**Algorithmes**

**Multi-model bibliography :** [**NLP 📝 GloVe, BERT, TF-IDF, LSTM... 📝 Explained (kaggle.com)**](https://www.kaggle.com/code/andreshg/nlp-glove-bert-tf-idf-lstm-explained) **-> contain full data process from cleaning to prediction/classification**

**Bert (transfert learning) - Résultat : emotions analyse from text**

code + logic architecture -> <https://www.kaggle.com/code/pavansanagapati/knowledge-graph-nlp-tutorial-bert-spacy-nltk>

[**Fast-Bert**](https://github.com/utterworks/fast-bert) is the deep learning library that allows developers and data scientists to train and deploy BERT and XLNet based models for natural language processing tasks beginning with Text Classification.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Zero-shot-classifier (transfert learning) - Résultat extract questions response directly from user text**

Notebook : [NLP.ipynb](https://colab.research.google.com/drive/1O7wruI6SQuE5iG5BaNepezA6s-3edybA?usp=sharing)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**RNN (deep learning) - Résultat : extract NER (**[**type d’entités**](https://editor.analyticsvidhya.com/uploads/19617Intro%20image.jpg)**)**

* **Architecture :** [**Recurrent Neural Network (RNN) Architecture Explained | by Sushmita Poudel | Medium**](https://medium.com/@poudelsushmita878/recurrent-neural-network-rnn-architecture-explained-1d69560541ef)
* **Currently work on this model :** [**shuwang127/NLP-RNN: Recurrent Neural Network for Natural Language Processing (github.com)**](https://github.com/shuwang127/NLP-RNN)
* **Result :** [RNN](https://drive.google.com/drive/folders/1jUCslpRJWzJ7ZIOAvBfkMVHozV6CGSQy?usp=sharing)**(NER classification) - we add so text classification model** [***Emotions classification with BERT***](https://www.kaggle.com/code/debarshichanda/bert-multi-label-text-classification)
* **54617362 num fehima**

**DNN (deep learning)**

steps / code → [Text Classification with Neural Networks - Atmosera](https://www.atmosera.com/blog/text-classification-with-neural-networks/) /

Définition: [What is Deep Neural Network (DNN)? | Deci](https://deci.ai/deep-learning-glossary/deep-neural-network-dnn/) /

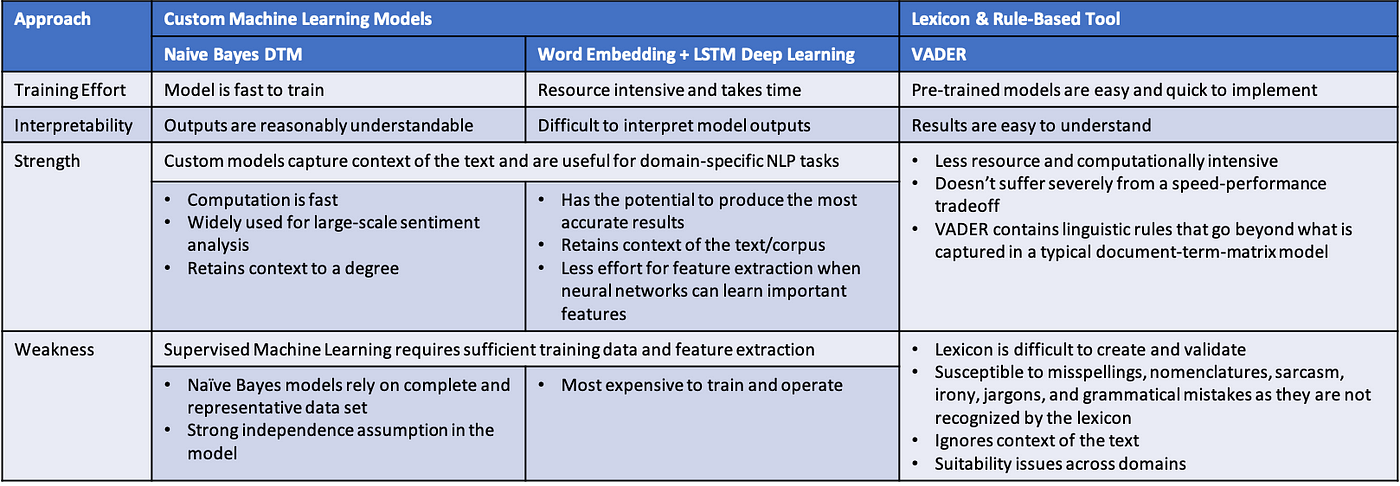
**Définition des termes NLP** : [Natural Language Processing (NLP) : Définition et principes (datascientest.com)](https://datascientest.com/introduction-au-nlp-natural-language-processing)

