



Comparaison de méthodes de réduction de dimension pour des analyses de données biologiques

Reunan Bellec & Malo Gillard
Enseignant référent : David Rousseau

21 mai 2019

Table des matières

1	Introduction	2
2	Principe des méthodes	3
2.1	Méthodes par sélection	5
2.1.1	Apprentissage supervisé	5
2.1.1.1	Random Forest	5
2.1.1.2	Relief	8
2.2	Méthodes par transformation	9
2.2.1	Apprentissage non-supervisé	9
2.2.1.1	ISOMAP	9
2.2.1.2	PCA	12
2.2.1.3	t-SNE	14
2.2.2	Apprentissage supervisé	16
2.2.2.1	LDA	16
3	Réduction de dimension sur des données biologiques	20
3.1	Introduction des données	20
3.2	Expertise	22
3.3	Résultats	23
4	Organisation du projet	26
5	Conclusion	28
	Bibliographie	31

Chapitre 1

Introduction

Le dérèglement climatique, le développement de nouvelles maladies des plantes, la maîtrise des rendements, amènent les états et l'ensemble des acteurs de l'agriculture en charge de la sélection variétale, à identifier des semences performantes, résistantes aux maladies, à des périodes de sécheresse ou de brusques variations environnementales durant leur développement. Ces travaux peuvent bénéficier d'avancées technologiques récentes en matière de traitements de l'information, applicables sur de larges populations de plantes. Une échelle particulièrement importante est celle de la graine, dont la qualité germinative conditionne la suite du développement de la plante. Dans ce projet, nous nous intéressons aux données issues d'une étude sur les graines de betterave sucrière, pour laquelle la France est l'un des plus gros producteurs au monde. L'objectif de cette expérience est d'élargir la variabilité génétique de la betterave, dans le but de la rendre plus compétitive, en doublant le rythme de croissance annuelle de son rendement en sucre. Des tests ont alors été réalisés sur différents génotypes de betteraves (200 individus) pour étudier leur germination. Les résultats obtenus sur les semences sélectionnées forment ainsi un gros jeu de données, où les échantillons étudiés sont décrits par divers paramètres tels que la surface, la longueur, la largeur, l'imbibition ou la vitesse de germination par exemple.

Compte tenu de la taille du jeu de données, il est nécessaire d'en réduire la dimension, c'est-à-dire diminuer le nombre de variables, pour pouvoir visualiser les données de manière intuitive, et ainsi pouvoir répondre aux problématiques de manière efficace. La problématique est ici de vérifier, par visualisation des données, que des biais d'expérimentation n'ont pas été introduit dans l'étude des génotypes. Ainsi, dans le cadre de ce travail d'étude et de recherche de première année de master Data Science, nous allons présenter différentes méthodes de réduction de dimension. Enfin, lorsque les principes des méthodes auront été compris, nous les appliquerons au jeu de données, puis nous analyserons brièvement les résultats afin de statuer sur la présence de biais au sein de l'expérience.

Chapitre 2

Principe des méthodes

Les objectifs de la réduction de dimension peuvent être de réduire le volume de stockage d'un jeu de données, de ne garder que des caractéristiques discriminantes pour éviter le sur-apprentissage ou simplement de visualiser les données en vue d'éventuelles études approfondies. Dans notre cas il s'agira de savoir s'il existe un biais d'expérience suffisamment important pour l'observer sur la représentation. De cette façon on verrait une hétérogénéité des points apparaître selon un paramètre expérimental théoriquement homogène.

Les méthodes de réduction de dimension peuvent être catégorisées de deux manières différentes. Tout d'abord, selon leur approche de l'information, certaines méthodes suppriment les dimensions faibles en information tandis que d'autres combinent les variables pour effacer la redondance et ainsi concentrer l'information. Cette dernière approche est celle des méthodes par transformation dans lesquelles les individus peuvent être représentés dans un nouvel espace, de dimension réduite, de sorte que les relations entre les nouveaux points soient aussi fidèles à ceux de la dimension d'origine que possible. Enfin, l'autre manière de catégoriser les méthodes est de séparer les méthodes supervisées des autres. Les méthodes d'apprentissage supervisées nécessitent que les données soient préalablement étiquetées tandis que pour les méthodes d'apprentissage non-supervisées, l'algorithme est autonome, il ne connaît pas d'exemples de résultats attendus. On notera la nécessité d'une expertise et d'une intuition supplémentaire pour l'exploitation des résultats issus de ces derniers. Ainsi, il est possible de classer chaque méthode selon les critères précédemment cités, comme dans la figure 2.1 ci-dessous par exemple.

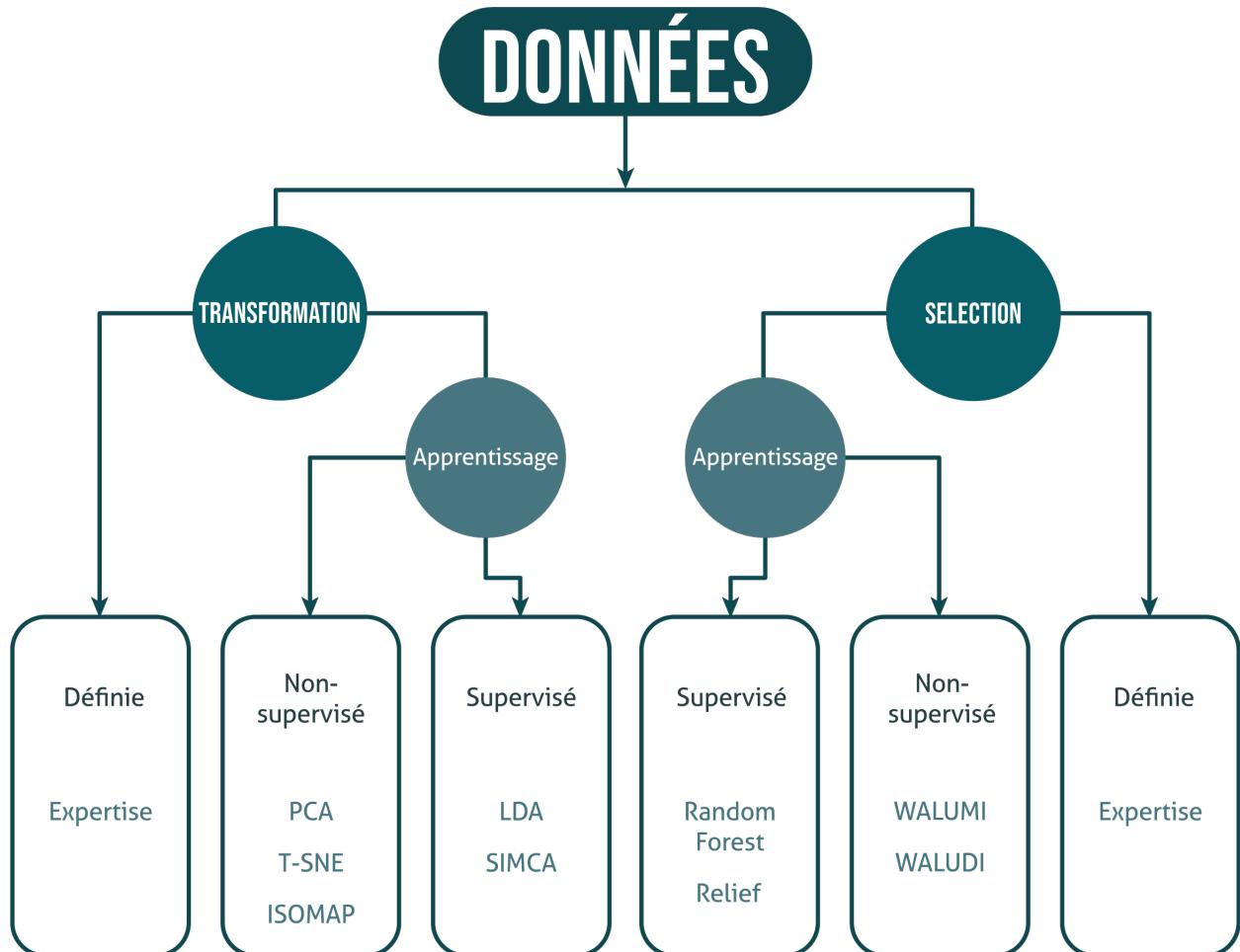


FIGURE 2.1 – Ce schéma résume simplement l'organisation de quelques méthodes d'apprentissages utilisées pour la réduction de dimension.

