

# Rapport STA203

## Analyse de données de raisins secs

---

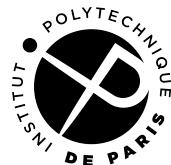
Thomas Chrétienne

Tom Cuel

24 mai 2025

# ENSTA

---



**IP PARIS**

# Table des matières

<b>Introduction</b>	<b>3</b>
<b>1 Partie I : Analyse non supervisée</b>	<b>4</b>
1.1 Analyse exploratoire . . . . .	4
1.2 Méthodologie de l'Analyse en Composantes Principales (ACP) . . . . .	8
1.3 Classification hiérarchique ascendante (CAH) . . . . .	14
1.4 Clustering hiérarchique sur les composantes principales. . . . .	17
<b>2 Partie II : Choisir un modèle</b>	<b>19</b>
2.1 Définition du modèle logistique et discussion sur le centrage-réduction . . . . .	19
2.2 Définition de l'échantillon d'apprentissage et projection sur le premier plan principal . . . . .	20
2.3 Estimation des modèles de régression logistique . . . . .	21
2.3.1 Modèle complet . . . . .	21
2.3.2 Modèle logistique sur les deux premières composantes principales . . . . .	23
2.3.3 Modèle logistique avec sélection de variables par critère AIC . . . . .	24
2.3.4 Régression logistique pénalisée (Lasso) . . . . .	26
2.3.5 Comparaison avec la régression Ridge . . . . .	28
2.4 SVM linéaire et noyau polynomial . . . . .	28
2.5 Analyse des courbes ROC et comparaison des modèles . . . . .	30
2.6 Comparaison des performances des modèles et choix final . . . . .	34
<b>3 Partie III : Un focus sur l'analyse discriminante</b>	<b>36</b>
3.1 Analyse discriminante à partir des deux premières composantes principales . . . . .	36
3.1.1 Définition du modèle et frontière de décision . . . . .	36
3.1.2 Écriture de la frontière via la diagonalisation de la matrice $\Sigma$ . . . . .	36
3.1.3 Visualisation des données et de la frontière de décision . . . . .	37
3.1.4 Évaluation de la performance sur le jeu de test et validation avec les fonctions R dédiées . . . . .	37
3.2 Courbe ROC et analyse discriminante quadratique . . . . .	38
3.2.1 Modèle PCA . . . . .	38
3.2.2 Analyse discriminante quadratique . . . . .	40
3.3 Analyse discriminante avec toutes les variables initiales . . . . .	42
<b>4 Bonus</b>	<b>45</b>
4.1 Modèle Random Forest . . . . .	45
4.2 Modèle Réseau de Neurones (nnet) . . . . .	46
4.3 Modèle Gradient Boosting (XGBoost) . . . . .	47
4.4 Résumé des performances bonus et comparaison avec les autres méthodes . . . . .	48

## Introduction

Ce rapport présente l'analyse statistique du jeu de données *Raisin.xlsx* contenant les caractéristiques morphologiques de deux variétés de raisins secs (Kecimen et Besni) cultivés en Turquie. L'objectif est d'identifier des patterns permettant de distinguer ces deux variétés à travers une analyse non supervisée puis supervisée.

Le jeu de données comprend 900 observations (450 pour chaque variété) et 7 variables quantitatives décrivant les caractéristiques morphologiques des grains. Notre analyse suivra trois étapes principales :

- Analyse non supervisée (ACP et classification hiérarchique)
- Analyse supervisée (régression logistique et SVM)
- Comparaison des méthodes et conclusion

# 1 Partie I : Analyse non supervisée

## 1.1 Analyse exploratoire

Le jeu de données est composé de 900 observations décrivant des grains de raisin selon 7 variables quantitatives (`Area`, `MajorAxisLength`, `MinorAxisLength`, `Eccentricity`, `ConvexArea`, `Extent`, `Perimeter`) et une variable qualitative `Class` indiquant la variété du raisin (`Kecimen` ou `Besni`).

**Structure des données.** Le tableau contient 900 lignes et 8 colonnes. Chaque observation correspond à une baie de raisin. La commande `str()` indique que toutes les variables quantitatives sont de type numérique et qu'il n'existe aucune valeur manquante.

**Statistiques descriptives.** Le résumé statistique (commande `summary(df)`) montre une grande disparité entre les variables. Par exemple :

- `Area` varie de 25 387 à 235 047 pixels, avec une moyenne de 87 804.
- `Eccentricity`, liée à la forme des objets, est comprise entre 0.35 et 0.96, ce qui montre une diversité morphologique notable.
- `Extent`, représentant le ratio entre l'aire de la baie et celle de son bounding box, est comprise dans un intervalle relativement resserré.