* [What are Deep Neural Networks?](https://www.datacamp.com/tutorial/introduction-to-deep-neural-networks#what-are-deep-neural-networks?-anart)
* [Why use a DNN?](https://www.datacamp.com/tutorial/introduction-to-deep-neural-networks#why-use-a-dnn?-after)
* [Deep Learning Tools and Libraries](https://www.datacamp.com/tutorial/introduction-to-deep-neural-networks#deep-learning-tools-and-libraries-const)
* [How to Build a Basic Deep Neural Network](https://www.datacamp.com/tutorial/introduction-to-deep-neural-networks#how-to-build-a-basic-deep-neural-network-inthi)
* [Understanding Deep Learning Models](https://www.datacamp.com/tutorial/introduction-to-deep-neural-networks#understanding-deep-learning-models-compu)
* [Challenges and Considerations in Deep Neural Network](https://www.datacamp.com/tutorial/introduction-to-deep-neural-networks#challenges-and-considerations-in-deep-neural-networks-wehav)
* **->** [**Introduction to Deep Neural Networks | DataCamp**](https://www.datacamp.com/tutorial/introduction-to-deep-neural-networks)
* **1-** [**BERT VS DNN Performance Colab (work to fix that with real example code)**](https://colab.research.google.com/drive/1KIzw-QtZ0zU9iOiH6AGjIS60LD1GTO05?usp=sharing)
* **2- DNN Simple codes**
* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
* **Articles**
* \* Transfert Learning ([Bert](https://arxiv.org/pdf/1810.04805.pdf))
* \* ([RNN](https://drive.google.com/file/d/16BiFJqEUXMq0uxxIRbGnRLWaKu-bPAap/view?usp=sharing) / [RNN-old](https://www.isca-archive.org/interspeech_2013/yao13b_interspeech.pdf) / [Netflix use case](https://drive.google.com/file/d/1MgzicMaD5VHG9dyGBal_W7nN9AZD8zHm/view?usp=sharing))
* \* Article DNN (<https://arxiv.org/pdf/1904.09535.pdf>)
* \* Zero-shot-classifier : [Zero-Shot Classification Using Transformers: Unlocking the Power of AI for Text-Based Tasks | by DataScience-ProF | Medium](https://medium.com/@TheDataScience-ProF/zero-shot-classification-using-transformers-unlocking-the-power-of-ai-for-text-based-tasks-e5118398ef17#:~:text=Zero%2Dshot%20classification%20is%20a,from%20those%20classes%20during%20training.)
* \* Chatbot using NLP : [CREATION OF A CHATBOT BASED ON NATURAL LANGUAGE PROCESSING FOR WHATSAPP (arxiv.org)](https://arxiv.org/pdf/2310.10675)
* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
* [Speech and Language Processing - Stanford University](https://drive.google.com/file/d/1gnNwtjf9NEuJaJSf0viM9sXwV_Hz_hN7/view?usp=sharing)
* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

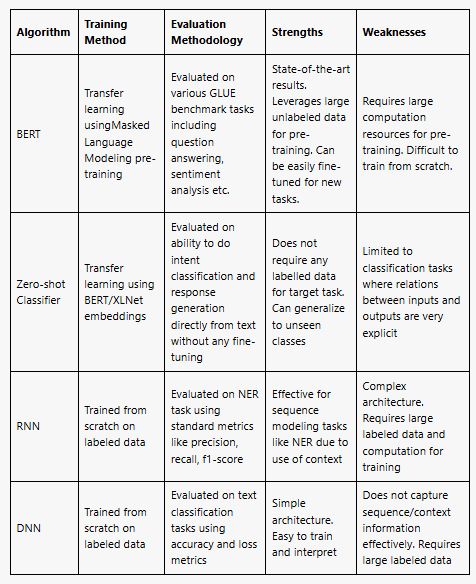
# **Strengths and weaknesses of common machine learning models**

