

# Réalisez un dashboard et assurez une veille technique

Élaborer un dashboard à partir d'une API créée précédemment  
Présenter un algorithme récent

Date de la soutenance : 04/06/2024

Antoine Arragon





# Objectifs

Répondre à 2 besoins de la société Prêt à Dépenser :

## 1 - Créer un tableau de bord

- A partir de la modélisation et de l'API effectuées précédemment;
- Visualisation aisée de la prédiction de défaut et du statut de la demande de crédit ;
- Informations supplémentaires sur le profil client ;
- Explication (globale et locale) de la prédiction du modèle

## 2 - Présenter une technique récente de NLP ou traitement d'images

- Se baser sur un jeu de données et un travail précédent ;
- Présenter les concepts importants ;
- Comparer avec une approche plus classique ;
- Réaliser une note méthodologique

Plan :



1. Présentation du dashboard
  - 1.1. Rappel des étapes précédentes : données - modélisation - API initiale
  - 1.2. Besoins du client
  - 1.3. Présentation du dashboard et de son déploiement
  
2. Présentation de l'algorithme RoBERTa
  - 2.1. Explication des principaux concepts de l'algorithme
  - 2.2. Présentation rapide du jeux de données et du travail précédent
  - 2.3. Proof of concept et comparaison avec BERT

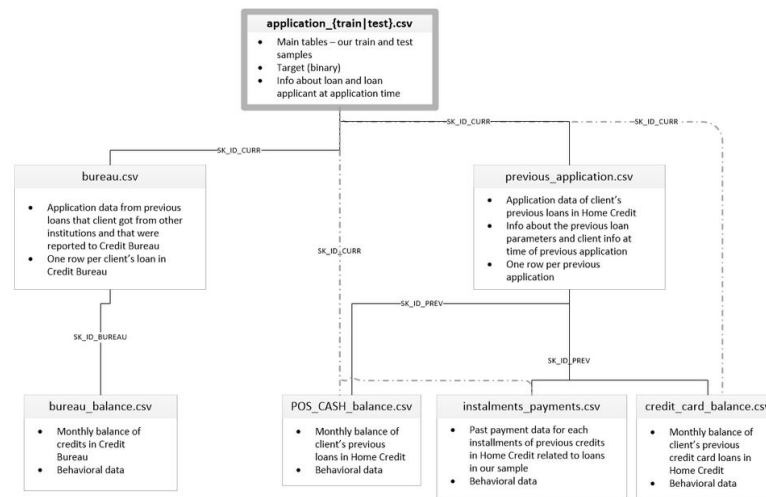
Limites et conclusion

# 1. Présentation du dashboard

## 1.1. Rappel des étapes précédentes : données - modélisation - API initiale

### Point rapide sur les données :

- Deux grands types d'informations :
  - Administratives : données relatives à l'âge, au sexe, au métier... des clients ;
  - Bancaires :
    - Informations provenant de l'historique interne de HomeCredit ;
    - Informations provenant d'autres institutions financières
- Regroupement des données par client, via différentes agrégations de variables;
- Dimensions du jeu de données final : 307505 lignes (clients) et 287 colonnes (variables).





# 1. Présentation du dashboard

## 1.1. Rappel des étapes précédentes : données - modélisation - API initiale

### Modélisation :

- Critère sélection modèle et hyperparamètres : coût minimal des erreurs de prédictions (en particulier des faux négatifs) ;
- Hypothèse : faux négatif 10 fois plus coûteux qu'un faux positif ;
- Meilleur résultat : LGBMClassifier ;
- Calibration ➤ confiance accrue dans probabilités prédites

### API initiale :

- Requêtes à partir d'ID clients ;
- Récupère prédictions du modèle ;
- Affiche probabilité de défaut & statut du crédit ;
- Affiche également quelques informations sur le profil client;



# 1. Présentation du dashboard

## 1.1. Rappel des étapes précédentes : données - modélisation - API initiale

Lien vers l'API déployée sur heroku : <https://application-credit-7ba79bc598e5.herokuapp.com/>

Route vers ID clients :

Informations renvoyées par l'API initiale :

← → ↻ [https://application-credit-7ba79bc598e5.herokuapp.com/list\\_ids](https://application-credit-7ba79bc598e5.herokuapp.com/list_ids)