La figure 1 donne une première visualisation des distributions par variable :

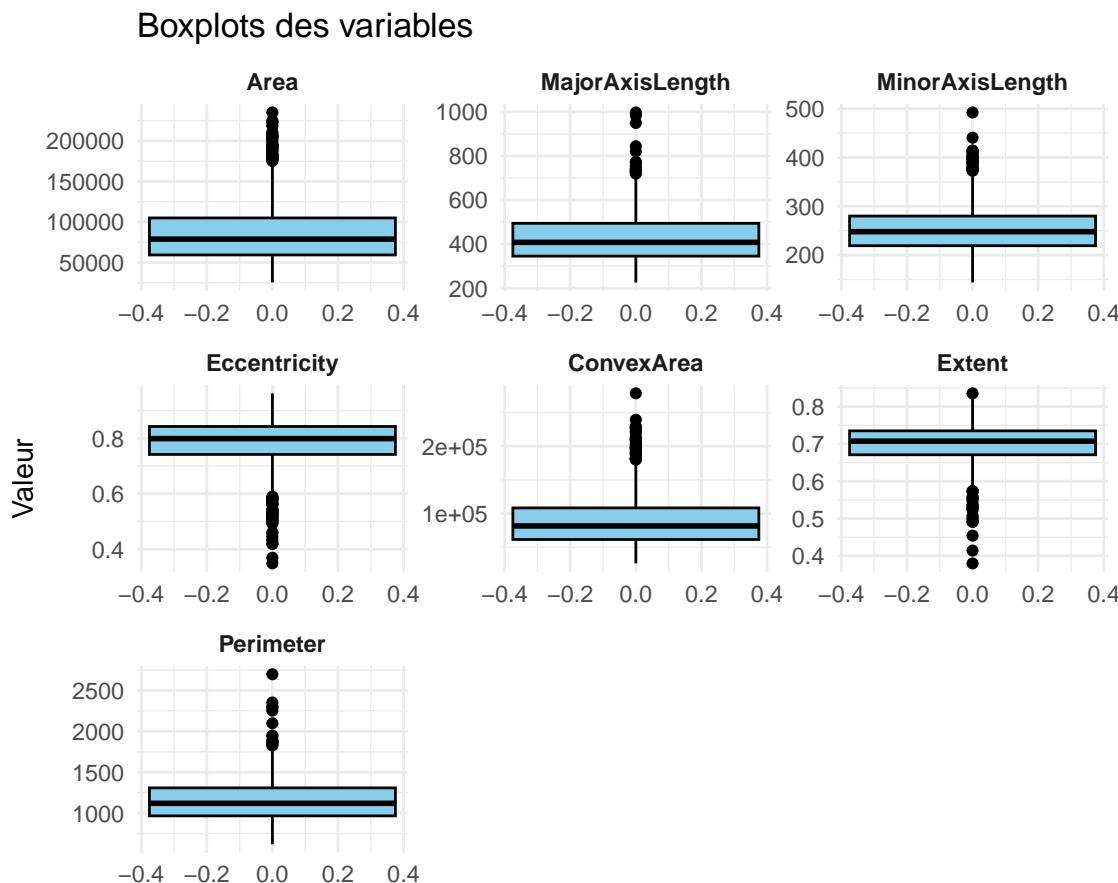


FIGURE 1 – Boxplots globaux des variables quantitatives

Les distributions sont très hétérogènes : `Area`, `ConvexArea` et `Perimeter` montrent des valeurs extrêmes, tandis que

**Extent** présente une distribution beaucoup plus resserrée. Cela indique que certaines variables ont des unités ou échelles très différentes.

**Analyse par classe.** On distingue deux classes de raisins réparties équitablement (450 observations chacune). La figure 2 compare les distributions des variables entre les deux classes.