Dans l'objectif d'une utilisation efficace et pertinente des méthodes d'apprentissage, il est nécessaire de comprendre le fonctionnement et les formules mathématiques sous-jacentes à chacune d'entre elles. Ainsi pour chacune de ces méthodes, nous commencerons par étudier le principe général pour savoir dans quelles situations celles-ci sont pertinentes. Nous nous attacherons ensuite à détailler le fonctionnement mathématique des méthodes et enfin des paramètres qui en découlent. Bien que des fonctions implémentées d'ajustement existent, il faut comprendre comment chacun des paramètres apparaît dans la méthode pour pouvoir les choisir de façon pertinente. Par manque de temps et de références pour certaines méthodes de la figure 2.1, nous nous concentrerons ici sur les méthodes suivantes : Random Forest, PCA, LDA, Isomap, t-SNE et Refief. Il existe néanmoins de nombreux ouvrages auxquels on pourra se référer dans l'optique d'approfondir le sujet,

tel que [1], [2] ou encore [3].

2.1 Méthodes par sélection

2.1.1 Apprentissage supervisé

2.1.1.1 Random Forest

A) Principe

La notion au coeur des random forest [4] est celle des arbres de décision. Ces derniers sont constitués de la manière suivante : les feuilles représentent les valeurs de la variable cible tandis que les branches représentent le chemin pour y arriver. Une variable explicative est choisie pour chaque noeud (embranchement), puis les données arrivées au noeud sont séparées. Le chemin final de la première variable choisie à la feuille est donc décrit par l'ensemble des choix des variables explicatives. On arrête de créer des noeuds lorsqu'un éventuel prochain noeud ne séparerait plus le sous-ensemble (défini au noeud précédent) ou lorsque la prédiction n'est plus améliorée. On s'intéresse, dans notre cas, aux arbres de classification (variable discrète à prédire).

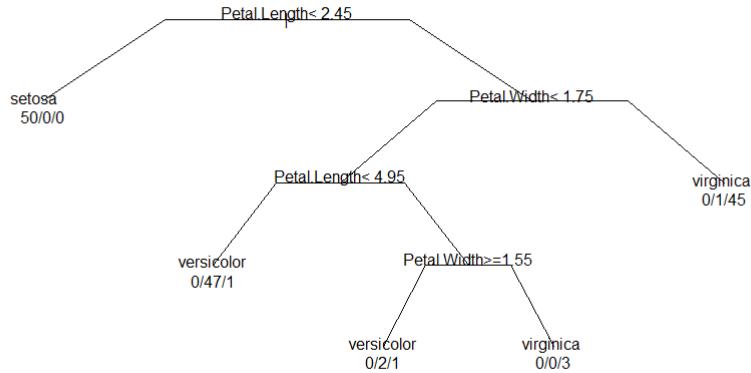


FIGURE 2.2 – Exemple d’abre de décision, calculé sur R avec le package "rpart" sur les données "iris"

Le choix de la variable de séparation des données est fait de sorte à maximiser une mesure de gain d’homogénéité (ou minimiser l’hétérogénéité selon le critère choisi) en tenant compte de tous les sous-ensembles à ce niveau de l’arbre. On veut en effet construire les sous-groupes de façon à ce qu’ils soient les plus homogènes du point de vue de la variable à prédire. Dans le cas où la variable choisie est continue, on effectue des tests selon différents points de coupures en évaluant le critère à chaque fois.

On note que si l’arbre contient autant de feuilles que d’individus, il sera grand et sans erreur de classification, mais peu efficace pour être utilisé à des fins de prédiction. Ainsi, il peut être nécessaire d’effectuer des méthodes de pré-élagage, qui consistent à ne pas garder un nœud si un des sous-ensembles résultants est trop petit ou si la mesure du critère est trop faible. Un post-élagage, méthode consistant à séparer l’échantillon d’apprentissage pour garder un échantillon d’élargissement, "pruning set", pour réduire l’arbre en optimisant ses performances. On peut aussi améliorer la qualité de prédiction par bagging, en sous-échantillonnant les données pour créer plusieurs arbres de décision, pour ensuite combiner les résultats de sorte qu’une variable prédite soit celle qui aura été prédite le plus fréquemment.

Le boosting consiste à pénaliser les observations de manière inversement proportionnelle à la pertinence de leur classification. Ainsi une observation difficile à

classer aura un poids plus élevé. De cette façon, de nouveaux arbres de décision sont générés, avec des erreurs de classification de plus en plus faibles. Bien que largement utilisée dans les algorithmes impliquant des arbres de décision, il s'agit d'une méthode courante en apprentissage supervisé.

L'importance des variables est déduite de la valeur du critère utilisé. Selon la valeur de cette dernière, on déduira l'importance de la variable, ce qui donnera une possibilité de les sélectionner ensuite. On notera que l'un des avantages de cette méthode est la transparence des séparations effectuées, contrairement à d'autres méthodes d'apprentissage comme les réseaux de neurones par exemple.

B) Mathématiques

Le coefficient de Gini cité ci-dessus se calcule de la manière suivante :

$$G = \sum_{i \neq j} \frac{|S_i||S_j|}{|S|^2} \quad (2.1)$$

où S_k représente le sous-ensemble des observations de la même classe k et S l'ensemble contenant toutes les observations. L'entropie de Shannon se calcule comme ci-après :

$$S = - \sum_{i \in I} p_i \log(p_i) \quad (2.2)$$

où I est l'ensemble des observations ayant la même valeur prédictive et p_i est la proportion d'exemples de P ayant pour classe résultante $i \in I$. L'objectif du choix du nœud est de s'approcher de zéro (sans forcément l'atteindre pour des raisons de sur-apprentissage).

C) Utilisation pratique et paramètres

Pour cette méthode, différents paramètres peuvent être définis : pour la forêt en elle-même, ou pour la construction des arbres de décision qui la constituent. La taille de l'échantillon et le nombre d'arbres concernent la construction de la forêt tandis que la taille des sous-ensembles, la profondeur maximale de l'arbre ou le critère choisi concernent les arbres qui la composent. Par exemple, on prêtera attention à ce que les arbres ne soient pas trop grands pour éviter le surapprentissage. On parle de surapprentissage lorsque le modèle correspond parfaitement aux données d'apprentissage mais qu'il engendre de fortes erreurs à la prédiction. À l'inverse, des arbres trop petits ne permettront pas d'enregistrer suffisamment d'informations des données d'apprentissage. Pour régler le nombre d'arbres, il est d'usage [5] d'effectuer plusieurs calculs du taux d'erreurs de classification, jusqu'à atteindre un nombre d'arbres au delà duquel le nombre d'erreur ne diminue plus.

2.1.1.2 Relief

A) Principe

L'algorithme Relief [6] a été publié pour la première fois par Kira et Rendell. Bien qu'aujourd'hui, on puisse parler de famille d'algorithmes [7], nous nous attacherons à l'algorithme d'origine et à son extension que nous utiliserons : ReliefF. En effet, initialement Relief prend en compte deux valeurs pour chaque variable prédictive, contrairement à ReliefF. Le principe de cet algorithme est de chercher les variables les plus pertinentes pour classer la variable cible, en calculant une mesure globale de la pertinence des caractéristiques des données en accumulant la différence des distances entre des exemples. Il prend en compte les données étiquetées pour ensuite rendre un score de pertinence pour chaque variable prédictive.

B) Mathématiques et déroulement de l'algorithme

L'algorithme suit un déroulement précis, tout d'abord, le vecteur de poids : W , utilisé pour la mesure de pertinence, est initialisé et vaut zéro. On sélectionne ensuite un point R_i de façon aléatoire dans l'ensemble des points, puis on recherche deux points les plus proches dans des classes différentes. En effet, un point est dans la même classe que le point sélectionné H , dit *hit*, tandis que l'autre se trouve dans l'autre classe M , dit *miss*. On actualise ensuite W pour chaque variable explicative X_i selon leurs valeurs de H, M et R_i , grâce à la formule

$$W_i = W_i - \frac{diff_i(R, H)}{k} + \frac{diff_i(R, M)}{k} \quad (2.3)$$

où k représente le $k^{\text{ième}}$ point choisi, et où $diff_i(I_1, I_2)$ se calcule de la façon suivante :

$$diff_i(I_1, I_2) = \frac{|X_i(I_1) - X_i(I_2)|}{\max(X_i) - \min(X_i)} \quad (2.4)$$

dans le cas d'une variable explicative continue et où $diff_i(I_1, I_2)$ vaut 0 ou 1 selon si R et H ont des valeurs différentes ou non. Si R et H ont des valeurs différentes pour une variable explicative, alors cette variable sépare des points qui sont étiquetées identiques. Il faut donc diminuer la qualité de cette variable, que l'on sauvegarde dans le vecteur de poids. Enfin, on répète le processus jusqu'au critère d'arrêt, qui peut concerner le temps de calcul ou le nombre d'itération.