Liste des id clients valides : [337756, 405321, 142233, 443286, 331778, 150748, 339028, 442617, 381637, 120657, 195799, 361221, 337439, 381855, 159897, 378516, 304463, 423094, 353064, 154762, 123258, 24 124477, 298554, 264884, 140623, 143312, 185990, 381321, 112205, 233457, 189277, 213028, 364657, 15 296178, 156281, 340541, 308300, 172233, 448307, 231498, 140578, 151772, 298689, 200090, 101538, 21 226588, 261042, 340475, 351618, 238993, 140713, 318894, 148125, 347327, 140773, 302269, 229582, 31 206191, 374271, 347597, 280523, 422795, 120568, 440779, 137460, 337904, 140667, 233152, 215850, 13 146780, 344045, 216425, 376221, 133141, 187491, 381175, 294349, 231643, 402106, 302416, 243450, 24 427475, 426583, 416568, 205840, 348278, 452997, 181361, 432898, 358569, 225279, 162283, 334891, 13 174499, 315183, 261708, 177303, 378768, 138242, 345828, 245404, 163722, 387439, 423716, 188162, 38

← → ↻ <https://application-credit-7ba79bc598e5.herokuapp.com/prediction/184348>

JSON Données brutes En-têtes

Enregistrer Copier Tout réduire Tout développer Filtre le JSON

▼ client\_infos:

- education: "Higher education"
- montant\_credit: 646920
- ratio\_revenu\_credit: 11.83
- revenu: 76500
- sexe: "F"
- source\_revenu: "Working"
- statut\_famille: "Married"
- âge: 26
- id: 184348
- probabilité\_défaut: 0.18
- statut: "non risqué"



# 1. Présentation du dashboard

## 1.2. Besoins du client

1. Transparence de la décision d'octroi ou non du crédit ;
2. Interface lisible et claire, information facilement accessible ;
3. Visualiser les principales informations descriptives relatives à un client ;
4. Interprétation des décisions du modèle ;
5. Comparaison du client sélectionné avec l'ensemble des clients ;
6. Informations sur le jeu de données



# 1. Présentation du dashboard

## 1.3. Présentation du dashboard et de son déploiement

Première étape : ajout des différents éléments demandés dans une API Flask plus complète (backend)

Deuxième étape : création d'un script différents pour l'application Streamlit (frontend)

- fait appel à l'API Flask, via son URL, pour récupérer les informations nécessaires ;
- Utilisation des différents outils de Streamlit pour créer une interface interactive :
  - liste déroulante ;
  - boutons ou case à cocher ;
  - mise en place d'une sidebar pour des informations toujours visibles et lisibles;

Troisième étape : déploiement via Github Actions sur Heroku

- <https://dashboard-credit-app-85ce4e23fc73.herokuapp.com/>



# 1. Présentation du dashboard

## 1.3. Présentation du dashboard et de son déploiement

Exemple de graphiques interactifs :

Top 9 des variables les plus importantes pour l'individu sélectionné



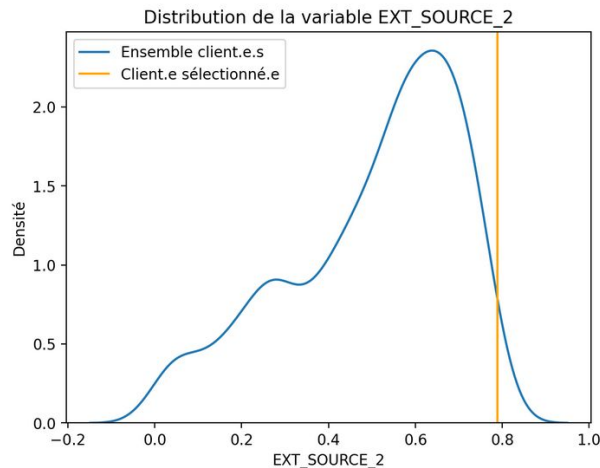
L'explication SHAP (SHapley Additive exPlanations) est une méthode de décomposition des prédictions de modèles de machine learning. Elle attribue une valeur d'importance à chaque variable pour chaque prédiction. Les valeurs SHAP positives indiquent une contribution positive à la prédiction (probabilité de défaut plus élevée), tandis que les valeurs négatives indiquent une contribution négative (probabilité de défaut moins élevée). Plus la valeur SHAP est élevée, plus la variable a un impact sur la prédiction. Les graphiques ci-dessus montrent les 10 variables les plus importantes pour le modèle et les 9 variables les plus importantes pour l'individu sélectionné.

