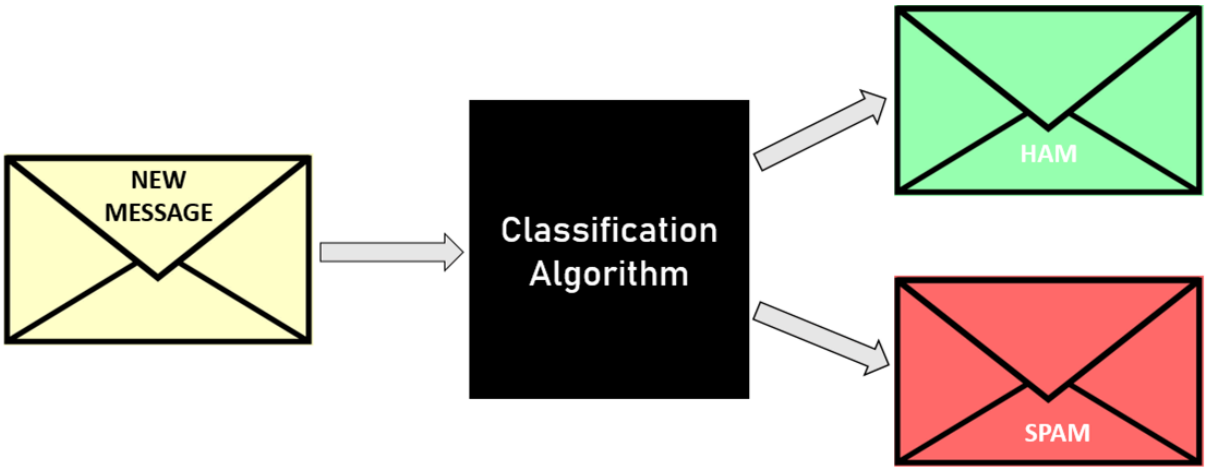
Contexte du projet :

L'objectif de notre projet est de développer un programme qui est capable de détecter si un message est un spam ou non. L'utilisateur pourra entrer un message, et en sortie, le programme indiquera si ce message est légitime (**ham**) ou suspecté d'être un **spam**.



Livrables :

- Créer un/des notebook reproductible, commenté et expliqué (IMPORTANT !)
- Créer un repo git et un espace sur github/gitlab pour le projet
- Faire une présentation (slides) qui explique votre démarche et les résultats obtenus avec :
    - le fonctionnement global du classifieur
    - la procédure suivie pour préparer les données et le preprocessing
    - la procédure suivie pour trouver un modèle adapté
    - le modèle d'IA sélectionné
- BONUS :
    - *Application streamlit qui fait de la prédiction en temps réel d'un message déposé par l'utilisateur
    - * Autres fonctionnalités avancées

Critères de performance

- Compréhension du jeux de données
- Capacité à préparer les données
- Performance des modèles de prédiction
- Capacité à apporter une solution dans le temps imparti
- Rédaction du notebook
- Qualité du synthèse du travail

Etapes du projet identifiées:

1) Récupération des données
2) Chargement et inspection initiale des données
3) Nettoyage des données
4) Préparation des donnés
5) Analyse des données
6) Choisir un modèle
7) Entrainement du modèle
8) Test du modèle
9) Création d'une interface web simple (optionnel)

Qu'est-ce qu'un spam ?

Un **spam** désigne toute communication non sollicitée, généralement envoyée en masse. Ces messages ont souvent pour objectif de **tromper**, **d'arnaquer** ou de **vendre**, et peuvent représenter une menace sérieuse pour la sécurité des utilisateurs.

Pourquoi les spams posent problème ?

Les spams posent de nombreux défis, notamment :

1. **Pollution numérique** : Ils encombrent les boîtes de réception, rendant la gestion des messages importants plus difficile.
2. **Liens dangereux** : Ils contiennent souvent des liens malveillants menant à des sites frauduleux ou infectés par des logiciels malveillants.
3. **Perte de temps** : Trier les messages authentiques des spams peut être fastidieux.
4. **Risques de sécurité** : Certains spams incluent des logiciels malveillants, comme des rançongiciels capables de bloquer ou compromettre nos appareils.