L'apport de ReliefF : [7] reliefF diffère de la manière suivante : plutôt que de chercher le point le plus proche qui est étiqueté différemment, l'algorithme le fait

pour chaque point d'étiquetage différent et inclut ce résultat pondéré au poids affecté. Soit k le nombre de modalités dans la variable à prédire. On a alors pour un R_i k hits H_j et k misses M_j . Cela permet donc de gérer les problèmes multi-classes.

Cette méthode dispose de peu de paramètres, excepté le nombre de dimensions d de l'espace d'arrivée, où les dimensions sélectionnées seront les d premières avec le meilleur score. Dans notre cas on choisira un espace de dimension adapté à la visualisation (entre 1 et 3 dimensions gardées).

2.2 Méthodes par transformation

2.2.1 Apprentissage non-supervisé

2.2.1.1 ISOMAP

A) Principe

Tout comme les méthodes PCA (cf 2.2.1.2) et T-SNE (cf 2.2.1.3), la méthode ISOMAP (*Isometric Mapping* [8]) est une méthode d'apprentissage non-supervisé. Cependant, contrairement à la méthode PCA, l'approche utilisée ici propose de mieux approximer la structure géométrique locale de l'ensemble de données à travers la réduction de dimension. Par exemple, supposons que notre jeu de données soit représenté en trois dimensions par une courbe en S (figure 2.3) :

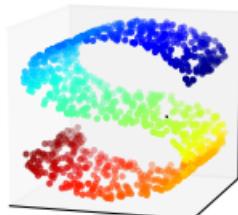


FIGURE 2.3 – Illustration d'une courbe en S (reproduit à partir de [9]).

En appliquant la méthode PCA à ce jeu de données, les résultats que nous obtiendrons (en passant de 3 à 2 dimensions) seront sous la forme de valeurs désorganisées. Au contraire, la méthode ISOMAP préservera la structure locale de l'ensemble de données après avoir appliqué la réduction de dimension. Nous pouvons voir sur l'image de droite de l'exemple ci-dessous (figure 2.4) que notre

S-curve a été "déroulée" de manière à pouvoir être représentée sur un plan.

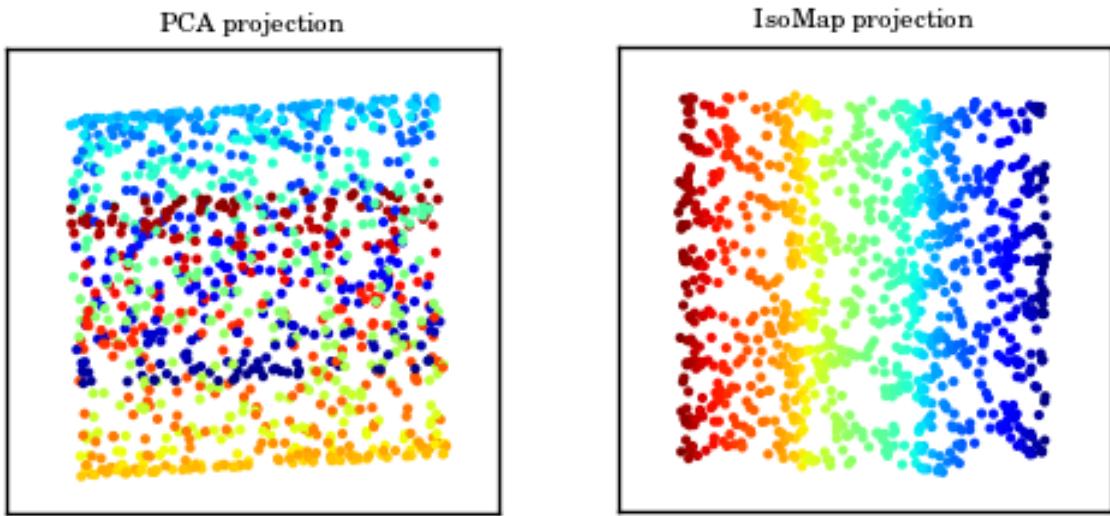


FIGURE 2.4 – Comparaison entre les méthodes PCA et ISOMAP (reproduit à partir de [9]).

La différence vient du fait que deux points peuvent être proches selon une distance euclidienne (utilisée dans le premier cas), mais très éloignés si on mesure la distance qui les sépare sur la **surface** définie par l'ensemble des points, appelée **distance géodésique** (ce que fait la méthode ISOMAP dans le deuxième cas). Néanmoins il faut prendre en compte le fait que nous ne connaissons pas la forme réelle de cette surface si nous n'avons qu'un ensemble discret de points, ce qui rend l'évaluation des distances géodésiques compliquée. Pour résoudre ce problème, la méthode ISOMAP va construire un graphe d'adjacence des points et approcher la distance géodésique en cherchant le chemin le plus court à travers ce graphe.

B) Explication mathématique

L'algorithme va se dérouler en 3 étapes.

Étape 1 : construction du graphe d'adjacence

Considérons que les données, n points de coordonnées x_1, \dots, x_n , sont représentées dans un ensemble X de dimension p . Pour construire le graphe d'adjacence, nous pouvons utiliser deux méthodes : pour chaque point x_i de X , soit chercher les k plus proches voisins x_1, \dots, x_k de x_i , soit, utiliser la distance euclidienne pour

trouver l'ensemble des points x_j situés dans un certain rayon r (en effet, pour des points voisins, la distance euclidienne fournit une approximation juste de la distance géodésique). Nous représentons ensuite les relations de voisinage par un graphe G : les noeuds sont les points x_i , et le poids de l'arête qui relie x_i à un point de son voisinage correspond à la distance euclidienne entre ces deux points (par défaut le poids de l'arête qui relie deux points ne faisant pas partie du même voisinage est fixé à ∞). Nous supposerons au préalable que notre structure géométrique est homogène, c'est-à-dire qu'il n'y a pas de groupes de points isolés.

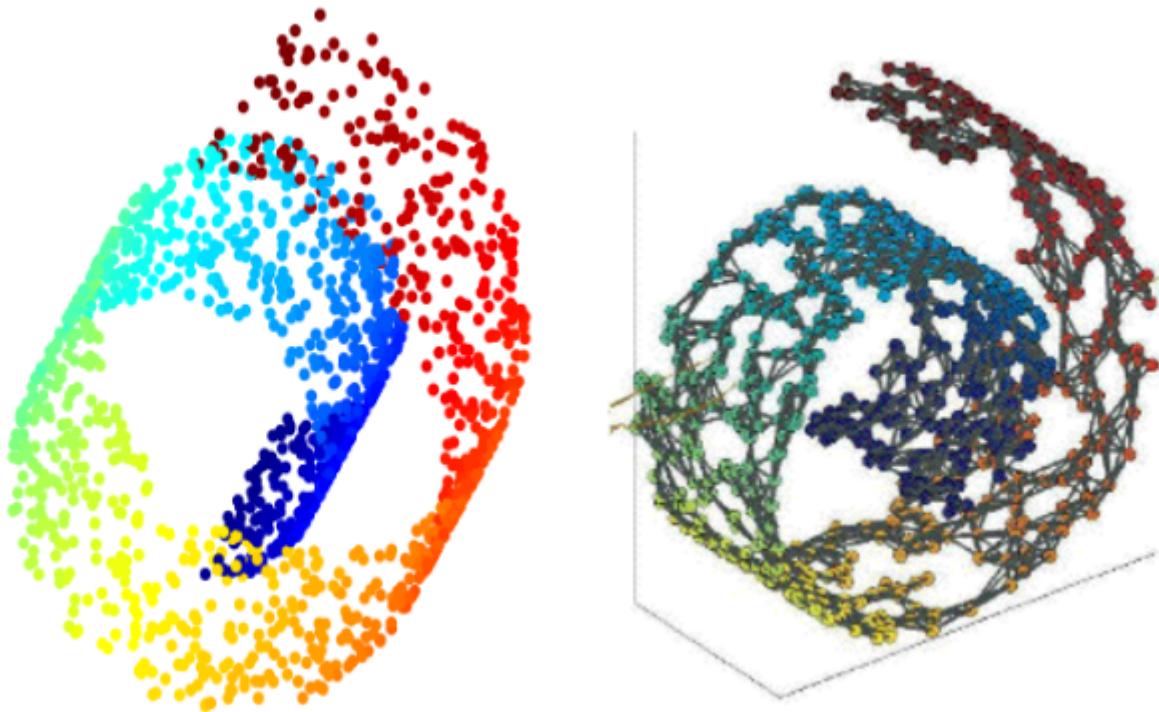


FIGURE 2.5 – Chaque point de la surface de ce *swiss-roll* est relié à ses k plus proches voisins (reproduit à partir de [10]).

Étape 2 : calcul des distances géodésiques

L'algorithme va calculer la distance géodésique $d_G(i, j)$ pour chaque paire de points (x_i, x_j) , d'après le graphe G , en appliquant un algorithme de recherche du chemin le plus court, comme par exemple l'algorithme **Dijkstra** [11]. Cela permet ainsi de construire la matrice des distances géodésiques D_G contenant tous les $d_G(i, j)$.

