DataScientest • com

Rapport Technique d'évaluation

Classification et extraction d'informations d'un document

Promotion : DATA SCIENTIST – Avril 2022

Participants :

Rym BEN HASSINE
Ferhat SADOUN
Bogdan POSEA
Eric GASNIER

Contexte

Description : Déclaration d'un sinistre sur le chatbot du site d'un assureur.

Certaines tâches peuvent être automatisées.

Etapes :

1) Déclenchement du service de déclaration d'un sinistre suivant le texte tapé dans le chatbot.
2) Vérification de l'identité de l'utilisateur :
    a. Le service demande un papier d'identité à l'utilisateur.
    b. Le système vérifie le papier d'identité (Carte d'Identité ou Passeport).
    c. Le système vérifie la date de validité de la pièce d'identité présentée.
    d. Le système récupère les noms/prénoms/etc… au format requis par le LCBFT (Lutte Contre le Blanchiment et le Financement du Terrorisme)
    e. L'assureur vérifie qu'il n'y a pas de soucis avec le registre correspondant.
3) Dépôt de la déclaration du sinistre et extraction des informations :
    a. Le service demande le dépôt de la déclaration.
    b. Le système vérifie le document. Est-ce bien le bon formulaire ?
    c. Le système extrait les informations importantes du document et formate les données selon un certain modèle.

Du point de vue technique :

Ce projet requiert des connaissances en extraction de textes et en classification de nature de documents.

Deux approches sont possibles :

1) Océrisation des documents (Reconnaissance Optique de Caractère - OCR), suivi d'un Traitement automatique du langage naturel (Natural Language Processing – NLP).
2) Traitement à partir d'images (Computer Vision – CV).

Ensuite, un traitement du texte des documents classifiés est nécessaire pour récupérer les informations recherchées.

Du point de vue économique :

L'intérêt d'utiliser l'intelligence artificielle dans ce cas est double :

1) L'opérateur chez l'assureur chargé de recueillir les informations effectue des tâches répétitives sans réelles valeurs ajoutées. L'opérateur peut ainsi se concentrer sur des tâches plus valorisantes.
2) Les tâches étant automatisées, elles génèrent un gain financier pour l'assureur.

Objectifs

Afin de définir un contexte réel, nous sommes entrés en contact avec une personne travaillant pour un grand groupe d'assurances. Le contexte choisi correspond à un appel récent d’offre de ce groupe.

Nous nous sommes donnés comme objectif de classifier au mieux les documents du jeu de données fournis en mettant l’accent sur la classification des pièces d’identité (cartes d’identité et passeports).

A travers cet objectif, nous avons cherché à explorer les diverses approches nous permettant la classification de documents, que ce soit par traitement du langage ou par Computer Vision.

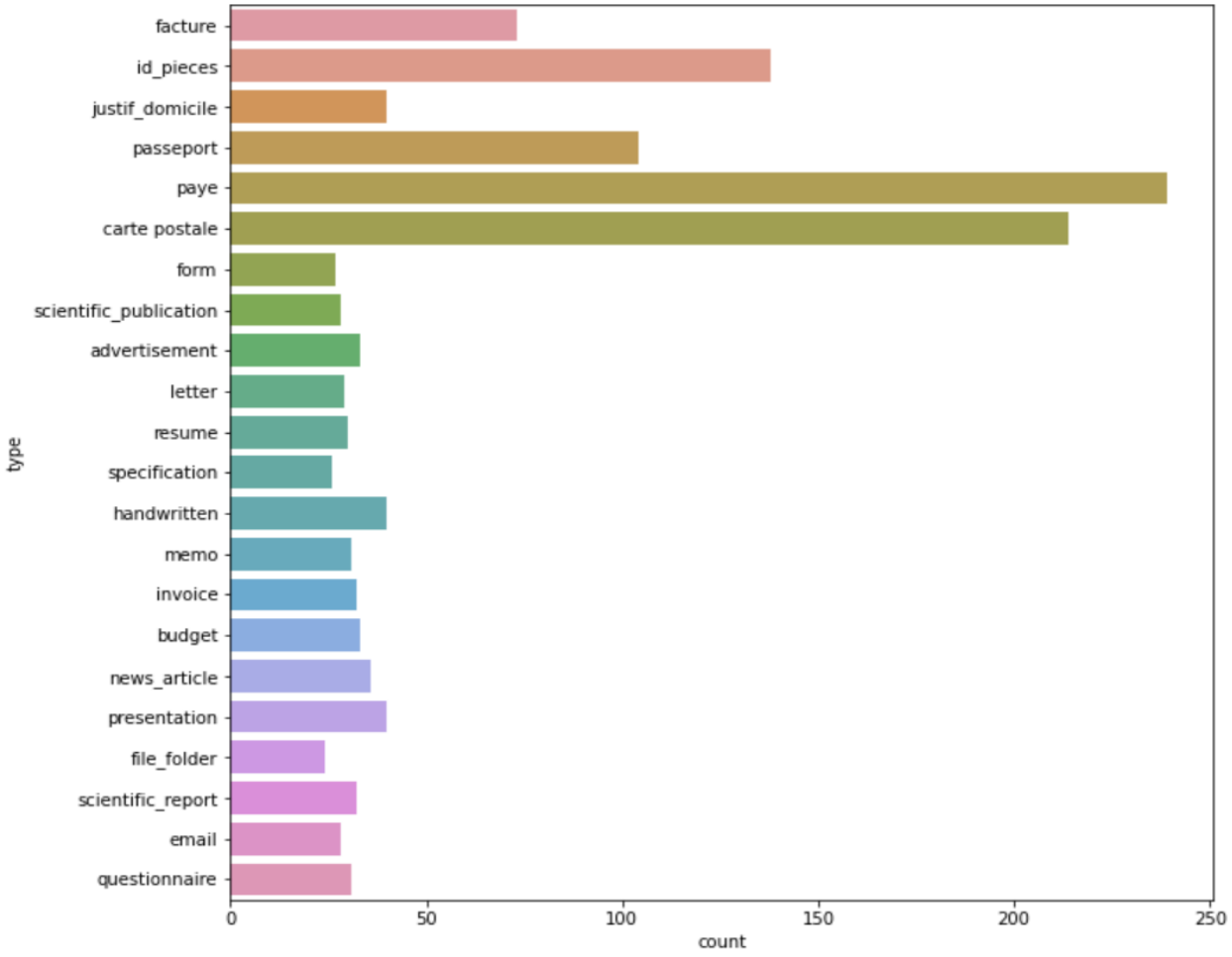
Enfin, nous avons cherché une piste réalisable pour extraire le texte des pièces d’identité.

Data

Cadre

Le jeu de données a été fourni par notre tuteur Thomas BOEHLER. Une partie des images provient d'anciens projets, une autre a été scrappée sur internet.

Le jeu de données comprend 1308 images labellisées selon 22 types. On en retrouve la répartition sur le graphique ci-dessous :



Les 5 principaux types de documents sont par ordre décroissant :

- ✓ Les bulletins de paie,
- ✓ Les cartes postales,
- ✓ Les carte d'identité,
- ✓ Les passeports,
- ✓ Les factures.

Ainsi, sachant que nous souhaitons mettre l'accent sur la classification des pièces d'identité, c'est un jeu de données intéressant car ces deux types de document sont bien représentés.

Projet

Classification du problème

Le problème de machine learning auquel nous sommes confrontés est un problème de classification par type de document (carte postale, passeport, …).

Pour comparer nos modèles, nous utiliserons plusieurs métriques :

- La précision des documents d'identité (pourcentage de pièces d'identité bien prédites par rapport à l'ensemble des pièces d'identité),
- Le rappel des documents d'identité (pourcentage de pièces d'identité bien prédites par rapport à l'ensemble des prédictions de pièces d'identité).
- Le F1-score permet de mesurer la précision et le rappel à la fois. Il s'agit de la moyenne harmonique de la précision et du rappel (recall en anglais).

Plan

Les deux approches de classification (« OCR + NLP » et « CV ») étant très différentes, nous les traiterons séparément. Nous suivrons alors le plan suivant :

I . EXPLORATION ET PRÉPARATION DES DONNÉES

a. Océrisation
b. Visualisation des données
c. Traduction
d. Regroupement des classes

II. APPRENTISSAGE SUPERVISÉ ET CLASSIFICATION

a. Gradient Boosting / Forêts aléatoires / Régression logistique / SVM
b. Multinomial Naive Bayes
c. Modèle de BERT

III. CLASSIFICATION PAR COMPUTER VISION

a. Exploration des données
b. Callbacks
c. Transfer Learning : choix du modèle le plus performant
d. Optimisation du ResNet50
e. Erreurs en image
f. Analyse / Conclusion