Les types de spams

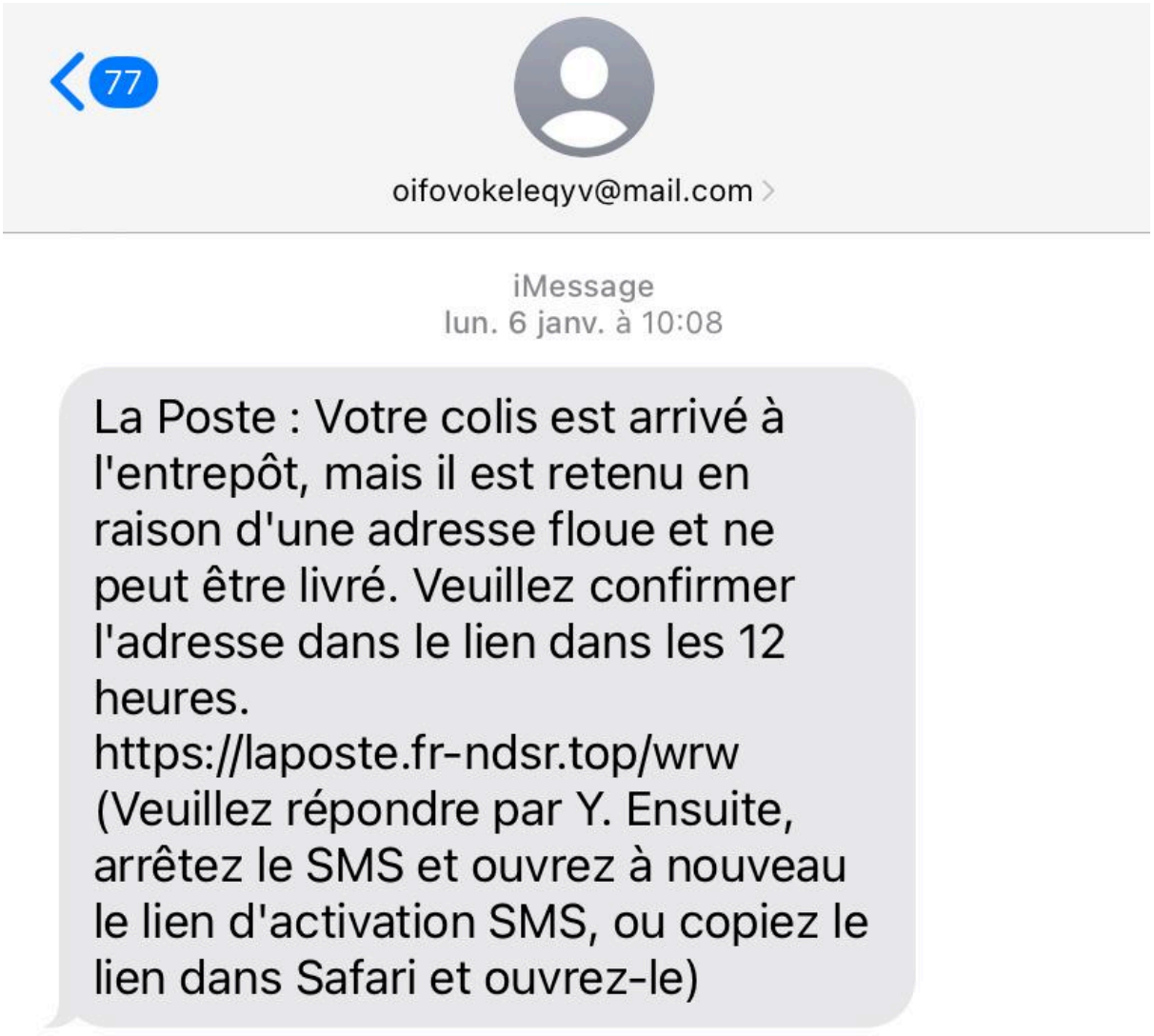
Les spams se présentent sous différentes formes, allant des messages commerciaux abusifs aux communications malveillantes. Voici les principales catégories :