Étape 3 : réduction de dimension

Nous appliquons **MDS** (*Multidimensional Scaling* ou *Positionnement multidimensionnel*) à partir de la matrice D_G pour se ramener à un espace Y de dimension inférieure d qui préserve le plus la géométrie du nuage de points original. Le positionnement multidimensionnel consiste à construire une matrice B à partir d'une matrice de distance (ici D_G) par **double centrage**, c'est à dire une matrice de la forme

$$B = \left(I - \frac{1}{n}J\right)D_G^2\left(I - \frac{1}{n}J\right) \quad (2.5)$$

où J est une matrice de taille $n \times n$ ne contenant que des uns. Par **décomposition en éléments propres** de B , nous récupérons les d plus grandes valeurs propres $\lambda_1, \dots, \lambda_d$ et v_1, \dots, v_d les vecteurs propres associés. Enfin, nous calculons la matrice

$$M = \Lambda_d^{1/2}V_d^T \quad (2.6)$$

où Λ_d est la matrice diagonale contenant les valeurs propres $\lambda_1, \dots, \lambda_d$ et V_d^T la matrice des vecteurs propres v_1, \dots, v_d transposée. Les colonnes de M constituent ainsi les nouvelles coordonnées y_1, \dots, y_n de nos points de départ dans l'espace Y .

C) Hyperparamètres

Lors de l'utilisation de la méthode ISOMAP, l'utilisateur doit spécifier manuellement à l'algorithme un **hyperparamètre** : le nombre k de plus proches voisins. Il est important de l'optimiser. En effet, si le choix de k nous donne un voisinage trop large, alors ce dernier risque d'inclure des points d'une autre partie de l'ensemble de points, dont la distance géodésique par rapport aux points du voisinage est potentiellement importante, et cela risque de causer des erreurs lors de la réduction de dimension (car les distances seront mal estimées). Au contraire, si k est trop petit, alors nous risquons de nous retrouver avec un ensemble fragmenté en plusieurs groupes de points déconnectés. Il existe plusieurs algorithmes pour optimiser k , comme par exemple l'algorithme **3NN Natural Nearest Neighbor**. [12]

2.2.1.2 PCA

A) Principe général

Le but de l'ACP [13] (*Analyse en Composantes Principales*, *PCA* en anglais) est de maximiser la variance. Plus précisément l'analyse en composante principale

va partir de variables corrélées entre elles, pour obtenir des variables non corrélées (composantes principales ou axes principaux) et ainsi éliminer l'information redondante. Les variables utilisées sont donc toutes quantitatives. Les nouvelles composantes synthétisant l'information seront rangées par ordre croissant de leur contenu en information. L'information considérée ici est l'inertie. La corrélation, mesure de liaison entre deux variables, est un point clé de cette méthode. On peut obtenir l'importance des axes par leur proportion de l'inertie associé, calculé à partir des valeurs propres. La notion de réduction de dimension intervient lors du choix du nombre de composantes pour obtenir la représentation la plus fidèle possible de l'espace d'origine.

B) Mathématiques

La problématique revient à trouver une matrice de rang donné la plus proche possible de la matrice originale. Lors de cette méthode le but est de projeter orhtogonalement le nuage de points dans un sous-espace de dimension réduite avec une représentation fidèle des distances initiales. On va alors décomposer la matrice X par **SVD** [14] :

$$X = U\Delta V = \sum_i s_i u_i v_i^t. \quad (2.7)$$

Ainsi, on cherchera les valeurs et les vecteurs propres de la matrice de variance-covariance $X^t D X$ où D est la métrique de l'espace des individus. On aura ainsi le sous espace $Vect(v_1, \dots, v_s)$ qui correspond au sous-espace de dimension s optimisant l'inertie projetée. L'inertie de ce sous-espace est :

$$s_1^2 + \dots + s_s^2. \quad (2.8)$$

Les projections sur v_s ,

$$F_s = XQu_s \quad (2.9)$$

correspondent aux composantes principales et permettent ainsi d'obtenir les données représentées dans un nouvel espace où les composantes contiennent l'information de façon décroissante.

C) Utilisation pratique

Pour cette méthode, les choix se font essentiellement après son application. En effet, de nombreuses sorties nous indiquent la qualité de la réduction de dimension. Ainsi, on peut habituellement avoir les valeurs de variances expliquées par axes

ainsi que les valeurs propres pour pouvoir définir les axes les plus importants. Si les valeurs de contributions et de cos2 (angle de projection sur l'axe) ne sont pas toujours disponibles directement dans les sorties de la méthode implémentée, il est nécessaire de les retrouver par le calcul. Par exemple, la qualité de représentation globale est décrite par partie d'inertie expliquée par l'axe de projection. L'inertie expliquée par les k premiers axes factoriels est donnée par :

$$\frac{\sum_{s=1}^k \lambda_s}{\sum_{s=1}^p \lambda_s}, \quad (2.10)$$

et la qualité de représentation, ou \cos^2 , d'un individu ou d'une variable suivant un axe ou un plan sa part d'inertie projetée, est donnée par :

$$\cos^2 = \frac{F_s^2(i)}{\sum_{s=1}^p F_s^2(k)}. \quad (2.11)$$

2.2.1.3 t-SNE

A) Principe

t-SNE (ou t-Stochastic Neighbour Embedding [15] [16]) fait partie des méthodes se concentrant sur le traitement des "manifolds", variétés géométriques (décris dans Isomap) intrinsèques aux données traitées. Elle permet de gérer le fait que deux points peuvent être proche selon la distance euclidienne classique, mais éloignés au regard de la variété. Cette méthode se concentre sur la structure locale des données et utilise les distributions de lois de probabilité comme mesure de liaison entre les points de telle sorte que des points proches dans l'espace original auront une forte probabilité d'être proche dans l'espace d'arrivée.

B) Mathématiques

Cette méthode consiste, pour chaque point, à associer aux autres points un poids gaussien selon la distance qui les sépare du point choisi. On note ces poids $p_{i|j}$, calculés de la manière suivante. On mesure la similarité entre deux points x_i et x_j de l'espace original :

$$p_{i|j} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k=1}^N \exp(-\|x_i - x_k\|^2 / 2\sigma_i^2)} \quad (2.12)$$

où N est le nombre d'individus, $k \in 1, \dots, N \neq i$ et nous verrons le calcul de σ_i plus loin. On voit facilement que plus les x_j sont éloignés de x_i plus la probabilité

devient faible. Une fois tous ces $p_{i|j}$ calculés, on cherche à construire une nouvelle distribution Q dans un espace de dimension inférieure, de sorte que les $q_{i|j}$ minimisent la divergence de Kullback-Leibler. La mesure de similarité dans le nouvel espace est définie par

$$q_{i|j} = \frac{\exp(-\|x_i - x_j\|^2)}{\sum_{k=1}^N \exp(-\|x_i - x_k\|^2)}. \quad (2.13)$$

Une fois ces calculs effectués on mesure la divergence de **Kullback-Lieber** [17]

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{i|j} \log \frac{p_{i|j}}{q_{i|j}} \quad (2.14)$$

que l'on minimise ensuite à l'aide d'une méthode d'optimisation (descente du gradient par exemple).

Le calcul de σ_i introduit la notion de perplexité. Il est nécessaire de calculer σ_i pour chaque x_i car on ne peut pas utiliser le même σ pour tous les points en raison des différences de densité de voisinage selon le point considéré. Une densité élevée à proximité du point aura une variance plus faible. Cette notion se rapproche des k plus proches voisins, et se calcule de la façon suivante :

$$\text{Perplexity}(P_i) = 2^{H(P_i)} \quad (2.15)$$

où $H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}$. On reconnaît l'entropie de Shannon, vu précédemment dans la section *Random Forest*, ici calculée pour la distribution de probabilité. L'algorithme t-SNE effectue alors une recherche dichotomique pour σ_i qui produit une distribution P_i avec la perplexité spécifiée. Les valeurs typiques de la perplexité que préconisent les auteurs de la publication [15] se situent entre 5 et 50.

C) Déroulement de l'algorithme

Algorithm 1 Algorithme t-SNE

Calcul de similarités pour chaque combinaison de points $x_i x_j$ selon la perplexité choisie.

