

Big Data & Machine Learning

SÉANCE 4 - RÉSEAUX DE
NEURONES & NLP

Organisation

Séance 1

- Introduction sur l'évolution du Big Data et du Machine Learning
- Modélisation avec le Framework **Spark** (technique de Random Forest)
- Modélisation avec **XGBoost** (Gradient Boosting)

Séance 2

- Le pourquoi du Framework DASK?
- Utilisation de **DASK** via une régression logistique et un Gradient Boosting
- Introduction à l'explicabilité / interprétabilité des modèles (techniques Shap et Lime)
- Introduction à **TPOT** (algo évolutionniste génétique de sélection de modèles)

Séance 3

- Implémentation des techniques de clustering (KMeans, KNN, Spectral)
- Comparaison CPU vs GPU (Framework **RAPIDS** de Nvidia)

Séance 4

- Distinction des différentes méthodes de réseaux de neurones (Perceptron, CNN, RNN)
- Classification de texte, analyse de sentiment, NLP

Les réseaux de neurones

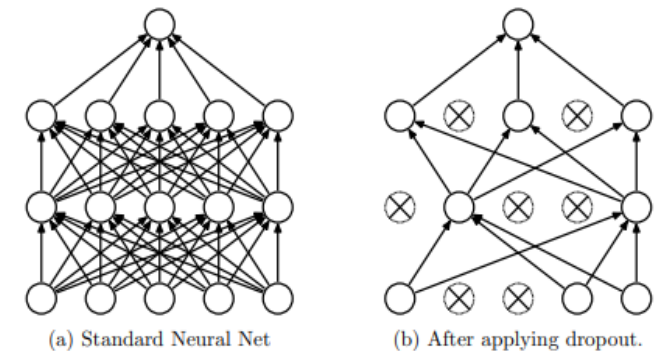
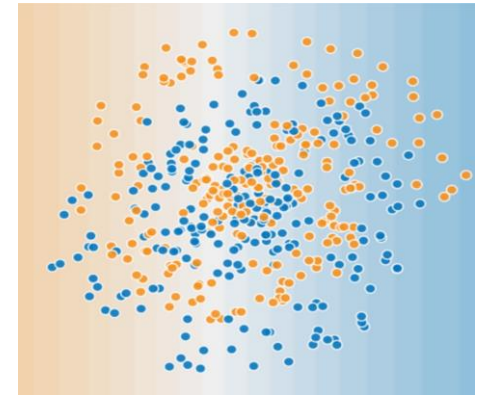
Introduction

Vise à répondre à un problème de classification non linéaire

Il n'est donc pas possible de prédire avec exactitude une étiquette avec un modèle de la forme $b + w_1x_1 + w_2x_2$

Quelques caractéristiques :

- Les poids sont initialisés selon une loi $N(0,1)$
- Les données en entrées doivent être normalisées (x-mean/std)
- A chaque **epoch**, plusieurs **batch** de données sont passés au NN : prédiction, feed forward
- Ensuite une **rétropropagation** est réalisée (descente de gradient), pour ajuster légèrement les poids de connexion, de façon à réduire l'erreur. Le gradient de l'erreur est donc propagée jusqu'à la première couche.
- Possibilité d'utiliser la fonction **drop_out** : supprime aléatoirement des unités de neurones, à chaque batch (évite l'overfitting, améliore les performances)
 - Il est conseillé de les utiliser sur les couches de fin, car les premières sont porteuses de beaucoup d'informations, les dernières peuvent être plus redondantes
- **Early stopping** également disponible pour arrêter l'entraînement si pas d'amélioration après X itérations, tout en permettant de conserver le meilleur modèle
- Le taux d'apprentissage



Les réseaux de neurones

PMC (Perceptron multicouche)

Le perceptron multicouche se compose d'une couche d'entrée, d'une couche de sortie et d'une ou plusieurs couches cachées.

