
RECONNAISSANCE DES PLANTES

Membres de l'équipe projet : Bernadette GASMI, Jean-Baptiste MACK, Morgan PERCHEC, Lionel SCHEIDER

28 juillet 2025



Jean-Baptiste MACK



Morgan PERCHEC



Bernadette GASMI



Lionel SCHNEIDER



Résumé : Dans le cadre de notre formation DataScientist, nous mettons en pratiques les techniques d'intelligence artificielle enseignées avec un passage obligé par le Machine Learning pour progressivement conclure vers le Deep Learning.

Mots clés :

Acronymes et sigles

Ce tableau présente les acronymes et sigles employés dans ce rapport.

Acronyme	Signification
ML	Machine Learning
DL	Deep Learning

1.	Introduction	6
1.1.	Cadrage du projet	6
1.1.1.	Expression du projet proposé par DataScientest.....	6
1.1.2.	Contexte du projet	7
1.1.3.	Scénario.....	8
1.1.4.	Etat des lieux.....	10
1.1.5.	Revue littéraire	10
1.1.6.	Présentation des datasets.....	11
1.1.7.	Objectifs	13
2.	Compréhension et manipulation des données.....	15
2.1.	Exploration des images brutes	16
2.1.1.	Exploration des données brutes dataset PlantVillage complet.....	16
2.1.2.	Exploration des données brutes dataset PlantVillage/Segmented.....	20
2.2.	Extraction des caractéristiques	25
3.	Explorer les caractéristiques	27
3.1.	Gestion des données manquantes	28
3.2.	Identification des outliers	28
3.3.	Distribution des caractéristiques	31
3.4.	Relations entre les caractéristiques	39
4.	Ré-échantillonnage	41
5.	Split train/test (et éventuellement validation)	41
6.	Data augmentation (train uniquement)	42
7.	Standardisation / normalisation des features (fit sur le train, transform sur test)	43

8.	Sélection des meilleures caractéristiques	43
9.	Conclusion	52
10.	Bibliographie	54
11.	Annexes.....	54
11.1.	Organisation du projet	55
11.2.	Reco_plante_Rapport exploration des données 24_07_2025.xls	55
11.3.	Exploration du dataset complet PlantVillage.....	56
11.4.	Analyse exploratoire des caractéristiques	61
11.5.	Environnement de développement	61

TABLE DES ILLUSTRATIONS

Figure 1 : Scénario global.....	9
Figure 2 : Sélection du dataset pour premier cycle	12
Tableau 1 : Tableau comparatif des datasets	13
Figure 3 : Structure du dossier PlantVillage.....	17
Figure 4 : Distribution du nombre d'images par espèce et modalité.....	17
Figure 5 : Comparaison du nombre d'images par modalité et espèce	18
Figure 6 : Nombre d'images par extension	18
Figure 7 : Dimensions (largeur et hauteur) des images	19
Tableau 2 : répartition des 38 classes de feuilles saines et malades	21
Figure 9 : Affichage des images inexploitables	22
Figure 10 : Images dupliquées	22
Figure 11 : Nombre d'images par espèce et distribution des classes.....	23
Figure 12 : Répartition des images saines vs malades par espèce, Distribution des classes saines, Distribution des classes malades	24
Figure 13 : Caractéristiques des feuilles de plantes.....	26
Figure 14 : données manquantes	28
Figure 15 : Identification des outliers par plante – identification des outliers par maladie.....	29
Figure 16 : Identification des outliers par feature et par classe	29
Figure 17 : Distribution des caractéristiques pour objectif 1 "Quelle est cette plante?"	31
Figure 18 : Distribution des caractéristiques pour objectif 2 1/3	33
Figure 19 : Distribution des caractéristiques pour objectif 2 2/3	33
Figure 20 : Distribution des caractéristiques pour objectif 2 3/3	34
Figure 21 : Distribution des caractéristiques pour objectif 3 "Quelle est la maladie ?".....	35
Figure 22 : Boxplots de la densité de contours selon les différentes maladies.....	37
Figure 23 : Histogramme de la distribution des caractéristiques par type de plantes	38
Figure 24 : Heatmaps des corrélations entre features	39

Tableau 3 : Tableau de synthèse des différentes approches possible pour analyser l'importance des features	44
Figure 25 : Heatmap des Fréquences de Sélection des Caractéristiques	48
Figure 26 : Heatmap des Importances des Caractéristiques.....	49
Figure 27 : Importance des caractéristiques par classe (SHAP).....	50

1. Introduction

1.1. Cadrage du projet

1.1.1. Expression du projet proposé par DataScientest

« L'objectif de ce projet est de localiser et classifier l'espèce d'une plante dans une image. Une fois la classification faite, vous pouvez retourner à l'utilisateur une description de celle-ci et identifier les possibles maladies. L'application sera donc capable à partir d'une image prise par l'appareil photo de donner une succession d'informations à l'utilisateur. »

Ressources à consulter :

Données :

- Identification de l'espèce :
<https://www.kaggle.com/vbookshelf/v2-plant-seedlings-dataset>,
<https://storage.googleapis.com/openimages/web/download.html>,
<https://cocodataset.org/#home>
- Maladie sur les plantes :
<https://www.kaggle.com/vipooooool/new-plant-diseases-dataset>,
<https://www.kaggle.com/saroz014/plant-disease>,
<https://www.kaggle.com/abdallahhalidev/plantvillage-dataset>

Bibliographie :

- <https://hal.archives-ouvertes.fr/hal-01629183/document>
- https://play.google.com/store/apps/details?id=org.plantnet&hl=en_US
- <https://www.plantsnPlantNet Plant Identification - Apps on Google Playap.com/>
-

Dans le cadre de ce projet, des recherches complémentaires ont été effectuées sur Internet afin de mieux comprendre les méthodes de classement, d'organisation et d'archivage de documents. Des ressources variées, comme des sites éducatifs, des forums spécialisés ou encore des blogs techniques, ont été consultées pour enrichir les connaissances théoriques et pratiques. Par ailleurs, des outils d'intelligence artificielle tels que ChatGPT et Mistral ont été utilisés pour poser des questions précises, obtenir des explications claires, explorer différentes approches et affiner certaines idées. Ces échanges ont permis de gagner du temps, de mieux structurer les étapes du projet et de répondre efficacement aux obstacles rencontrés.

Partant de cette démarche, nous avons bâti notre projet, présenté en détail dans la suite de ce rapport.

1.1.2. Contexte du projet

Contexte

- Contexte d'insertion du projet dans votre métier.
- Du point de vue technique.
- Du point de vue économique.
- Du point de vue scientifique.

Ce projet s'inscrit dans le contexte de changement climatique, il y aura très probablement des besoins croissants pour des diagnostics automatisés et rapides afin d'apporter une aide à la filière agriculture et forestière :

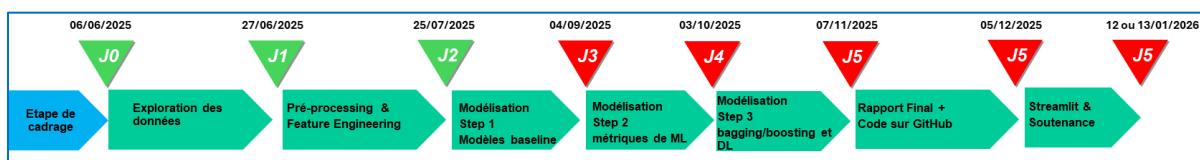
- la nécessité de surveiller la présence de maladies au plus tôt voir l'apparition de nouvelles maladies
- l'enjeu à venir est d'adapter les modèles aux différents stades de croissance des feuilles (semis, jeunes, matures, séchées) et d'intégrer la régionalité et la variabilité climatique.

Organisation projet

Voir annexe 11.1

Cycle de vie du projet

Le cycle de vie du projet est le suivant :



Les chapitres suivants s'appuient sur les décisions de chaque jalon. Cette version du rapport fait état de notre avancement jusqu'au passage du jalon2.

Risques projet

Risque 1 : Beaucoup d'itérations

- Cause : Le démarrage du projet arrive trop tôt par rapport au planning des apprentissages nécessaires pour le projet.
- Mitigation : Lire les modules des cours par anticipation

Risque 2 : Erreurs probables sur les interprétations

- Cause : La durée entre J0 –J2 très courte
- Mitigation : Suites aux analyses exploratoires, les recommandations ne sont pas toutes appliquées mais elles ne sont pas perdues et elles seront considérées dans les runs suivants.

1.1.3. Scénario

La figure suivante illustre le scénario global établi à partir de l'objectif du projet.

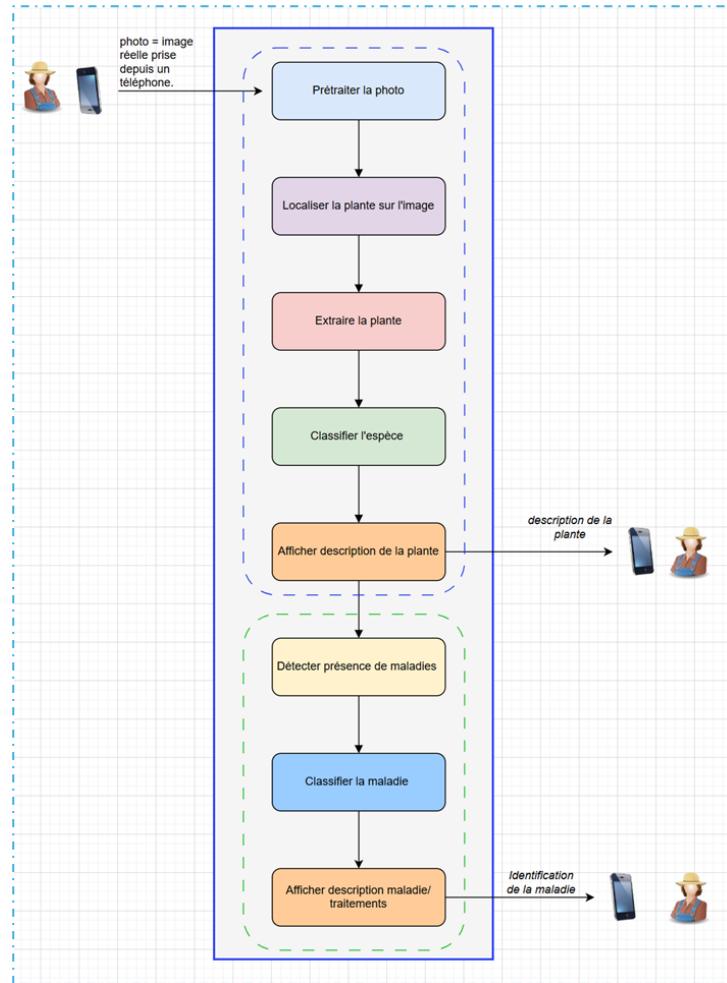


Figure 1 : Scénario global

Nous avons identifié 3 scénarios possibles :

Scénario1 : Détection de plantes invasives ou nuisibles parmi des plantules de maïs

- Contexte : agriculture
- Dispositif : Application mobile
- Attentes :
 - Détection rapide et fiable de plantes invasives
 - Détection de maladies ou carences
 - Système d'alerte
- Contraintes :
 - Faible fréquence des espèces recherchées
 - Végétation dense, forte variabilité interindividuelle

Scénario2 : Surveillance de la croissance de plantes de maïs

- Contexte : agriculture
- Dispositif : Application mobile
- Attentes :
 - Reconnaissance des stades de croissance
 - Estimation du stress hydrique

- Détection de maladies ou carences
- Contraintes :
 - Faible fréquence des espèces recherchées
 - Végétation dense, forte variabilité interindividuelle

Scénario3 : A partir d'une photo d'une feuille cadrée

- Identifier la classe/espèce de la plante
- Dire si la plante est malade ou saine
- Si elle est malade, donner le nom de la maladie

Lors de la revue du jalon1, la décision a été prise de retenir le scénario3, et de prioriser les étapes du scénario global : Classifier l'espèce, Déetecter la présence de maladie et Classifier la maladie, les autres étapes seront développées ultérieurement suivant notre avancement.

1.1.4. Etat des lieux

Nous avons effectué un état des lieux pour les applications similaires à l'objectif proposé par DataScientest, les datasets et les critères de qualité

L'annexe A présente des applications d'identification des plantes et des applications qui permettent d'identifier des maladies potentielles

L'annexe B présente des datasets d'images de plantes avec leurs caractéristiques

L'annexe C présente des critères de qualités parmi lesquels nous ferons une sélection pour le projet

1.1.5. Revue littéraire

Nous avons parcouru la publication proposée par DataScientest et exploré d'autres publications. La publication suivante a retenu particulièrement notre attention car c'est une revue systématique des méthodes les plus performantes de Machine Learning, de traitement d'image, et d'algorithmes de Deep Learning appliqués à la reconnaissance des espèces végétales, avec citation des datasets utilisés publiée en 2024.

[https://www.researchgate.net/publication/384455442 A systematic review of deep learning techniques for plant diseases](https://www.researchgate.net/publication/384455442_A_systematic_review_of_deep_learning_techniques_for_plant_diseases) :

Cette publication confirme le Deep Learning comme technique la plus performante. Cependant, lors du passage de jalon1, il a été convenu d'effectuer tout le cycle de vie du projet (étapes Machine Learning, Deep Learning) dans un esprit de mise en application des techniques apprises dans le cursus DataScientist.

La revue littéraire et l'état des lieux nous ont permis :

- de comprendre les enjeux, de définir les objectifs du projet dans le détail, avec précision et les risques compte tenu de la complexité du sujet,

- de choisir les datasets qui seront utilisés compte tenu de leurs caractéristiques
- d'envisager des pré-sélections de modèles de Machine Learning et Deep Learning

1.1.6. Présentation des datasets

Les 6 datasets proposés par DataScientest sont présentés ci-dessous :

V2 Plant Seedlings : ~ 4 800 images de semis couvrant 12 espèces courantes, acquises en conditions contrôlées.

Open Images V6 : ~ 9 000 000 d'images annotées avec plus de 16 000 classes d'objets, incluant de nombreuses plantes et feuilles.

COCO (Common Objects in Context) : ~330 000 images avec 80 catégories d'objets, prises dans des scènes naturelles variées.

New Plant Diseases : ~ 87 000 images de feuilles malades couvrant 38 maladies sur 14 espèces, avec annotations « malade vs sain » et étiquettes de maladie détaillées.

Plant Disease : ~ 54 000 images segmentées (une feuille par photo) de 38 maladies + sain sur 14 cultures (dérivé de PlantVillage).

PlantVillage : ~ > 54 000 photos de feuilles (saines et malades) réparties sur 38 classes de maladies, capturées en conditions variées.

A partir des 6 datasets, nous avons effectués plusieurs sélections successives afin de n'en retenir qu'un, PlantVillage, pour exercer un premier cycle d'activités aboutissant à la sélection des meilleures caractéristiques, avant entraînement des modèles d'apprentissage.

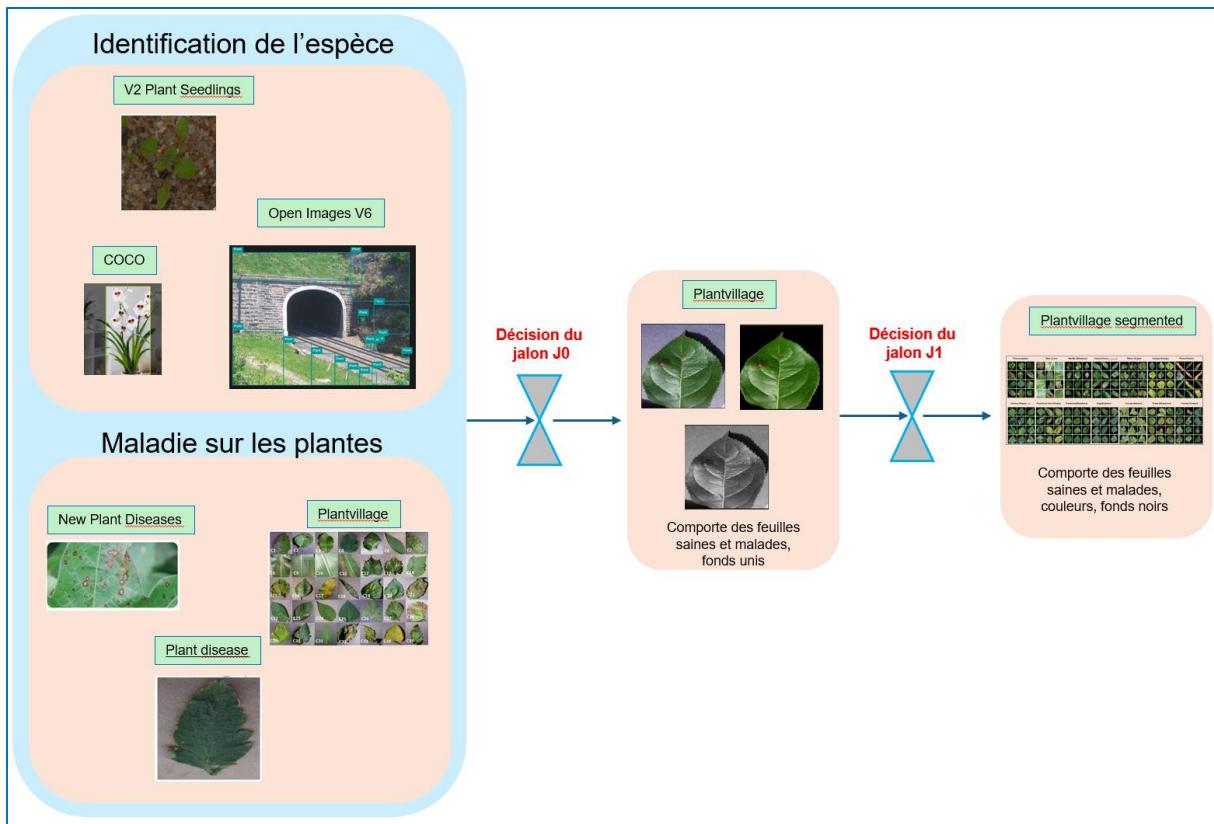


Figure 2 : Sélection du dataset pour premier cycle

Les 3 datasets V2 Plant Seedlings, Open Images V6 et COCO sont mis de côté car plus généralistes et permettent la détection d'une plante dans une image. On trouve en effet des images de plantes au stade de croissance, avec des environnements différents et nous avons trouvé peu d'espèces communes entre ces 3 datasets.