Calcul de la matrice de similarité symétrique : $p_{ij} = \frac{p_{i|j} + p_{j|i}}{2N}$

Choix aléatoires d'une solution Y dans l'espace d'arrivée

Calcul de similarités pour chaque combinaison de points $x_i x_j$ selon la perplexité choisie.

while Y suffisamment proche de X selon la divergence de Kullback Liebler ou que le nombre d'itérations inférieur au seuil. **do**

 Calcul la matrice similarité pour Y

 Utiliser la descente du gradient, où la divergence de Kullback Liebler est utilisée comme fonction de coût, pour trouver un nouveau Y

end while

Outre le choix de la perplexité, on note aussi le choix de la métrique pour le calcul des poids. Lors d'une utilisation de l'algorithme, il est possible d'accéder à la valeur finale de divergence de **Kullback-Lieber** [17], ce qui donne un indicateur de la qualité de transformation du jeu de données. Les algorithmes qui utilisent t-SNE utilisent aussi des algorithmes d'approximation comme **Barnes-Hut** [18] par exemple. Bien que dispensables, ces algorithmes permettent de diminuer le temps de calcul.

2.2.2 Apprentissage supervisé

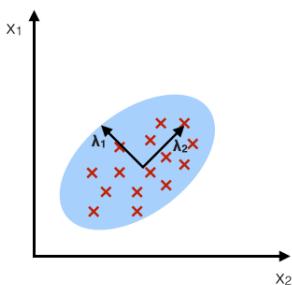
2.2.2.1 LDA

A) Principe

L'analyse discriminante linéaire (*LDA* : *Linear Discriminant Analysis* [19]) est une méthode statistique développée par Fisher et qui permet la réduction de dimension ainsi que la classification d'objets ou d'individus d'un tableau de données. Son objectif est de chercher les combinaisons linéaires des variables de ce tableau permettant de séparer au mieux les individus. Très similaire à la PCA, la particularité de cette méthode est qu'elle va, en plus de chercher les axes principaux qui maximisent la variance entre les individus, s'intéresser aux axes maximisant la variance entre les classes auxquelles appartiennent les individus (la variance *inter-classes*) et minimisant la variance entre les individu d'une même classe (la variance *intra-classe*) : on cherche donc une description des individus qui conduise à la meilleure discrimination entre les classes. Nous pouvons voir sur l'exemple suivant (figure 2.6) les objectifs respectifs des méthodes PCA et LDA :

PCA:

component axes that maximize the variance



LDA:

maximizing the component axes for class-separation

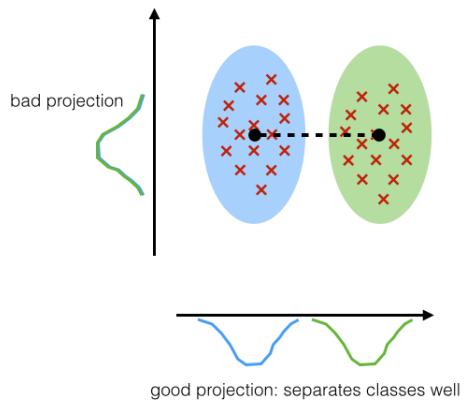


FIGURE 2.6 – Comparaison PCA/LDA (reproduit à partir de [20]).

Sur la figure de droite, nous constatons qu'effectuer une projection sur l'axe horizontal nous permettrait de bien séparer nos individus selon la classe à laquelle ils appartiennent (bleue ou verte). Au contraire, une projection des points sur l'axe vertical serait une mauvaise idée, car il n'y aurait aucun moyen de connaître leur classe.

B) Explication mathématique

Le principe général de la méthode est de construire une première variable, que nous appellerons variable discriminante, comme combinaison linéaire des variables initiales. Celle-ci devra minimiser la variance intra-classe et maximiser la variance inter-classes (nous revenons sur ces notions dans l'étape 2 de l'explication). La variable discriminante suivante sera construite de manière non corrélée à la première tout en vérifiant ces conditions, et ainsi de suite. Une analyse discriminante linéaire se déroule en 4 étapes.

Étape 1 : calcul des centres de gravité des classes

Considérons un échantillon constitué de n individus appartenant à des classes $1, \dots, q$ sur lesquels sont mesurées p variables quantitatives, dans un tableau que l'on note X . Les n individus sont représentés par des points $X_i^T = (x_i^1, \dots, x_i^p)$ dans l'espace \mathbb{R}^p pour $i = 1, \dots, n$. Nous commençons par calculer les moyennes m_i des individus dans les groupes (c'est à dire les centres de gravité des nuages constitués

des individus de la classe k avec $k = 1, \dots, q$), pour chaque variable quantitative. Nous déterminons ensuite la moyenne des valeurs de chaque individu, pour chaque classe. À la fin, nous obtenons q vecteurs, contenant chacun p éléments :

$$m_1 = \begin{pmatrix} \mu_{1(Var_1)} \\ \vdots \\ \mu_{1(Var_p)} \end{pmatrix}, \dots, m_q = \begin{pmatrix} \mu_{q(Var_1)} \\ \vdots \\ \mu_{q(Var_p)} \end{pmatrix} \quad (2.16)$$

avec μ_k les moyennes des individus de la classe k pour chaque variable quantitative.

Étape 2 : calcul des matrices de variance-covariance

Il faut ensuite calculer la matrice de variance-covariance S_k de taille $p \times p$ des nuages de points de chaque classe à partir des centres de gravité m_k calculés précédemment,

$$S_k = \sum_{i \in I(k)} \frac{p_i}{p_k} (X_i - m_k)(X_i - m_k)^T \quad (2.17)$$

où p_i est le poids d'un individu i , p_k le poids la classe k et $I(k)$ représente l'ensemble des indices i des individus de la classe k . Cela nous permet d'obtenir :

- la matrice de variance-covariance **inter-classes** (*between*, i.e. entre les centres de gravités des classes) de taille $p \times p$,

$$B = \sum_{i=1}^q p_k (m_k - m)(m_k - m)^T \quad (2.18)$$

où $m = \sum_{i=1}^n \frac{1}{n} X_i$ est le centre de gravité du nuage de points.

- la matrice de variance-covariance **intra-classes** (*within*, i.e. entre les individus de chaque classe) de taille $p \times p$,

$$W = \sum_{i=1}^q p_k S_k. \quad (2.19)$$

Étape 3 : décomposition en éléments propres de $W^{-1}B$

Nous réalisons la décomposition en éléments propres de la matrice $W^{-1}B$. Nous en déduisons p valeurs propres $\lambda_1, \dots, \lambda_p$ et les vecteurs propres associés v_1, \dots, v_p .

Étape 4 : choix des variables discriminantes

Nous souhaitons projeter nos données dans un sous-espace où les individus seront le plus possible séparés selon leur classe, mais également réduire la dimension de notre espace de variables. Les axes du nouvel espace de variables seront représentés par nos vecteurs propres. Cependant, ces derniers définissent seulement les directions des axes, il faut donc s'intéresser aux valeurs propres correspondantes pour savoir quels vecteurs propres portent le plus d'information, et ainsi savoir lesquels nous allons garder pour la suite. Nous allons donc trier les valeurs propres $\lambda_1, \dots, \lambda_p$ dans l'ordre décroissant : les vecteurs propres associés aux valeurs propres les plus grandes sont ceux qui contiendront le plus d'information.

Nous construisons ensuite la matrice V de taille $p \times d$ composée des d vecteurs propres avec le plus d'information, et que nous gardons pour construire nos variables discriminantes. En effet, soit F_D notre première variable discriminante. Alors comme cette variable est une combinaison linéaire de nos variables initiales, elle est de la forme

$$F_D = Xa \quad (2.20)$$

où

$$X = (X_1, \dots, X_p) \quad (2.21)$$

correspond à nos variables quantitatives et a à un vecteur colonne, appelée la *fonction linéaire discriminante*. Cette fonction correspond en fait à un vecteur propre calculé précédemment : le vecteur propre contenant le plus d'information, qui est donc la première fonction discriminante. La première variable discriminante est finalement donnée par

$$F_D = a_1 X_1 + \dots + a_p X_p \quad (2.22)$$

avec a_1, \dots, a_p les coefficients du vecteur a . On procède de même pour avoir les fonctions discriminantes suivantes. Enfin, nous obtenons par l'opération

$$Y = X \times V \quad (2.23)$$

les nouvelles coordonnées des n individus dans le nouvel espace de variables Y de dimension d .

Chapitre 3

Réduction de dimension sur des données biologiques

3.1 Introduction des données

Dans le but d'analyser la germination, de la semence sèche jusqu'à la percée de la radicule, on utilise l'imagerie automatisée sur des semences semées sur des Tables de Jacobsen (bancs de germination). Les essais de germination sont menés à 5°C dans un module climatique sur 2 tables de Jacobsen. Les sondes de température de régulation et surveillance des essais ont été étalonnées avant le début de l'expérimentation. Les 2 tables de Jacobsen ont été cartographiées pour la température en 5 points. Chaque table dispose de 4 caméras sous lesquels sont semées 600 semences par caméra par zone de 5x5 semences. Il y a 24 zones par série d'image et 25 semences par zone. Les zones et semences sont codés « ligne-colonne ». Trois semis ont été réalisés en parallèle sur les 2 tables dans l'ordre des populations. Les images acquises pendant 21 jours ont été analysées et à partir des mesures sur images, l'heure de germination a été détectée sur chaque semence, en plus de plusieurs caractéristiques complémentaires. Une étude statistique de la croissance des plantules (fig 3.1 et 3.2) a ainsi été réalisée dans le but d'analyser si des lots de plantules au patrimoine génétique ou avec des caractéristiques germinatives mesurées par ailleurs se distinguent dans cette phase de développement.