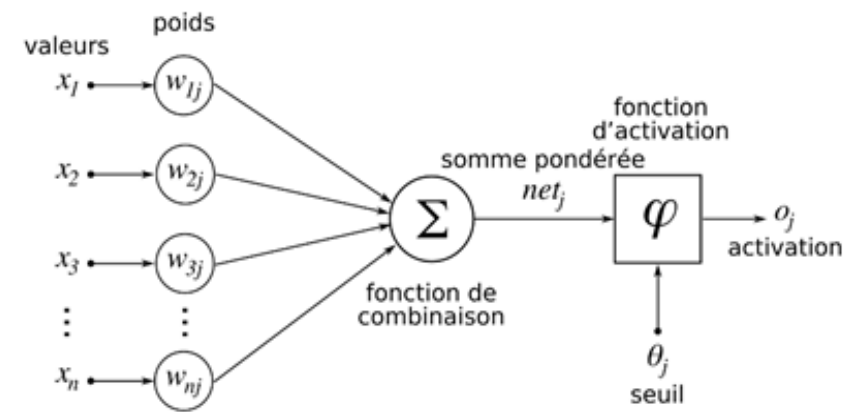
Chaque neurone de la couche cachée prend en entrée un vecteur de données, transforme les valeurs de la couche précédente en une somme pondérée, suivie d'une fonction d'activation non linéaire f (sigmoïde, tangente hyperbolique, RELU, ELU...)

$$Y = \sum(\text{weight} * \text{input}) + \text{bias}$$

En théorie, il suffit donc d'ajouter un nombre de neurones suffisant au niveau de la couche cachée pour approximer n'importe quelle fonction non linéaire.

Caractéristiques Keras :

- Ajout des différentes couches (sigmoïde, RELU, ELU...), appelées Dense()
- La première doit comporter le nombre de features en entrée
- Chacune doit comporter le nombre de neurones
- Ajout d'une couche/fonction **softmax** si besoin
- Ajout de l'**optimiseur** à utiliser (SDG, adam...), avec le taux d'apprentissage
- Ajout d'une fonction de coût / d'erreur : MSE, MAE...
- Ajout de métriques : accuracy...



Les réseaux de neurones

Les fonctions d'activation

Instabilité des gradients

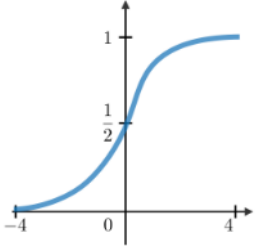
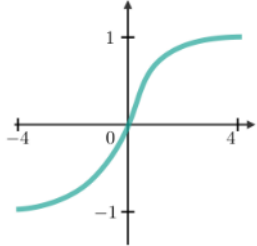
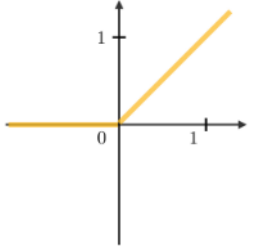
Disparitions des gradients

Explosion des gradients

A l'inverse, les gradients qui deviennent de plus en plus gros, où de nombreuses couches reçoivent des poids importants, ce qui fait diverger l'algorithme. Principalement pour les réseaux récurrents.

Mort des RELU

Il arrive aussi pour la fonction ReLU qu'au cours de l'entraînement, les neurones arrêtent de produire autre chose que 0. En particulier si fort taux d'apprentissage. Se produit si la descente de gradient modifie les poids d'un neurone qui donne une somme pondérée négative de ses entrées, celui-ci commencera à produire 0 en sortie.

Sigmoïde	Tanh	RELU
$g(z) = \frac{1}{1 + e^{-z}}$	$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	$g(z) = \max(0, z)$
		

Les réseaux de neurones

CNN (Convolution)

Le CNN est utilisé pour de la reconnaissance / classification d'images. Détection d'objets, reconnaissance faciale...

CNN classification prend une image en entrée, et la classe dans une catégorie (animal, objet, humain...) Il analysera une matrice de pixels, qui dépend de la résolution de l'image.