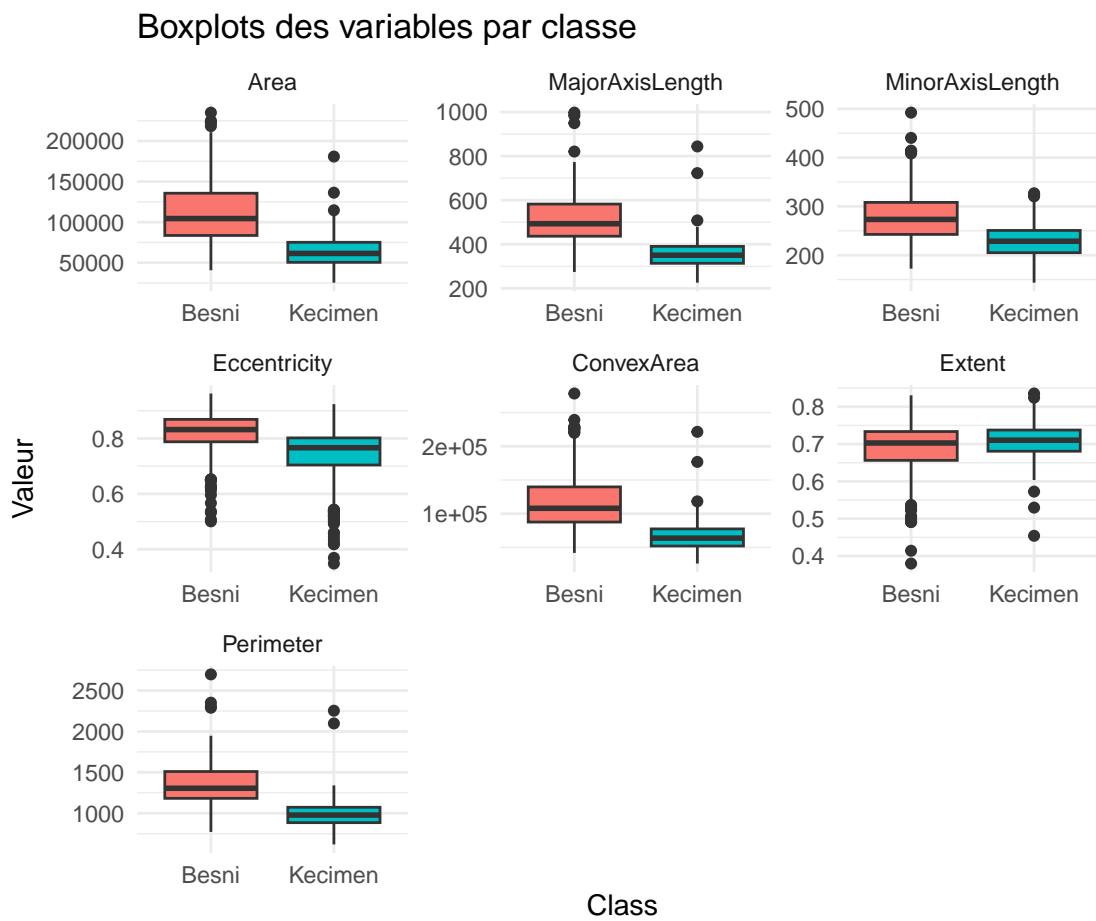


FIGURE 2 – Boxplots des variables par classe

Certaines variables montrent des différences nettes entre les classes. Par exemple :

- **Kecimen** possède généralement des valeurs plus élevées pour **Area**, **ConvexArea** et **Perimeter**.
- **Besni** semble avoir des valeurs plus faibles pour **Eccentricity** et **MajorAxisLength**.

Cela suggère que la distinction entre les deux classes pourrait être exploitée par des méthodes supervisées.

**Corrélations entre variables.** La matrice de corrélation présentée en figure 3 montre de fortes redondances :

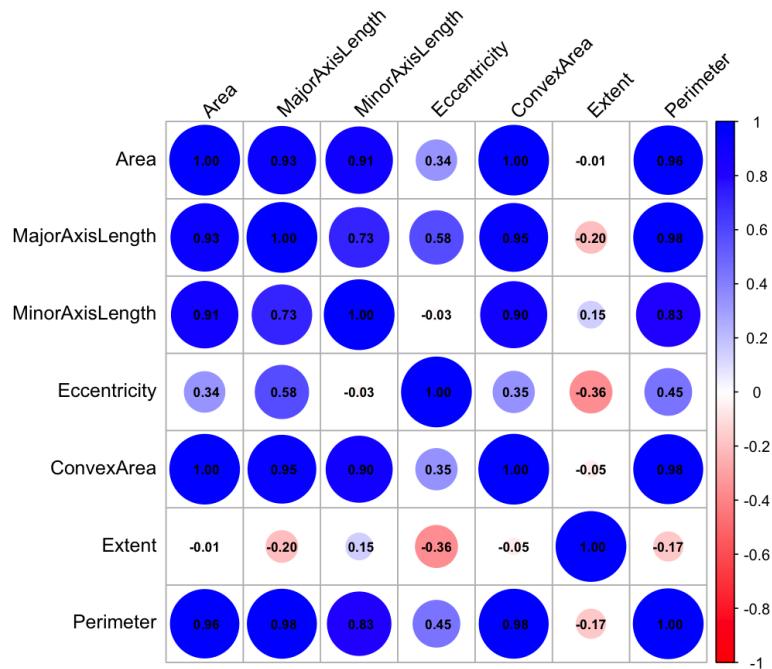


FIGURE 3 – Matrice de corrélation entre les variables

On note notamment :

- Des corrélations très fortes entre **Area**, **MajorAxisLength**, **MinorAxisLength**, **ConvexArea** et **Perimeter** ( $r > 0.95$ ).
- Une faible corrélation entre **Extent** et les autres variables.
- **Eccentricity** montre un comportement indépendant, ce qui pourrait en faire une variable discriminante.