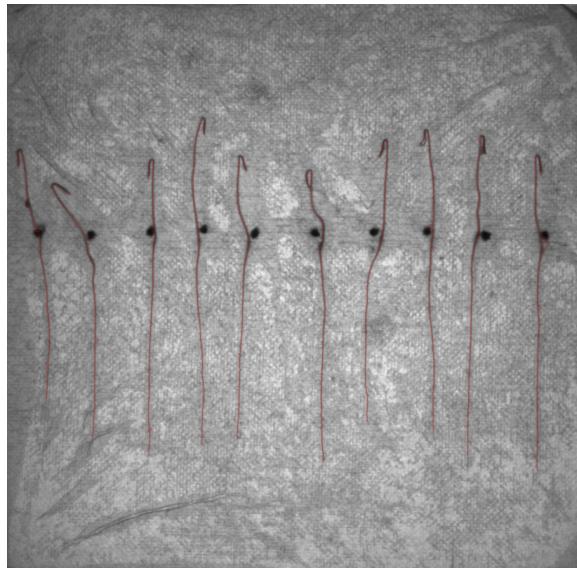


FIGURE 3.1 – Dix plantules à un stade de développement donné [21].

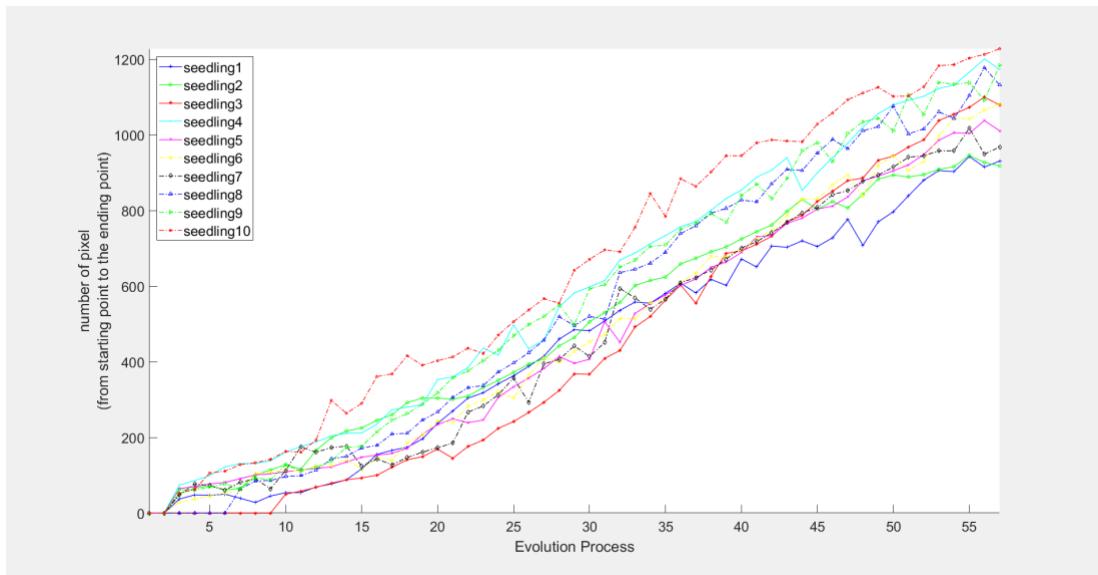


FIGURE 3.2 – Évolution de la longueur des dix plantules au cours du temps [21].

Nous nous sommes particulièrement intéressés à l'étude de la croissance des plantules au cours des 21 premiers jours prévue par le projet (fig 3.3), après les semences, en fonction de plusieurs variables : quantitatives (ex : vitesses de croissance entre les jours 15 et 21, etc) et qualitatives (Bancs, Population, Echantillon, Caméra, Semis et Zone).



AKER WP4 Phenotyping

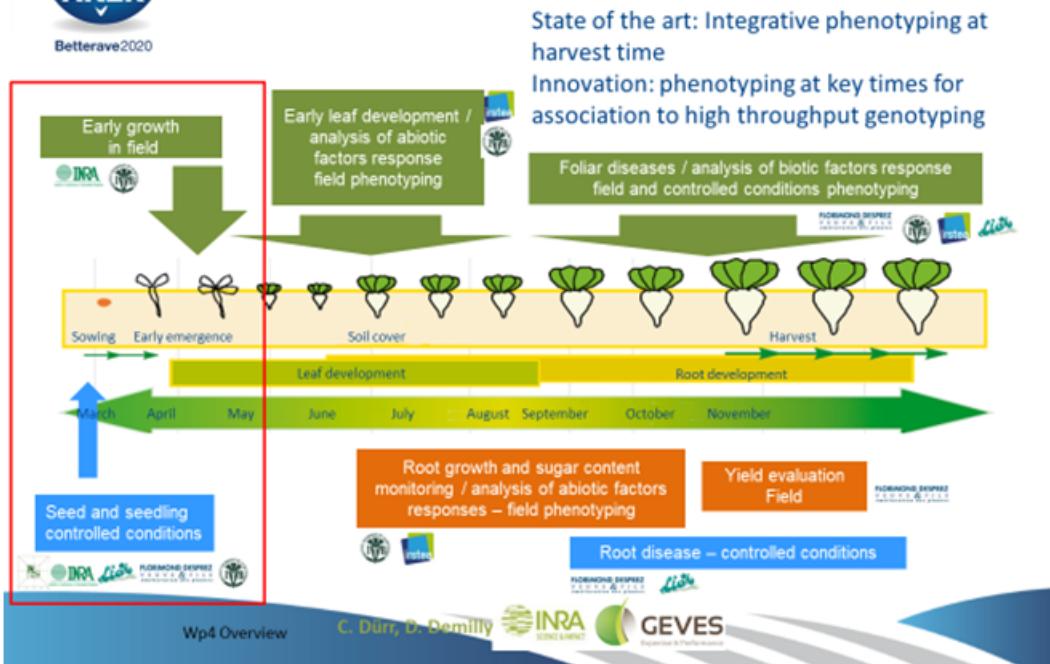


FIGURE 3.3 – Dans l’encadré rouge figure la partie du projet où les données que nous étudions ont été acquises [21].

3.2 Expertise

La première tâche à réaliser avant de travailler sur les données au travers des différentes méthodes de réduction de dimension est l’expertise, c’est à dire la préparation, le nettoyage des données. Nous avons réfléchi avec un point de vue différent de celui du data scientist : qu’est-il bon de garder ou d’écartier dans le jeu de données, faut-il rajouter des variables explicatives, et pourquoi. Comme nous nous intéressons à l’évolution des germinations jour par jour, nous avons écarté les variables qui s’exprimaient en heures, c’est à dire les variables **5°C TMG (h)** et **5°C T50 (h)**, et gardé celles qui s’exprimaient en jours, **5°C TMG (j)** et **5°C T50 (j)**. Cependant, cette dernière comprenait un nombre important d’entrées "Nan" (plus de la moitié), ce qui signifie qu’il n’a pas été possible de calculer le délai nécessaire pour obtenir 50% de germination pour plus de la moitié des individus. Nous l’avons donc également écartée car elle présentait peu d’intérêt dans notre étude. Nous avons ensuite rajouté 6 variables '**v15-16j**', '**v16-17j**', '**v17-18j**',

'v18-19j', 'v19-20j', et 'v20-21j', à partir des variables '15j', '16j', '17j', '18j', '19j', '20j', '21j' et 'Aire sous la courbe' (que nous avons ensuite écartées de notre étude, car elles n'apportaient pas d'information utile), pour exprimer la vitesse de germination de chaque individu jour après jour à partir du jour 15.

3.3 Résultats

Les premiers résultats ci-dessous, obtenus grâce aux codes disponibles sur le github [22], illustrent graphiquement l'application de différentes méthodes de réduction de dimension sur nos données, pour la variable qualitative "camera". En effet lors de l'étude effectuée sur la germination des plantules de betterave, les deux tables de Jacobsen disposaient de plusieurs caméras, qui sont ici étiquetées de 0 à 3. Nous remarquons que tous les points sont mélangés quelle que soit la méthode utilisée : nous ne retrouvons pas de groupe de points appartenant à la même classe. Il n'y a donc pas vraiment de séparation entre les individus dont la germination était étudiée par la caméra 0 et ceux dont la germination était étudiée par la caméra 2 par exemple. Par conséquent, ces résultats ne nous permettent pas de conclure quant à l'impact des caméras utilisées sur les plantules.