Veillez sélectionner une variable à comparer

EXT\_SOURCE\_2

Pour l'individu 443286, la valeur de EXT\_SOURCE\_2 est 0.79. Moyenne pour l'ensemble des client.e.s : 0.5.

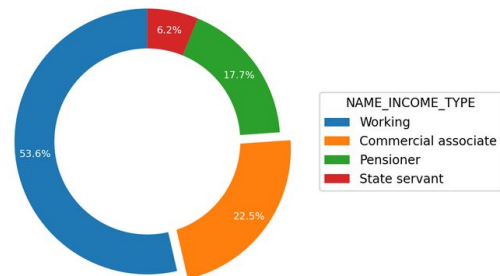
Les différences de valeurs pour une même variable entre ce graphique et le graphique précédent sont dues à la normalisation des données nécessaire à la modélisation.



Veillez sélectionner une variable à comparer

NAME\_INCOME\_TYPE

Répartition de la variable NAME\_INCOME\_TYPE





# **1. Présentation du dashboard**

## **1.3. Présentation du dashboard et de son déploiement**

Présentation du dashboard et de ses différents éléments et graphiques

## 2. Présentation de l'algorithme RoBERTa

### 2.1. Présentation des principaux concepts de l'algorithme

L'architecture Transformer (2017) :

- Encoder :
  - 6 layers avec 2 sub-layers pour chaque :
    - Multi-head attention
    - Feed Forward
- Decoder :
  - 6 layers avec 3 sub-layers pour chaque :
    - Multi-head attention
    - Masked Multi-head attention
    - Feed Forward

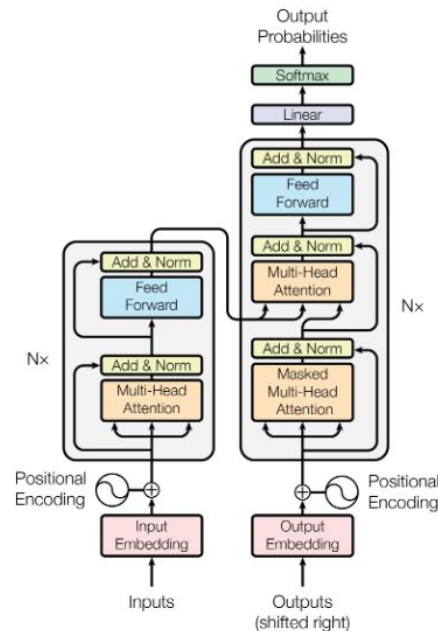


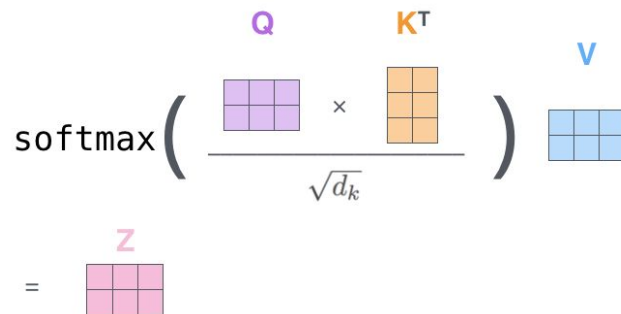
Figure 1: The Transformer - model architecture.

## 2. Présentation de l'algorithme RoBERTa

### 2.1. Présentation des principaux concepts de l'algorithme

Focus sur l'attention :

- 1) Word Embedding
- 2) Pour chaque mot on calcule 3 vecteurs en multipliant l'embedding par des matrices dont les poids sont initialisés aléatoirement :
  - a) Q : Queries
  - b) K : Keys
  - c) V : Values
- 3) Produit scalaire Q.K (ex :  $q_1 \cdot k_i$ ) = score
- 4) Normalisation du score par softmax
- 5) Multiplication de chaque vecteur v par le score normalisé
- 6) Addition des vecteurs de valeurs pondérés.


$$\text{softmax} \left( \frac{\begin{matrix} \text{Q} \\ \text{3x3 matrix} \end{matrix} \times \begin{matrix} \text{K}^T \\ \text{3x3 matrix} \end{matrix}}{\sqrt{d_k}} \right) \begin{matrix} \text{V} \\ \text{3x3 matrix} \end{matrix} = \begin{matrix} \text{Z} \\ \text{3x3 matrix} \end{matrix}$$