New Plant Disease est structuré en 3 dossiers : Train (38 dossiers), Valid (38 dossiers), et Test (33 dossiers). Il est recréé à l'aide d'une augmentation hors ligne à partir du dataset d'origine Plantvillage.

Grâce à la réduction de la complexité, ce choix nous permet :

- D'organiser le projet (espace de travail, outils, gestion de configuration)
- Définir l'architecture du code
- S'approprier les connaissances apprises en formation sur un cas réel

Toute la suite du document est basée sur l'utilisation du dataset PlantVillage pour répondre aux objectifs du projet.

Lors de la prochaine étape du projet, nous rajouterons les 2 datasets : Plant Disease et New Plant Diseases, ce qui permettra de couvrir une plus grande diversité de conditions réelles (fonds naturels, variations d'éclairage et d'angles) et d'augmenter la couverture des maladies et espèces rares, améliorant la robustesse de nos modèles face à des cas peu représentés. Le recouplement devrait se faire au niveau des classes (espèces/maladies), mais pas d'un

chevauchement systématique de fichiers image. Cela nous rapprochera d'une application finale opérationnelle.

Le tableau suivant est un comparatif des 3 datasets permettant la détection des maladies.

Caractéristique	PlantVillage	Plant Disease	New Plant Diseases
Taille du dataset	~54,000 images 38 classes (espèce + maladie ou healthy)	~54,000 images	~87000 images
Nombre d'espèces	14 espèces de plantes	38 classes de maladies sur plusieurs espèces	> 60 espèces de plantes
Nombre de maladies	26 maladies (avec healthy classes)	>50 maladies (certaines plantes avec plusieurs maladies)	~120 (certaines rares) maladies classes
Type d'image	Images avec des fonds unis	Images variées, prises sur le terrain	Images prises en milieu naturel
Annotation	Espèce + maladie + healthy	Espèce + maladie	Espèce + maladie + conditions environnementales
Format	JPG / PNG	JPG	JPG / JPEG / PNG
Accessibilité	Ouvert (Kaggle, GitHub)	Ouvert (Kaggle)	Ouvert Dataset (Google Search, Zenodo, etc.)

Tableau 1 : Tableau comparatif des datasets

1.1.7. Objectifs

Lors de la revue jalon1, nous avons convenu de partir sur les 3 objectifs suivants :

Objectif 1 – Classification de l'espèce – Quelle est cette plante ?

Développer un modèle (multi-classe) capable de reconnaître automatiquement l'espèce de chaque feuille (orange, maïs, tomates, etc.) à partir de ses caractéristiques de forme, couleur et texture.

Objectif 2 – Détection de l'état de santé – La plante est-elle malade ?

Mettre en place un classifieur binaire pour distinguer les feuilles saines de celles présentant des symptômes de maladie.

Objectif 3 – Identification du type de maladie – Quelle est la maladie de la plante ?

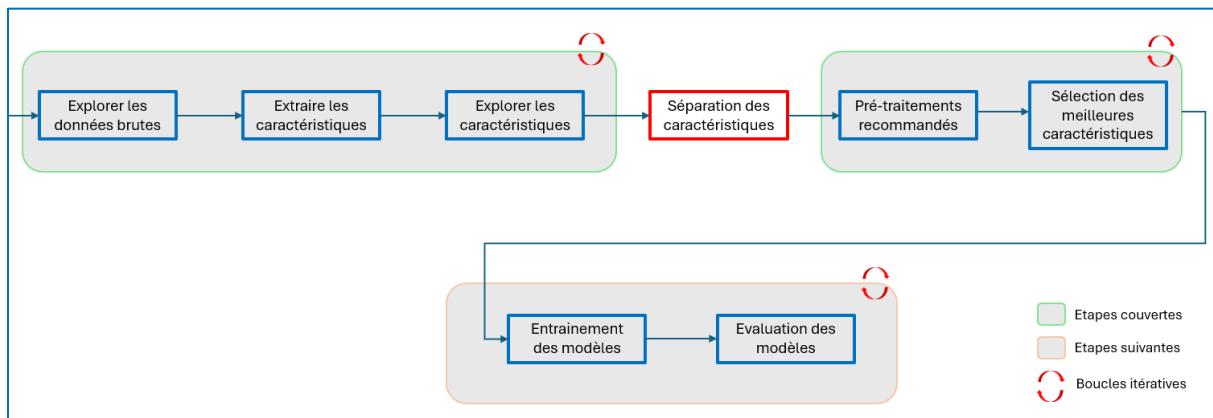
Pour les feuilles diagnostiquées « malade », développer un modèle (multi-classe) déterminant précisément la maladie (tavelure, mildiou, rouille, etc.)

Notre projet se situe donc dans un contexte d'apprentissage supervisé. Nos variables cibles sont : le nom de la plante, l'état de santé et le nom de la maladie. Les variables explicatives seront les caractéristiques des feuilles (forme, couleur, contour...).

2. Compréhension et manipulation des données

Les étapes d'exploration des données et de traitement sont les plus déterminantes pour l'élaboration des modèles. Elles s'effectuent de manière itérative au fur et à mesure des résultats des observations.

Dans la suite du document, les étapes effectuées sont présentées dans l'ordre présenté ci-dessous. Le dataset considéré est PlantVillage/segmented.



Bloc1 – Explorer les données brutes :

- Exploration des images brutes
- Analyse exploratoire sur l'ensemble du dataset
- Redimensionnement uniforme des images
- Extraction des labels (nom_plante, nom_maladie, Est_Saine)
- Vérification et suppression des doublons

Bloc2 – Extraire les caractéristiques

- Extraction des caractéristiques par image

Bloc3 – Explorer les caractéristiques

- Exploration statistique des features extraits (analyse, visualisation)
- Analyse des outliers
- Ré-échantillonnage (sur le train uniquement)

Bloc4 – Séparation des caractéristiques

- Split train/test/ valid

Bloc5 - Prétraitements recommandés

- Data augmentation (sur le train uniquement)
- Standardisation / normalisation des features

Bloc6 - Sélection des meilleures caractéristiques

A partir de cette base d'activités, le premier "run" nous permet de nous roder :

- sur l'architecture du code et sa gestion ainsi que les outils à disposition,
- la répartition du travail entre nous,
- la compréhension de nos données,
- la structuration en 3 objectifs et leur gestion,
- d'identifier tout un ensemble de points d'attention à traiter

Dans l'étape suivante, on fera plusieurs runs sur ce premier set d'activités mais plus rapidement car on sera rodé.

2.1. Exploration des images brutes

2.1.1. Exploration des données brutes dataset **PlantVillage complet**

Le dataset PlantVillage est structuré en 3 dossiers : color, grayscale et segmented. Chaque dossier comporte 54306 images, réparties dans 38 sous-dossiers (les 38 classes). Le nom des sous-dossiers correspond à : “**Nom de la plante_Nom du pathogène**” (dans la suite du document, nous parlerons de maladie par abus de langage) ou “**Nom de la plante_Healthy**”. Il y a 14 espèces de plantes.

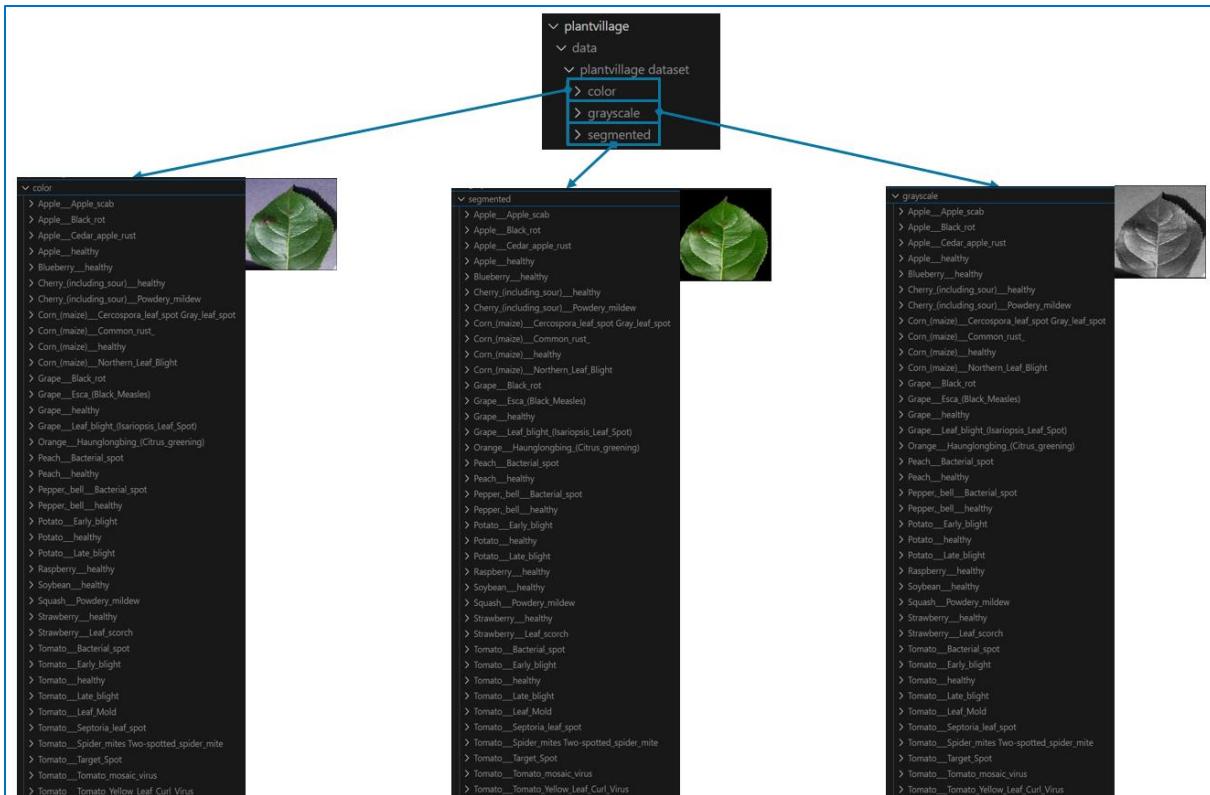


Figure 3 : Structure du dossier PlantVillage

Quelques analyses statistiques fournissent un aperçu du contenu de ce dataset. L'annexe 11.3 présente plus de détails.

Distribution du nombre d'images par espèce et modalité :



Figure 4 : Distribution du nombre d'images par espèce et modalité

L'histogramme révèle que la plupart des espèces disposent de 0 à 2 000 images, avec un pic autour de 1 000 images. Quelques-unes se distinguent par un volume beaucoup plus élevé, frôlant les 5 000 images. Seule la modalité « segmented » est affichée (voir annexe pour les 2 autres modalités). Cette forte variabilité dans le nombre d'images par espèce risque de déséquilibrer l'entraînement des modèles de classification. Il sera donc crucial d'envisager des techniques de rééchantillonnage et de vérifier si les autres modalités offrent une couverture plus uniforme des espèces.

Comparaison du nombre d'images par modalité et espèce :

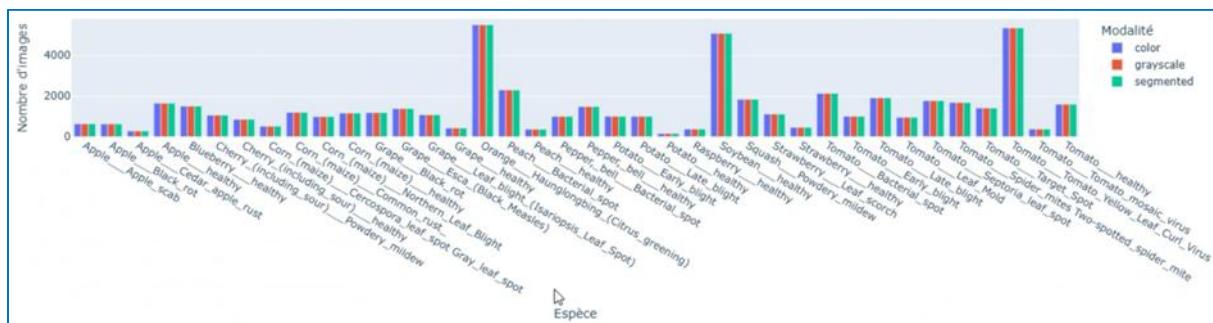


Figure 5 : Comparaison du nombre d'images par modalité et espèce

Le nombre d'images par espèce varie fortement, de quelques dizaines à plus de 4 000. Pour chaque espèce, les volumes sont quasiment identiques en « color », « grayscale » et « segmented », montrant un jeu de données équilibré au niveau des modalités. Les maladies de la tomate se distinguent par un nombre particulièrement élevé d'images ($> 4\,000$). À l'inverse, certaines espèces sont faiblement représentées (ex : Raspberry). Cela risque de biaiser l'apprentissage des modèles sur ces classes. Il faudra utiliser des techniques d'augmentation ou de suréchantillonnage pour les espèces rares.

Nombre d'images par extension :

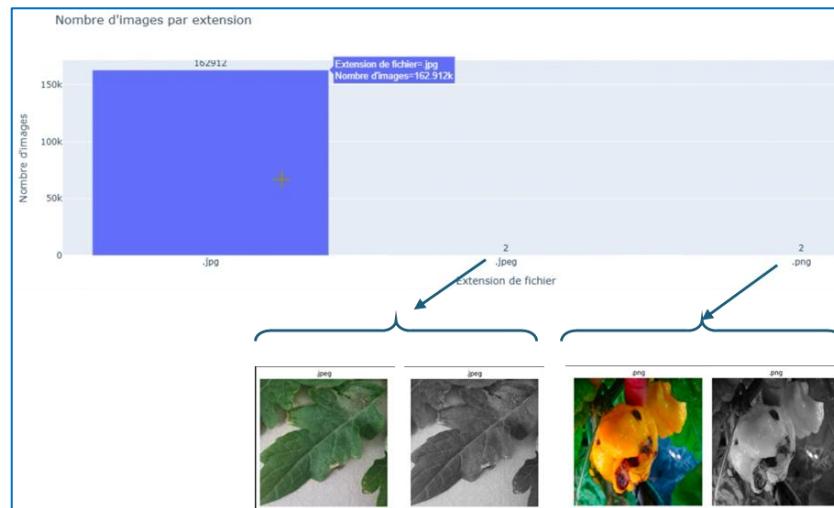


Figure 6 : Nombre d'images par extension

L'extension .jpg a un nombre très élevé d'images, soit 162,912 images. Cela indique que la majorité des images dans ce dataset sont au format JPEG. Les extensions .jpeg et .png ont chacune seulement 2 images. Il faudra définir le traitement à appliquer à ces 4 images.

Largeur vs Hauteur des images selon l'extension :

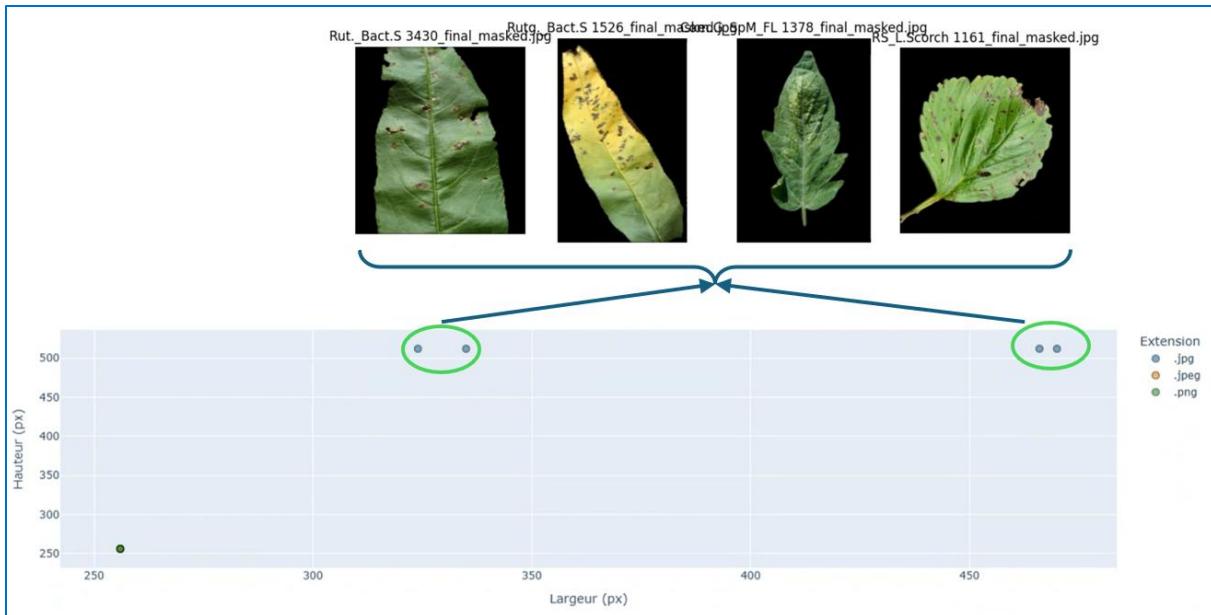


Figure 7 : Dimensions (largeur et hauteur) des images

La quasi-totalité des images sont au format .jpg, les extensions .jpeg et .png étant très marginales. Les dimensions de ces 4 images se regroupent majoritairement autour de 500×500 px. Quelques points à environ 250×300 px témoignent d'une légère variation dans les tailles. Se concentrer sur le format .jpg simplifiera l'analyse dans un premier temps, mais il faudra regarder de plus près les rares .jpeg et .png.

2.1.2. Exploration des données brutes dataset PlantVillage/Segmented

Ce chapitre présente des observations visuelles et des statistiques du dataset PlantVillage/Segmented ainsi que les traitements effectués image par image. Ceci afin de disposer d'un dataset « propre » pour la phase d'extraction des caractéristiques.

Le dataset PlantVillage/Segmented comporte 54306 images, réparties dans 38 sous-dossiers (les 38 classes). Le nom des sous-dossiers correspond à : "Nom de la plante_Nom du pathogène" (dans la suite du document, nous parlerons de maladie par abus de langage) ou "Nom de la plante_Healthy". Il y a 14 espèces de plantes.