**Visualisation bivariée.** La figure 4 présente une matrice de scatterplots avec coloration par classe, générée via `ggpairs` :

Matrice de scatterplots colorée par classe

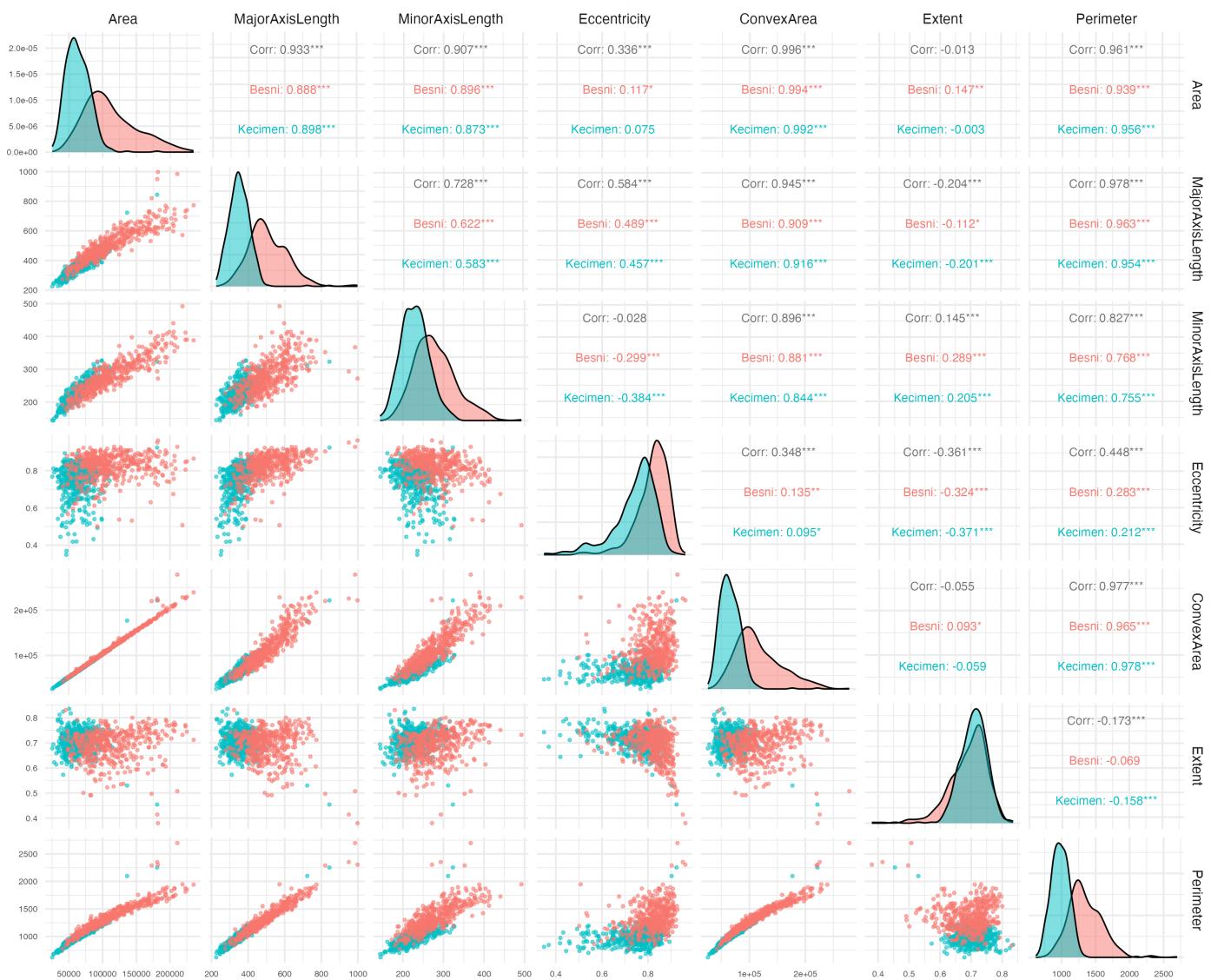


FIGURE 4 – Matrice de scatterplots colorée par classe

Certaines paires de variables montrent une bonne ségrégation des classes, notamment :

- Area vs MinorAxisLength,
- ConvexArea vs MajorAxisLength.

En revanche, d'autres paires comme celles impliquant Extent ou Eccentricity présentent des chevauchements importants.