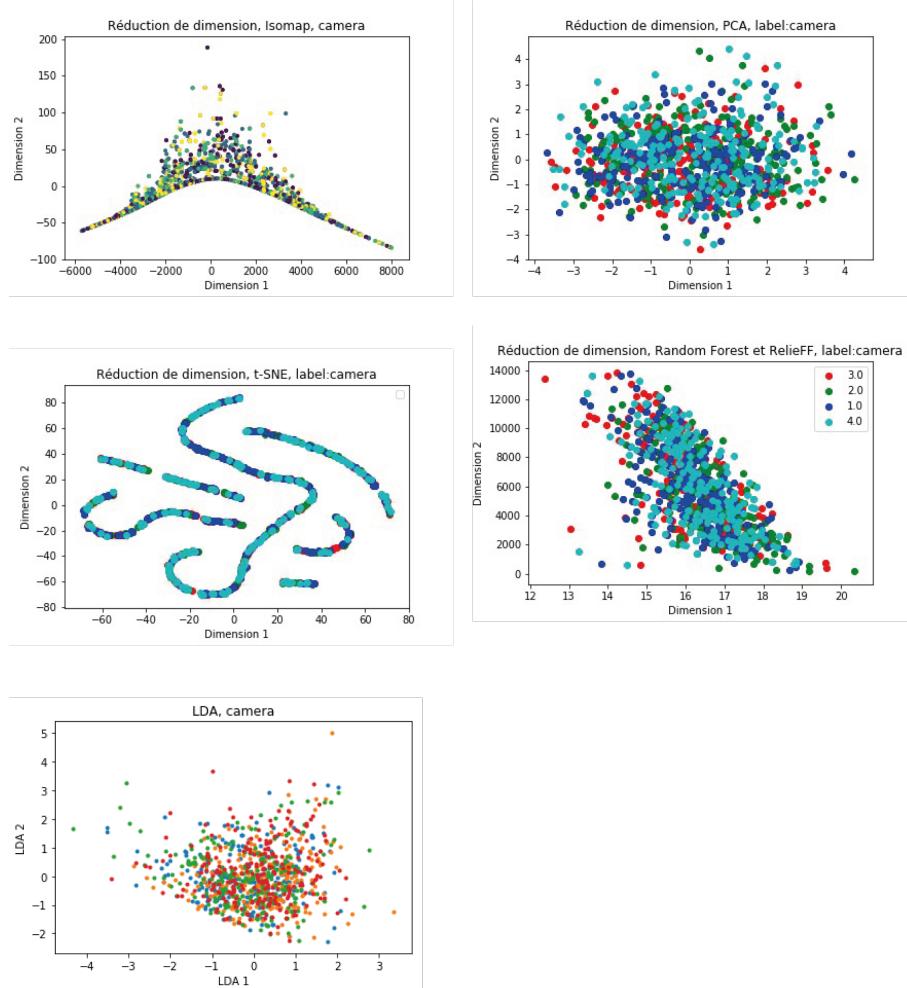


FIGURE 3.4 – Résultats obtenus avec les méthodes de réduction de dimension, pour la variable qualitative "camera" [22].

Intéressons-nous aux résultats pour la variable "Bancs" sur la figure ci-dessous. Ici les classes correspondent à 0 et 1, puisque les essais de germination ont été menés sur deux bancs de germination. Là encore, rien ne nous permet d'affirmer qu'un banc a eu plus d'influence que l'autre sur la germination des plantules, puisque tous les individus sont mélangés sur les graphiques. Nous noterons que pour deux classes, la méthode LDA ne nous permet qu'une représentation en 1 dimension, ce qui n'est pas très pertinent pour l'interprétation dans notre cas.

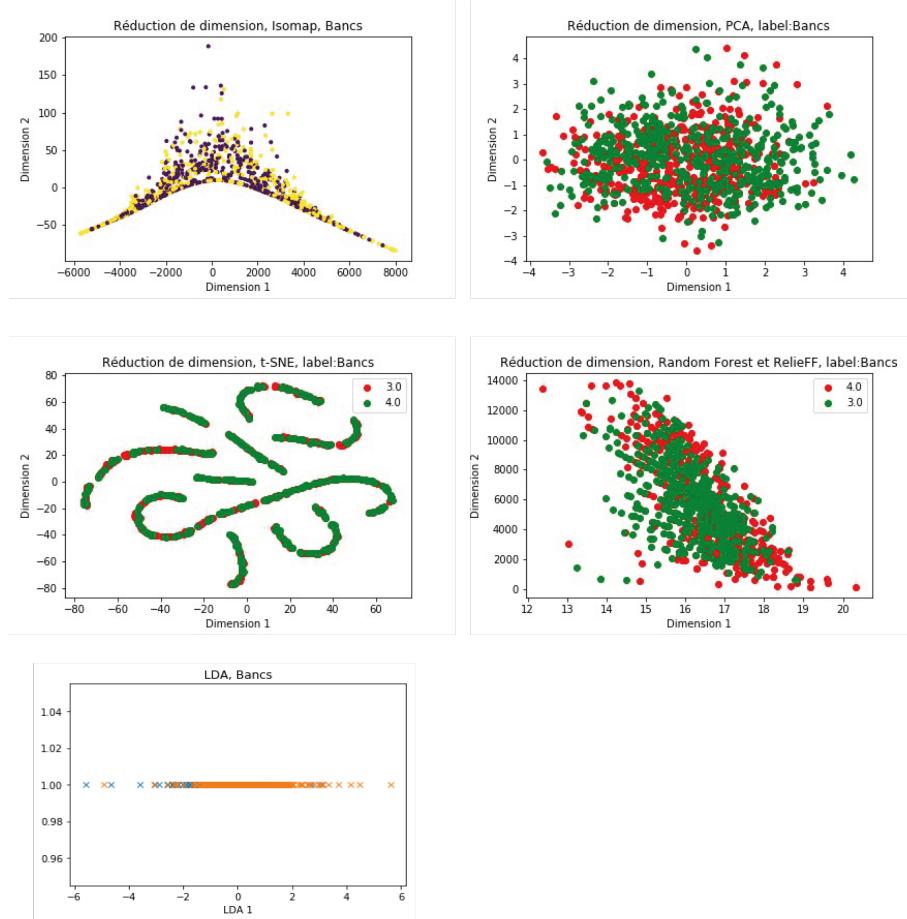


FIGURE 3.5 – Résultats obtenus avec les méthodes de réduction de dimension, pour la variable qualitative "Bancs" [22].

On note de plus que les résultats obtenus pour t-SNE ne sont pas très pertinents. En effet, en plus de ne pas séparer les groupes, les résultats ne sont pas stables selon le point de départ et ce malgré différentes valeurs de paramètres (perplexité et métrique confondus) et la divergence de Kullback-Leibler n'a pas pu être abaissée en dessous de 0.20, ce qui est très élevé. Les graphes pour la méthode Random Forest et pour la méthode Relief sont les mêmes car les variables sélectionnées, à savoir 'Aire sous la courbe' et 'Aire sous la courbe' sont les mêmes.