Ci-dessous un extrait des images du dataset Plantvillage/segmented pour les 14 types de plantes :

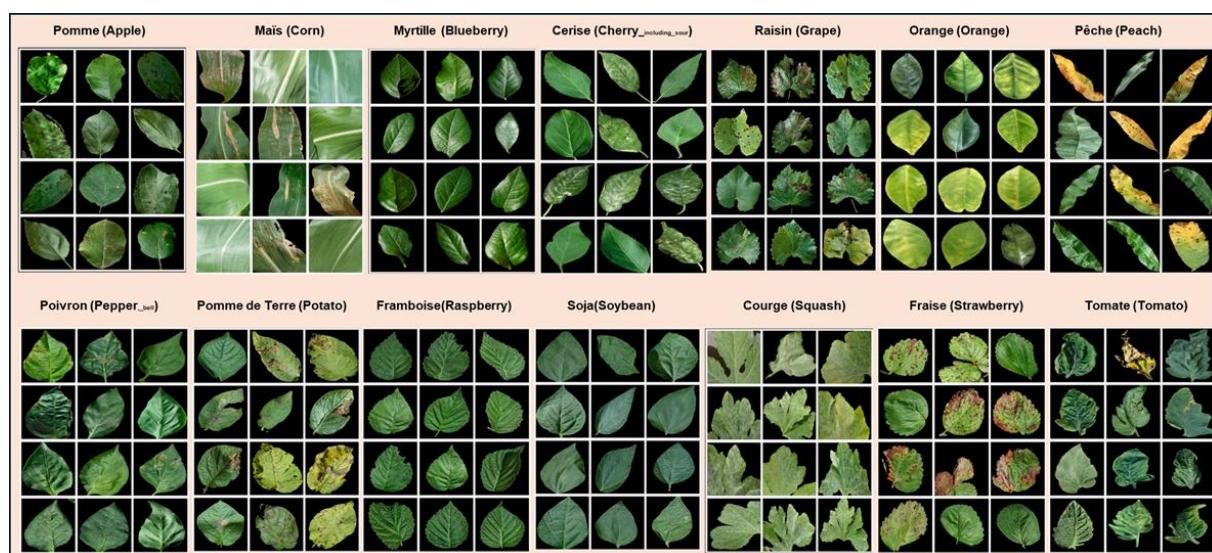


Figure 8 : extrait des images du dataset Plantvillage/segmented pour les 14 types de plantes

Le tableau 2 montre la répartition des 38 classes de feuilles saines et malades.

Observations : Il met en évidence une forte hétérogénéité du dataset : certaines espèces, comme la tomate ou la pomme, sont représentées par de nombreuses maladies, tandis que d'autres, comme le blueberry ou le Soybean, ne sont présentes qu'à l'état sain. Cette distribution très déséquilibrée expose le projet à des risques de biais d'apprentissage et de mauvaise détection des classes minoritaires ou absentes, ce qui impacte directement la robustesse des modèles sur nos trois objectifs : identification de la plante, détection de la maladie, et identification du type de maladie.

Actions à envisager :

- Rééquilibrage des classes : data augmentation, oversampling pour les maladies rares, sous-échantillonnage des classes sur-représentées.
- Métriques adaptées : f1-score par classe, recall, confusion matrix plutôt qu'accuracy brute.

- Stratégies multi-tâches :

- Peut-être séparer la détection “plante saine/malade” des espèces où la distinction est possible,
- Adapter le modèle à la structure spécifique de chaque espèce.

Dossier plantes saines	Dossier plantes malades
Apple__healthy	Apple__Apple_scab Apple__Black_rot Apple__Cedar_apple_rust
Blueberry__healthy	
Cherry_(including_sour)___healthy	Cherry_(including_sour)___Powdery_mildew
Corn_(maize)___healthy	Corn_(maize)___Cercospora_leaf_spot_Gray_leaf_spot Corn_(maize)___Common_rust_ Corn_(maize)___Northern_Leaf_Blight
Grape__healthy	Grape__Black_rot Grape__Esca_(Black_Measles) Grape__Leaf_blight_(Isariopsis_Leaf_Spot)
	Orange__Haunglongbing_(Citrus_greening)
Peach__healthy	Peach__Bacterial_spot
Pepper,_bell__healthy	Pepper,_bell__Bacterial_spot
Potato__healthy	Potato__Early_blight Potato__Late_blight
Raspberry__healthy	
Soybean__healthy	
	Squash__Powdery_mildew
Strawberry__healthy	Strawberry__Leaf_scorch
Tomato__healthy	Tomato__Bacterial_spot Tomato__Early_blight Tomato__Late_blight Tomato__Leaf_Mold Tomato__Septoria_leaf_spot Tomato__Spider_mites_Two-spotted_spider_mite Tomato__Target_Spot Tomato__Tomato_Yellow_Leaf_Curl_Virus Tomato__Tomato_mosaic_virus

Tableau 2 : répartition des 38 classes de feuilles saines et malades

Taille du dossier segmented qui comporte toutes les images est : 593 Mo

Suppression des images inexploitables

Nous avons identifié 18 images quasi noires (dont une complètement noire). A l'affichage de ces images, nous avons décidé de les supprimer. Cette suppression étant tracée, si besoin nous pourrons toujours revenir sur cette décision si nécessaire.

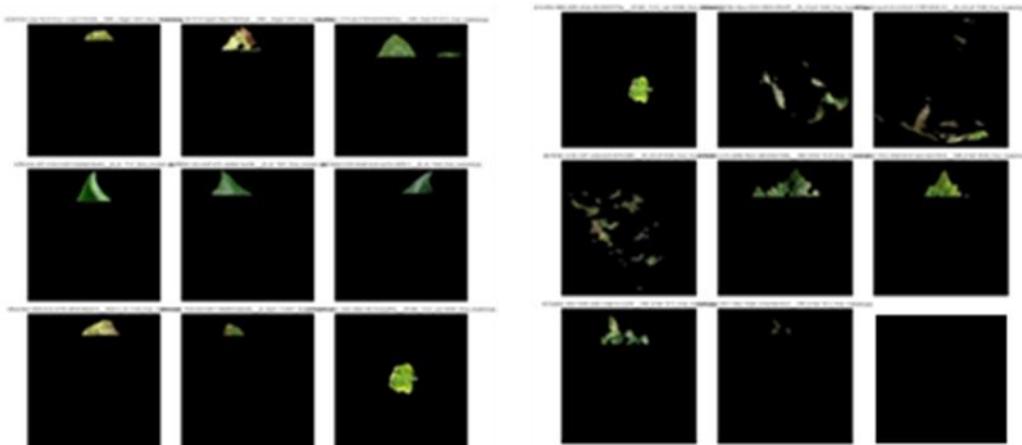


Figure 9 : Affichage des images inexploitables

Détection de doublons d'image et suppression

Les deux histogrammes (Images dupliquées par espèce et Images dupliquées par maladie) ci-dessous présentent les doublons d'image contenus dans le dataset. Il y a 21 doublons d'image. Nous avons décidé de les supprimer.

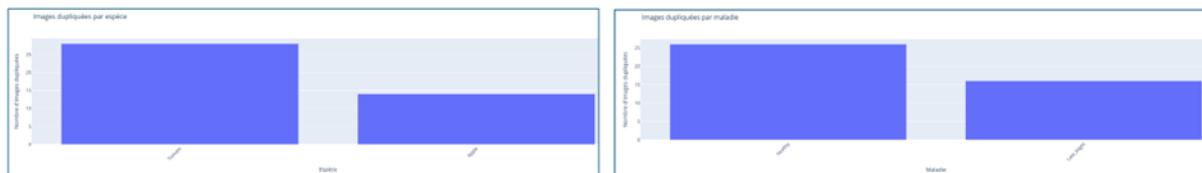


Figure 10 : Images dupliquées

Le nombre total d'images passe à 54267.

Analyse statistique du dataset Segmented

L'analyse de ces histogrammes : observations, risques et conclusions pour chacun des 3 objectifs retenus du projet.

Nombre d'images par espèce et distribution des classes :

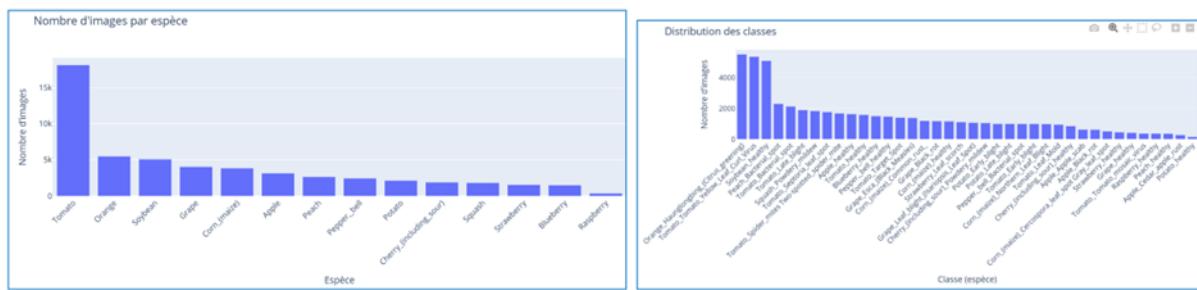


Figure 11 : Nombre d'images par espèce et distribution des classes

Observations :

Le dataset montre un très fort déséquilibre entre espèces : Tomato domine avec près de 18 000 images, tandis que Raspberry n'en compte qu'environ 300. Orange et Grape figurent aussi parmi les plus abondantes, alors que Blueberry ou Peach restent peu représentées.

Au sein de certaines espèces (Tomato, Orange, Grape...), les images malades sont majoritaires, alors que pour d'autres (Pepper_bell, Potato, Squash...) les exemples sains l'emportent. Les maladies courantes comme Orange_Huanglongbing ou Tomato Yellow Leaf Curl Virus totalisent chacune près de 5 000 images.

En revanche, de nombreuses maladies rares forment une longue traîne avec moins de 300–500 images.

Cet équilibre inégal entre espèces et classes de santé justifie l'usage de rééchantillonnage et d'augmentation de données pour stabiliser l'entraînement des modèles.

Risques :

Le déséquilibre massif vers Tomato ($\approx 18\,000$ images) expose le modèle à un sur-apprentissage sur cette espèce, au détriment des espèces rares comme Raspberry. Les classes sous-représentées manquent de diversité d'exemples pour capturer leurs signatures visuelles spécifiques. Un modèle binaire global, ignorant l'espèce, sera biaisé vers la majorité d'images de chaque espèce, faussant les décisions. Les déséquilibres intra-espèce (prévalence inégale de maladies selon l'espèce) empêchent de fixer un seuil unique de classification fiable.

La domination de certaines plantes entraîne une tendance du modèle à prédire systématiquement la classe de la plus représentée, au risque de négliger les catégories minoritaires.

Conclusions :

On priviliege un split stratifié sur (nom_plante, Est_Saine) pour préserver les proportions d'espèces et de classes saines/malades. Un rééchantillonnage ciblé (sous-sampling de Tomato, oversampling ou data augmentation des espèces et classes sous-représentées) compensera les déséquilibres. L'usage de class weights équilibrés dans le modèle renforcera la prise en compte des catégories rares. Les espèces ou maladies très peu fréquentes (< seuil, ex. 300 images) peuvent être regroupées sous une étiquette "Other" pour limiter le bruit. Enfin,

une approche hiérarchique ou multi-étapes (classification binaire par espèce ou intégration de nom_plante en entrée) affinera la performance globale.

Répartition des images saines vs malades par espèce, Distribution des classes saines, Distribution des classes malades

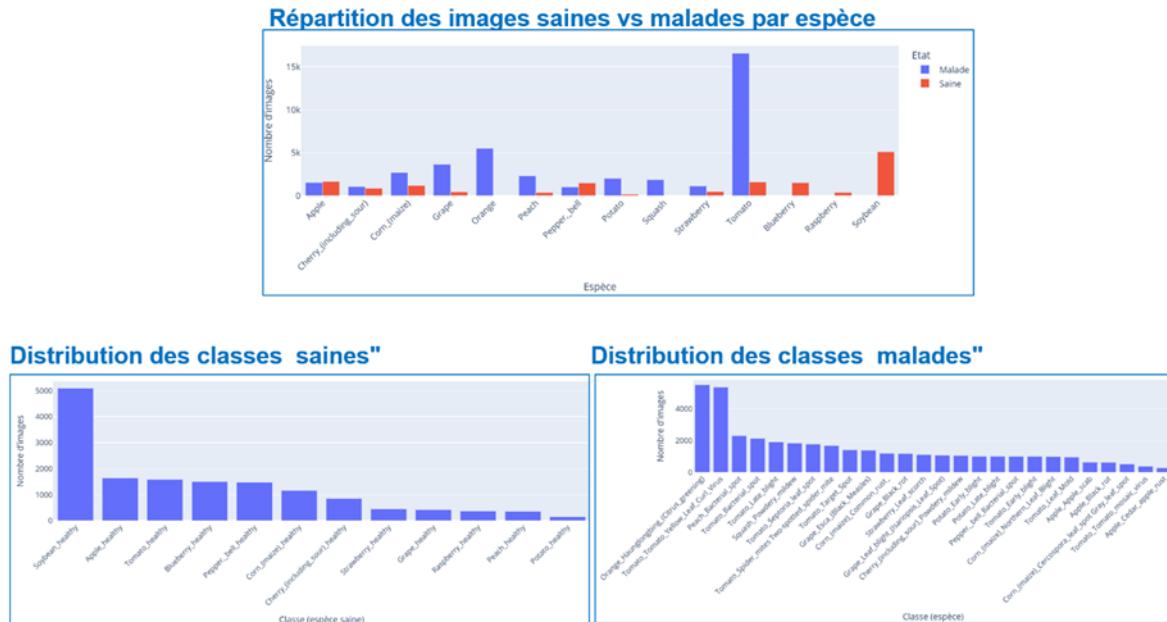


Figure 12 : Répartition des images saines vs malades par espèce, Distribution des classes saines, Distribution des classes malades

Observations :

Certaines espèces (Tomato, Orange, Grape, Corn, Peach...) présentent une majorité d'images malades (bleu > rouge), tandis que d'autres (Pepper_bell, Potato, Raspberry, Squash, Strawberry...) sont majoritairement saines. Parmi les saines, Soybean_healthy domine avec ~ 5 100 images, suivie d'Apple_healthy, Tomato_healthy et Blueberry_healthy (~ 1 500–1 600), alors que Potato_healthy est très sous-représentée (~ 120). Du côté des maladies, Orange_Huanglongbing et Tomato_Yellow_Leaf_Curl_Virus culminent à ~ 5 000 images, puis un palier de pathologies intermédiaires à 1 800–2 300. La majorité des autres maladies se situe dans une longue traîne avec moins de 500, voire moins de 100 images.

Risques :

Intégrer l'état de santé dans un classifieur d'espèce risquerait de le biaiser, car chaque plante présente un déséquilibre sain/malade propre (Tomato presque tout malade, Pepper presque tout sain). Entraîner un modèle binaire "sain vs malade" sans distinguer les espèces conduira à des seuils inadaptés face aux distributions extrêmes (trop de "sains" pour Soybean, pas assez pour Potato ; inverse pour Tomato vs Pepper). Les maladies globales, dominées par quelques pathologies très fréquentes, feront qu'un classifieur multiclasse brut privilégiera systématiquement les 3–5 classes majeures et ignorera les rares. Les espèces et maladies

sous-représentées manquent de données pour apprendre une signature visuelle robuste, augmentant les risques de faux positifs et négatifs.

Conclusions :

Le modèle d'espèce doit être entraîné sans la feature Est_Saine, en stratifiant le split sur nom_plante (ou conjointement sur nom_plante + Est_Saine pour chaque sous-tâche) afin de préserver les proportions réelles. On appliquera des class weights ou un over-/undersampling ciblé (p. ex. augmenter Potato_healthy, sous-échantillonner Tomato_malade) pour corriger les biais d'état majoritaire par espèce. Pour la classification des maladies, seules celles dépassant un seuil minimal d'images (ex. 300) seraient conservées, les autres regroupées en "Other_disease ». Des augmentations de données spécifiques renforceront les classes minoritaires au sein de chaque espèce ou maladie retenue. Enfin, une approche hiérarchique (classification binaire par espèce puis multiconfiance maladie) ou l'ajout de nom_plante en entrée assure une meilleure robustesse face aux déséquilibres.

Redimensionnement uniforme des images

Sur les 54267 images, la taille la plus fréquente est 256x 256 et il n'y a que 4 images de tailles différentes : [((256, 256), 54263), ((466, 512), 1), ((324, 512), 1), ((335, 512), 1), ((470, 512), 1)]

Ces 4 images ont été redimensionnées à 256 x 256.

Extraction des labels

Pour couvrir les 3 objectifs du projet, nous avons identifié les cibles suivantes :

- nom_plant
- est_Saine
- nom_maladie

2.2. Extraction des caractéristiques

En Machine Learning, les algorithmes de classification ne travaillent pas directement sur les images brutes composées de pixels. Il est donc nécessaire de convertir chaque image en un vecteur de données numériques, en extrayant des caractéristiques visuelles (ou features) pertinentes. Ces caractéristiques doivent résumer l'information essentielle contenue dans l'image, afin de permettre aux modèles d'apprentissage de distinguer efficacement les différentes classes, en l'occurrence les espèces de plantes. Elles peuvent représenter divers aspects comme la forme, la couleur, la texture ou la structure spatiale de l'image. Le choix et la qualité de ces descripteurs ont un impact direct sur la performance finale du modèle.