The self-attention calculation in matrix form

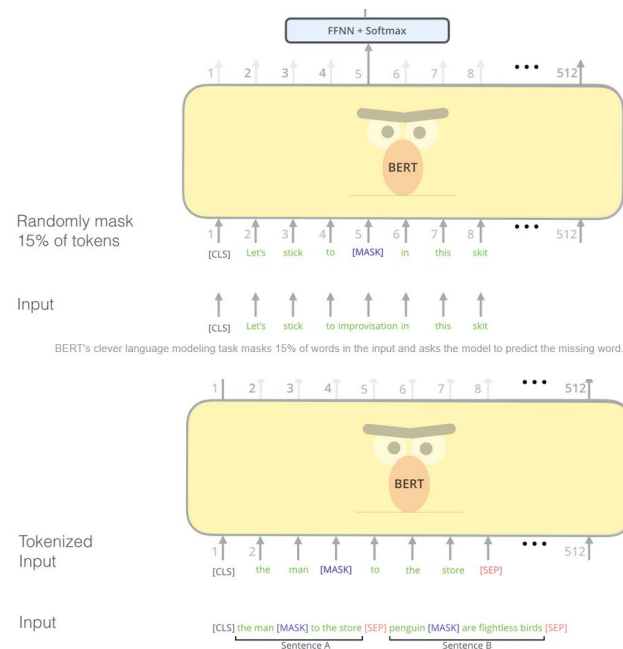
Source :  
<http://jalammar.github.io/illustrated-transformer/>

## 2. Présentation de l'algorithme RoBERTa

### 2.1. Présentation des principaux concepts de l'algorithme

Le modèle BERT (2018) :

- 1) Focus sur la compréhension du contexte du langage ;
- 2) Basé sur l'architecture Transformer mais n'utilise que des encodeurs (12 pour BERT-base et 24 pour BERT-large ;
- 3) Pré-entraînement conséquent et sur 2 tâches :
  - a) Masked language Model
  - b) Next Sentence prediction
- 4) Meilleurs résultats que modèles précédents sur de nombreuses tâches de NLP



Source : <https://jalamar.github.io/illustrated-bert/>

## 2. Présentation de l'algorithme RoBERTa

### 2.1. Présentation des principaux concepts de l'algorithme

Le modèle RoBERTa (2019), une optimisation du modèle BERT :

- 1) 10 fois plus de données d'entraînement (160 Gb vs 16) ;
- 2) Encodage tokens différents - vocabulaire + riche ;
- 3) MLM dynamique ;
- 4) Batch\_size plus élevé lors de l'entraînement ;
- 5) Suppression de la tâche de NSP.

Hyperparam	RoBERTa <sub>LARGE</sub>	RoBERTa <sub>BASE</sub>
Number of Layers	24	12
Hidden size	1024	768
FFN inner hidden size	4096	3072
Attention heads	16	12
Attention head size	64	64
Dropout	0.1	0.1
Attention Dropout	0.1	0.1
Warmup Steps	30k	24k
Peak Learning Rate	4e-4	6e-4
Batch Size	8k	8k
Weight Decay	0.01	0.01
Max Steps	500k	500k
Learning Rate Decay	Linear	Linear
Adam $\epsilon$	1e-6	1e-6
Adam $\beta_1$	0.9	0.9
Adam $\beta_2$	0.98	0.98
Gradient Clipping	0.0	0.0

Table 9: Hyperparameters for pretraining RoBERTa<sub>LARGE</sub> and RoBERTa<sub>BASE</sub>.

	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	WNLI	Avg
<i>Single-task single models on dev</i>										
BERT <sub>LARGE</sub>	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
XLNet <sub>LARGE</sub>	89.8/-	93.9	91.8	83.8	95.6	89.2	63.6	91.8	-	-
RoBERTa	<b>90.2/90.2</b>	<b>94.7</b>	<b>92.2</b>	<b>86.6</b>	<b>96.4</b>	<b>90.9</b>	<b>68.0</b>	<b>92.4</b>	<b>91.3</b>	-
<i>Ensembles on test (from leaderboard as of July 25, 2019)</i>										
ALICE	88.2/87.9	95.7	<b>90.7</b>	83.5	95.2	92.6	<b>68.6</b>	91.1	80.8	86.3
MT-DNN	87.9/87.4	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0	87.6
XLNet	90.2/89.8	98.6	90.3	86.3	<b>96.8</b>	<b>93.0</b>	67.8	91.6	<b>90.4</b>	88.4
RoBERTa	<b>90.8/90.2</b>	<b>98.9</b>	90.2	<b>88.2</b>	96.7	92.3	67.8	<b>92.2</b>	89.0	<b>88.5</b>