Chapitre 4

Organisation du projet

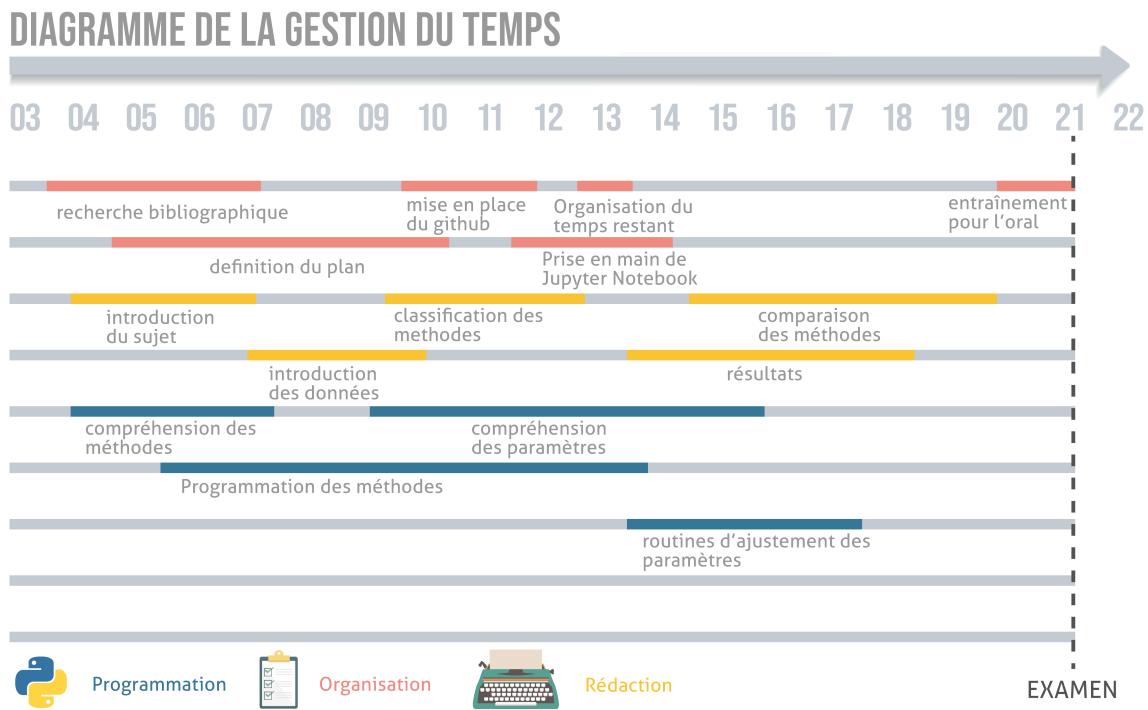


FIGURE 4.1 – Diagramme synthétisant l’organisation du projet.

Nous avons d’abord commencé par une recherche bibliographique (en essayant de trouver les publications d’origine ou, à défaut, les publications les plus citées) afin de comprendre ce qui différenciait les méthodes présentées par le sujet. Nous avons ensuite appréhendé les données, pour pouvoir appliquer de façon pertinente les méthodes découvertes plus tôt. Dans un souci d’organisation, nous avons choisi d’utiliser un logiciel de gestion de version, *git*, avec un hébergement sur *github*

(<https://github.com/etudiantdatascience>), pour que chacun puisse participer au projet tout en gardant un œil sur l'avancée de l'autre. Dans cette même optique nous avons transféré nos codes au format *Jupyter Notebook*, pour pouvoir consulter nos codes et résultats de manière simple et rapide. Durant cette période, une grande partie du temps de travail a été consacrée à la compréhension des paramètres des fonctions, à la préparation des données et à d'éventuelles routines d'ajustement lorsque cela était possible. Nous avons ensuite pu rendre compte des résultats selon les méthodes. Enfin, nous avons pu terminer la rédaction du rapport en *LateX*, bien qu'une rédaction régulière a eu lieu au cours des semaines de recherches et de programmation des méthodes.

Chapitre 5

Conclusion

Dans l'objectif de visualiser les données biologiques pour observer un éventuel regroupement des variables qualitatives intrinsèques au protocole d'analyse de la germination, nous avons cherché à réduire la dimension du jeu de données selon différentes méthodes d'apprentissage. Ces méthodes nous ont permis de ne constater aucune différence visuelle, qui aurait mené à un ajustement direct du protocole expérimental. Pour aboutir à une solution satisfaisante, il va donc être nécessaire d'effectuer des tests statistiques, pour conclure ou non à une différence significative dans les résultats causée par un biais expérimental. On notera aussi que, malgré des différences dans les méthodes étudiées, celles-ci comportent toutes leurs spécificités. Que ce soit dans leur utilisation préconisée ou dans leurs avantages, les résultats ont tous été semblables pour l'ensemble des méthodes.

Bibliographie

- [1] E. Biernat, M. Lutz, and Y. LeCun. *Data science : fondamentaux et études de cas : Machine learning avec Python et R.* Eyrolles, 2015.
- [2] Christopher J. C. Burges. Dimension reduction : A guided tour. *Foundations and Trends® in Machine Learning*, 2(4) :275–365, 2010.
- [3] Sébastien Guérif. *Réduction de dimension en apprentissage numérique non supervisé*. PhD thesis, 2006. Thèse de doctorat dirigée par Bennani, Younès Informatique Paris 13 2006.
- [4] Leo Breiman. Random forests. *Mach. Learn.*, 45(1) :5–32, October 2001.
- [5] Philipp Probst, Anne-Laure Boulesteix, and Marvin Wright. Hyperparameters and tuning strategies for random forest, 04 2018.
- [6] Kenji Kira and Larry A. Rendell. A practical approach to feature selection. In *Proceedings of the Ninth International Workshop on Machine Learning*, ML92, pages 249–256, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.
- [7] Sadek Benhammada. *Etude comparative de méthodes de sélection de caractéristiques en apprentissage automatique. Proposition d'une variante.* PhD thesis, 06 2007.
- [8] J. C. Langford J. B. Tenenbaum, V. de Silva. A global geometric framework for nonlinear dimensionality reduction. *SCIENCE*, jan 2002.
- [9] Comparison of pca and manifold learning. http://www.astroml.org/book_figures/chapter7/fig_S_manifold_PCA.html.
- [10] Nonlinear dimensionality reduction. http://vision.cse.psu.edu/seminars/talks/PRML/David_NDR_lecture.pdf.
- [11] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, june 1959.
- [12] Natural nearest neighbor for isomap algorithm without free-parameter. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.889.6032&rep=rep1&type=pdf>.

- [13] Alaa Tharwat. Principal component analysis - a tutorial. *International Journal of Applied Pattern Recognition*, 3 :197, 01 2016.
- [14] G. W. Stewart. On the early history of the singular value decomposition. *SIAM Review*, 35(4) :551–566, 1993.
- [15] Laurens van der Maaten and Geoffrey E. Hinton. Visualizing data using t-sne. 2008.
- [16] Laurens van der Maaten. Learning a parametric embedding by preserving local structure. In David van Dyk and Max Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 384–391, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 16–18 Apr 2009. PMLR.
- [17] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1) :79–86, 03 1951.
- [18] Josh Barnes and Piet Hut. A hierarchical $O(n \log n)$ force-calculation algorithm. *Nature*, 324 :446–449, 12 1986.
- [19] G. J. McLachlan. *Discriminant Analysis and Statistical Pattern Recognition*. 2004.
- [20] Linear discriminant analysis – bit by bit. [http://sebastianraschka.com/
Articles/2014_python_lda.html](http://sebastianraschka.com/Articles/2014_python_lda.html).
- [21] Diaporama d'introduction au sujet. [https://github.com/
etudiantdatascience/TER-reduction-dimension/tree/master/
Illustrations](https://github.com/etudiantdatascience/TER-reduction-dimension/tree/master/Illustrations).
- [22] Codes python en jupyter notebook. [https://github.com/
etudiantdatascience/TER-reduction-dimension/tree/master/
CodePython](https://github.com/etudiantdatascience/TER-reduction-dimension/tree/master/CodePython).
- [23] Isomap avec scikit.learn. [https://scikit-learn.org/stable/modules/
generated/sklearn.manifold.Isomap.html](https://scikit-learn.org/stable/modules/generated/sklearn.manifold.Isomap.html).
- [24] Analyses de données. [http://michaelbernier.github.io/IMN430/
imn430-chap03.pdf](http://michaelbernier.github.io/IMN430/imn430-chap03.pdf).
- [25] Unsupervised learning : Dimensionality reduction and visualization. [http://www.xavierdupre.fr/app/ensae_teaching_cs/helpsphinx/
notebooks/06_unsupervised_dimreduction.html](http://www.xavierdupre.fr/app/ensae_teaching_cs/helpsphinx/notebooks/06_unsupervised_dimreduction.html).
- [26] Isomap for dimensionality reduction in python. [http://benalexkeen.com/
isomap-for-dimensionality-reduction-in-python/](http://benalexkeen.com/isomap-for-dimensionality-reduction-in-python/).
- [27] Dimension reduction - isomap. [https://blog.paperspace.com/
dimension-reduction-with-isomap/](https://blog.paperspace.com/dimension-reduction-with-isomap/).

- [28] Implementing lda in python with scikit-learn. <https://stackabuse.com/implementing-lda-in-python-with-scikit-learn/>.
- [29] Linear discriminant analysis using python. <http://www.thejavageek.com/2018/04/30/linear-discriminant-analysis-using-python/>.
- [30] Visualizing multidimensional data in python. <http://www.apnorton.com/blog/2016/12/19/Visualizing-Multidimensional-Data-in-Python/>.
- [31] Using linear discriminant analysis (lda) for data explore : Step by step. <https://www.apsl.net/blog/2017/07/18/using-linear-discriminant-analysis-lda-data-explore-step-step/>.
- [32] Analyse discriminante linéaire. <http://wwwabi.snv.jussieu.fr/erocha/webthese/ADL.html>.
- [33] Analyse discriminante linéaire. https://www.researchgate.net/publication/223824802_Selection_of_the_optimal_parameter_value_for_the_Isomap_algorithm.