Pour chaque image de plante, nous avons extrait un ensemble de descripteurs manuels visant à capturer différents aspects visuels discriminants. Ces caractéristiques sont regroupées en plusieurs catégories :

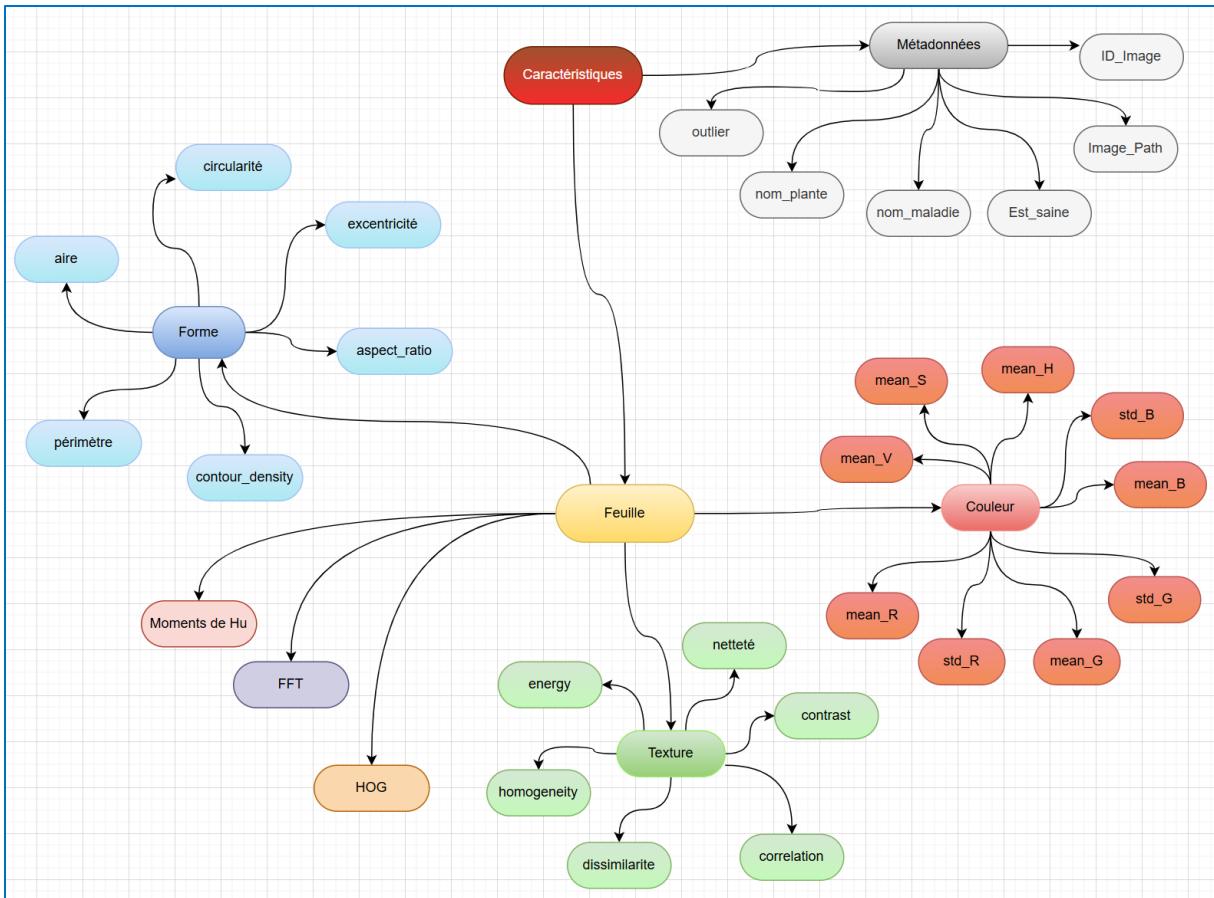


Figure 13 : Caractéristiques des feuilles de plantes

- **Caractéristiques morphologiques** : superficie (aire), périmètre, circularité, excentricité, rapport d'aspect (aspect ratio) et densité de contours. Ces indicateurs décrivent la forme globale des objets présents sur l'image.
- **Caractéristiques colorimétriques** : moyennes et écarts-types des canaux RGB (mean/std_R, G, B), ainsi que les moyennes des composantes HSV (mean_H, S, V), permettant de représenter les couleurs dominantes et leur variation.
- **Caractéristiques de texture** : netteté, contraste, energy, homogeneity, dissimilarity, correlation, calculées à partir de matrices de co-occurrence, décrivent les variations locales d'intensité dans l'image.
- **Descripteurs invariants** : les moments de Hu (hu_1 à hu_7) permettent de capturer la forme de manière invariante à la rotation, à la translation et au changement d'échelle.

- **Descripteurs fréquentiels** : les coefficients extraits de la transformée de Fourier (fft_energy, fft_entropy, low/high frequency power) communiquent une information sur la répartition spectrale des détails dans l'image.
- **Descripteurs de gradient** : les descripteurs HOG (moyenne, écart-type, entropie) résument les orientations dominantes des gradients, utiles pour capturer les structures visuelles.

Ces descripteurs sont concaténés pour former un vecteur unique par image, servant ensuite d'entrée aux algorithmes de classification. L'objectif est d'évaluer leur capacité à discriminer les différentes espèces de plantes présentes dans notre jeu de données.

En complément de cette extraction, nous avons recensé l'ensemble des caractéristiques générées dans un tableau synthétique. Ce tableau précise pour chaque descripteur : sa source ou librairie d'origine, ainsi que sa fonction ou utilité dans le cadre de notre projet. Il permet ainsi de mieux comprendre le rôle de chaque variable dans la chaîne de traitement, qu'il s'agisse de mesurer une propriété géométrique, colorimétrique ou texturale, ou bien d'assurer un filtrage ou un encodage préalable.

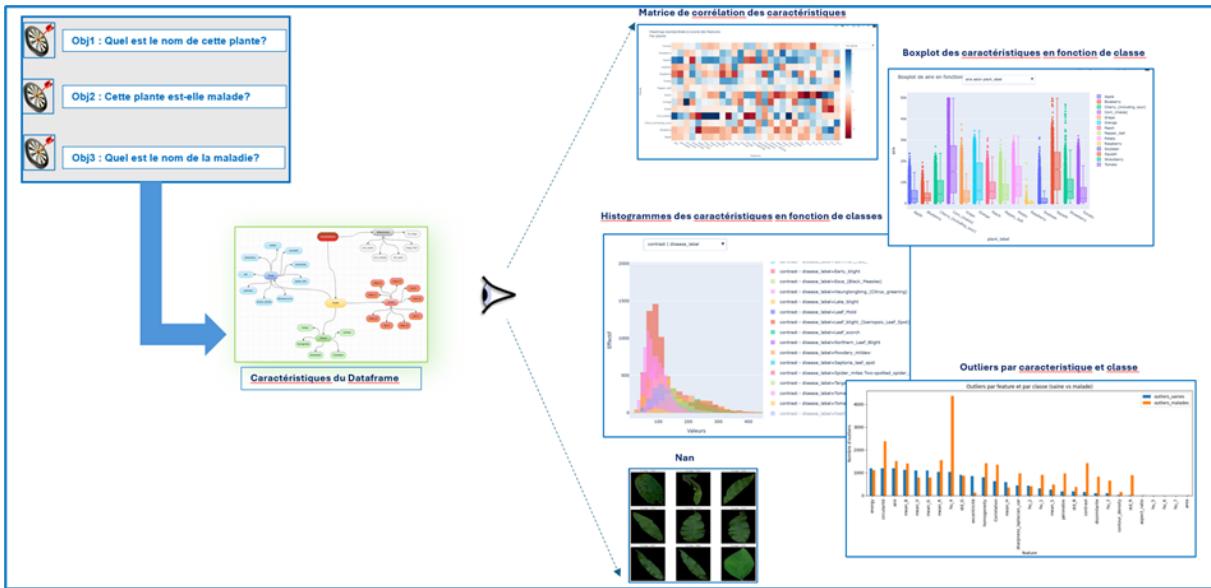
Certaines variables sont directement issues du traitement d'image (par exemple via OpenCV, NumPy, skimage.feature ou la transformée de Fourier), tandis que d'autres jouent un rôle de support (identifiant de la plante, label, cible, ou métadonnée de structure). Cette organisation facilite l'analyse, la traçabilité, ainsi que la future sélection des features les plus discriminantes pour la phase de classification.

Ce tableau complet est disponible en annexe 11.2 pour consultation détaillée. Le fichier Excel sera livré avec ce rendu1.

3. Explorer les caractéristiques

L'exploration des caractéristiques extraites des images repose sur des Visualisations et des Statistiques en fonction des 3 objectifs. Des versions HTML sont fournies avec ce rendu_1.

L'objectif de ce chapitre est de montrer le potentiel des graphiques pour nous supporter dans l'analyse des résultats de performance et boucles d'itération sur des points plus ciblés. Dans la suite, nous avons piochés quelques exemples.



3.1. Gestion des données manquantes

Les colonnes de forme (dimensions, aire, périmètre, circularité, excentricité, aspect_ratio) ne sont renseignées qu'à hauteur de 54 258 lignes (au lieu des 54 267 images totales). Pour 9 images, la fonction extract_shape_features a renvoyé None (et Pandas en a donc fait des NaN).

Cela signifie qu'aucun contour n'est détecté (seuil mal calé) : pour ces images : [12835, 13932, 15526, 15725, 15731, 16335, 16475, 16862, 42064] :

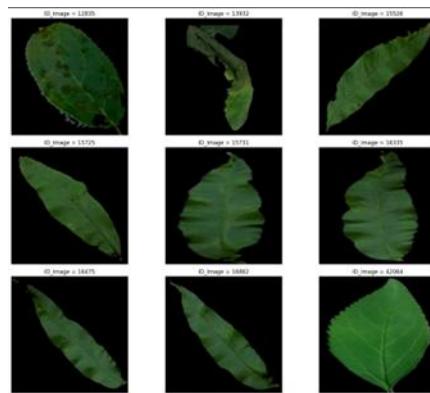


Figure 14 : données manquantes

Dans un premier temps, nous les avons supprimées. Cependant, dans la prochaine étape, il faudra vérifier pourquoi ces images n'ont pas de contour.

3.2. Identification des outliers

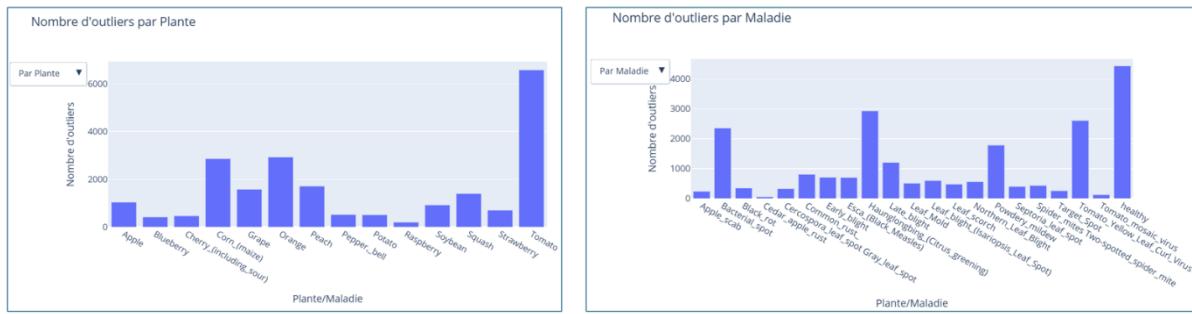


Figure 15 : Identification des outliers par plante – identification des outliers par maladie

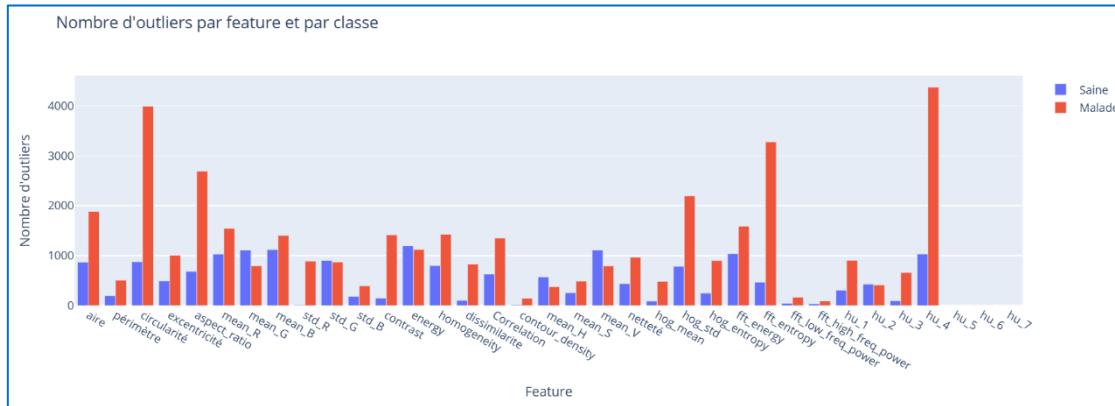


Figure 16 : Identification des outliers par feature et par classe

Détection des outliers via la méthode IQR pour chaque variable.

Nombre de lignes avec au moins un outlier 21900.

Observations :

Le taux d'outliers est élevé : 40,34%. Le graphique présente la répartition du nombre d'outliers par feature, séparée selon les classes "saine" et "malade". On constate que, pour la majorité des variables, la classe "malade" affiche significativement plus d'outliers que la classe "saine". Certaines features, comme périmètre, fft_energy ou hu_4, se distinguent particulièrement avec un nombre élevé d'outliers chez les malades. Quelques variables montrent cependant une distribution plus équilibrée ou peu d'outliers dans les deux classes.

Risques :

La surreprésentation d'outliers chez les malades confirme l'hétérogénéité importante au sein de cette classe, mais cela peut aussi révéler des problèmes potentiels de qualité de données ou de mesures extrêmes spécifiques à certains cas. Si ces outliers sont dus à des erreurs de saisie ou à des artefacts, ils risquent de biaiser les analyses statistiques et les modèles prédictifs, en faussant la détection des vraies différences entre classes. Enfin, la présence d'un nombre très faible ou nul d'outliers sur certaines features pourrait signaler des variables peu discriminantes ou mal calibrées. Les features avec beaucoup d'outliers risquent de perturber certains algorithmes sensibles aux valeurs extrêmes. Supprimer les outliers, sans discernement, risque d'enlever des cas légitimes et importants. Possible présence de valeurs aberrantes parmi les outliers détectés,

Conclusions :

- Ce graphique met en évidence l'importance de traiter et d'analyser spécifiquement les outliers, notamment pour la classe "malade" où ils sont très présents. Avant toute modélisation, il sera essentiel d'investiguer l'origine de ces valeurs extrêmes, de décider de leur prise en compte (suppression, correction, transformation) et de vérifier leur impact sur la robustesse des résultats. Une gestion rigoureuse des outliers contribuera à améliorer la fiabilité des analyses et la performance des modèles de classification ultérieurs. Analyse approfondie des outliers : Inspecter visuellement et statistiquement les valeurs extrêmes pour distinguer les vraies valeurs aberrantes (bruit, erreur) des observations atypiques mais valides.
- Traitement adapté selon le contexte :
 - Adapter le prétraitement (suppression, transformation) ou choisir des modèles robustes aux outliers.
 - Tester l'impact du traitement des outliers sur la performance des modèles (avec/sans, transformation...).

- Prioriser les features discriminantes : Les variables qui montrent beaucoup d'outliers chez les malades pourraient être de bons marqueurs pour la classification santé/maladie.

Dans le chapitre sur les Représentations visuelles, les boxplots sont également des représentations pertinentes pour explorer la nature des outliers.

3.3. Distribution des caractéristiques

Un jeu d'histogrammes interactifs permet l'exploration de la distribution des caractéristiques suivant l'objectif auquel le modèle devra répondre.

Objectif1 : Quelle est cette plante ?



Figure 17 : Distribution des caractéristiques pour objectif 1 "Quelle est cette plante?"

Observations :

Variabilité importante entre espèces : Les distributions de certaines features (aire, couleur, texture...) varient fortement d'une plante à l'autre, ce qui suggère que plusieurs caractéristiques sont potentiellement très discriminantes pour l'identification de la plante.

Features très redondantes ou corrélées : Certaines familles de variables (ex : canaux couleurs, moments de Hu) présentent des corrélations élevées entre elles, ce qui peut indiquer une certaine redondance à traiter lors de la sélection des variables.

Présence d'outliers marquée pour quelques caractéristiques : Quelques features montrent beaucoup de valeurs extrêmes pour certaines plantes, ce qui peut refléter soit une vraie variabilité biologique, soit des problèmes de mesure ou d'extraction.

Caractéristiques quasi constantes ou peu informatives pour certaines classes : Certaines variables varient peu selon l'espèce, ce qui les rend peu utiles pour la classification de la plante (par exemple, une texture ou une couleur presque identique dans toutes les classes).

Risques :

Un fort déséquilibre entre espèces peut biaiser le modèle, qui aura tendance à privilégier les classes majoritaires lors de la prédiction. Les espèces sous-représentées risquent d'être mal reconnues, augmentant le taux d'erreur pour ces classes. Les performances globales du modèle peuvent être trompeuses et masquer des faiblesses sur les minorités. Ce déséquilibre complique la généralisation et la robustesse de la solution sur l'ensemble des plantes du jeu de données.

Conclusions :

On observe que la variabilité de certaines caractéristiques entre espèces offre de bons leviers pour la classification. Toutefois, la redondance, la présence d'outliers et des variables peu informatives devront être prises en compte pour optimiser la robustesse et la performance du modèle pour l'objectif "identifier la plante".

Objectif2 : Cette plante est-elle saine ?