IV. EXTRACTION D'INFORMATION

V. DIFFICULTÉS RENCONTRÉES LORS DU PROJET

VI. BILAN & SUITE DU PROJET

ANNEXES

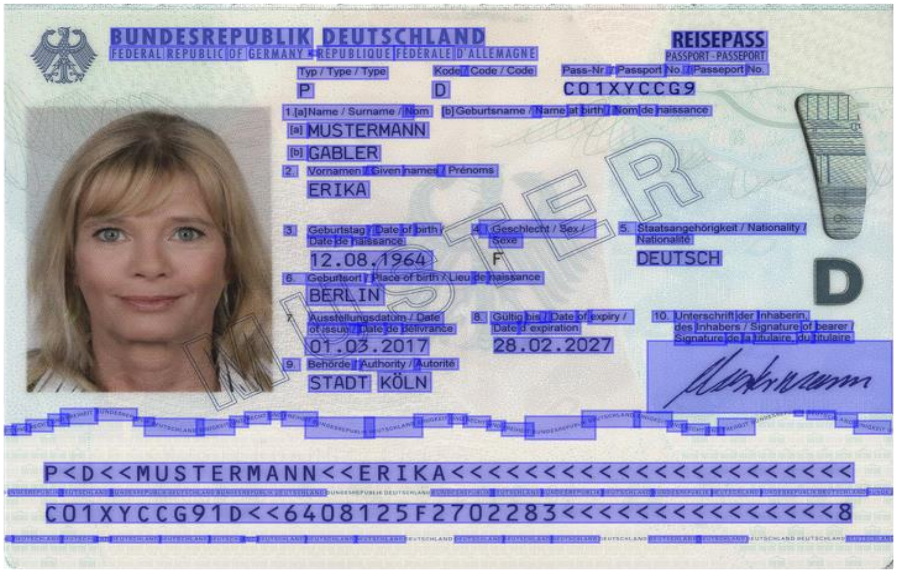
Bibliographie

Description des fichiers de code

I. EXPLORATION ET PRÉPARATION DES DONNÉES

a. Océrisation

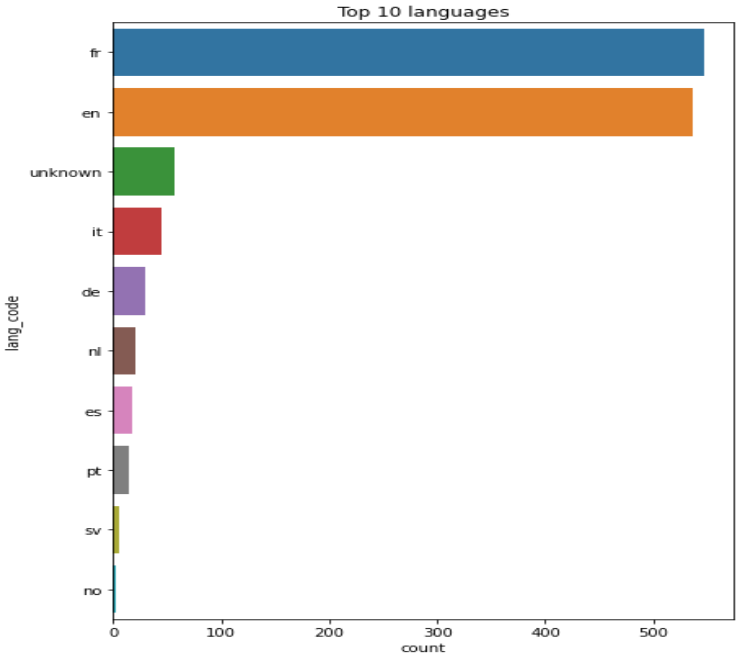
Pour l'océrisation (extraction du texte à partir de l'image) nous avons utilisé la bibliothèque doctr.

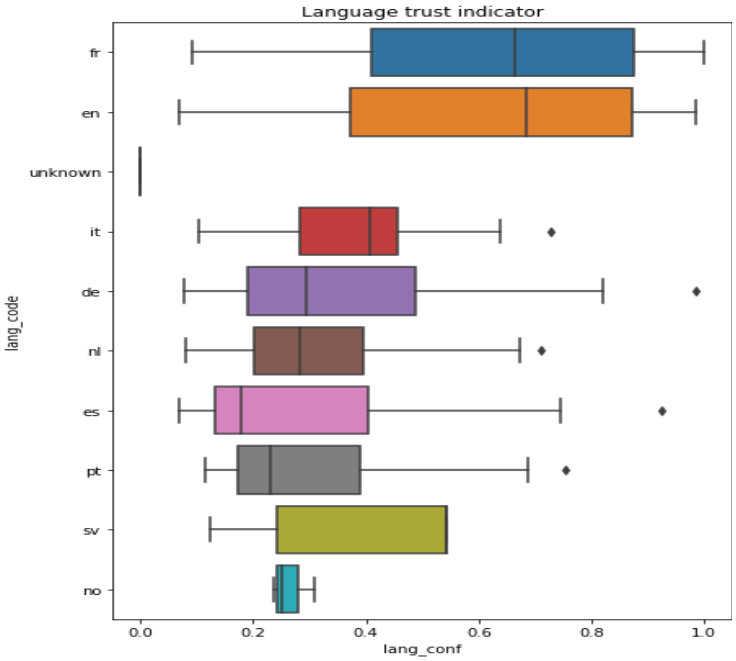


Une fois le texte extrait, nous avons réalisé une détection automatique de langue en utilisant la bibliothèque fastext. Une comparaison préalable a été faite pour comparer les deux bibliothèques de détection de langue : polyglot et fastext.

b. Visualisation des données

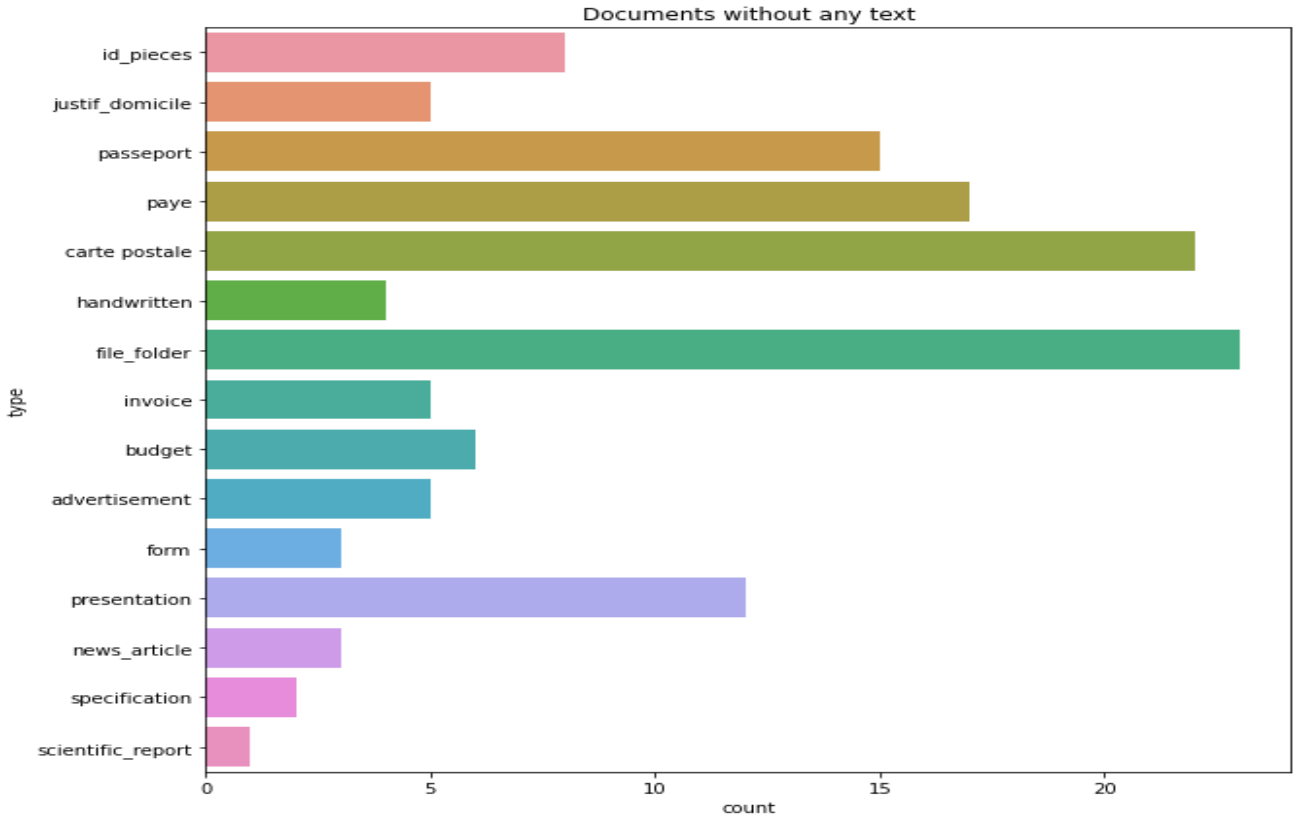
Parmi les langues les plus utilisées, nous retrouvons l'anglais et le français.



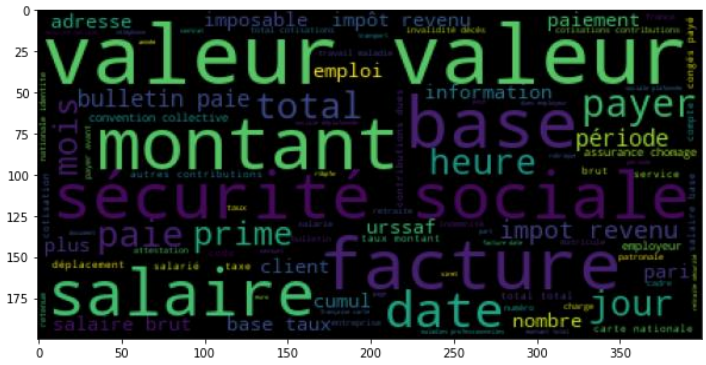


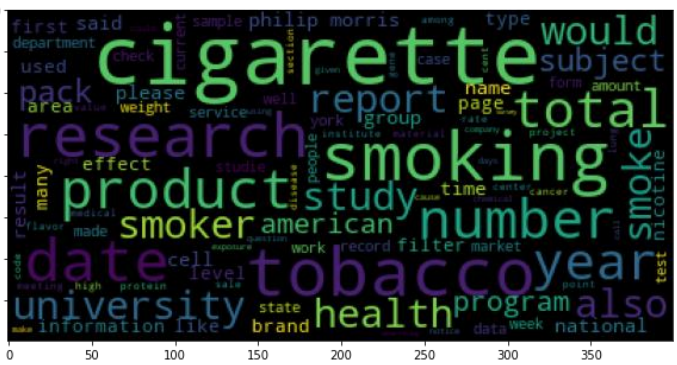
A ce stade, nous observons des documents sans texte. Deux possibilités :

- Pour certains, aucun texte n'est présent sur l'image,
- Pour d'autres, l'océrisation n'as pas réussi à extraire le texte.



Ci-dessous, les mots les plus utilisés pour les documents français et anglais :





Nous observons les points suivants :

- ✓ Les langues principales utilisées dans le document sont françaises et anglaises,
- ✓ Sur 153 images l'extraction de texte a échoué,
- ✓ Les mots les plus utilisées : valeurs (fr), cigarette(en).

c. Traduction

Pour être capable de traiter de manière uniforme l'ensemble des données nous n'allons nous concentrer que sur les documents français et anglais. Une traduction française vers anglaise a été effectuée en utilisant la bibliothèque *transformer* avec l'aide du modèle Helsinki-NLP/opus-mt-fr-en. Pour la traduction anglais francais on à utilisé la pipeline *translation_en_to_fr*

Exemple de traduction :

```
Source:  FACTURE Adresse CP Ville Téléphone
Translated: Invoice address Cp City Phone
```

Nous avons conservé le texte brut après la traduction et créé deux nouvelles colonnes dans le jeu de données afin de normaliser le texte traduit (suppression de stop words, ponctuations, tokenisation).

d. Regroupement des classes

Après traduction, en analysant les labels, nous nous sommes aperçus de certaines similitudes. Ainsi, nous avons regroupé certaines classes.