1. **Spams commerciaux** :
    - Messages publicitaires envoyés sans le consentement préalable des destinataires, souvent en violation des lois en vigueur.
2. **Spams malveillants** :
    - **Phishing (hameçonnage)** : Tentatives de dérober des informations personnelles ou bancaires.
    - **Arnaques financières** : Sollicitations pour des dons fictifs ou des promesses de gains faciles.
    - **Logiciels malveillants** : Messages contenant des pièces jointes ou des liens infectés par des virus, des chevaux de Troie ou des rançongiciels.

Exemple concret de phishing

Un exemple récent illustre bien la menace des spams :

Après avoir passé une commande en ligne, j'ai reçu un message étrange d'un expéditeur inconnu :



Ce message imite un service officiel pour inciter l'utilisateur à cliquer sur un lien douteux. Une fois le lien ouvert, des informations personnelles pourraient être volées ou l'appareil infecté par un malware.

Comment reconnaître un spam ?

Les spams partagent souvent des caractéristiques communes :

1. **Offres ou gains incroyables** : Promesses de remboursements inattendus, de gains miraculeux ou de cadeaux gratuits.
2. **Sens de l'urgence** : Messages pressant l'utilisateur d'agir immédiatement sous peine de sanction (défaut de paiement, fermeture de compte, etc.).
3. **Demandes de données personnelles** : Sollicitations pour mettre à jour ou confirmer des informations sensibles (mots de passe, coordonnées bancaires, etc.).
4. **Pièces jointes suspectes** : Encouragement à télécharger des fichiers potentiellement malveillants.
5. **Arnaques émotionnelles** : Appels à l'aide prétendument envoyés par un proche en détresse ou des appels aux dons frauduleux.
6. **Propositions douteuses** : Invitations à participer à des jeux-concours ou à des pyramides financières.

Récupération des données :

On a déjà un dataset mis à disposition accessible via ces liens (l'un ou l'autre):

- https://archive.ics.uci.edu/ml/datasets/sms+spam+collection
- https://www.kaggle.com/uciml/sms-spam-collection-dataset

Il contient **5 574 messages** en anglais tous dejà étiquetés **ham** (légitime) ou **spam**.

En consultant le dataset via le lien fourni sur Kaggle, nous avons pu observer un **déséquilibre significatif** dans la distribution des données : le pourcentage de messages spam est beaucoup plus faible que celui des messages légitimes (ham).

Ce déséquilibre pourrait introduire un biais dans l'apprentissage du modèle, en le rendant moins performant pour la détection des spams. Pour obtenir de **meilleurs résultats**, on a décider qu'il faudrait **enrichir la base de données** en r**echerchant et intégrant de nouvelles données** contenant des messages **spam supplémentaires** afin d'augmenter la taille globale du dataset et de rééquilibrer la répartition entre les labels.

Plutôt que de supprimer des exemples existants, ce qui aurait conduit à une perte d'informations précieuses, nous avons privilégié cette stratégie d'enrichissement pour conserver un maximum de diversité et de représentativité dans les données d'entraînement. Cette stratégie a permis non seulement de résoudre le problème de déséquilibre, mais aussi de rendre le modèle **plus robuste** en augmentant la diversité des cas étudiés. En intégrant une variété plus large de spams, le modèle est mieux préparé à reconnaître des motifs plus subtils ou variés, **améliorant ainsi sa capacité à généralise**r.

Recherche de données supplémentaires

Trouver des datasets supplémentaires contenant des spams de SMS a été très difficile. Mais en nous basant sur la définition d'un spam, nous avons décidé d'intégrer des **spams d'e-mails** comme source complémentaire de données.