* [Strengths and weaknesses of common machine learning models | Download ScientificDiagram (researchgate.net)](https://www.researchgate.net/figure/Strengths-and-weaknesses-of-common-machine-learning-models_tbl1_341573546)
* **Methodology Crisp-DM (Rapport Methodology):** [**What is CRISP DM? - Data Science Process Alliance (datascience-pm.com)**](https://www.datascience-pm.com/crisp-dm-2/)
* **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***
* **About NLP AND BART:**
* NLP deals with enabling machines to understand human language in text or speech form. This involves tasks like text classification, sentiment analysis, information retrieval, question answering, etc.
* BERT (Bidirectional Encoder Representations from Transformers) is a state-of-the-art NLP model developed by Google. It applies bidirectional training of the Transformer attention model.
* Previous models analyzed text sequences either left-to-right or with a combined left-right and right-left approach. BERT allows truly bidirectional training which gives it a deeper understanding of language context.
* BERT uses a technique called Masked LM where some tokens in the input are randomly masked and the model tries to predict the original values based on context. This allows bidirectional training that was previously not possible.
* BERT has achieved state-of-the-art results on many NLP tasks like question answering, natural language inference, etc. demonstrating the power of its approach.
* The key innovation of BERT is applying bidirectional training of the Transformer model to language modeling, compared to prior work that only looked at the text sequence in one direction. This allows it to better capture context and flow of language.
* **Second Step: How to apply choosen model in our user case (3d products recommendation + NLP)**

Text Classification approach

****

**Lien Tableau Comparatif pour les 4 algo :** [**https://poe.com/s/qAy7alGqYNYZzLNgFygB**](https://poe.com/s/qAy7alGqYNYZzLNgFygB)

****

def clean\_text(text):

'''Make text lowercase, remove text in square brackets,remove links,remove punctuation

and remove words containing numbers.'''

text = str(text).lower()

text = re.sub('\[.\*?\]', '', text)

text = re.sub('https?://\S+|www\.\S+', '', text)

text = re.sub('<.\*?>+', '', text)

text = re.sub('[%s]' % re.escape(string.punctuation), '', text)

text = re.sub('\n', '', text)

text = re.sub('\w\*\d\w\*', '', text)

return text

def preprocess\_data(text):

# Clean puntuation, urls, and so on

text = clean\_text(text)

# Remove stopwords

text = ' '.join(word for word in text.split(' ') if word not in stop\_words)

# Stemm all the words in the sentence

text = ' '.join(stemmer.stem(word) for word in text.split(' '))

return text

**Target-encoding:**

from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

le.fit(df['target'])

df['target\_encoded'] = le.transform(df['target'])

df.head()

**Top Tokens Visualisation (words in the imputed text)**

twitter\_mask = np.array(Image.open('/kaggle/input/masksforwordclouds/twitter\_mask3.jpg'))

wc = WordCloud(

background\_color='white',

max\_words=200,

mask=twitter\_mask,

)

**wc.generate(' '.join(text for text in df.loc[df['target'] == 'ham', 'message\_clean']))**

**//this line will generate words**

plt.figure(figsize=(18,10))

plt.title('Top words for HAM messages',

fontdict={'size': 22, 'verticalalignment': 'bottom'})

plt.imshow(wc)

plt.axis("off")

plt.show()

| Algorithme | Méthode d'entraînement | Méthodologie d'évaluation | Forces | Faiblesses |
| --- | --- | --- | --- | --- |
| BERT | Apprentissage transféré en utilisant le pré-entraînement du modèle de langage masqué | Évalué sur diverses tâches de référence GLUE incluant question-réponse, analyse de sentiment etc. | Meilleurs résultats de l'état de l'art. Tire parti des grandes données non étiquetées pour le pré-entraînement. Peut être facilement ré-entraîné pour de nouvelles tâches. | Nécessite d'importantes ressources de calcul pour le pré-entraînement. Difficile à entraîner à partir de zéro. |
| Classifieur zéro-shot | Apprentissage transféré en utilisant les plongements BERT/XLNet | Évalué sur la capacité à effectuer la classification d'intentions et la génération de réponses directement à partir du texte sans ré-entraînement | Ne nécessite aucune donnée étiquetée pour la tâche cible. Peut généraliser à des classes non vues | Limité aux tâches de classification où les relations entre les entrées et les sorties sont très explicites |
| RNN | Entraîné à partir de zéro sur des données étiquetées | Évalué sur la tâche NER en utilisant les métriques standards comme la précision, le rappel, le F1-score | Efficace pour la modélisation de séquences comme NER grâce à l'utilisation du contexte | Architecture complexe. Nécessite de grandes données étiquetées et de calculs pour l'entraînement |
| DNN | Entraîné à partir de zéro sur des données étiquetées | Évalué sur les tâches de classification de texte en utilisant la précision et les métriques de perte | Architecture simple. Facile à entraîner et à interpréter | Ne capture pas efficacement l'information de séquence/contexte. Nécessite de grandes données étiquetées |