**Alternative avec scatterplotMatrix.** La figure 5 propose une alternative générée par le package car :

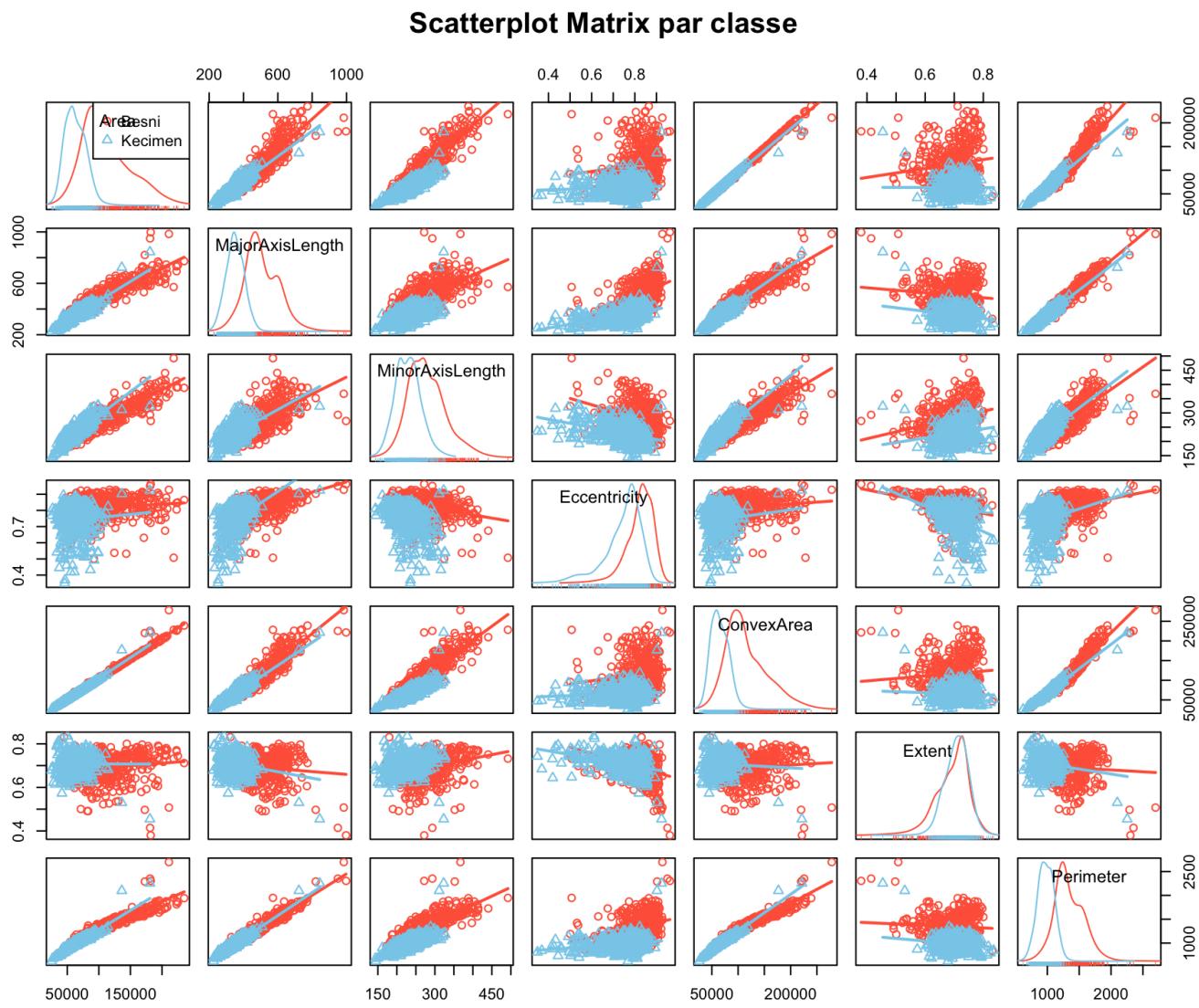


FIGURE 5 – Matrice de scatterplots (scatterplotMatrix)

Ceci confirme les corrélations linéaires fortes entre certaines variables ainsi que les tendances de séparation partielle entre classes.

**Variabilité.** Les variances calculées pour chaque variable confirment les observations visuelles :

- Très forte variance pour **Area** et **ConvexArea** (de l'ordre de  $10^9$ ).
- Faible variance pour **Eccentricity** et **Extent**, qui sont donc moins discriminantes en valeur absolue.

## 1.2 Méthodologie de l'Analyse en Composantes Principales (ACP)

Nous allons ici dérouler pas à pas la méthodologie de l'Analyse en Composantes Principales (ACP) appliquée à notre jeu de données.

**Prétraitement des données.** Avant de lancer l'ACP, les variables quantitatives ont été automatiquement centrées et réduites afin d'assurer que toutes contribuent équitablement à la construction des composantes principales, indépendamment de leurs échelles initiales, que l'on avait vu bien différentes précédemment.

**Valeurs propres et pourcentages d'inertie.** L'ACP a permis d'extraire  $p - 1 = 6$  composantes principales (le jeu de données comporte  $p = 7$  variables quantitatives). Le tableau 1 donne les valeurs propres associées à ces axes, représentant la variance expliquée par chacun.