Justification :

- Les spams, qu'ils soient sous forme de SMS ou d'e-mails, partagent des caractéristiques communes :
    - Contenu trompeur ou nuisible.
    - Appels à l'action urgents : *"Cliquez ici", "Agissez maintenant"*.
    - Usage de mots-clés typiques : *Gratuit, Gagner, Récompense, Urgent*.
    - Liens malveillants ou raccourcis.
    - Langage inhabituel : Majuscules excessives, fautes d'orthographe ou grammaire, ponctuation exagérée.

En d'autres termes, **un spam reste un spam**, quel que soit son format ou son canal. Ainsi, utiliser des spams d'e-mails pour enrichir le dataset permet de :

- Augmenter la quantité de données étiquetées comme spams.
- Rééquilibrer la répartition entre les classes (**spam** et **ham**).

Nous avons identifié trois datasets pertinents pour notre analyse :

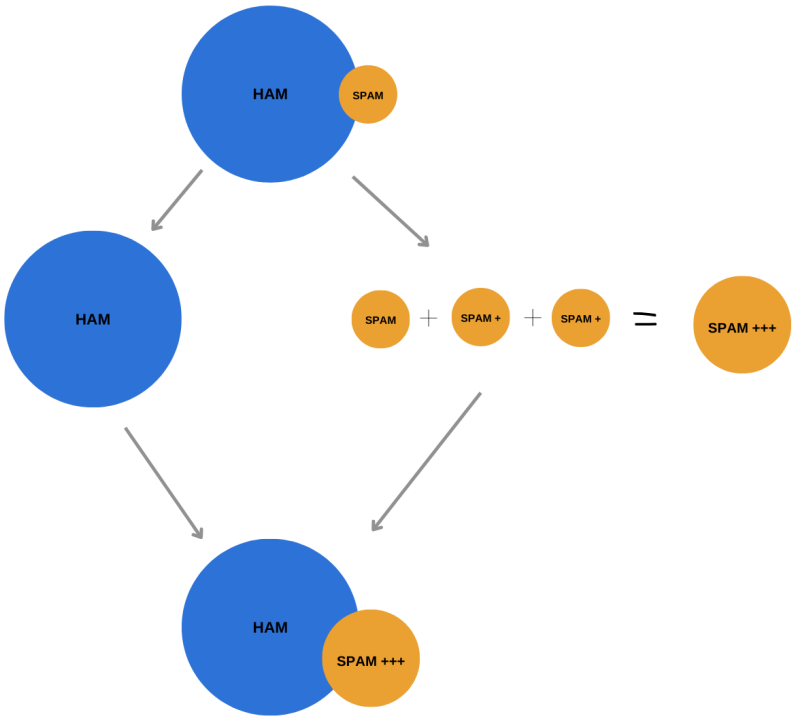
- **Nazario.csv :**
  https://storage.googleapis.com/kaggle-data-sets/5074342/8502378/bundle/archive.zip?X-Goog-Algorithm=GOOG4-RSA-SHA256&X-Goog-Credential=gcp-kaggle-com%40kaggle-161607.iam.gserviceaccount.com%2F20250109%2Fauto%2Fstorage%2Fgoog4_request&X-Goog-Date=20250109T132707Z&X-Goog-Expires=259200&X-Goog-SignedHeaders=host&X-Goog-Signature=147d80e909f1e851fa50c21d7f7e05cf598414bb5c0589867a8905919fa25d5e384a6a7d09bc484cde5811f91e7b9fdd97bb5443807a7c76e51aab8e0e40d2246cbf93f940824613c160c27218f54f222a43622770768369b62fc2d51ee4bc4c98acbbe1b8757c1b41d7e292e3ba89c260533b250d38acbea139d7f73875e947f8a19a2745958fc2a5fc529f396aa69807dd6e8225fc7bb9a1df793166cb56fb8bb36cb465e087871f7cf15e50ef53a34ed036f628879efe7b46b9ecfbc18c4e7299113956c461e7a5db0f74220d134508fb36741a01552fd719ff2d6c765bd0f15f38bf8c1ab52fb11aa99c683ecc570a23e03d4f30b7385b0488e14905b19d