## 2. Présentation de l'algorithme RoBERTa

### 2.2. Présentation rapide du jeu de données et du travail précédent

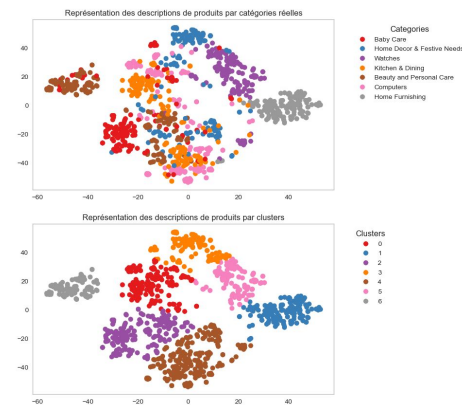
#### 1. Jeu de données :

- Fichier csv contenant des données de l'entreprise "Flipkart" ;
- 1050 produits répartis en équitablement en 7 catégories (150 pdts/catégorie) ;
- Regroupement du nom des produits et de leur description puis encodage des catégories :

	name_desc_product_clean	categorie_encoded
0	Elegance Polyester Multicolor Abstract Eyelet ...	4
1	Sathiyas Cotton Bath Towel Specifications of S...	0
2	Eurospa Cotton Terry Face Towel Set Key Featur...	0
3	SANTOSH ROYAL FASHION Cotton Printed King size...	4
4	Jaipur Print Cotton Floral King sized Double B...	4

#### 2. Etude de faisabilité - classification

- Introduction au preprocessing de textes ;
- Comparaison approches BoW et Word Embedding - 1e utilisation de BERT ;
- Extractions de features et clustering :





## 2. Présentation de l'algorithme RoBERTa

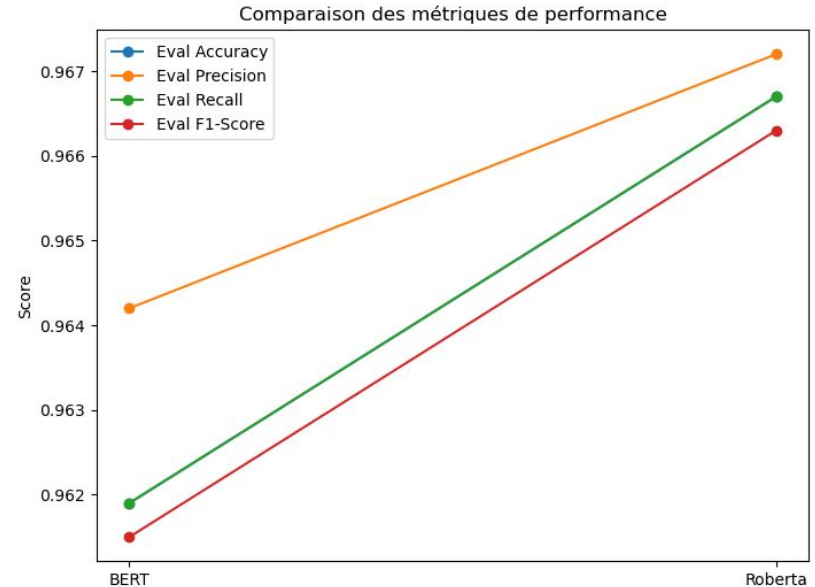
### 2.3. Proof of Concept et comparaison avec BERT

#### Problématique :

- Classification supervisée - prédire les catégories de produits à partir de leur nom et description ;
- Comparaison de 2 modèles : BERT et RoBERTa ;
- Métriques utilisées : accuracy, précision, rappel, f1\_score ;

#### Etapes :

- Nettoyage rapide du texte ;
- Tokenisation ;
- Configuration - Entraînement des modèles ;
- Evaluation