Caractéristiques :

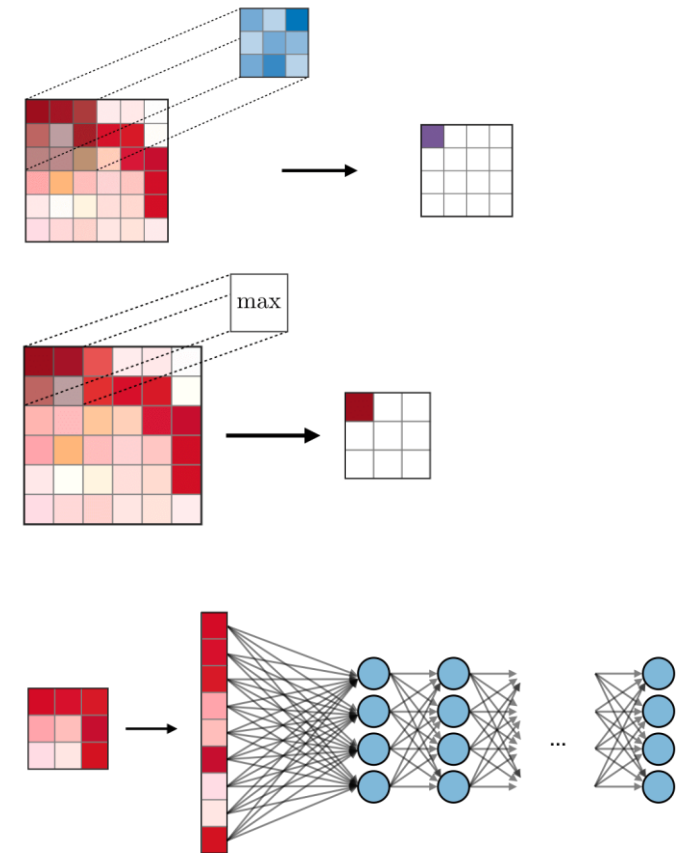
- **Convolution** : On recherche une ou des caractéristique(s) dans l'image et on dresse une matrice de correspondance (la caractéristique peut se trouver plusieurs fois dans l'image)
 - Un motif dans une image, la détection d'un pattern.
 - Utilisation d'une taille de fenêtre + pas de la fenêtre/avancement
- **Pooling** : On réduit la matrice de pixel, en ne gardant qu'à chaque fenêtre la valeur maximum (la plus importante) ou la valeur moyenne (de toutes les valeurs).
 - Utilisation d'une taille de fenêtre + pas de la fenêtre/avancement
- **Flatten** : mise à plat d'une matrice de valeurs en un vecteur pour la couche ReLU
- **ReLU** : Toute valeur négative est remplacée par 0
- Rétropropagation
- Les dernières couches après la mise à plat des matrices, peut s'apparenter à la méthode du perceptron

Les réseaux de neurones

CNN (Convolution)

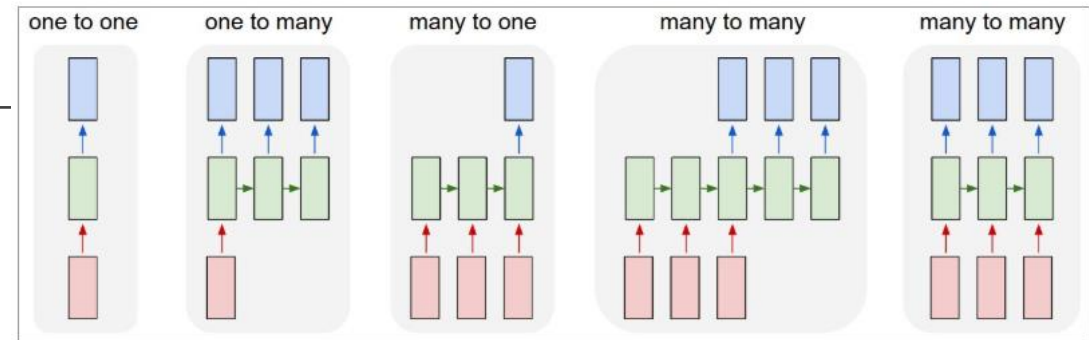
Les couches de convolution et de pooling représente le **preprocessing** pour les analyses d'images