Ancienne classe	Nouvelle classe
advertisement	other_types
form	other_types
handwritten	other_types
letter	other_types
memo	other_types
presentation	other_types
news article	scientific_doc
scientific publication	scientific_doc
scientific report	scientific_doc

Par la suite pour l'approche NLP nous n'utilisons que les données en langue anglaise et française. Après le nettoyage notre dataset se réduit à 1177 lignes (1308 initialement).

La nouvelle structure de notre dataset est la suivante :

- **text_ocr** - le texte brut extrait par l'océrisation
- **text** - le texte normalisé (lowercase, regex tokenizer, delete stop words)
- **text_fr** - le texte traduit en français avec un minimum de normalisation (delete numbers, delete dates)
- **text_en** - le texte traduit en anglais avec un minimum de normalisation (delete numbers, delete dates)
- **text_fr_norm** - le texte normalisé en français (lowercase, regex tokenizer, delete stop words, delete punctuations)
- **text_en_norm** - le texte normalisé en anglais (lowercase, regex tokenizer, delete stop words, delete punctuations)
- **target** - target
- **target_min** - avec le regroupement des classes

II. APPRENTISSAGE SUPERVISÉ ET CLASSIFICATION

a. Gradient Boosting / Forêts aléatoires / Régression logistique / SVM

Présentation des modèles de classification utilisés:

Dans cette partie, nous allons aborder la classification en tant que problème d'apprentissage supervisé en utilisant les quatres algorithmes suivants:

- Gradient Boosting Classifier: il s'agit d'une technique d'entraînement séquentiel. Ce modèle optimise la fonction de perte en générant des apprenants de base de manière séquentielle, de sorte que l'apprenant de base actuel soit toujours plus efficace que le précédent. Cette méthode tente de générer des résultats précis au départ au lieu de corriger les erreurs tout au long du processus, comme AdaBoost. Pour cette raison, l'algorithme peut conduire à des résultats précis.

- Les forêts aléatoires: Les forêts aléatoires se composent d'un grand nombre d'arbres décisionnels individuels qui fonctionnent comme un ensemble. Chaque arbre individuel de la forêt aléatoire émet une prédiction de classe et la classe ayant reçu le plus de votes devient la prédiction de notre modèle.

- La Régression Logistique: La régression logistique est une méthode bien connue en statistiques qui sert à prédire la probabilité d'un résultat et qui est fréquemment utilisée pour les tâches de classification. L'algorithme prédit la probabilité d'occurrence d'un événement en ajustant les données à une fonction logistique.

- Les Machines à Vecteurs de Support: Le principe des SVM consiste à ramener un problème de classification à un hyperplan dans lequel les données sont séparées en plusieurs classes dont la frontière est la plus éloignée possible des points de données. Le concept de frontière implique que les données soient linéairement séparables. Pour y parvenir, les supports vector machines font appel des fonctions mathématiques permettant de projeter et séparer les données dans l'espace vectoriel, les "vecteurs de support" étant les données les plus proches de la frontière. C'est la frontière la plus éloignée de tous les points d'entraînement qui est optimale, et qui présente donc la meilleure capacité de généralisation.

- Résultats obtenus:

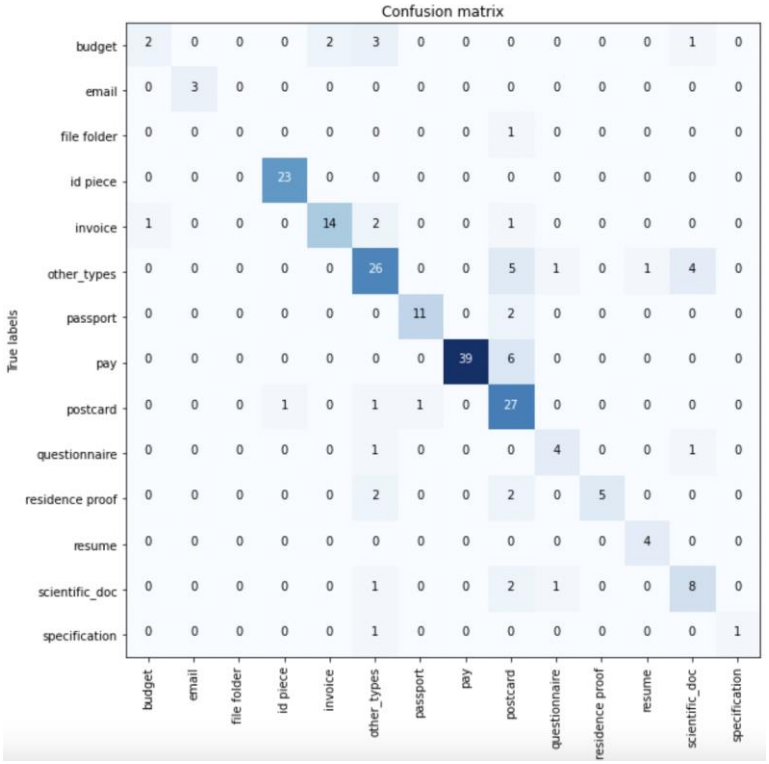
Pour chacun de ces modèles,comme certaines classes sont sous-représentées, nous avons rajouté une fonction Random Over Sampler pour générer de nouveaux échantillons aléatoires.
Cette technique permet de rééquilibrer la distribution des classes pour un ensemble de données déséquilibré.
Voici un tableau contenant les précisions globales obtenues pour les quatres modèles avant et après l'ajout de la fonction Random Over Sampler :

Accuracy	Gradient Boosting	Random Forest	Logistic Regression	SVM
Avant l'oversampling	0,73	**0,77**	0,73	0,76
Après l'oversampling	0,73	0,75	**0,79**	0,76

Nous notons que la meilleure performance est obtenue avec le modèle de régression logistique après le rééchantillonnage et nous permet d'obtenir un score de 79%.

D’après la matrice de confusion et le rapport de classification du modèle de Régression Logistique après rééchantillonnage (ci-dessous):

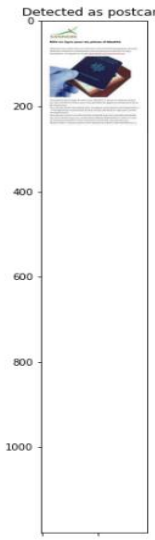
- Parmi les 24 cartes d’identité prédites, 23 sont effectivement des cartes d’identité, soit une précision de 96%.
- Toutes les cartes d’identité ont été prédites comme cartes d’identités , soit un rappel de 100% .
- Parmi les 12 prédictions de passeports, 11 sont effectivement des passeports, soit une précision de 92%.
- Parmi les 13 passeports, 11 ont été prédits correctement, soit un rappel de 85%. 2 passeports ont été prédit comme des cartes postales.
- Le modèle nous donne un F1-score de 98% pour les cartes d’identités et de 88% pour les passeports.

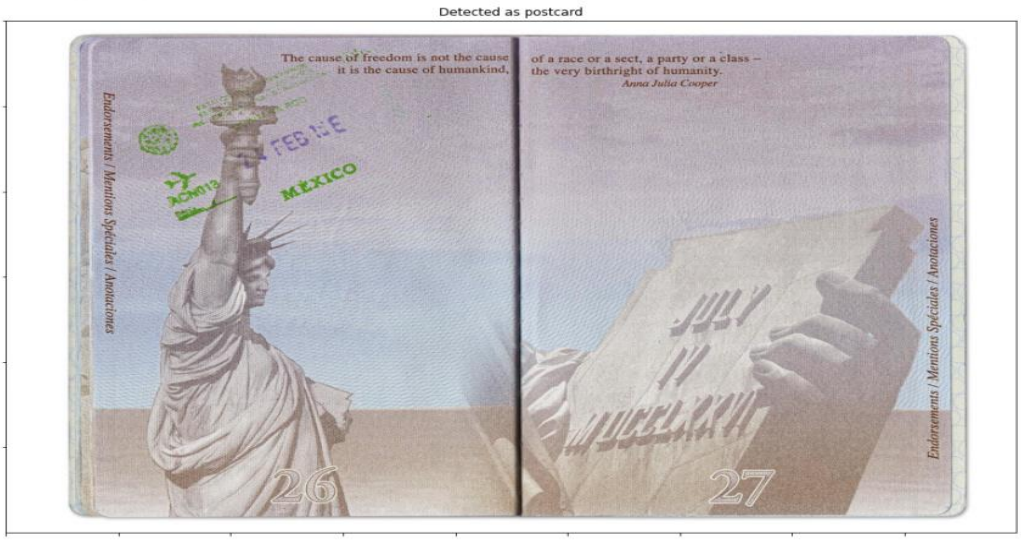


	precision	recall	f1-score	support
budget	0.67	0.25	0.36	8
email	1.00	1.00	1.00	3
file folder	0.00	0.00	0.00	1
id piece	0.96	1.00	0.98	23
invoice	0.88	0.78	0.82	18
other_types	0.70	0.70	0.70	37
passport	0.92	0.85	0.88	13
pay	1.00	0.87	0.93	45
postcard	0.59	0.90	0.71	30
questionnaire	0.67	0.67	0.67	6
residence proof	1.00	0.56	0.71	9
resume	0.80	1.00	0.89	4
scientific_doc	0.57	0.67	0.62	12
specification	1.00	0.50	0.67	2
accuracy			0.79	211
macro avg	0.77	0.70	0.71	211
weighted avg	0.81	0.79	0.79	211

- Erreur en image pour le modèle de Régression Logique après rééchantillonnage aléatoire:

Voici en image les 2 passeports prédits comme carte postales. En regardant les images,nous constatons qu'il est difficile d'identifier qu'il s'agit de passeport.