- **spam_ham_dataset.csv :**
  https://storage.googleapis.com/kaggle-data-sets/109196/260807/bundle/archive.zip?X-Goog-Algorithm=GOOG4-RSA-SHA256&X-Goog-Credential=gcp-kaggle-com%40kaggle-161607.iam.gserviceaccount.com%2F20250109%2Fauto%2Fstorage%2Fgoog4_request&X-Goog-Date=20250109T090038Z&X-Goog-Expires=259200&X-Goog-SignedHeaders=host&X-Goog-Signature=5d47839ba114efec8698afb869a3b5a3d2531e3da1b8f6f0bd35e4b87a9e299f9f32a6439d86e3b0bb086d7a9712a75a0bd5ca1444ab18d6203f46076a858ddef4807a9a14e4b2700c2258374f13b1a2f0b4f223b0c120b88f39e884796eeaf055aef140dfceb9938119bc60b38f7b9d067115e34ce5491211fe724eb4be596e73cad922bee0353bcffcdd5ab4e793daa252ff86203e3497fd36ffe830d53d26423c0fae73b7c9db5d9348f9da4af493d116ef7fe88884c84045730d6fab4381cf4359c403173acbdbb5db8789aad865fc6fb532c08e12b0a4e33cec43b3082093b16e05d6111a0287f0ee19f94257659811b213e98c72bea03fe560412ad2ba

- **emails.csv :**
  https://storage.googleapis.com/kaggle-data-sets/3690036/6399975/bundle/archive.zip?X-Goog-Algorithm=GOOG4-RSA-SHA256&X-Goog-Credential=gcp-kaggle-com%40kaggle-161607.iam.gserviceaccount.com%2F20250108%2Fauto%2Fstorage%2Fgoog4_request&X-Goog-Date=20250108T143448Z&X-Goog-Expires=259200&X-Goog-SignedHeaders=host&X-Goog-Signature=15df70d1e380b34688ee8b82323007a5ecf2ed44c92a57b0535b98618d2b63c71b211e4872a3b3ae5aeaba03d6ad7a63068bb0ddafa39db652d3c498484bb0a8bb6ff7f46145775f5e3b456cfe44d383c20a86bf12168de422d7d3e26a221941b4b84c78653f66e285b9123701494d5101256dddb38a020a9624f0b344d758dd62ec454adab1cab1414adb4a7d7d97466e3dd3907db305daad16c65c946d64d4078b98ba8d746043a510e6b2b82bd82d7f85ea3370fcf1c95325f621db568ba09967629ad383d2f0205f4a50e1f5a11f9265dee769983b22c2a26cf57466d66fc772cfd3e00bded3302a83e7e43b2219d229e7568640c444671682b2bace6b8c

Nettoyage et transformation des données

Pour rendre notre dataset exploitable, nous avons entrepris un processus de nettoyage et de transformation des données, avec pour objectif de standardiser les informations et d'enrichir le dataset.



Étape 0 : Récupération des données pertinentes et harmonisation des datasets

L'objectif global était d'extraire les informations pertinentes (messages et labels) et d'adapter les datasets pour les rendre fusionnables et comparable entres eux.

On a d'abord pris soin de separer les spams et les hams pour chaques dataset. en conservant uniquement les hams du dataset original et tout les spams provenant du dataset original ou additionnaux. On va les traiter dans des datasets differents pour chaque.

Chaque dataset a donc suivi un algorithme de préparation different mais tout en respectant des principes communs :

- identifier et nommer les colonnes importantes.
    - (`message` et `label`)
- Normaliser la structure et le format des données.
    - Dans la colonne `label`, remplacer le terme *spam* par la valeur *1*.
    - Pour les jeux de données issus de courriels, ne conserver que le contenu du message (en ignorant l'objet, le destinataire et l'expéditeur).