- La couche de **convolution** peut être vu comme un filtre / produit scalaire (avec sa matrice de poids + biais)
- La couche de **pooling** est une opération de sous-échantillonnage, après une couche de convolution
 - Chaque opération de pooling sélectionne la valeur maximale de la surface
- La couche **flatten**, permet une mise à plat de la matrice en un vecteur, permettant une classification un NN classique



Les réseaux de neurones

RNN (Récurrent)

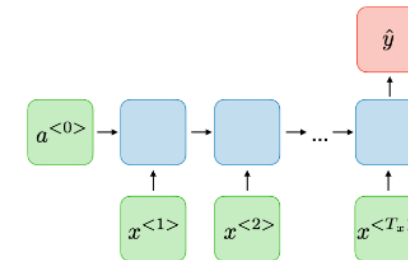


Création de séquences utilisées pour le traitement du langage naturel, ou la reconnaissance vocale (LSTM, GRU, Transformer...)

Classification de sentiment

Les étapes de preprocessing sont ici :

- En amont du NN : Tokenizer, stop words...
- **Embedding** : Création d'une relation géométrique entre les mots (vecteurs de mots), relation sémantique. Les mots relativement proches possèdent une forte relation
- Les sequences de **RNN**



NLP

Les bibliothèques utilisées



- **spaCy** : word2vec, NER, pretrained models (+60 languages), tokenisation, word vectors, lemmatisation, POS, sentiment analysis...
 - Plusieurs modules : *_core_news_sm, *_core_news_md, *_core_news_lg, *_core_news_trf (transformer) => différentes taille de key word, vectors...
- **NLTK** : stop words / punctuation list, tokenisation, NER, parse tree, API sur standford/coreNLP
 - Lib très bas niveau, là où spaCy permet de faire en quelques lignes ce que NLTK fait en 20 lignes
 - Toutes les fonctionnalités NLP sont quasiment implémentables via cette lib
- **stanfordnlp** / coreNLP : Pos tagging, tokenizer, lemmatisation, NER, dépendances entre les mots (suj / veb...), coréférences (Pierre, il, lui...)
- **Polyglot** : tokenisation, NER, sentiment analysis (polarité)...
- **Worldcloud** : Génération de nuages de mots à partir de fichier texte...
- **Gensim**



spaCy

NLP

Quelques méthodes

NER (name entity recognition) : catégorisation des mots dans des classes (nom, organisation, adresse, num tel, quantité, pourcentage, valeur monétaire...)

POS (part of speech tagger) : tag chaque mot avec des propriétés grammaticales (genre, nombre, [verbe, article, nom]...)

Word embedding / vecteurs de mots : les mots apparaissant dans des contextes similaires auront des vecteurs relativement proches dans un espace vectoriel (ex : chien et chat)

Elon Musk PERSON apparently wasn't aware that his company SpaceX had a Facebook ORG page. The SpaceX and Tesla PRODUCT CEO has responded to a comment on Twitter ORG calling for him to take down the SpaceX, Tesla and Elon Musk ORG official pages in support of the #deletefacebook movement by first ORDINAL acknowledging he didn't know one existed, and then following up with promises that he would indeed take them down.

He's done just that, as the SpaceX NORP Facebook page is now gone, after having been live earlier today DATE (as you can see from the screenshot included taken at around 12:10 PM ET) TIME .

NLP

Quelques méthodes

- **Stop Words** : retirer via un dictionnaire de mots, les mots communs porteur de peu d'informations : articles, conjonctions de coordination, pronoms...
- **TF-IDF** : importance d'un mot dans un document, relative à un corpus de documents

TF = $2/38$ (2 = nombre de mots python, sur 38 mots présents dans le document)

IDF = $\log(\text{base } 2) \ 3/2$: (3 documents total, et le mot python est présent que dans 2 documents)

Poids final :

- Doc 1 = $2/38 \cdot \log(3/2) = 0,0092$
- Doc 2 = $0/38 \cdot \log(3/2) = 0$
- Doc 3 = $1/40 \cdot \log(3/2) = 0,0044$

Le doc 1 est donc plus pertinent par rapport au mot python

Exercice

NLP avec spaCy

- [Analyser les tweets de Trump](#)
- [Lien du Notebook Google Colab](#)