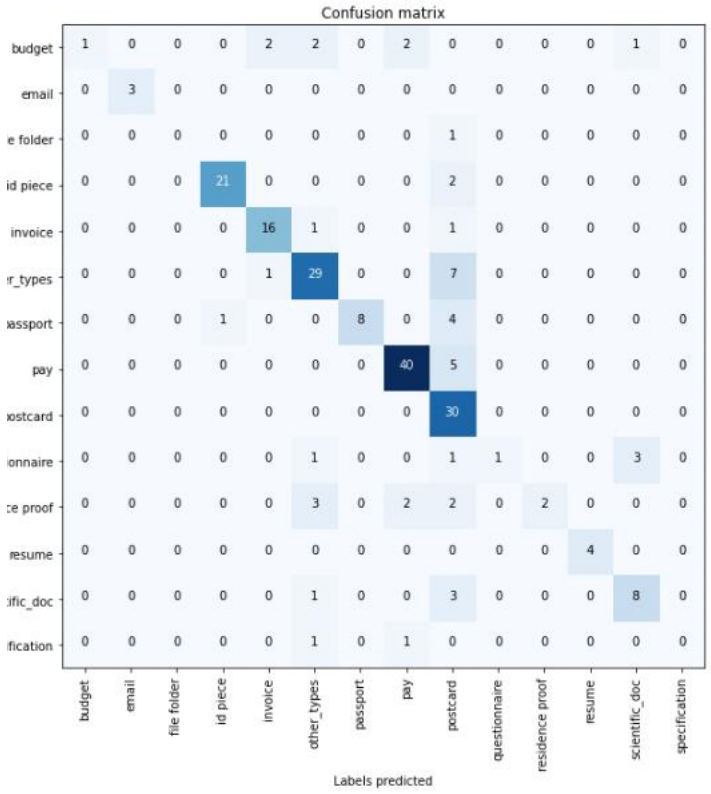


d

Le modèle Random Forest nous donne également un bon score de 77% sans réechantillonage aléatoire.

D'après la matrice de confusion et le rapport de classification du modèle Random Forest (voir les figures ci-dessous):

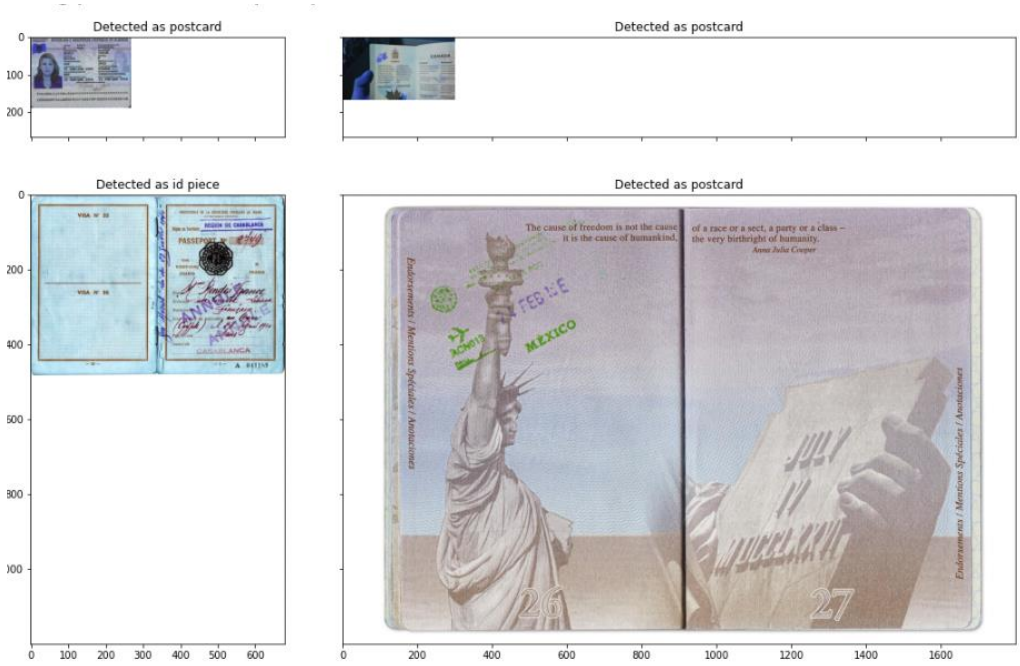
- Parmi les 22 cartes d'identité prédites, 21 cartes d'identité sont effectivement des cartes d'identité, soit une précision de 95%.
- Parmi les 23 cartes d'identité, 2 ont été prédites comme cartes postales, soit un rappel de 91%.
- Les 8 prédictions de passeport sont effectivement des passeports, soit une précision de 100%,
- Parmi les mauvaises prédictions de passeports, 1 a été prédit comme carte d'identité et 4 autres comme cartes postales.
- Le modèle nous donne un F1-score de 93% pour les cartes d'identité et de 76% pour les passeports.



	precision	recall	f1-score	support
budget	1.00	0.12	0.22	8
email	1.00	1.00	1.00	3
file folder	0.00	0.00	0.00	1
id piece	0.95	0.91	0.93	23
invoice	0.84	0.89	0.86	18
other_types	0.76	0.78	0.77	37
passport	1.00	0.62	0.76	13
pay	0.89	0.89	0.89	45
postcard	0.54	1.00	0.70	30
questionnaire	1.00	0.17	0.29	6
residence proof	1.00	0.22	0.36	9
resume	1.00	1.00	1.00	4
scientific_doc	0.67	0.67	0.67	12
specification	0.00	0.00	0.00	2
accuracy			0.77	211
macro avg	0.76	0.59	0.60	211
weighted avg	0.82	0.77	0.75	211

- Erreur en image du modèle Random Forest:

Parmi les mauvaises prédictions de passeports, un passeport a été prédit comme carte d'identité et 4 autres comme cartes postales. Voici les erreurs en images.



- Voting classifier

Nous avons ensuite utilisé un voting classifier avec un ‘hard’ voting.
Le voting classifier est un modèle qui s'entraîne sur un ensemble de modèles et prédit une sortie (classe) en fonction de la probabilité la plus élevée de la classe choisie comme sortie.
Il agrège simplement les résultats de chaque modèle transmis au Voting Classifier et prédit la classe de sortie en fonction de la plus grande majorité des votes.
Nous avons utilisé les quatres modèles de base vu précédemment. Voici les scores de ces modèles:

- Gradient Boosting Classifier: 0.61
- Random Forest Classifier: 0.71
- Logistic Regression: 0.75
- SVM: 0.73
- Voting Classifier: 0.75

D’après les scores obtenus, nous remarquons que le Voting Classifier nous donne une performance équivalente à celle de Regression Logistique.
Par conséquent, nous considérons qu’il n’est pas utile de garder le voting classifier.

En conclusion, parmi les quatres modèles précédents, il est préférable de choisir le modèle random forest car même si le modèle de régression logistique après rééchantillonage nous permet d’améliorer la performance, le rééchantillonnage aléatoire peut augmenter la probabilité d'un overfitting. En effet, le rééchantillonage permet de faire des copies exactes des exemples de la classe minoritaire.

b. Multinomial Naive Bayes

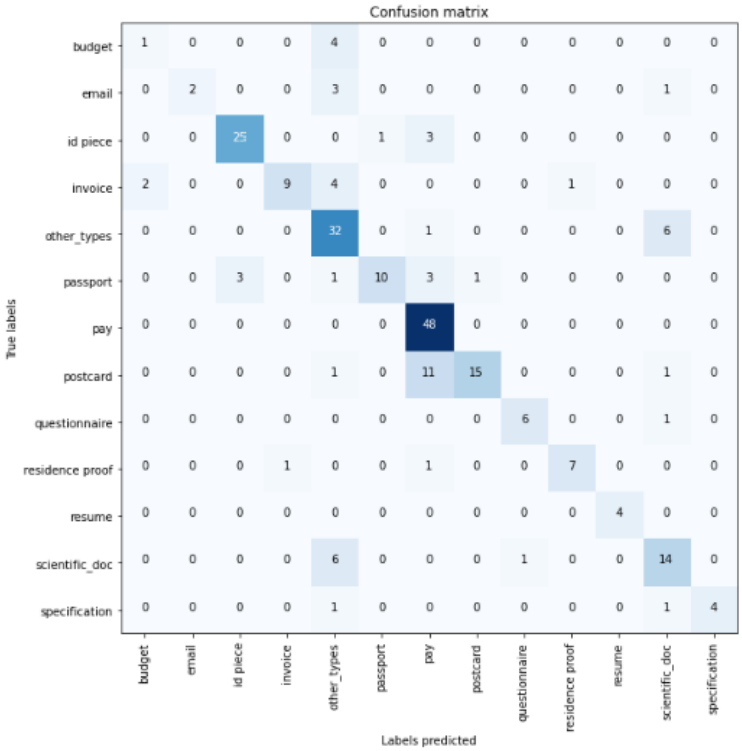
Naive Bayes Classifier est un algorithme populaire en Machine Learning. C’est un algorithme du Supervised Learning utilisé pour la classification. Il est particulièrement utile pour les problématiques de classification de texte.

On utilise l'approche Bag-of-words qui consiste à attribuer à chaque document un vecteur de longueur égale à la taille du vocabulaire du corpus de textes analysés. Pour choisir les hyperparamètres on fait appel à la fonction GridSearchCV qui utilise notamment la validation croisée.