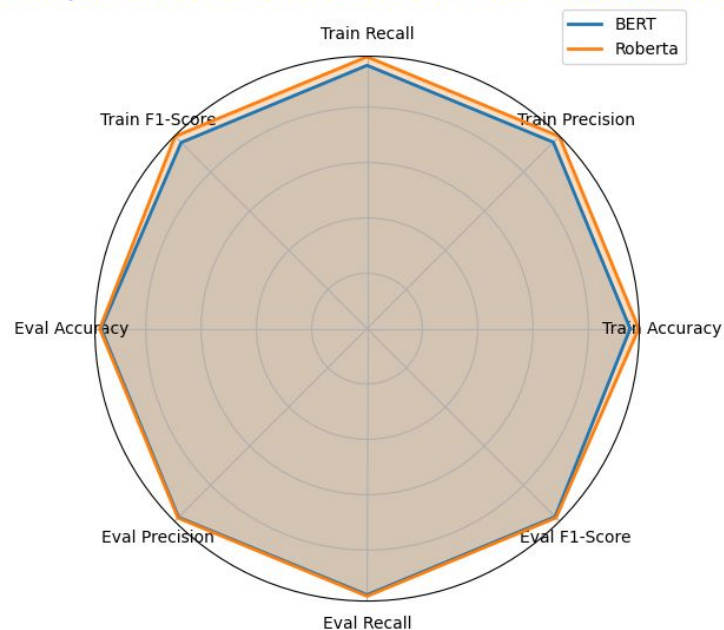


## 2. Présentation de l'algorithme RoBERTa

### 2.3. Proof of Concept et comparaison avec BERT

Pour cette tâche de classification, les résultats ont été bons et sensiblement similaires entre les 2 modèles :

#### Comparaison des modèles (BERT vs RoBERTa)





## Limites et conclusion

Partie tableau de bord :

- Manque d'informations sur besoins réels des usagers ➤ réunions nécessaires ;
- Sans doute plus de lisibilité sur différentes pages ;
- Améliorations nécessaires sur l'accessibilité du WCAG.

Partie Veille technique :

- Manque de connaissances du domaine et besoin d'approfondissement ;
- Recherche d'hyperparamètres et adaptation au problème spécifique ;
- L'interprétabilité est manquante ;
- D'autres modèles à tester ?

Conclusion : s'agissant du tableau de bord ou de la partie veille technique, il s'agit d'une ébauche, d'une première itération, qu'il sera nécessaire de retravailler et approfondir.

Merci de votre attention.





# Bibliographie / Webographie

## Articles initiaux :

- "Attention Is All You Need" - 2017 : <https://arxiv.org/abs/1706.03762>
- "RoBERTa: A Robustly Optimized BERT Pretraining Approach" - 2019 : <https://arxiv.org/abs/1907.11692>
- "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" - 2018 : <https://arxiv.org/abs/1810.04805>

## Ressources HuggingFace :

- [https://huggingface.co/docs/transformers/model\\_doc/roberta](https://huggingface.co/docs/transformers/model_doc/roberta)
- <https://huggingface.co/tasks/text-classification>
- [https://colab.research.google.com/github/DhavalTaunk08/NLP\\_scripts/blob/master/sentiment\\_analysis\\_using\\_roberta.ipynb](https://colab.research.google.com/github/DhavalTaunk08/NLP_scripts/blob/master/sentiment_analysis_using_roberta.ipynb)
- [https://colab.research.google.com/github/huggingface/notebooks/blob/main/examples/text\\_classification.ipynb#scrollTo=Bliy8zgjlRjY](https://colab.research.google.com/github/huggingface/notebooks/blob/main/examples/text_classification.ipynb#scrollTo=Bliy8zgjlRjY)
- <https://huggingface.co/blog/bert-101>
- [https://huggingface.co/docs/transformers/v4.41.2/en/model\\_doc/bert#transformers.BertModel](https://huggingface.co/docs/transformers/v4.41.2/en/model_doc/bert#transformers.BertModel)
- <https://huggingface.co/learn/nlp-course/chapter0/1>

## Articles de blogs :

- <https://machinelearningmastery.com/how-does-attention-work-in-encoder-decoder-recurrent-neural-networks/>
- <https://machinelearningmastery.com/natural-language-processing/>
- <https://machinelearningmastery.com/the-transformer-model/>
- <https://machinelearningmastery.com/what-is-attention/>
- <https://machinelearningmastery.com/the-attention-mechanism-from-scratch/>
- <https://machinelearningmastery.com/encoder-decoder-recurrent-neural-network-models-neural-machine-translation/>
- <https://machinelearningmastery.com/a-brief-introduction-to-bert/>
- <https://towardsdatascience.com/contextual-transformer-embeddings-using-self-attention-explained-with-diagrams-and-python-code-d7a9f0f4d94e>
- <https://towardsdatascience.com/a-complete-guide-to-bert-with-code-9f87602e4a11>
- <https://jalammar.github.io/illustrated-bert/>
- <http://jalammar.github.io/illustrated-transformer/>
- <https://lesdieuxducode.com/blog/2019/4/bert--le-transformer-model-qui-sentraine-et-qui-represente>