TABLE 1 – Valeurs propres et pourcentages d'inertie cumulée

Composante	Valeur propre	% d'inertie	% d'inertie cumulée
Dim.1	4.83	69.03	69.03
Dim.2	1.45	20.76	89.79
Dim.3	0.63	8.98	98.77
Dim.4	0.06	0.81	99.58
Dim.5	0.02	0.31	99.89
Dim.6	0.01	0.09	99.99
Dim.7	0.00	0.01	100.00

La règle du coude appliquée à l'éboulis des valeurs propres (Figure 6) suggère de conserver les trois premières composantes principales, qui expliquent ensemble environ 98,8% de la variance totale. L'inertie moyenne est indiquée en rouge pointillée.

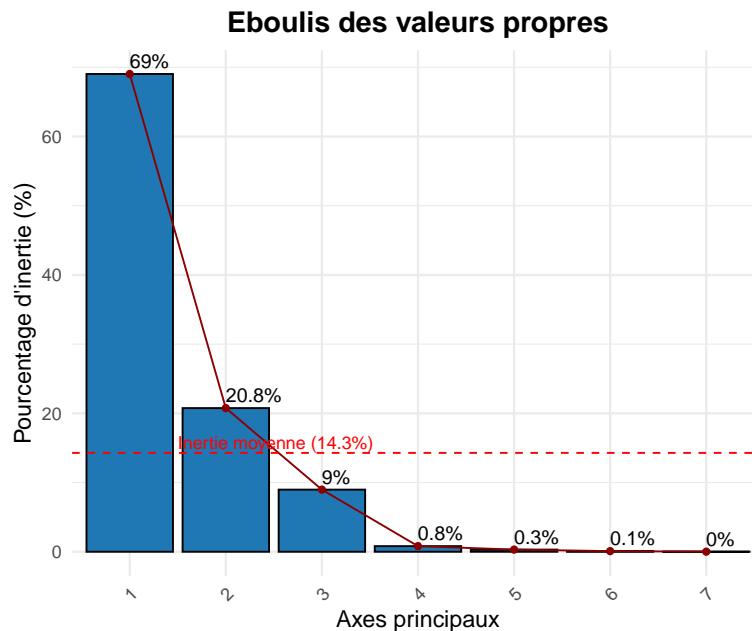


FIGURE 6 – Éboulis des valeurs propres

**Projection des individus.** Les Figures 7 et 8 présentent la projection des individus sur les plans factoriels formés respectivement par les axes (Dim 1, Dim 2) et (Dim 2, Dim 3), avec coloration selon la classe d'appartenance. Des ellipses de confiance permettent de visualiser la dispersion intra-classe.