On entraîne plusieurs modèles avec le texte français, anglais en ajoutant ou non le over sampling pour équilibrer le jeux de données (RandomOverSample) et en utilisant comme target le set réduit des classes (target_min) ou le l’ensemble total des classes. A la fin on compare la meilleure combinaison.

	index	precision	recall	f1-score	support	classifier	category
265	accuracy	0.677725	0.677725	0.677725	0.677725	MNB	classic target english
289	accuracy	0.654028	0.654028	0.654028	0.654028	MNB	classic target english ros
312	accuracy	0.655660	0.655660	0.655660	0.655660	MNB	classic target french
336	accuracy	0.660377	0.660377	0.660377	0.660377	MNB	classic target french ros
359	accuracy	0.716102	0.716102	0.716102	0.716102	MNB	classic target
383	accuracy	0.673729	0.673729	0.673729	0.673729	MNB	classic target ros
400	accuracy	0.739336	0.739336	0.739336	0.739336	MNB	classic target_min english
417	accuracy	0.720379	0.720379	0.720379	0.720379	MNB	classic target_min english ros
433	accuracy	0.688679	0.688679	0.688679	0.688679	MNB	classic target_min french
449	accuracy	0.669811	0.669811	0.669811	0.669811	MNB	classic target_min french ros
465	accuracy	0.750000	0.750000	0.750000	0.750000	MNB	classic target_min
482	accuracy	0.694915	0.694915	0.694915	0.694915	MNB	classic target_min ros

Nous obtenons le meilleur score avec le set réduit des classes sur le texte traduit en anglais (13).



	precision	recall	f1-score	support
budget	0.33	0.20	0.25	5
email	1.00	0.33	0.50	6
id piece	0.89	0.86	0.88	29
invoice	0.90	0.56	0.69	16
other_types	0.62	0.82	0.70	39
passport	0.91	0.56	0.69	18
pay	0.72	1.00	0.83	48
postcard	0.94	0.54	0.68	28
questionnaire	0.86	0.86	0.86	7
residence proof	0.88	0.78	0.82	9
resume	1.00	1.00	1.00	4
scientific_doc	0.58	0.67	0.62	21
specification	1.00	0.67	0.80	6
accuracy			0.75	236
macro avg	0.82	0.68	0.72	236
weighted avg	0.78	0.75	0.74	236

On remarque une bonne précision sur les passeport (0.91) et pièce d'identité (0.89) avec un recall score de 0.56 sur passeport et 0.86 sur les pièces d'identités.

c. Bert (Bidirectional Encoder Representations from Transformers)

BERT fait appel à Transformer, un mécanisme d'attention qui apprend les relations contextuelles entre les mots (ou les sous-mots) d'un texte. Dans sa forme classique, Transformer comprend deux mécanismes distincts : un encodeur qui lit le texte en entrée et un décodeur qui produit une prédiction pour la tâche. Puisque l'objectif de BERT est de générer un modèle de langage, seul le mécanisme d'encodage est nécessaire.

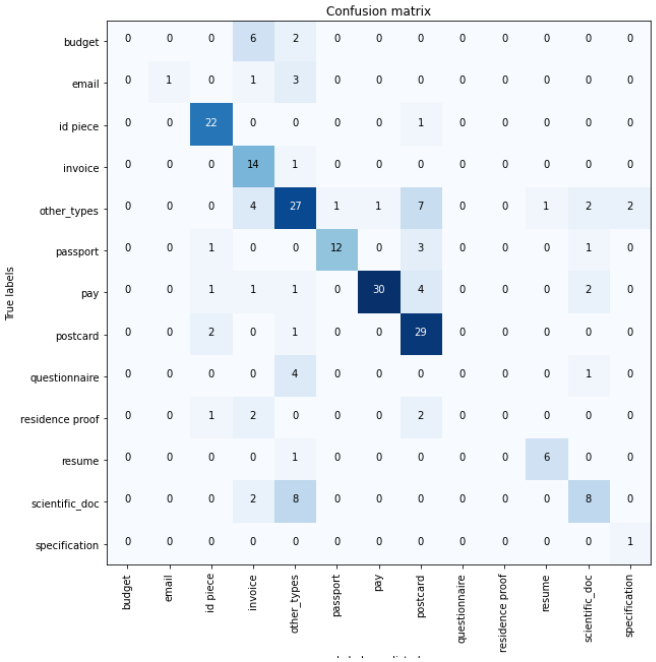
Nous utiliserons le BertTokenizerFast pré-entraîné. Dans cette étude, nous ne tiendrons pas compte des caractères minuscules et majuscules présents. Pour cette raison, nous convertirons toutes les séquences en caractères minuscules

Maintenant que notre jeu de données est fin prêt, nous allons choisir le modèle "**bert-base-uncased**" disponible depuis la bibliothèque *transformer* pour commencer la modélisation.

Pour cette étape, nous utiliserons le modèle BERT. Ce modèle est conçu pour être pré-entraîné sur des représentations bi-directionelles à partir d'un texte non labellisé en conditionnant sur le contexte de droite et de gauche (bi-directionnel) sur chaque couche du modèle.

L'avantage principal de ce modèle est qu'il peut tout simplement être ajusté (fine-tuning) en ajoutant une simple couche de sortie dépendamment de la tâche désirée sans pour autant modifier la structure principale du modèle.

En regardant les résultats :



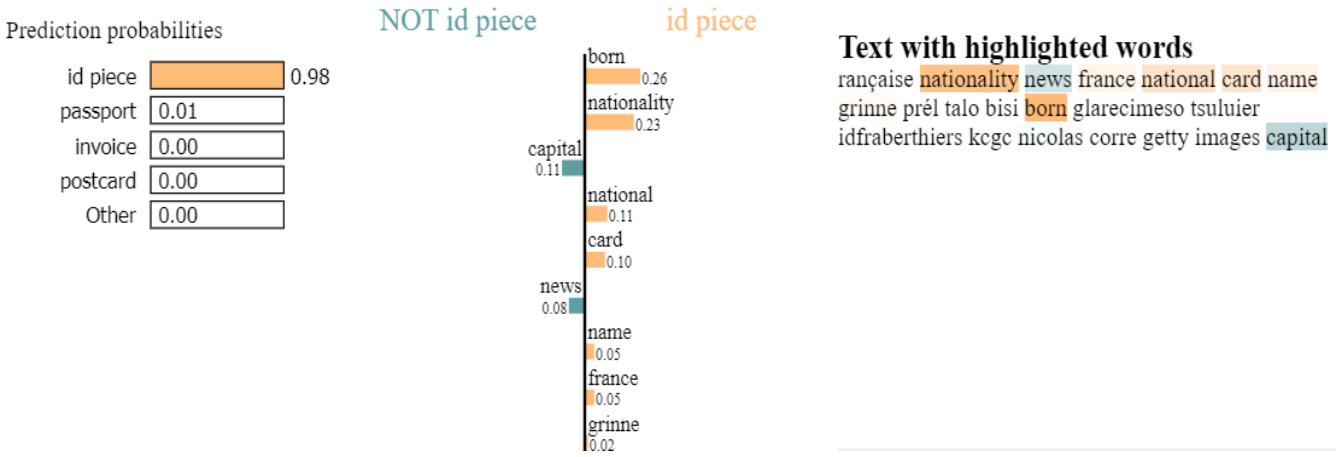
	precision	recall	f1-score	support
budget	0.00	0.00	0.00	8
email	1.00	0.20	0.33	5
id piece	0.81	0.96	0.88	23
invoice	0.47	0.93	0.62	15
other_types	0.56	0.60	0.58	45
passport	0.92	0.71	0.80	17
pay	0.97	0.77	0.86	39
postcard	0.63	0.91	0.74	32
questionnaire	0.00	0.00	0.00	5
residence proof	0.00	0.00	0.00	5
resume	0.86	0.86	0.86	7
scientific_doc	0.57	0.44	0.50	18
specification	0.33	1.00	0.50	1
accuracy			0.68	220
macro avg	0.55	0.57	0.51	220
weighted avg	0.66	0.68	0.65	220

On remarque une très bonne précision sur les passeports (0.92) et pièces d'identité (0.81) avec un recall score de 0.71 sur passeport et 0.96 sur les pièces d'identités.

Le modèle se trompe souvent avec la détection des passeports comme carte postale. On observe aussi une erreur dans le label une pièce d'identité labellisé “passport”.

d. Interprétabilité des résultats

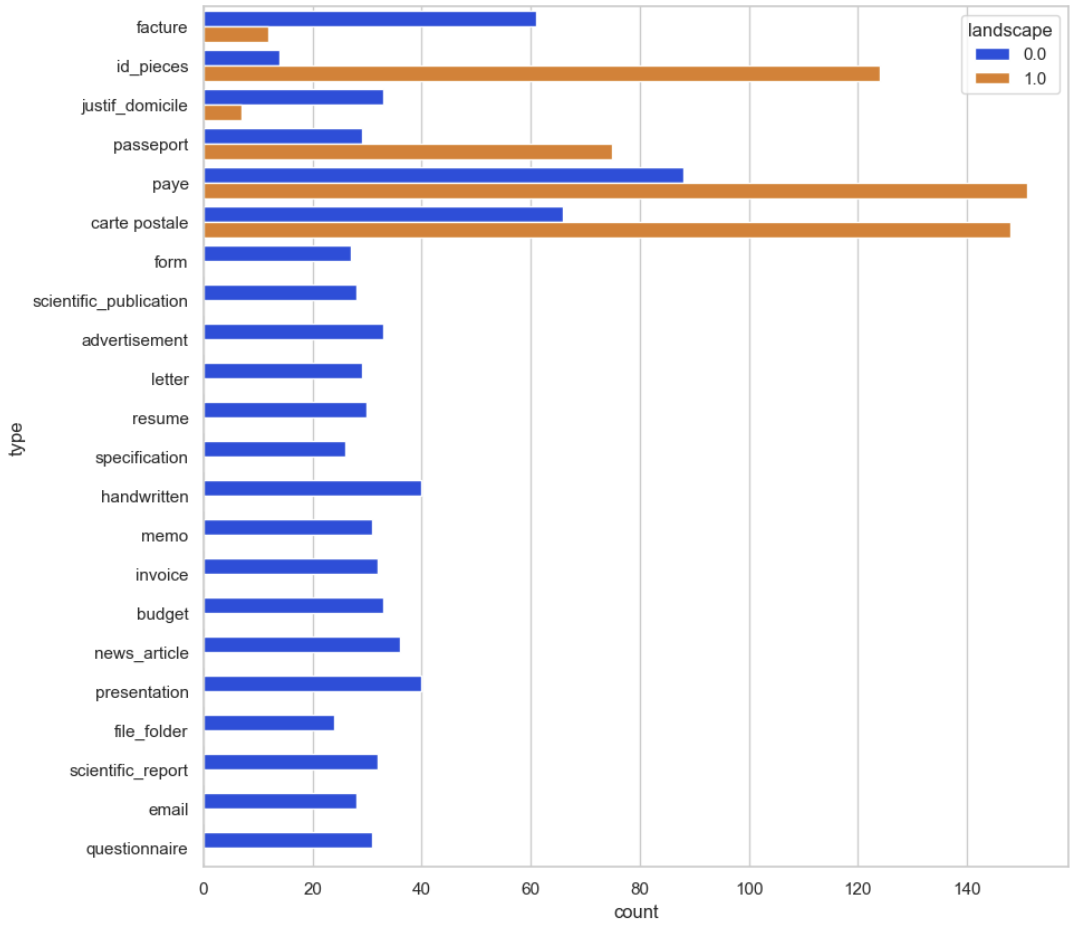
On utilise LIME (en anglais, Local Interpretable Model-agnostic Explanations) est un modèle local qui cherche à expliquer la prédiction d'un individu par analyse de son voisinage.