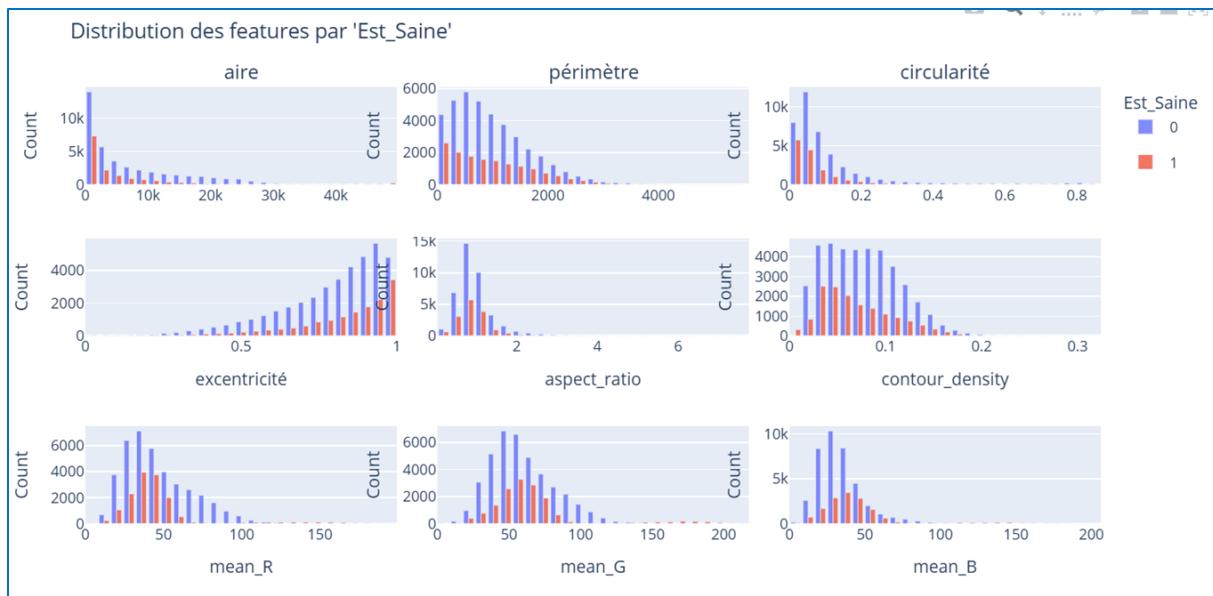


Figure 18 : Distribution des caractéristiques pour objectif 2 1/3

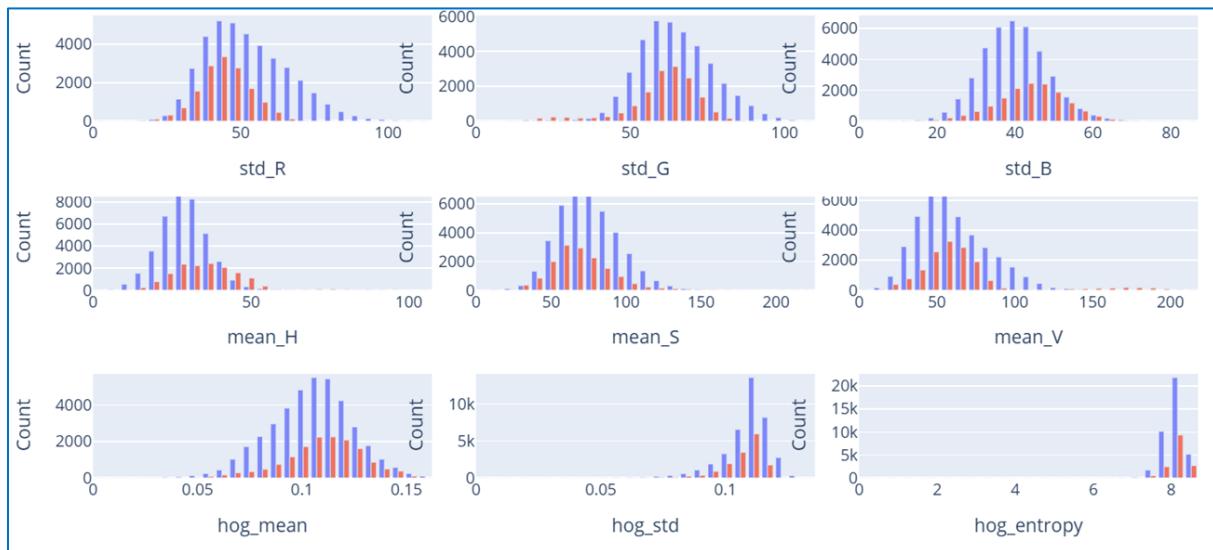


Figure 19 : Distribution des caractéristiques pour objectif 2 2/3

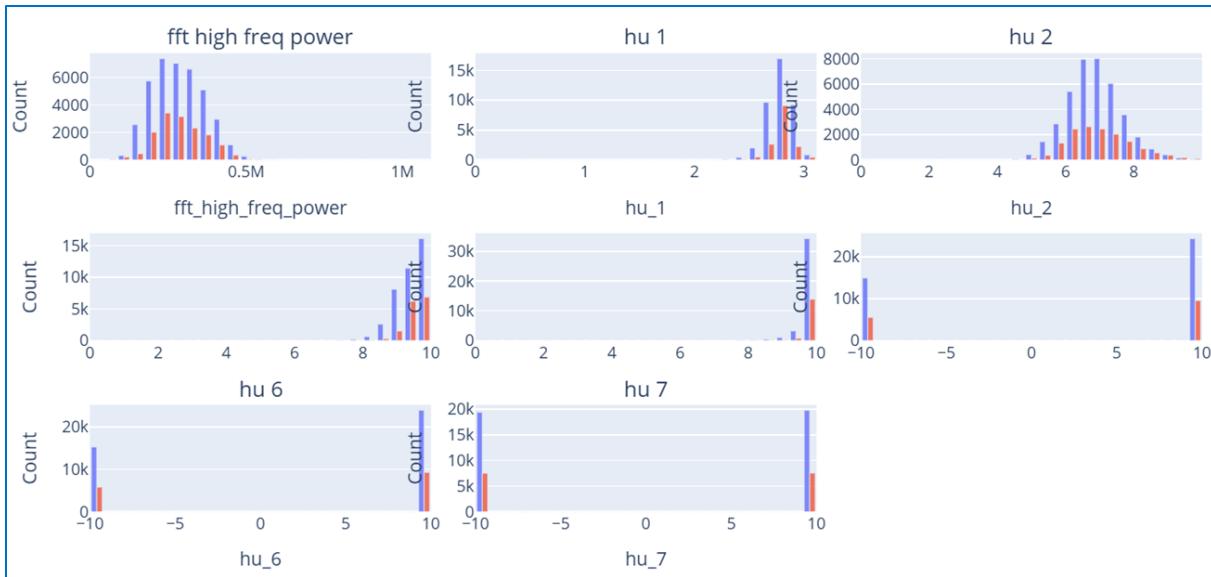


Figure 20 : Distribution des caractéristiques pour objectif 2 3/3

Observations :

Observations

Les variables morphologiques (aire, périmètre, circularité) et colorimétriques (mean R, G, B) montrent des différences nettes entre les classes saine (bleu) et malade (rouge). Les plantes malades présentent souvent une aire et une circularité plus faibles, avec des valeurs de teinte (H) et saturation (S) distinctes. Les mesures de texture (hog, fft, hu) confirment cette séparation.

Risques :

Les plantes avec ces caractéristiques tendent vers une altération de structure et de couleur indicative de stress ou infection. Une mauvaise identification pourrait survenir si certaines valeurs se chevauchent, surtout dans les variables de texture. Les zones intermédiaires peuvent nécessiter des tests supplémentaires (pH, humidité).

Conclusions :

Le profil global du graphique correspond davantage aux distributions observées pour les plantes malades. Les écarts sur la couleur et la morphologie suggèrent un état non sain. Cette plante est donc probablement malade.

Objectif3 : Quelle est cette maladie ?

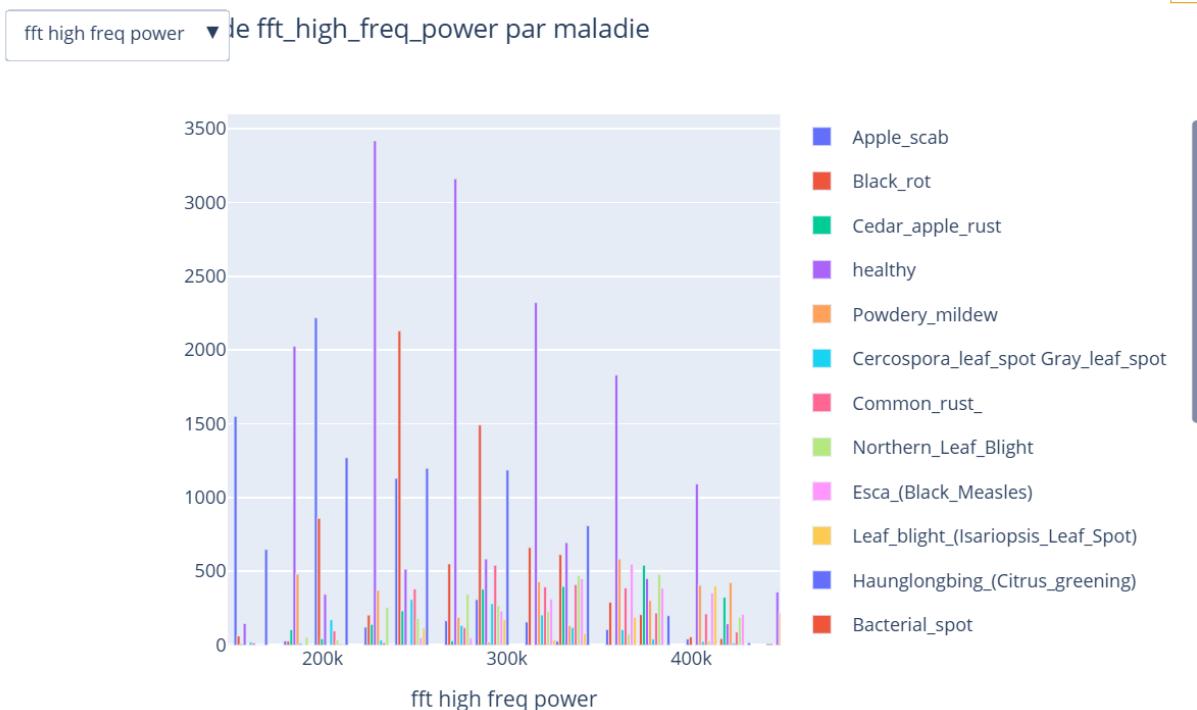


Figure 21 : Distribution des caractéristiques pour objectif 3 "Quelle est la maladie ?"

Observations

Les graphiques présentent la répartition des images par maladie, toutes plantes confondues. On constate que certaines maladies (ex : Apple_scab, Powdery_mildew, Leaf_blight_, Haunglongbing_) sont beaucoup plus représentées dans le jeu de données, tandis que d'autres ne comptent que quelques exemples. Ce déséquilibre entre maladies est très marqué, certaines pathologies rares n'apparaissant que sur une poignée d'images.

Risques

Un fort déséquilibre entre maladies expose le modèle à un risque de biais : il apprendra

surtout à reconnaître les maladies majoritaires et aura du mal à prédire correctement les maladies rares, même si elles sont importantes en pratique. Il y a donc un risque élevé de sous-performance sur les maladies peu représentées, ce qui peut fausser l'interprétation des résultats ou masquer l'apparition de pathologies émergentes.

Conclusion

Pour répondre correctement à l'objectif 3 (identifier la maladie), il sera essentiel de corriger ou d'atténuer ce déséquilibre (par exemple via de l'augmentation de données, un sous-échantillonnage, ou des pondérations lors de l'apprentissage). Cela garantira que le modèle accorde une attention équitable à chaque maladie et que la détection des maladies rares soit suffisamment fiable pour un usage terrain.

Représentations visuelles des caractéristiques :

Ces visualisations permettent d'identifier des variables clés qui séparent bien les plantes saines des malades. Elles permettent également de repérer les comportements atypiques (valeurs extrêmes) qui méritent une investigation (plante en transition de santé, mesure erronée, etc.). Aider au choix des variables pour un futur modèle de classification visant à prédire si une plante est malade.

Le nom du dossier contenant les fichiers HTML des Boxplots : Boxplots

- **Boxplot1 et Boxplot2** : ils présentent une grande quantité de données et plusieurs classes, mais la lisibilité est réduite : difficile d'identifier visuellement des différences marquées entre malades et non-malades.
- **Boxplot3** : intéressant car il met en évidence des médianes différentes entre maladies et une variabilité interne bien visible, mais il est légèrement moins clair que Boxplot2 pour distinguer les distributions.
- **Boxplot4** : plus simple et lisible, mais les différences entre classes sont moins marquées, limitant son intérêt pour identifier une plante malade.

- **Boxplot5** : il met en évidence de fortes dispersions avec des valeurs extrêmes (outliers), mais sans séparation claire par maladie.
- **Boxplot6** : comparable au Boxplot3 en termes de clarté, il offre une bonne visualisation de la dispersion mais moins de contraste entre classes.
- **Boxplot7** : très riche en points et en variabilité, mais les différences entre classes sont moins nettes, rendant l'interprétation plus complexe.
- **Boxplot8** : plus condensé visuellement et facile à lire, mais la séparation des distributions entre maladies y est peu marquée.

Explorons un des boxplots : le **Boxplot2** particulièrement intéressant à commenter pour l'objectif 3 : “*Quelle est cette maladie ?*”.

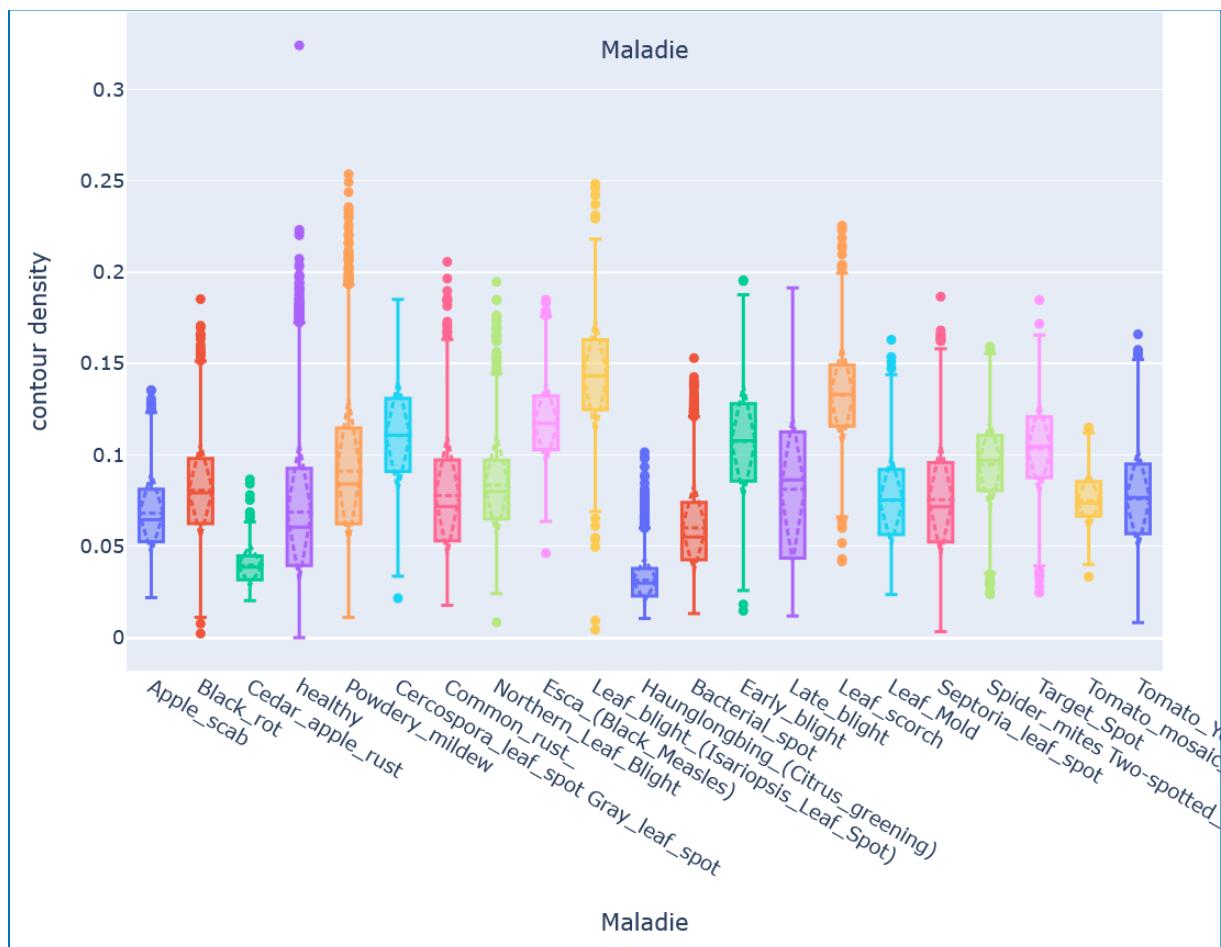


Figure 22 : Boxplots de la densité de contours selon les différentes maladies

Le Boxplot2 met en évidence la variable `contour_density` pour distinguer deux maladies du pommier : `Apple_scab` et `Black_rot`. Ces maladies présentent des symptômes visuellement contrastés sur un même type de feuilles de pommier, limitant les biais liés aux différences d'espèces et facilitant l'interprétation des écarts de médiane et de dispersion. Cette comparaison illustre clairement la capacité des boxplots à révéler des variables discriminantes pour l'objectif 2, qui vise à isoler les caractéristiques clés selon le type de maladie. Un axe d'amélioration consistera à générer des boxplots par plante, afin d'explorer d'autres maladies (ex. `Leaf_blight` ou `Haunglongbing`) tout en conservant la cohérence inter-espèces.