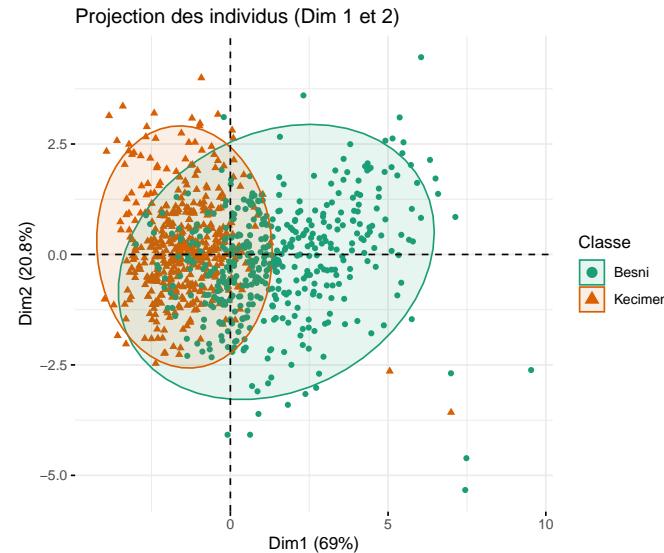


FIGURE 7 – Projection des individus selon Dim 1 et Dim 2

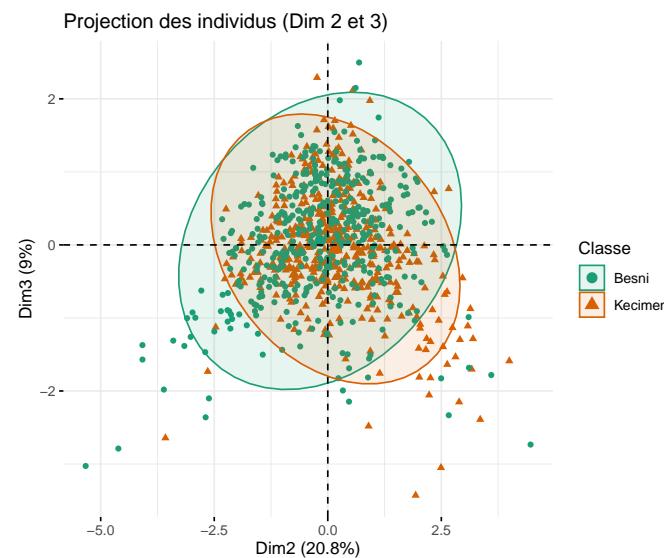


FIGURE 8 – Projection des individus selon Dim 2 et Dim 3

On observe une bonne séparation des classes selon les deux premiers axes, avec notamment une concentration des individus dans certaines zones bien distinctes. Cela justifie l'interprétation approfondie de ces deux dimensions, tandis que la séparation entre les dimensions 2 et 3 n'est pas très claire entre les différents individus.

**Cercle des corrélations.** Le cercle des corrélations permet d'étudier la représentation des variables sur les axes principaux. Les Figures 9 et 10 montrent les corrélations entre les variables et les composantes principales.

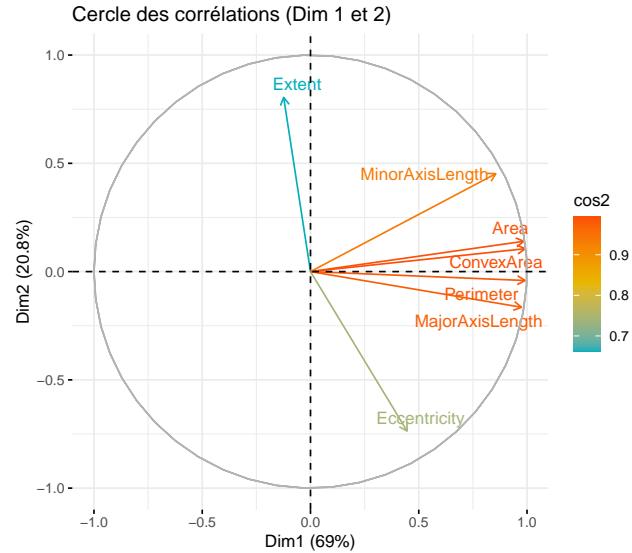


FIGURE 9 – Cercle des corrélations (Dim 1 et 2)

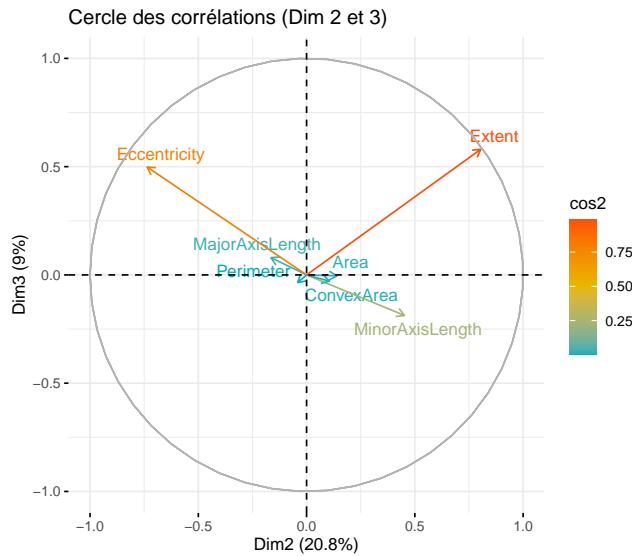


FIGURE 10 – Cercle des corrélations (Dim 2 et 3)

Les variables **Area**, **ConvexArea**, et **Perimeter** sont fortement corrélées à l'axe 1. **Extent** et **MinorAxisLength** sont mieux représentées sur l'axe 2, tandis que **Eccentricity** a une forte contribution à la troisième dimension. Ces résultats orientent l'interprétation thématique des axes :

- Dim 1 : grandeur des objets (liée aux mesures de surface et de périmètre) ;
- Dim 2 : proportion et compacité ;
- Dim 3 : forme géométrique (notamment l'excentricité).

**Synthèse des représentations.** Les graphiques combinés en Figures 11 et 12 superposent les individus et les variables dans les plans factoriels (Dim 1-2 et Dim 2-3), offrant une vue d'ensemble structurée.