Après l'analyse des quelques exemples de notre dataset le modèle se base sur les bons mots pour prendre la décision.

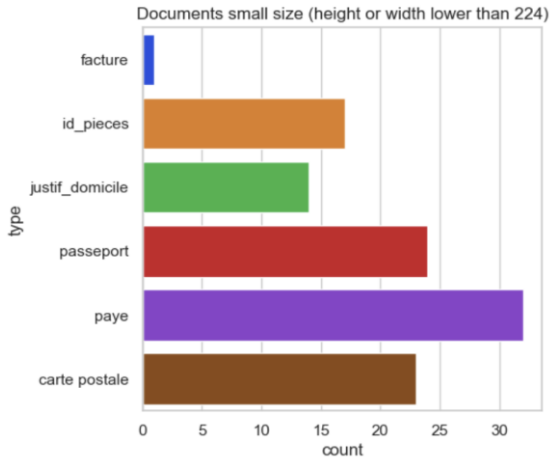
III. CLASSIFICATION PAR COMPUTER VISION

a. Exploration des données



Nous observons les points suivants :

- ✓ Toutes les images sont en couleur (3 canaux).
- ✓ L'orientation des documents n'est pas toujours la même (voir graphique ci-dessus).
- ✓ 7% des images ont une hauteur inférieure à 224 et 2.6% une largeur inférieure à 224.
- ✓ Parmi ces documents de petite taille, nous retrouvons : les bulletins de paie, les cartes postales, les cartes d'identité et les passeports (voir graphique ci-dessous).



Nous devrons prendre en compte ces observations en Computer Vision dans les modèles que nous allons tester. En particulier, une résolution supérieure à (224, 224) n'apporte sans doute pas d'améliorations aux résultats de classification.

b. Callbacks (ou fonctions de rappel)

Deux callbacks ont été retenus pour les algorithmes :

- Réduction automatique du taux d'apprentissage.
- Arrêt automatique de l'apprentissage.

Le premier callback fonctionne ainsi. Si la fonction de perte du jeu de validation ne diminue pas après un certain nombre d'époques alors le taux d'apprentissage est divisé par 10. Cela permet à l'algorithme de converger plus efficacement vers les poids optimaux.

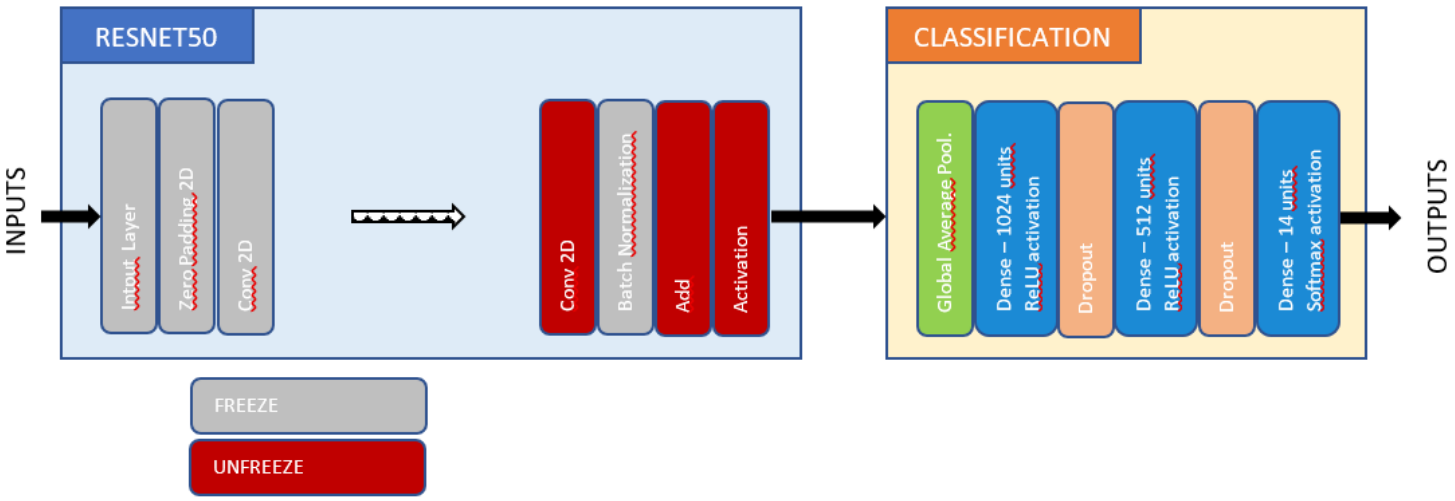
Le second callback stoppe l'apprentissage si la fonction de perte du jeu de validation ne diminue pas après un certain nombre d'époques.

c. Transfer Learning : choix du modèle le plus performant

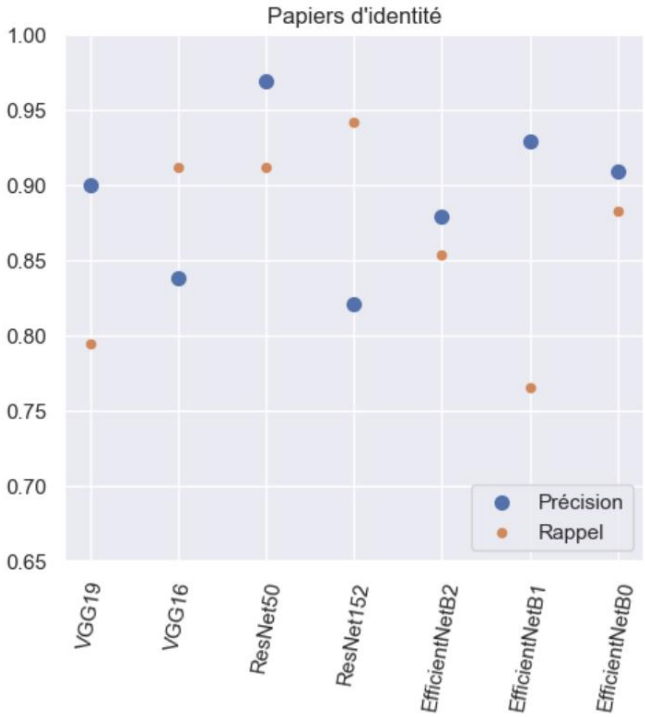
Le principe du « Transfer Learning » consiste à reprendre des algorithmes pré-entraînés sur de nombreuses images en en conservant les poids. Nous complétons ensuite le modèle par quelques couches afin d'obtenir la sortie voulue, à savoir les types des documents dans notre cas.

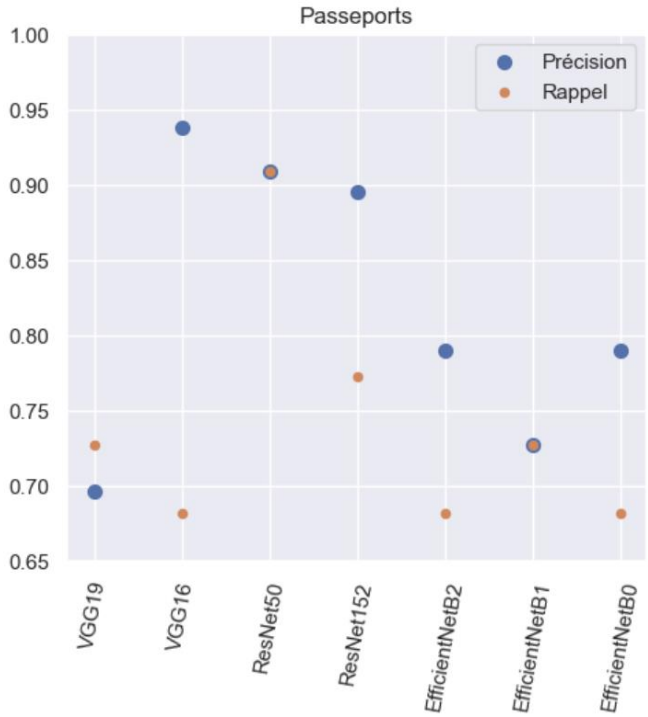
Pour l'exécution des algorithmes, les deux callbacks précédemment définis sont utilisés ainsi qu'une partie « Fine Tuning ». Comme son nom l'indique, cette dernière partie permet d'affiner le résultat. Le fonctionnement est donc le suivant :

1) Entraînement de l'algorithme avec les poids gelés du modèle pré-entraîné.
2) Récupération du taux d'apprentissage du premier algorithme en fin d'apprentissage.
3) Dégel des dernières couches du modèle pré-entraîné.
4) Nouvel entraînement de l'algorithme.

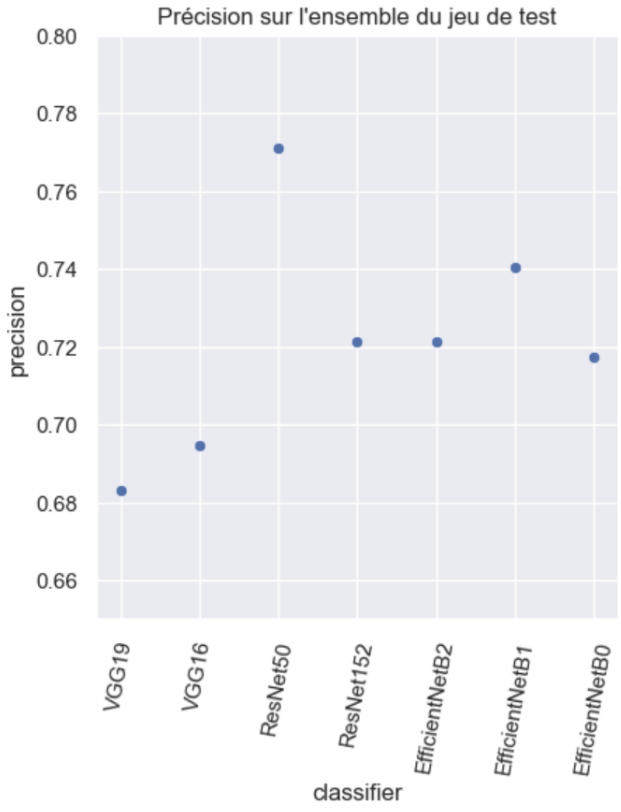


Après entraînement des modèles, nous obtenons les précisions et rappels ci-dessous pour les pièces d'identité :





On observe que l'algorithme ResNet50 nous donne de meilleures performances en termes de précision et de rappel concernant les documents d'identité.