Aperçu de la structure souhaité:

	label	text
1363	1	are you ready to get it ? hello ! viagra is ...
1364	1	would you like a $ 250 gas card ? don ' t let...
1365	1	immediate reply needed dear sir , i am dr ja...
1366	1	wanna see me get fisted ? fist bang will sho...
1367	1	hot stock info : drgv announces another press ...

Étape 1 : Standardisation des données

Nous avons commencé par utiliser des **expressions régulières** pour standardiser les messages. Cela impliquait de remplacer des éléments spécifiques comme :

- Les adresses e-mails, les URLs, et numéros de téléphone par le token `[ENTITY]`.
- Les mots explicites par le token `[EXPLICIT]`.

Cette standardisation présentait plusieurs avantages :

1. **Focus sur le contenu textuel** : En éliminant les informations spécifiques comme les URLs ou les e-mails, le modèle pouvait se concentrer sur les mots et phrases qui caractérisent les spams.
2. **Réduction de la longueur excessive des messages** : Les longues URLs dans les spams e-mails allongeaient considérablement les textes. En les remplaçant par des tokens, nous avons uniformisé la longueur des messages.

Nous avons ensuite supprimer les lignes qui dépassait 250 caractères pour les spams supplémentaires (uniquement ceux issus des e-mails). Pourquoi 250 ? Cette limite est proche de celle des spams SMS initiaux, ce qui nous permettait de récupérer les messages d'e-mails les plus concis et semblable aux spams SMS.

Sans token :

```
'hilarious prank call service  please visit http : / / ukprankcalls . com to play a hilarious joke on your mates !'
```

Avec token :

```
'hilarious prank call service please visit [entity]'
```

Étape 3 : Enrichissement des données avec de nouvelles colonnes

Pour ne pas perdre d'informations importantes et enrichir le dataset, nous avons ajouté de nouvelles colonnes contenant des caractéristiques textuelles pertinentes :

- **Longueur des messages (avec et sans espace)**
- **Nombre de mots**
- **Caractères spéciaux (espaces non compris)**
- **Nombre de majuscules**

Ces informations enrichissent le dataset avec des signaux clés que le modèle peut exploiter pour mieux distinguer les spams des messages légitimes.

Étape 4 : Suppression des caractères spéciaux inutiles

Nous avons ensuite supprimé les **caractères spéciaux superflus**, tout en préservant les tokens `[ENTITY]`.

structure des datasets à ce stade:

	label	text	is_eng	len	len_no_spaces	n_words	n_special_chars	n_capital_letters	explicit
9	1	sends them to their final destination designat...	True	188	150	38	6	0	0
17	1	emile the cablefilterz will allow you to recei...	True	179	148	32	9	6	0
20	1	muniz govenment don t want me to sell undergro...	True	190	157	34	7	6	0
22	1	tinydrive 2 2 gb 5 x 2 25 x 2 5 inches h x w x...	True	169	127	43	17	12	0
27	1	80 smart ink system [entity]	True	165	142	24	120	6	0
33	1	visit [entity]	True	16	14	3	3	6	0

Étape 5 : Fusion des datasets

Une fois que nos spams (anciens et nouveaux) et nos hams étaient prêts, nous avons entrepris de les **fusionner** en un seul gros fichier CSV.

Étape 6 : Analyse et ajout de métriques liées à [ENTITY]

Nous avons ensuite comptabilisé le nombre d'apparitions de `[ENTITY]` dans chaque message. Pourquoi ? Les spams réutilisent souvent des éléments comme les liens ou les adresses mail pour rediriger les victimes. Une forte présence de `[ENTITY]` peut révéler un spam très insistant ou structuré autour de redirections suspectes. Cela devient donc un **signal** pertinent pour le modèle de détection.