Histogramme présentant la distribution des caractéristiques par type de plantes :

Le nom du fichier HTML : `distributions_features_targets.html`

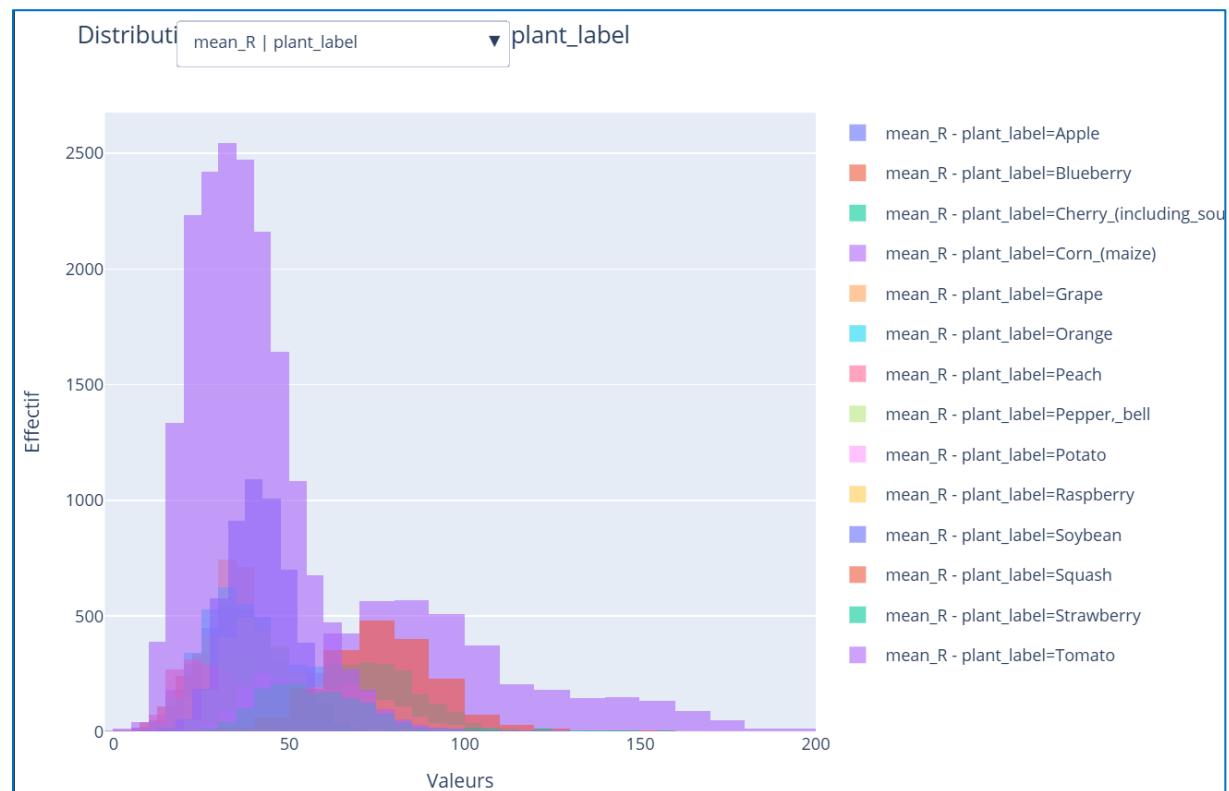


Figure 23 : Histogramme de la distribution des caractéristiques par type de plantes

3.4. Relations entre les caractéristiques

Heatmaps présentant les corrélations entre features:

Le nom du fichier HTML : heatmap_features_grouped.html

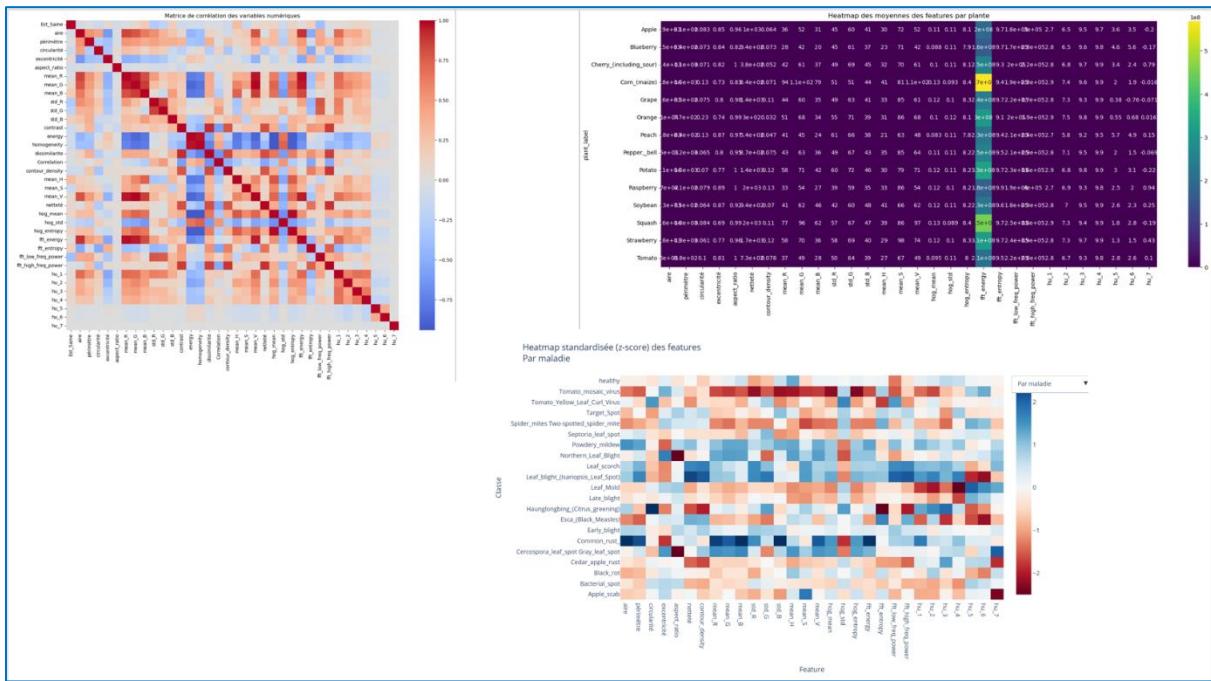


Figure 24 : Heatmaps des corrélations entre features

Observations :

- On observe une grande variabilité des profils z-score des features entre les différentes espèces de plantes.
- Certaines plantes présentent des profils très contrastés (zones rouges/bleues marquées), ce qui indique des caractéristiques spécifiques fortes.
- Aucune plante n'a exactement la même "empreinte" de feature, c'est intéressant pour la classification d'espèces.
- Le contraste est beaucoup plus net : chaque feature est soit nettement au-dessus, soit nettement au-dessous de la moyenne globale.
- La plupart des features discriminent de façon binaire entre sain et malade : fort potentiel pour des modèles de classification binaire.

- Il existe des patterns différenciés entre maladies, avec des features fortement déviées pour certaines pathologies.
- Les profils “healthy” (sains) se distinguent assez bien, avec des scores souvent proches de la moyenne (blancs).
- Certaines maladies affichent des signatures très marquées sur certaines features, ce qui peut aider à leur identification automatique.

Risques :

- Certains groupes de features semblent évoluer ensemble. A priori, il y a de la redondance.
Risque de surapprentissage
- Variabilité intra-classe : Pour certaines plantes ou maladies, la couleur des cases oscille beaucoup, ce qui peut indiquer une hétérogénéité importante dans la classe. Cela pourrait compliquer la classification fine.
- Effet de standardisation : Comme l'échelle est relative (z-score), une forte déviation sur une feature peut être causée soit par une vraie différence, soit par une variance très faible pour cette feature.
- Binarité excessive (statut sain/malade) : Si les différences sont trop franches et systématiques sur toutes les features, risque d'avoir des modèles “trop parfaits” sur le jeu d'entraînement et moins robustes en situation réelle

Conclusions :

- Sélection de features pertinentes : Utiliser l'information des heatmaps pour prioriser les features qui montrent des différences nettes entre classes (plante, maladie, statut) tout en surveillant les redondances.
- Analyse approfondie des profils atypiques : Vérifier pourquoi certaines plantes ou maladies ont des patterns très différents ou très fluctuants, et si cela correspond à des outliers, ou une vraie diversité biologique.
- Affinement des modèles :

- Pour la classification binaire (sain/malade), de nombreuses features sont prometteuses : tester différents algorithmes.
- Pour la classification multi-classes (espèce, maladie), travailler la réduction de dimensionnalité et l'ingénierie de features pour optimiser la séparation entre groupes.

Nous n'avons pas eu le temps d'utiliser l'Analyse en Composantes Principales (PCA). Pour la prochaine itération, il sera pertinent de réaliser une PCA sur l'ensemble des variables quantitatives extraites, après standardisation, afin de visualiser la structure globale des données. Cette analyse permettra d'explorer la capacité des features à séparer les plantes selon leur maladie ou leur état de santé via des scatterplots des deux premières composantes principales. Enfin, il sera intéressant de reproduire cette approche séparément par espèce pour mieux comprendre la distinction entre maladies au sein d'une même plante.

4. Ré-échantillonnage

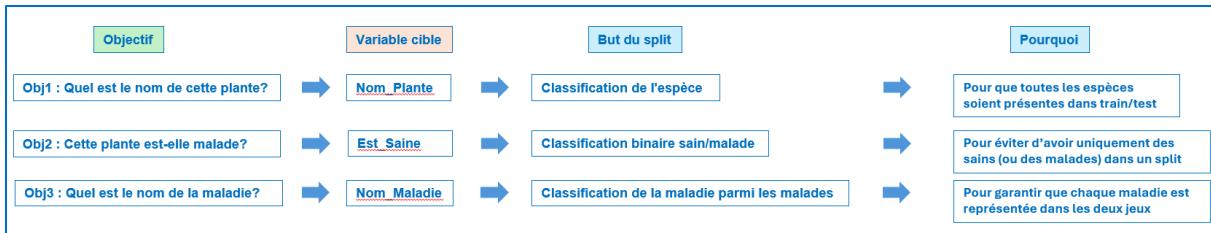
Pour donner suite au constat de déséquilibre de classe, nous avons réalisé un rééchantillonnage avant le split pour équilibrer les classes :

- oversampling ciblé des espèces/maladies minoritaires
- undersampling léger des classes sur-représentées

5. Split train/test (et éventuellement validation)

Pour garantir la qualité de notre modèle, nous avons séparé le dataset en trois parties : un ensemble d'entraînement pour apprendre, un ensemble de validation pour régler et améliorer le modèle, et un ensemble de test pour évaluer ses performances réelles sur des données totalement nouvelles.

Le déséquilibre des classes constaté nous impose par ailleurs des découpages du jeu de données (train, val, test) stratifiés, c'est-à-dire en respectant la proportion des classes cibles pour chaque objectif :



6. Data augmentation (train uniquement)

Afin d'améliorer la robustesse et la généralisation de nos modèles nous avons, réalisé de la data-augmentation, sur train uniquement, après découpage.

Ceci afin de :

- Réduire le déséquilibre (même après ré-échantillonnage certaines classes restent peu nombreuses).
- Enrichir la diversité des cas et forcer le modèle à travailler sur des variantes
- Limiter le sur-apprentissage en exposant le modèle à du bruit pour diminuer sa tendance à sur-apprendre

Nous avons appliqué une combinaison des méthodes suivantes :

- Flip (retournement horizontal/vertical)
- Rotation (ex : -30°, +30°...)
- Crop aléatoire (recadrage)
- Translation (décalage)
- Modification de la luminosité, contraste, bruit
- Zoom in/out
- Transformation couleur (changement de teinte, saturation)
- Gaussian blur, sharpness

7. Standardisation / normalisation des features (fit sur le train, transform sur test)

Lors de notre premier run nous avons relevé un taux d'outliers élevé. Sans traitement de ces outliers certains modèles ne pourront être utilisés.

L'analyse des distributions de certaines variables nous a informés que certaines variables avaient des ordres de grandeur très différents (ex : aire entre 0 et 50000, circularité entre 0 et 1).

Le constat du nombre élevé d'outliers pour certaines variables (hu_4, circularité, aire, mean_R, mean_B, etc.) implique que les méthodes "classiques" de scaling (standardisation, min-max) seront fortement influencées par ces valeurs extrêmes.

La présence d'outliers nous impose donc le choix de RobustScaler.

Afin de garantir que nos modèles ne soient pas biaisés par des valeurs extrêmes ou des ordres de grandeurs différents nous avons :

- Appliqué un RobustScaler fit uniquement sur l'ensemble d'entraînement
- Transformé ensuite les ensembles de validation et de test avec ce même scaler préalablement ajusté.

8. Sélection des meilleures caractéristiques

Cette phase consiste à conserver les caractéristiques les plus pertinentes pour la tâche de classification. On cherche à :

- Réduire la dimension du dataset,
- Éliminer le bruit ou les variables non informatives,
- Améliorer la performance et la vitesse des modèles.

Voici un tableau de synthèse des différentes approches possibles pour analyser l'importance des features.

Méthode	Type	Principe	Classification binaire	Classification multi-classes	Avantages	Inconvénients
Variance Threshold	Filtre	Supprime les features dont la variance est inférieure à un seuil	✓	✓	Très rapide à exécuter	Ne tient pas compte de la cible
Corrélation (Pearson)	Filtre	Retire les features fortement corrélées entre elles	✓	✓	Réduit multicolinéarité	Seulement linéaire
Chi-2	Filtre	Test d'indépendance entre feature catégorielle et cible	✓	Limitée (binaires ou k-classes >2)	Simple et efficace pour variables discrètes	Ne gère pas bien les variables continues
ANOVA F-test	Filtre	Compare moyennes des groupes de la cible	-	✓	Approprié pour variables numériques	Hypothèses normales
Mutual Information	Filtre	Mesure l'information partagée entre feature et cible	✓	✓	Capture relations non linéaires	Plus coûteux qu'un test univarié
Recursive Feature Elimination	Wrapper	Enlève itérativement la moins importante via un estimateur	✓	✓	Prend en compte l'interaction features-cible	Lourd en calcul
Sélection séquentielle (SFS/SBS)	Wrapper	Ajoute ou retire séquentiellement les meilleures features	✓	✓	Flexible (forward/backward)	Risque de surapprentissage, coûteux
L1 Regularization (Lasso)	Intégré	Pénalise les coefficients pour en annuler certains	✓	✓ (via "one-vs-rest")	Rapide, intégré au modèle	Seuil dépendant du paramètre de pénalité
Importance d'arbre (Random Forest)	Intégré	Mesure la réduction d'impureté apportée par chaque feature	✓	✓	Gère variables mixtes, robuste au bruit	Biais vers variables à plus de niveaux
Gradient Boosting Importances	Intégré	Comme pour les forêts, mais via boosting	✓	✓	Meilleure précision souvent	Plus lent que Random Forest

Tableau 3 : Tableau de synthèse des différentes approches possible pour analyser l'importance des features

Filtre = sélection avant apprentissage, indépendante du modèle.

Wrapper = sélection autour du modèle, itérative et évaluée par performance.

Intégré = sélection pendant l'apprentissage, via des pénalités ou des métriques internes au modèle

Si on reprend le cours de DataScienceTest

« Chapitre du cours DataScience Test :

Les différentes méthodes de Feature selection peuvent donner des résultats différents en raison de leurs approches, de leurs hypothèses sous-jacentes et de la nature des données. En ce qui concerne la question de la méthode à privilégier, il n'y a pas de réponse unique, car cela dépendra du contexte spécifique de votre problème, de vos données et de vos objectifs. Il est souvent recommandé d'explorer plusieurs méthodes de Feature selection, d'évaluer leurs performances sur un ensemble de validation ou à l'aide de techniques de validation croisée, et de choisir celle qui donne les meilleurs résultats en termes de performances du modèle. Il peut également être utile de combiner plusieurs méthodes de sélection de features pour tirer parti de leurs avantages respectifs et pour obtenir des résultats plus robustes. »

Approche recommandée

Exécuter au moins une méthode de chaque famille, puis analyser leurs résultats au regard des scores obtenus pour valider ou infirmer vos hypothèses.

Bonnes pratiques de code

Adopter une architecture modulaire afin d'assurer une reproductibilité rapide et une évolutivité aisée. Dans la suite nous nous sommes concentrés sur la détection d'espèces.

On a décidé de prendre les sélecteurs suivants :

```
# Univariate: F-test, keep top 10 features
'univariate_f': SelectKBest(score_func=f_classif, k=10),

# RFE avec RandomForest
'rfe_rf': RFE(
    estimator=RandomForestClassifier(n_estimators=100, random_state=random_state),
    n_features_to_select=10
),

# Model-based: L1 LogReg
'sfm_l1': SelectFromModel(
    estimator=LogisticRegression(
        penalty='l1', solver='liblinear', random_state=random_state
    ),
    threshold='median'
),

# Model-based: Tree
'sfm_tree': SelectFromModel(
    estimator=RandomForestClassifier(n_estimators=100, random_state=random_state),
    threshold='median'
)
```

On couvre ainsi chaque type et on a pris les valeurs conseillées d'hyper paramètres par défaut.

Nous avons préféré investir sur la modularité du code, car notre groupe avait une conviction c'est que le processus idéal pour ce type de projet est itératif. Travailler le coût de chaque itération et sécuriser son résultat nous a paru la bonne approche pour permettre de factualiser nos résultats finaux.

Ainsi on peut faire évoluer notre démarche au fur et à mesure de notre compréhension sans en payer un cout de réalisation trop fort.

La méthode evaluate_selectors est ainsi faite pour créer un pipeline pour chacun des sélecteurs. On a choisi un seul classifier qui est RandomForestClassifier, ce qui rend

comparable les différents résultats.

Le RandomForest a été choisi pour plusieurs raisons,

1. **Robustesse** - Les forêts aléatoires sont robustes face au sur-apprentissage
2. **Gestion des classes déséquilibrées** via `class_weight='balanced'`, ce qui reste encore le cas malgré le pré traitement d'oversampling
3. **Mesure d'importance intégrée** – ce qui va être utile pour nos graphiques d'analyse
4. **Capacité à gérer des relations non-linéaires**

On a gardé l'approche RobustScaler comme énoncée plus haut pour la standardisation.

```
# Pour chaque sélecteur
for name, selector in selectors.items():
    print(f"\nÉvaluation du sélecteur: {name}")

# Créer le pipeline
pipe = Pipeline([
    ('scaler', scaler),
    ('feat_sel', selector),
    ('clf', classifier)
])

# Création d'une validation croisée stratifiée explicite
stratified_cv = StratifiedKFold(n_splits=cv, shuffle=True, random_state=random_state)

# Évaluation par validation croisée stratifiée
cv_results = cross_validate(
    pipe, X, y, cv=stratified_cv, # Utilisation explicite de StratifiedKFold
    scoring=['accuracy', 'f1_macro', 'precision_macro', 'recall_macro'],
    return_train_score=True,
    return_estimator=True # Pour récupérer les modèles entraînés
)
```

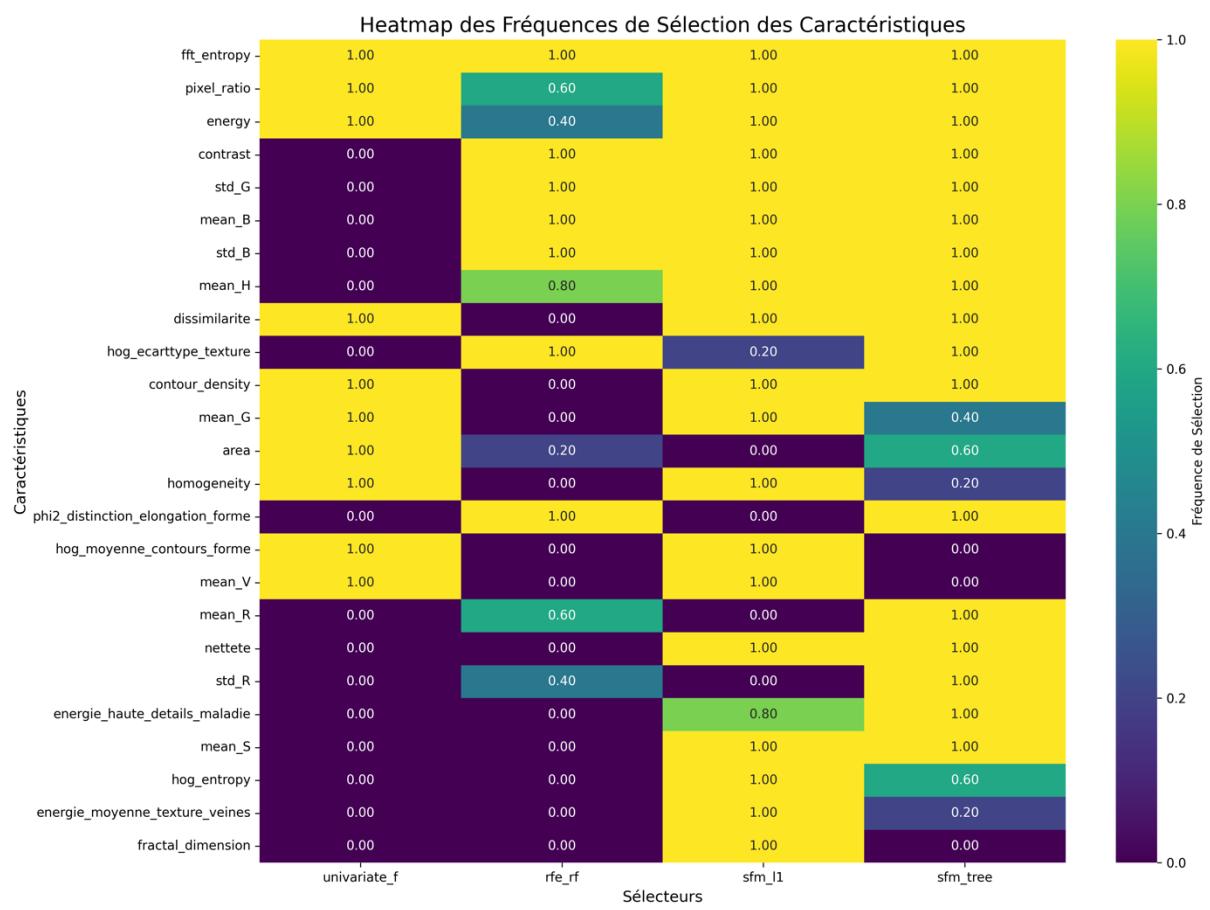


Figure 25 : Heatmap des Fréquences de Sélection des Caractéristiques

Voici une première heatmap qui permet d'observer les choix de feature par fréquence, on retrouve le fait que chaque type d'approche ne favorise pas une solution commune.