Si nous observons la précision globale sur l'ensemble du jeu de test, la tendance observée précédemment se confirme.

Le transfer learning en utilisant l'algorithme ResNet50 offre de meilleures performances (environ 5% de mieux que les algorithmes de la famille EfficientNet.

Nous choisirons donc par la suite cet algorithme afin d'affiner nos recherches de prédiction.

d. Optimisation du ResNet50

Afin d'optimiser les performances de l'algorithme sélectionné, nous avons réalisé deux étapes :

- ✓ Première étape : Augmenter les données d'apprentissage.
- ✓ Deuxième étape : Corriger le labeling des documents d'identité.

Première étape :

Afin d'augmenter les données d'apprentissage, nous avons utilisé pour moitié les données de notre jeu d'entraînement et pour moitié les mêmes données de notre jeu d'entraînement mais cette fois-ci augmentées.

Pour l'augmentation de données, nous avons choisi la méthode *ImageDataGenerator* de tensorflow avec les paramètres ci-dessous :

- Rotation des images de 90 degrés,
- Largeur de l'image pouvant variée de 5%,
- Hauteur de l'image pouvant variée de 5%,
- Image renversée horizontalement,
- Zoom de 5%.

Ces images supplémentaires permettent de rendre l'algorithme plus robuste et de prendre en compte le fait que l'utilisateur final ne fournira pas forcément un document bien proportionné et dans le bon sens.

Deuxième étape :

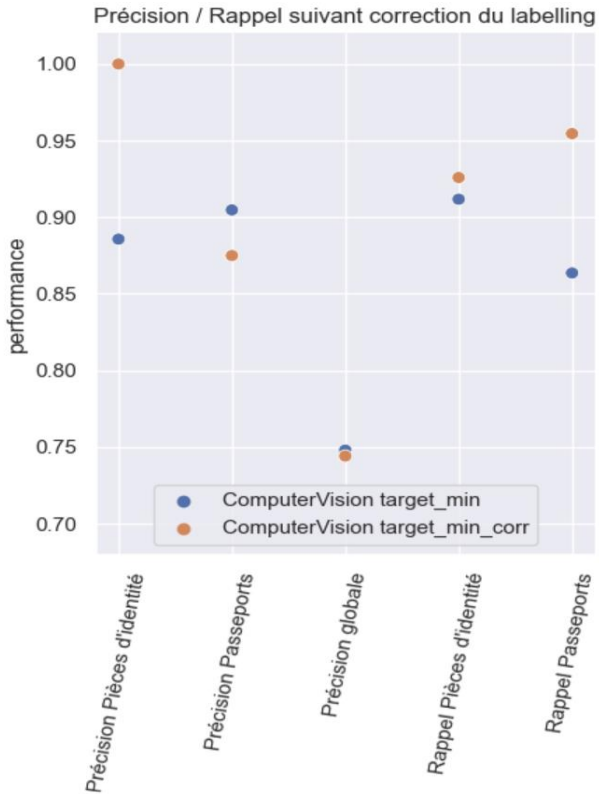
En explorant les documents d'identité, nous nous sommes aperçus d'erreurs de labeling :

- Deux documents sont de type "Pièces d'identité" au lieu de "Passeports",
- Deux documents sont de type "Passeports" au lieu de "Pièce d'identité",
- Parmi les "Passeports", un document est de type inconnu,
- Parmi les "Pièces d'identité", nous retrouvons deux permis de conduire,
- Parmi les "Pièces d'identité", nous retrouvons dix-huit documents avec sur l'image un passeport et une carte d'identité.

Afin que le dataset soit plus exploitable, nous avons conservé les deux catégories "Passeports" et "Pièces d'identité" en conservant :

- Dans la catégorie "Passeports" uniquement des passeports,
- Dans la catégorie "Pièces d'identité" uniquement des cartes d'identité.

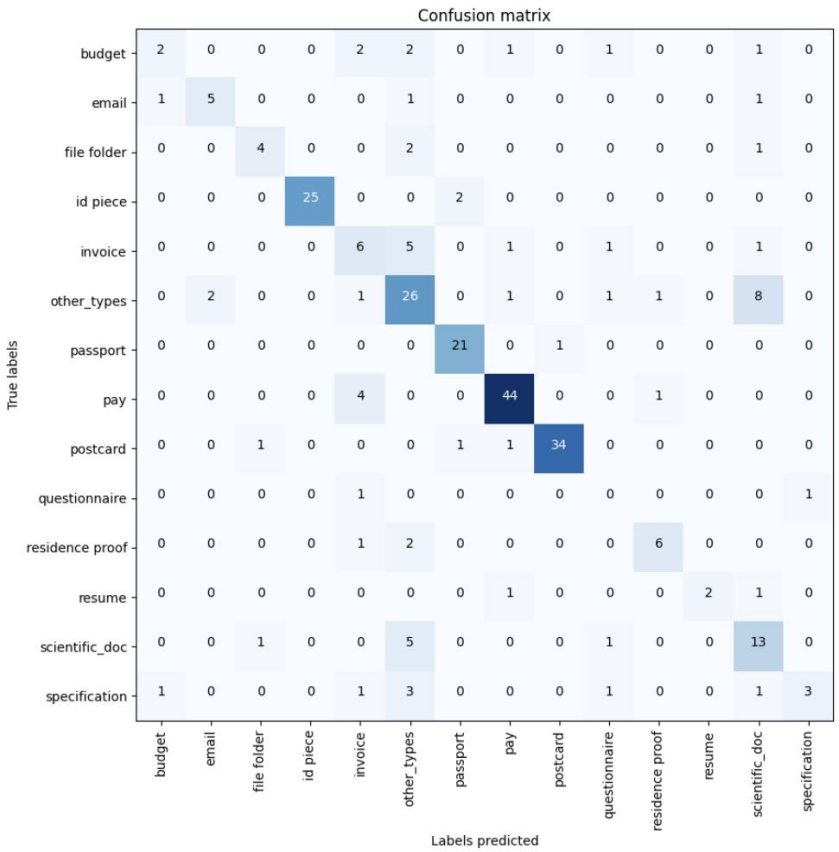
Nous pouvons constater à travers le graphique ci-dessous une nette amélioration des précisions / rappels pour les documents d'identité (environ 4.6% du f1-score en moyenne).



Etude réalisée avec l'augmentation de données :

- En bleu, sans correction de label,
- En orange, avec correction de label.

En prenant en compte la correction des labels, nous obtenons la matrice de confusion et le rapport de classification ci-dessous :



	precision	recall	f1-score	support
budget	0.33	0.11	0.17	9
email	0.83	0.62	0.71	8
file folder	0.83	0.71	0.77	7
id piece	1.00	0.93	0.96	27
invoice	0.37	0.50	0.42	14
other_types	0.60	0.68	0.64	40
passport	0.88	0.95	0.91	22
pay	0.91	0.84	0.87	49
postcard	0.95	0.95	0.95	37
questionnaire	0.00	0.00	0.00	2
residence proof	0.86	0.67	0.75	9
resume	0.67	0.50	0.57	4
scientific_doc	0.50	0.65	0.57	20
specification	0.57	0.40	0.47	10
accuracy			0.74	258

La matrice de confusion nous permet de mieux comprendre les erreurs de prédictions.

Ainsi, nous pouvons constater que :

- Toutes les cartes d'identité prédites sont effectivement des cartes d'identité, soit une précision de 100%,
- Parmi les 27 cartes d'identité, 2 ont été prédites en tant que passeport, soit un rappel de 93% (25 / 27).
- Parmi les 24 prédictions de passeport 21 sont effectivement des passeports, soit une précision de 88%,
- Parmi les 22 passeports, 21 ont été prédits correctement, soit un rappel de 95%.

e. Erreurs en image

Les 4 erreurs de prédiction des pièces d'identité et des passeports sont les suivantes :

➔ 2 cartes d'identités détectées en tant que passeport

➔ 1 passeport détecté en tant que carte postale

➔ 1 carte postale détectée en tant que passeport

f. Analyse / Conclusion

Seulement 4 images ont été mal détectées parmi 27 cartes d'identité et 22 passeports. Compte-tenu de la qualité du jeu de données, cela reste une bonne performance.

Parmi les erreurs de détection, l'algorithme a du mal à reconnaître les cartes d'identité étrangères. Ceci est fort compréhensible car elles sont peu représentées dans le jeu de données.

IV. EXTRACTION D'INFORMATION

Pour l'extraction d'information à partir des pièces d'identités on utilise le texte extrait via le processus d'océrisation sur lequel on applique des expressions régulières pour localiser les zones de textes qui nous intéressent. En fonction de type de document on applique différentes expressions régulières

Les information extraites dans cet exemple

ID NUMBER: ['**D2H6862M2**']
LAST NAME: [**'MARTIN'**]
FIRST NAME: ['**Maelis-Gaelle, Marie** ']

V. DIFFICULTÉS RENCONTRÉES LORS DU PROJET

Preprocessing

- Données multilingue qui a nécessité la mise en place d'un système de traduction.
- Nous avons rencontré certaines difficultés sur des textes longs. Afin de finaliser le processus une traduction ligne par ligne à été mise en place pour les textes longs.
- Temps d'exécution d'océrisation assez long.