Étape 7 : Prétraitement textuel avancé

Pour continuer à affiner nos données, nous avons réalisé :

1. **La tokenisation** : Découper chaque texte en mots, symboles, ou chiffres, afin de faciliter l'analyse textuelle.
2. **La suppression des stop words** : Retirer les mots très fréquents mais peu informatifs (ex. "the", "and", "of") pour concentrer le modèle sur les termes importants.
3. **Le stemming** : Réduire les mots à leur racine (ex. "winning", "won" deviennent tous "win"), ce qui permet de traiter les variations morphologiques comme un même concept.

Étape 8 : Analyse des mots fréquents et création de caractéristiques discriminantes

En étudiant la fréquence des mots, nous avons identifié certains termes caractéristiques des spams comme `"free"`, `"click"`, `"claim"`, `"offer"`, `"win"`, `"prize"` et `"want"`.

Pour chacun de ces mots, nous avons créé une colonne supplémentaire afin de compter leurs occurrences dans le message. Par exemple, la colonne `w_free` indique combien de fois le mot `"free"` apparaît. Ces nouvelles features fonctionnent comme des indicateurs forts de probabilité de spam.

Étape 9 : Finalisation et enregistrement

Toutes ces étapes terminées, nous avons enregistré notre **dataset final**. Les messages sont maintenant nettoyés, normalisés et agrémentés de nombreuses colonnes complémentaires : tout est prêt pour l'entraînement du modèle de classification.

Étape 10 : Préparer les données pour le modèle

Arrivés à ce stade, nous avons séparé :

- **Les données textuelles**, qui devaient être transformées grâce au **TF-IDF**. Cette méthode attribue un poids à chaque mot en fonction de sa fréquence dans un message par rapport à sa fréquence globale dans tout le corpus.
- **Les données numériques**, prêtes à être utilisées directement (longueurs, comptes de majuscules, etc.).

La fusion de **TF-IDF** (partie textuelle) avec ces **caractéristiques numériques** a donné un tableau final très complet, composé de centaines ou milliers de dimensions décrivant chaque message.

Étape 11 : Faire face au déséquilibre des classes

Même après avoir ajouté plus de spams, nous avions toujours un léger déséquilibre. Pour y remédier, nous avons appliqué **SMOTE** (Synthetic Minority Oversampling Technique), qui crée artificiellement de nouveaux spams en interpolant les données existantes. Cette technique a considérablement amélioré la répartition des classes dans notre jeu d'entraînement, rendant le modèle plus juste et moins enclin à rater des spams rares.

Étape 13 : Entraînement et évaluation du modèle

Avec nos données prêtes, nous avons opté pour un **Random Forest Classifier**, un algorithme puissant et polyvalent. Nous avons coupé nos données en deux sous-ensembles :

- **90 %** pour l'entraînement (pour ne pas trop réduire le nombre de cas disponibles)
- **10 %** pour le test

Le modèle a atteint un **score de précision** élevé sur l'ensemble de test, ce qui était encourageant. Cependant, la simple précision ne suffit pas toujours à juger la qualité d'un classifieur spam : il faut aussi regarder la capacité à **ne pas laisser passer** un spam (rappel) et à **ne pas classer un ham comme spam** (précision sur la classe spam).

Étape 14 : Analyser les erreurs pour s'améliorer

Pour creuser plus loin, nous avons inspecté **les faux positifs** (hams classés en spams) et **les faux négatifs** (spams classés en hams). Concrètement, nous avons ressorti les messages bruts à partir du dataset initial (`df_raw`) pour comprendre ce qui pouvait tromper le modèle.

- Les **faux positifs** étaient très rare. Parfois des messages légitimes, mais contenant des mots alléchants ("free", "win") ou un style rappelant celui d'un spam.
- Les **faux négatifs** se composaient de spams normaux manqué et notamment de spams à caractère explicite, beaucoup plus rares dans notre dataset mais présents. Le modèle manquait de données similaires pour bien apprendre à les repérer.