On a choisi les 25 features en fonction de la moyenne arithmétique des scores d'importance à travers toutes les méthodes de sélection.

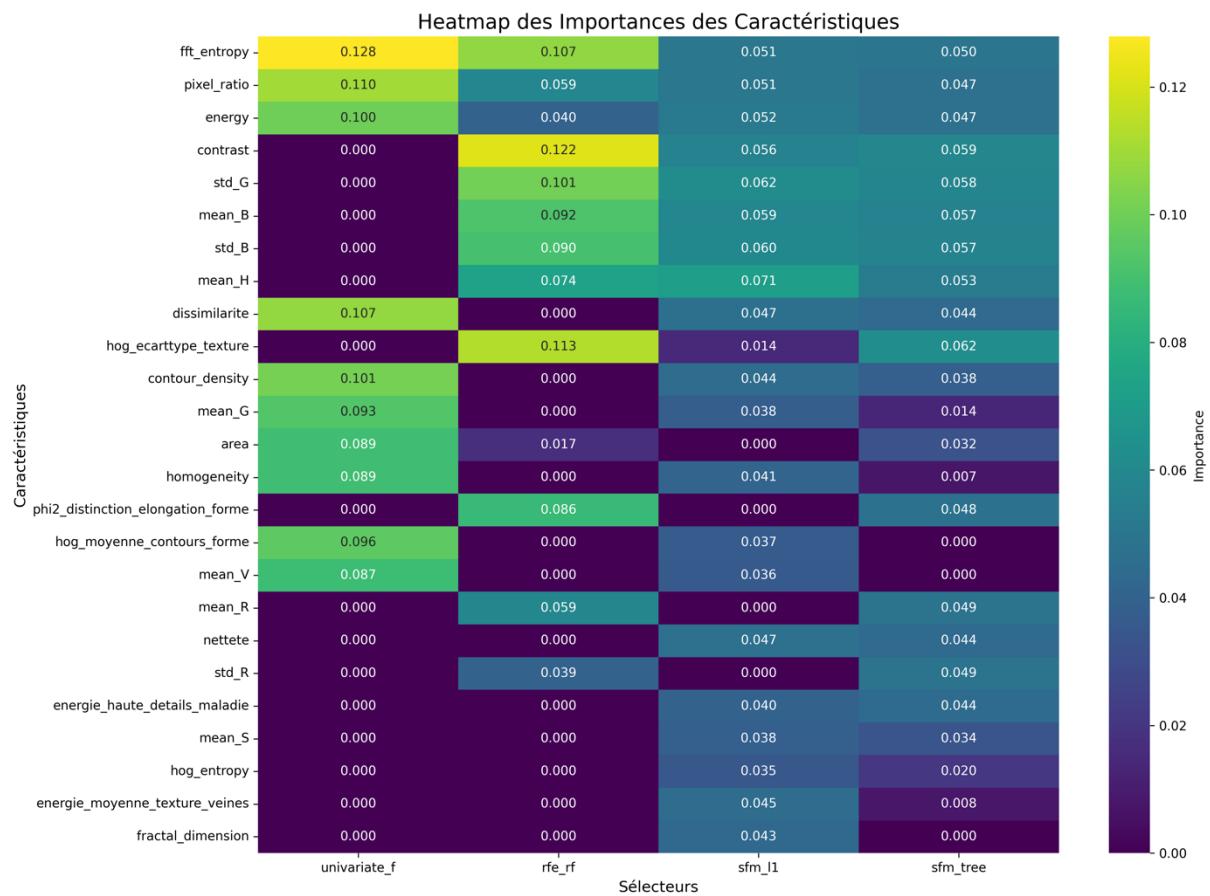


Figure 26 : Heatmap des Importances des Caractéristiques

On voit bien que le choix du modèle va impacter nos choix de feature, pour l'instant on intuite une solution par arbre mais l'analyse des scores via des exécutions par pipelines nous aidera à le factualiser. On pourra ainsi faire varier les différentes architectures, hyper paramètres et confronter les résultats pour valider nos choix.

Dans un cadre professionnel et non pas scolaire, nous aurions pu envisager aussi de faire travailler AutoMI afin de gagner du temps dans cette démarche d'essai erreur sur plusieurs modèles et hyper paramètre.

On a malgré tout voulu y voir plus clair avec une approche arbre, et nous avons testé un SHAP via TreeExplainer afin de mieux comprendre le rôle pour chacune des classes.

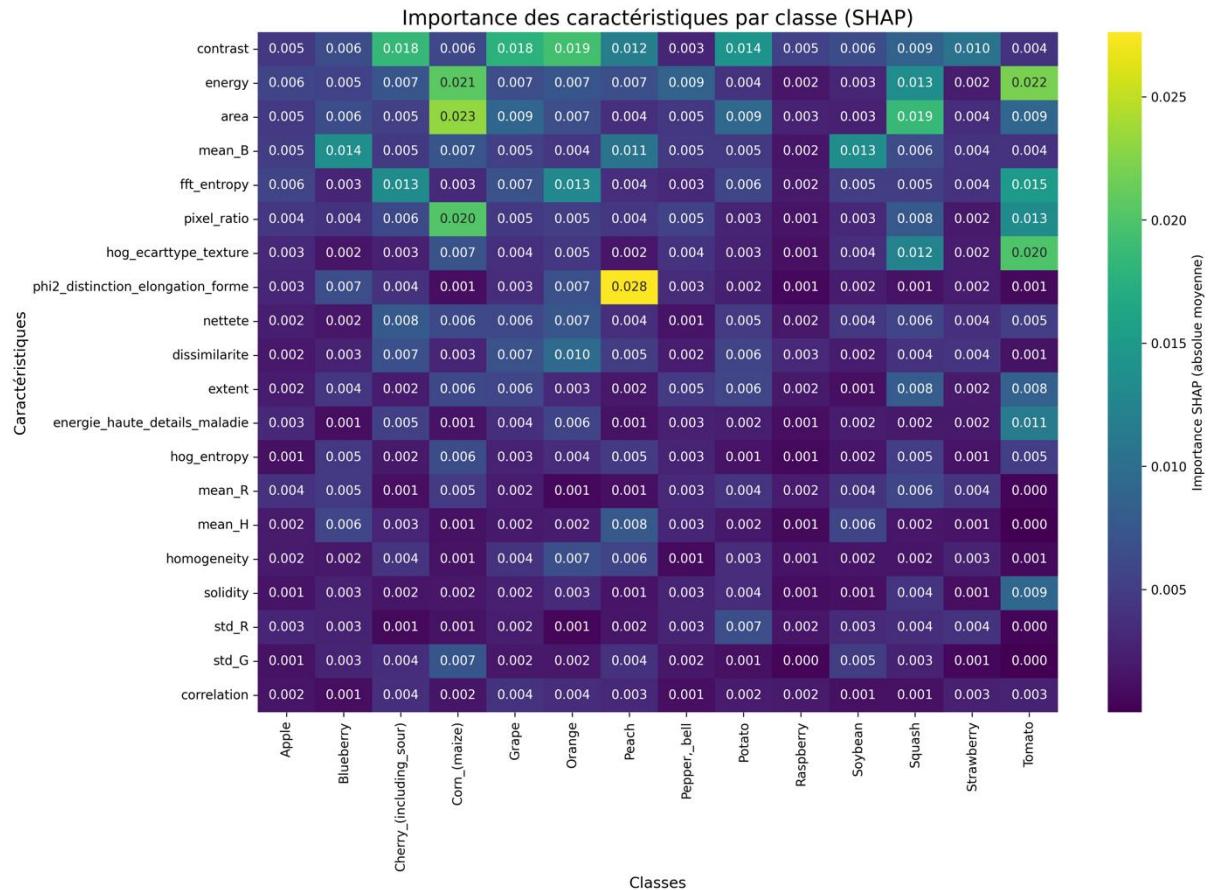


Figure 27 : Importance des caractéristiques par classe (SHAP)

Certaines features jouent un rôle plus ou moins important en fonction des classes.

Nous allons adapter en fonction des modélisations que nous souhaitons explorer le choix de nos features. On pourra explorer via SHAP avec le bon explainer et favoriser les features en fonction de l'importance pour les classes, faire un premier filtre.

Puis reprendre une exploration avec le sélecteur correspondant à notre modèle cible pour raffiner, et lancer des campagnes afin de collecter les résultats et ainsi regarder pour enlever

le bruit et optimiser le F1 score. L'approche consistera à tester nos hypothèses puis à affirmer ou infirmer celles-ci.

On se propose de revoir notre stratégie et faire 2 modèles multi-classe, un pour la détection de l'espèce puis un pour la détection de maladies avec cette idée que l'état sain est une classe comme les autres.

L'avantage de cette approche est de favoriser la réutilisabilité du code et de la démarche et nous permettra dans un temps raisonnable de rechercher la meilleure modélisation. Il faut prendre en compte le fait que nous ne travaillons que sur un dataset (plantvillage) et que lorsque les datasets seront mergés ces résultats vont forcément évoluer.

Nous allons malgré tout creuser le besoin de chainer les deux modèles.

9. Conclusion

Au terme de cette première phase du projet, nous avons suivi une démarche structurée en six blocs successifs, allant de l'exploration des données brutes jusqu'à la sélection des meilleures caractéristiques. Ce travail nous a permis de poser des bases solides pour la suite des analyses.

Dans un premier temps, l'exploration des six jeux de données disponibles a abouti à la sélection d'un dataset, facilitant ainsi la prise en main des outils de Machine Learning et la gestion de la complexité du sujet. Grâce à l'extraction et à l'ingénierie de nouvelles caractéristiques, nous avons enrichi notre dataframe et affiné notre compréhension des données. Les analyses exploratoires menées ont permis de soulever plusieurs points d'attention, dont certains ont déjà été intégrés à notre processus.

La séparation des données a été réalisée de manière stratifiée, garantissant une répartition représentative des classes lors des phases d'apprentissage et de test. Ce premier "run" a été particulièrement instructif :

- Il a permis de définir l'architecture du code, d'organiser notre travail collaboratif, et de choisir les outils adaptés.
- Il a favorisé une meilleure compréhension de nos données, de leurs spécificités et de leurs contraintes.
- Il a permis de poser les jalons d'une méthodologie reproductible et adaptable pour la suite du projet.

Ce premier bilan va nous permettre une optimisation progressive de notre démarche. Lors des prochaines itérations, nous capitaliserons sur l'expérience acquise pour accélérer les cycles d'expérimentation. Nous pourrons ainsi envisager de nouveaux choix méthodologiques, comme :

- L'intégration et l'autorisation des outliers et valeurs extrêmes,

- La modification de la gestion de la variable cible, en passant d'une vision binaire à une classification multi-classes,
- L'ajout progressif de nouveaux jeux de données,
- L'enrichissement dynamique des features en fonction des résultats observés.

En conclusion, cette première étape a été essentielle pour structurer notre approche, clarifier les objectifs, et préparer des analyses plus approfondies. La méthodologie appliquée jusqu'ici servira de socle à une exploration plus large et à des expérimentations ciblées, dans le but d'améliorer la pertinence et la performance de nos futurs modèles.

10. Bibliographie

Publications :

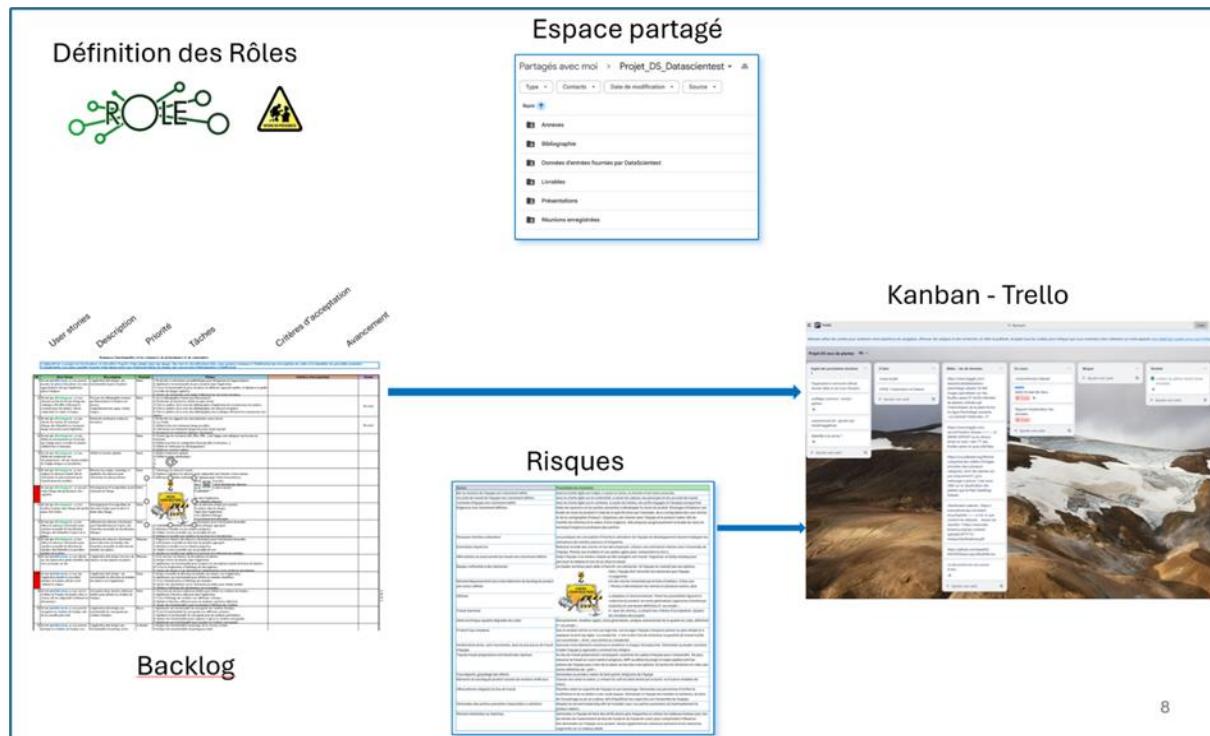
- **[2024] - Trends in Machine and Deep Learning Techniques for Plant Disease Identification: A Systematic Review** - Diana-Carmen Rodríguez-Lira , Diana-Margarita Córdova-Esparza , José M. Álvarez-Alvarado Juan Terven , Julio-Alejandro Romero-González, JuvenalRodríguez-Reséndiz
- **[2024] - A systematic review of machine learning and deep learning approaches in plant species detection** - Deepti Barhate , Sunil Pathak, Bhupesh Kumar Singh , Amit Jain , Ashutosh Kumar Dubey
- **[2024] - A systematic review of deep learning techniques for plant diseases** - Ishak Pacal · Ismail Kunduracioglu · Mehmet Hakki Alma · Muhammet Deveci · Seifedine Kadry · Jan Nedoma · Vlastimil Slany · Radek Martinek
- **[2024] - ASystematic Literature Review of Machine Learning and Deep Learning Approaches for Spectral Image Classification in Agricultural Applications Using Aerial Photography** – Usman Khan, MuhammadKhalidKhan, MuhammadAyubLatif, MuhammadNaveed, MuhammadMansoorAlam, SalmanA.Khan, MazlihamMohdSu'ud

Livres :

- **[2022] - The StatQuest Illustrated Guide To Machine Learning** – Josh Starmer -
- **[2017] – Machine Learning avec Scikit-Learn** – Aurélien Géron - Dunod