Computer Vision :

- Temps d'exécution des algorithmes de Deep Learning assez long sur CPU.
  - ➢ Configuration de l'ordinateur tel que la GPU de la carte NVIDIA soit utilisée lors des entraînements.
- Dégradation des performances lors de la phase de tuning.
  - ➢ Reprise du dernier "learning rate" en fin de la première phase d'apprentissage.
  - ➢ Lors du "dégel" des poids des dernières couches des algorithmes ResNet et EfficientNet, il est important de ne pas toucher aux poids des couches "Batch Normalization".
- Difficultés de réglage des Call Back.
  - ➢ Nombreux essais pour trouver les paramètres optimaux.

VI. BILAN & SUITE DU PROJET

A travers le projet, nous avons étudié de nombreux algorithmes (en traitement du langage et en computer vision) pour parvenir à répondre à notre problématique, à savoir classifier au mieux les pièces d'identité (cartes d'identité et passeports).

Compte tenu de notre jeu de données, l'algorithme ResNet50 semble le mieux répondre à cette problématique avec des précisions et rappels de l'ordre de 93%.

Cependant, il faut noter que le jeu de données utilisé pour le projet n'était pas de qualité suffisante pour apporter une réponse complète au contexte énoncé en début de rapport.
En effet, pour une étude plus pertinente, il faudrait un jeu de données avec:

- Plus de documents pour entraîner les modèles,
- Une plus grande diversité dans les pièces d'identité,
- Un scan de la page avec nom, prénom, photo pour les pièces d'identité,
- Etc…

Un plus grand nombre de documents et des documents plus diversifiés permettrait de gagner en robustesse. Un meilleur cadrage des pièces d'identité avec le scan de la "bonne" page permettrait une meilleure océrisation et ainsi améliorerait considérablement les performances des algorithmes en traitement du langage. Des algorithmes tels que Logistic Regression ou BERT deviendraient alors de bons candidats pour répondre à la problématique.

Ainsi, l'étape suivante serait de reprendre les algorithmes détaillés dans notre rapport dans un contexte entreprise avec des jeux de données de meilleure qualité.

ANNEXES

Bibliographie

NLP

- Hugging Face - https://huggingface.co/
- Sklearn - https://scikit-learn.org/stable/
- Datascientest - https://datascientest.com/
- Doctr - https://mindee.com/product/doctr

Computer Vision :

- Recherche des résolutions d'entraînement sur le site de keras pour les divers algorithmes de Transfer Learning : https://keras.io/api/applications/resnet/
- Résolution de bug sur le site https://stackoverflow.com/ pour la mise en place de la méthode ImageDataGenerator (utilisée pour l'augmentation de données).

Description des fichiers de code

L'ensemble des fichiers décrits ici se trouve sur le repository git https://github.com/DataScientest-Studio/py_docr dans le répertoire notebooks.

FICHIER	DESCRIPTION	Entrées / Sorties
01 - Database exploration.ipynb	Exploration du jeu de données: ● Nettoyage du jeu de données ● Ajout de features (height, …) ● Analyses	Entrée: *other_data.csv* Sortie: *data_with_meta.csv*
01.a - Data exploration of ID labels.ipynb	Exploration des pièces d'identité: ● Vérification des labels ● Séparation des pièces d'identité en 2 catégories (carte d'identité et passeport)	Entrée: ● *data_with_meta.csv* ● *data.csv* Sortie: ● *data_with_meta_corr.csv* ● *data_corr.csv*
02 - Image conversion.ipynb	Extraction de textes par OCR	Entrée: *data_with_meta.csv* Sortie: *data_with_text.csv*
03 - Language distribution.ipynb	Détection de la langue et ajout d'une nouvelle feature	Entrée: *data_with_text.csv* Sortie: data_with_text_langue.csv
04 - Text processing.ipynb	Preprocessing du texte océrisé: ● Suppression des stopwords ● Analyse des mots trouvés (wordcloud, …)	Entrée: data_with_text_langue.csv Sortie: *data_with_text_final.csv*
04.a - Analysis by group of words.ipynb	Analyse par groupe de mots	
04.b - Text translation.ipynb	Traduction des textes	Entrée: *data_with_text_final.csv* Sortie: *data_with_text_translated.csv*
04.d - Hugging_face.ipynb	Test hugging face	Entrée: *new_data.csv*

		Sortie: *df_fr_new.csv*
04.e - lemmatisation-traduction-francais.ipynb	Test lemmatisation	Entrée: *df_fr_new.csv* Sortie: *df_en_to_fr.csv* Entrée: *df_en_to_fr.csv* Sortie: *df_en_to_fr_lemma.csv*
04.f - lemmatisation and classification.ipynb	Test lemmatisation	Entrée: *df_en_to_fr_lemma.csv*
04.z - Final preprocessing data.ipynb	Regroupement en 14 classes (target_min)	Entrée: *data_with_text_translated.csv* Sortie: *data.csv*
05.a - Gradient boosting.ipynb	Prédictions avec l'algorithme Gradient Boosting (target_min - 14 classes) + Test après rééchantillonnage	Entrée: *data.csv*
05.b - Random Forest.ipynb	Prédictions avec l'algorithme des Forêts Aléatoires (target_min - 14 classes) + Test après rééchantillonnage	Entrée: *data.csv*
05.c - Logistic Regression.ipynb	Prédictions avec l'algorithme de Régression Logistique (target_min - 14 classes) + Test après rééchantillonnage	Entrée: *data.csv*
05.d- SVM.ipynb	Prédictions avec l'algorithme SVM (target_min - 14 classes) + Test après rééchantillonnage	Entrée: *data.csv*
05.e - Random over sampler by lang.ipynb	Prédictions avec l'algorithme Gradient Boosting après rééchantillonnage des données (22 classes) Analyse des prédictions par langue	Entrée: *data_with_text_final.csv*
05.f - VotingClassifier.ipynb	Prédictions avec l'algorithme VotingClassifier	Entrée: *data.csv*
05.y - MNB -MultinomialNB.ipynb	Prédictions avec l'algorithme MNB	Entrée: *data.csv*
05.z - SVC - Support Vector Machines.ipynb	Test avec l'algorithme SVM	Entrée: *data.csv*
06 - Word Embedding.ipynb	Prédictions avec un réseaux de neurone Embedding (target_min - 14 classes)	Entrée: *data.csv*
06.a - word2vec.ipynb	Vectorisation avec l'algorithme word2vec Prédictions avec un réseaux de neurone Dense (target_min - 14 classes)	Entrée: *data.csv*
06.b - doc2vec.ipynb	Vectorisation avec l'algorithme doc2vec Prédictions avec un réseaux de neurone Dense (target_min - 14 classes)	Entrée: *data_with_text.csv*
07 - BERT modelisation.ipynb	Prédictions avec modèle BERT sur l'ensemble des classes	Entrée: *data.csv*
07.a - BERT modelisation target min .ipynb	Prédictions avec modèle BERT sur le set restreint des classes	Entrée: *data.csv*
08 - Computer Vision - LENET.ipynb	Entrainement/Prédiction: • Modèle Lenet • Call Back • target_min - 14 classes	Entrée: *data.csv*
08.a - Computer Vision - LENET_ImageDataGenerator.ipynb	Entrainement/Prédiction: • Modèle Lenet • Augmented Data avec ImageDataGenerator • 4 classes	Entrée: *new_data.csv*

08.b - Computer Vision - LENET_albumentations.ipynb	Entrainement/Prédiction: ● Modèle Lenet ● Augmented Data avec albumentations ● 4 classes	Entrée: *new_data.csv*
09 - Computer Vision - VGG16.ipynb	Entrainement/Prédiction: ● Modèle VGG16 ● Call Back + Fine Tuning ● target_min - 14 classes	Entrée: *data.csv*
09.a - Computer Vision - VGG19.ipynb	Entrainement/Prédiction: ● Modèle VGG19 ● Call Back + Fine Tuning ● target_min - 14 classes	Entrée: *data.csv*
09.b - Computer Vision - EfficientNetB0.ipynb	Entrainement/Prédiction: ● Modèle EfficientNetB0 ● Call Back + Fine Tuning ● target_min - 14 classes	Entrée: *data.csv*
09.c - Computer Vision - EfficientNetB1.ipynb	Entrainement/Prédiction: ● Modèle EfficientNetB1 ● Call Back + Fine Tuning ● target_min - 14 classes	Entrée: *data.csv*
09.d - Computer Vision - EfficientNetB2.ipynb	Entrainement/Prédiction: ● Modèle EfficientNetB2 ● Call Back + Fine Tuning ● target_min - 14 classes	Entrée: *data.csv*
09.e - Computer Vision - ResNet50.ipynb	Entrainement/Prédiction: ● Modèle ResNet50 ● Call Back + Fine Tuning ● target_min - 14 classes	Entrée: *data.csv*
09.f - Computer Vision - ResNet152.ipynb	Entrainement/Prédiction: ● Modèle ResNet152 ● Call Back + Fine Tuning ● target_min - 14 classes	Entrée: *data.csv*
09.g - Computer Vision - ResNet50 - Augmented Data.ipynb	Entrainement/Prédiction: ● Modèle ResNet50 ● Augmented Data ● Call Back + Fine Tuning ● target_min - 14 classes	Entrée: *data.csv*
09.h - Computer Vision - ResNet50 - Augmented Data - Corr	Entrainement/Prédiction: ● Modèle ResNet50 ● Augmented Data ● Call Back + Fine Tuning ● Correction du labeling ● target_min_corr - 14 classes	Entrée: *data_corr.csv*
09.i - Computer Vision - ResNet50 - Augmented Data - Limited types.ipynb	Entrainement/Prédiction: ● Modèle ResNet50 ● Augmented Data ● Call Back + Fine Tuning ● 7 classes	Entrée: *data_with_meta.csv*
09.j - Computer Vision - ResNet50 - Augmented Data - Limited types - Corr.ipynb	Entrainement/Prédiction: ● Modèle ResNet50 ● Augmented Data ● Call Back + Fine Tuning ● Correction du labeling ● 7 classes	Entrée: *data_with_meta_corr.csv*
09.k - Performances modeles Transfer Learning.ipynb	Comparaison des performances des modèles de Computer Vision	Entrée: *classification_report.csv*

10 - LIME interpretability.ipynb	Interprétabilité de Lime	Entrée: *data.csv*
11 - Data extraction.ipynb	Extraction de données	Entrée: *data.csv*
helpers.py	Méthodes utilisées par les différents programmes afin de nous éviter de les redéfinir à chaque utilisation.	