11. Annexes

11.1. Organisation du projet



8

Nous réalisons des points hebdomadaires

11.2. Reco_plante_Rapport exploration des données 24_07_2025.xls

N° Col	Nom de la colonne	Description	Disponibilité de la variable et comment la remplir	Type informatique	Taux de NA	Gestion des NA	Distribution des valeurs	Remarques sur la colonne	Librairie / Source	Fonction
Qui représente cette variable en quelques mots ?										
1	fit_image	Nombre de l'image	remplie mais pas dans le champ en arrière-plan	int4, float4 etc...	0 %	Si "totaux" déclinaut	Quelle mode de [coche] - gestion des NA favorisez vous ?	utemps libre à renseigner		
2	Image_Path	Chemin de l'image	oui	int4	0	Aucun	Uniques ou plusieurs			utilisé pour le chargement de la lien au fichier trait.
3	nom_plante	Espece de la plante	oui	objet	0 %	Aucun	Catégorie - sup à 10 catégories			utile pour identifier la plante concernée
4	nom_categorie	Catégorie de la plante	oui	objet	-2 %	Aucun	Catégorie - sup à 10 catégories			utile pour identifier la plante concernée
5	Etat_Sante	Etat de la feuille (saine + malade =0)	oui	bool	0	Remplacer par la valeur moyenne	Type de maladie détectées sur la feuille			si la plante est saine ou malade (obligatoire pour prédictif)
6	nt_hex	Flag indiquant si l'image est une image normale (1) ou non (0)	oui	bool	1 %	Remplacer par 0.5	Indique si la feuille est saine ou non			si l'image est globalement norme ou maladie (utilité de donner) Sert au filtre
7	Texte_hex	Texte de la feuille	oui	Text	0 %	Supprimer la ligne si NA	Taille de la feuille			Surface occupée par la feuille ou la partie détectée
8	area	Surface de la feuille détectée via contour (px²)	oui	float4	-1 %	Supprimer la ligne si NA	Longueur du contour de la feuille			Longeur du contour de la forme détectée
9	perimetre	Longueur du contour de la feuille (px)	oui	float4	-1 %	Supprimer la ligne si NA	Perimetre de la feuille			Perimetre de la forme détectée
10	ratio_perimetre	Ratio perimetre/area	oui	float4	-1 %	Supprimer la ligne si NA	Entrez 0 et 1			Meilleur ratio entre le perimetre et l'aire
11	asymmetrie_ratio	Ratio asymétrie de la forme (0 <= ratio <= 1 : lignes Dimensionnelles)	oui	float4	-1 %	Supprimer la ligne si NA	Asymétrie de la forme			Meilleur ratio entre les formes d'épaisseur
12	mean_1	Moyenne du canal Rouge de la feuille (Intensity Normalized (0-1))	oui	float4	-1 %	Remplacer par moyenne	Valores 0 et 1			Meilleur des intensités sur le canal Rouge, Vert, Bleu. Donne la tonne dominante
13	mean_2	Moyenne du canal Vert de la feuille (Intensity Normalized (0-1))	oui	float4	-1 %	Remplacer par moyenne	Entrez 0 et 1			Meilleure des intensités sur le canal Rouge, Vert, Bleu. Donne la tonne dominante
14	mean_G	Moyenne du Canal Vert de la feuille (Intensity normalisée (0-1))	oui	float4	-1 %	Remplacer par moyenne	Coleur moyenne verte de la feuille			Meilleure des intensités sur le canal Rouge, Vert, Bleu. Donne la tonne dominante
15	mean_B	Moyenne du canal Bleu de la feuille (Intensity Normalized (0-1))	oui	float4	-1 %	Remplacer par moyenne	Entrez 0 et 1			Meilleure des intensités sur le canal Rouge, Vert, Bleu. Donne la tonne dominante
16	std_R	Ecart-type du canal Rouge (Intensity Normalized (0-1))	oui	float4	-1 %	Remplacer par moyenne	Variation du rouge sur la feuille			Variation réel type des couleurs – mesure la diversité chromatique
17	std_G	Ecart-type du canal Vert (Intensity Normalized (0-1))	oui	float4	-1 %	Remplacer par moyenne	Entrez 0 et 1			Variation réel type des couleurs – mesure la diversité chromatique
18	std_B	Ecart-type du canal Bleu (Intensity Normalized (0-1))	oui	float4	-1 %	Remplacer par moyenne	Variation du bleu sur la feuille			Variation réel type des couleurs – mesure la diversité chromatique
19	mean_H	Moyenne du canal Tonk (Hue Degres (0-179))	oui	float4	-1 %	Remplacer par moyenne	0-179 degrés			Moyenne dans l'intervalle HSV (Hue Saturation Value) plus robuste aux variations de tonalité
20	mean_S	Moyenne du canal Saturation (0-255)	oui	float4	-1 %	Remplacer par moyenne	0-255			Moyenne dans l'intervalle HSV (Hue Saturation Value) plus robuste aux variations de tonalité
21	mean_V	Moyenne du canal Value (0-255)	oui	float4	-1 %	Remplacer par moyenne	0-255			Moyenne dans l'intervalle HSV (Hue Saturation Value) plus robuste aux variations de tonalité
22	variance	Variance de la Laponie (champs mesures intensité Dimensionnelles)	oui	float4	-1 %	Supprimer la ligne si NA	Valores positivos			Meilleure de la clarté des détails dans l'image (meilleur calculé par les gradients du Laplacien)
23	GLCM_Dissimilarity	GLCM dissimilitude (champs mesures intensité Dimensionnelles)	oui	float4	-1 %	Remplacer par mediane	0-1			Homogénéité des textures
24	energy	GLCM energy - variance entre les cases de la matrice de cooccurrence (Dimensionnelles)	oui	float4	-1 %	Remplacer par mediane	Valores positivos			Energy des textures de gris
25	homogeneity	GLCM homogeneity - variance uniforme ou varianc (Dimensionnelles)	oui	float4	-1 %	Remplacer par mediane	0-1			Homogénéité des textures
26	GLCM_Contrast	GLCM contrast - variance entre les cases de la matrice de cooccurrence (Dimensionnelles)	oui	float4	-1 %	Remplacer par mediane	Valores positivos			Contrastation des textures de gris
27	Correlation	GLCM correlation - dépendance linéaire des intensités voisines (Dimensionnelles)	oui	float4	-1 %	Remplacer par mediane	1 à 1			Corrélation des niveaux de gris voisins
28	correl_coeff	Densité de corrélation - pondérée 1/area (Dimensions)	oui	float4	-1 %	Supprimer la ligne si NA	pondérées / area			Rapport contrast / surface
29	1_moment	1 moment de la distribution de probabilité (Dimensions)	oui	float4	-1 %	Remplacer par mediane	Valores continuas			Rapport contrast / surface (pour le moment)
30	nt_1	Moment de la loi normale (champs descripteurs matériels) (Dimensions)	oui	float4	-1 %	Remplacer par mediane	Descriptor de forme 2			Descriptor de forme 2 (pour le moment)
31	nt_2	Moment de la loi normale (champs descripteurs matériels) (Dimensions)	oui	float4	-1 %	Remplacer par mediane	Descriptor de forme 3			Descriptor de forme 3 (pour le moment)
32	nt_4	Moment de la loi normale (champs descripteurs matériels) (Dimensions)	oui	float4	-1 %	Remplacer par mediane	Descriptor de forme 4			Descriptor de forme 4 (pour le moment)
33	nt_5	Moment de la loi normale (champs descripteurs matériels) (Dimensions)	oui	float4	-1 %	Remplacer par mediane	Descriptor de forme 5			Descriptor de forme 5 (pour le moment)
34	nt_6	Moment de la loi normale (champs descripteurs matériels) (Dimensions)	oui	float4	-1 %	Remplacer par mediane	Descriptor de forme 6			Descriptor de forme 6 (pour le moment)
35	nt_7	Moment de la loi normale (champs descripteurs matériels) (Dimensions)	oui	float4	-1 %	Remplacer par mediane	Descriptor de forme 7			Descriptor de forme 7 (pour le moment)
36	hog_mean	Moyenne des valeurs HOG (Histogramme des gradients orientés)	oui	float4	-1 %	Remplacer par moyenne	Valores positivos			Descriptor de grandeur des gradients
37	hog_std	Ecart-type des histogrammes HOG	oui	float4	-1 %	Remplacer par moyenne	Valores positivos			Mesure de la complexité de la texture
38	hog_entropy	Entrée des histogrammes HOG	oui	float4	-1 %	Remplacer par moyenne	Valores positivos			Force de l'hétérogénéité
39	nt_mean	Energie spectrale de la transformation de Fourier	oui	float4	-1 %	Remplacer par moyenne	Valores positivos			Force de l'hétérogénéité
40	nt_mean_sq	Entrée de la transformation de Fourier	oui	float4	-1 %	Remplacer par moyenne	Valores positivos			Force de l'hétérogénéité
41	nt_low_frec_power	Puissance dans la bande de basse fréquence (Transformée de Fourier)	oui	float4	-1 %	Remplacer par moyenne	Valores positivos			Informations globales (formes, zones homogènes)
42	nt_high_frec_power	Puissance dans la bande de haute fréquence (Transformée de Fourier)	oui	float4	-1 %	Remplacer par moyenne	Valores positivos			Informations basses fréquences (contours et détails fins)
43										
44	plant_Apple									
45	plant_Bacterial_spot									
47	plant_Tomato									
48	plant_Apricot_healthy									
49	plant_Bacterial_spots									
50	disease_Bacterial_spot									
51	disease_Tomato_mosaic_virus									
52	disease_Healthy									

11.3. Exploration du dataset complet PlantVillage

Quelques analyses statistiques fournissent un aperçu de ce dataset:

Distribution du nombre d'images par espèce et modalité :



1.Distribution des Images par Espèce :

- La majorité des espèces ont entre 0 et 2000 images. Il y a un pic notable autour de 1000 images par espèce, ce qui suggère que beaucoup d'espèces sont représentées par environ ce nombre d'images.
- Il y a quelques espèces avec un nombre d'images significativement plus élevé, comme on peut le voir par les barres autour de 5000 images.

2.Modalités des Images :

- Le graphique utilise une légende pour indiquer que toutes les barres représentent la modalité "segmented". Cela signifie que les données présentées concernent uniquement les images segmentées.
- Les autres modalités ("color" et "greyscale") ne sont pas représentées dans cet histogramme particulier.

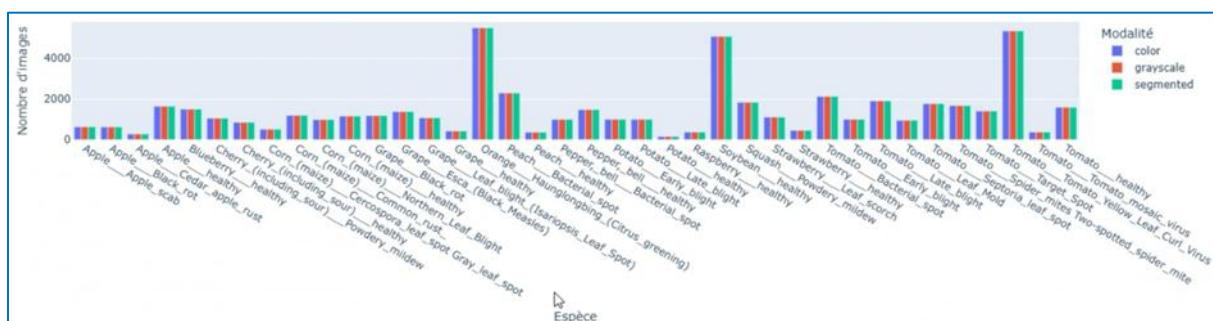
3.Variabilité du Nombre d'Images :

- Il y a une grande variabilité dans le nombre d'images par espèce. Certaines espèces ont très peu d'images, tandis que d'autres en ont beaucoup plus.
- Cette variabilité peut avoir des implications pour l'entraînement de modèles de Machine Learning, car les espèces avec moins d'images peuvent être moins bien représentées et donc plus difficiles à classer correctement.

4.Implications pour le Projet :

- Pour un projet de classification d'images, il serait important de prendre en compte cette variabilité. Des techniques comme le rééchantillonnage pourraient être utilisées pour augmenter le nombre d'images des espèces moins représentées.
 - Il pourrait également être utile de vérifier si les autres modalités ("color" et "greyscale") présentent une distribution similaire ou si elles offrent une couverture plus équilibrée des espèces.

Comparaison du nombre d'images par modalité et espèce :



1. Nombre d'Images par Espèce :

- Le nombre d'images varie considérablement d'une espèce à l'autre. Certaines espèces ont un nombre relativement faible d'images, tandis que d'autres en ont un nombre très élevé, dépassant parfois 4000 images.

2. Répartition des Modalités :

- Chaque espèce est représentée par trois barres de couleurs différentes, correspondant aux trois modalités : "color" (bleu), "greyscale" (rouge), et "segmented" (vert).
 - Pour chaque espèce, le nombre d'images semble être similaire à travers les trois modalités, ce qui suggère que le dataset est bien équilibré en termes de modalités pour chaque espèce.

3. Espèces avec un Grand Nombre d'Images :

- Certaines espèces, comme "Tomato_Early_blight", "Tomato_healthy", "Tomato_Leaf_Mold", "Tomato_Septoria_leaf_spot", et "Tomato_Spider_mites", ont un nombre particulièrement élevé d'images. Cela pourrait indiquer que ces espèces sont plus couramment étudiées ou plus faciles à imager.
 - Les espèces de tomates semblent être particulièrement bien représentées dans ce dataset.

4 Espèces avec un Petit Nombre d'Images

- D'autres espèces, comme "Apple_Cedar_apple_rust", "Apple_healthy", "Blueberry_healthy" et "Cherry_healthy" ont un nombre relativement faible d'images

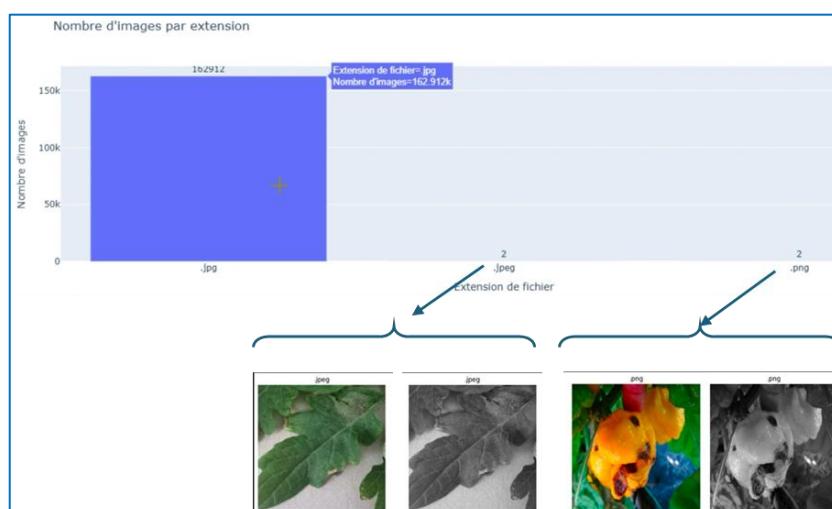
Cela pourrait indiquer que ces espèces sont moins courantes ou plus difficiles à imager.

5.Implications pour l'Analyse :

- La variabilité dans le nombre d'images par espèce peut avoir des implications pour l'entraînement de modèles de machine learning. Les espèces avec un grand nombre d'images peuvent être mieux représentées dans les modèles, tandis que celles avec moins d'images peuvent nécessiter des techniques d'augmentation de données pour améliorer leur représentation.
- L'équilibre entre les modalités pour chaque espèce est un bon signe pour la cohérence des données et peut faciliter les comparaisons et analyses entre les différentes modalités.

En résumé, ce graphique montre une grande variabilité dans le nombre d'images par espèce, avec certaines espèces de tomates étant particulièrement bien représentées. L'équilibre entre les modalités pour chaque espèce est un point fort du dataset.

Nombre d'images par extension :



1.Extension .jpg :

- L'extension .jpg a un nombre très élevé d'images, soit 162,912 images. Cela indique que la majorité des images dans ce dataset sont au format JPEG.

2.Extensions .jpeg et .png :

- Les extensions .jpeg et .png ont chacune seulement 2 images.

3.Implications pour le Dataset :

- La prédominance des fichiers .jpg peut indiquer que le dataset est principalement composé de photographies ou d'images similaires qui bénéficient de la compression JPEG.

Largeur vs Hauteur des images selon l'extension :

1.Extensions de Fichier :

- Le graphique utilise trois couleurs différentes pour représenter les extensions de fichier : bleu pour .jpg, orange pour .jpeg, et vert pour .png.

2.Distribution des Tailles d'Images :

- La majorité des images ont une largeur d'environ 500 pixels et une hauteur d'environ 500 pixels. Cela suggère que la plupart des images dans ce dataset sont carrées ou presque carrées.
- Il y a quelques images avec une largeur d'environ 250 pixels et une hauteur d'environ 300 pixels, indiquant une légère variation dans les dimensions des images.

3.Prédominance des Images .jpg :

- Les points bleus, représentant les images .jpg, sont les plus nombreux, ce qui confirme que la majorité des images dans le dataset sont au format .jpg.
- Les images .jpeg et .png sont très peu représentées, avec seulement quelques points orange et verts visibles sur le graphique.

4.Uniformité des Dimensions :

- La concentration des points autour de la largeur et de la hauteur de 500 pixels indique une certaine uniformité dans les dimensions des images. Cela peut être utile pour le traitement et l'analyse des images, car cela réduit la nécessité de redimensionner les images à une taille commune.

5.Implications pour le Traitement des Images :

- L'uniformité des dimensions des images peut faciliter le traitement par lots et l'application de modèles de Machine Learning, car les images ont déjà des tailles similaires.
- La prédominance des images .jpg suggère que les efforts de traitement et d'analyse peuvent se concentrer sur ce format, bien qu'il soit important de ne pas négliger les quelques images dans d'autres formats.

En résumé, ce graphique montre que la plupart des images dans le dataset sont au format .jpg et ont des dimensions uniformes autour de 500 pixels de largeur et de hauteur. Cela peut simplifier le traitement et l'analyse des images.

11.4. Analyse exploratoire des caractéristiques

L'analyse exploratoire est supportée par des graphiques (histogrammes, boxplot, heatmap...).

Ils sont générés avec plotly, avec des menus déroulants. Ils sont enregistrés dans des fichiers HTML pour explorer plus précisément les caractéristiques :

-  heatmap_features_grouped.html
-  distributions_features_targets.html
-  Boxplot8.html
-  Boxplot7.html
-  Boxplot6.html
-  Boxplot5.html
-  Boxplot4.html
-  Boxplot3.html
-  Boxplot2.html
-  Boxplot1.html
-  objectif3_histos.html
-  objectif2_histos.html
-  objectif1_histos.html
-  outliers_par_feature.html

11.5. Environnement de développement

La gestion de configuration s'effectue avec Github.

Nous avons créé un environnement nommé conda_env