FIGURE 11 – Individus et variables sur les axes 1 et 2

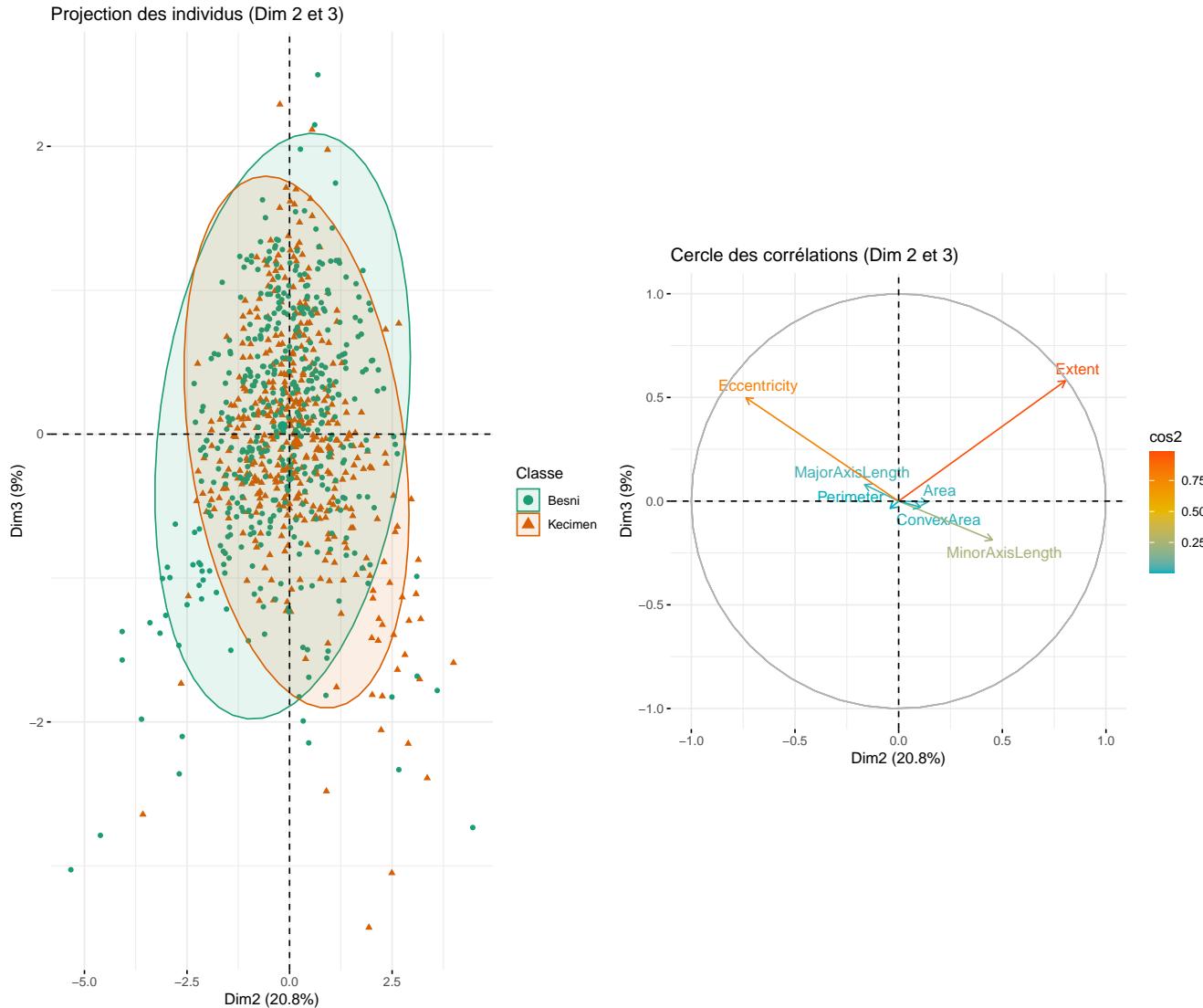


FIGURE 12 – Individus et variables sur les axes 2 et 3

**Analyse des contributions et qualité de représentation.** Le tableau 2 détaille les contributions des variables aux trois premiers axes. Les plus contributives sont (*comme on pouvait s'y attendre avec l'analyse préliminaire*) :

- Area, ConvexArea, Perimeter sur Dim 1 ;
- Extent, MinorAxisLength, Eccentricity sur Dim 2 ;
- Extent, Eccentricity sur Dim 3 .

TABLE 2 – Contributions (%) des variables aux axes 1 à 3

Variable	Dim.1	Dim.2	Dim.3
Area	20.10	1.35	0.00
MajorAxisLength	19.65	1.87	1.01
MinorAxisLength	15.16	14.06	5.57
Eccentricity	4.12	37.31	39.50
ConvexArea	20.33	0.77	0.13
Extent	0.32	44.53	53.58
Perimeter	20.32	0.12	0.20

Ainsi, l'ACP a permis une forte réduction dimensionnelle en concentrant l'information sur les trois premiers axes. Ces derniers révèlent une structure logique dans les données, facilitant leur visualisation et leur interprétation. Les résultats orienteront l'analyse discriminante ultérieure et peuvent être vus plus en détails lors de la compilation du fichier R.

On peut aussi ajouter la figure suivante qui montre où sont situés sur le plan principal de l'ACP les individus avec les plus grandes contributions dans chaque dimension.

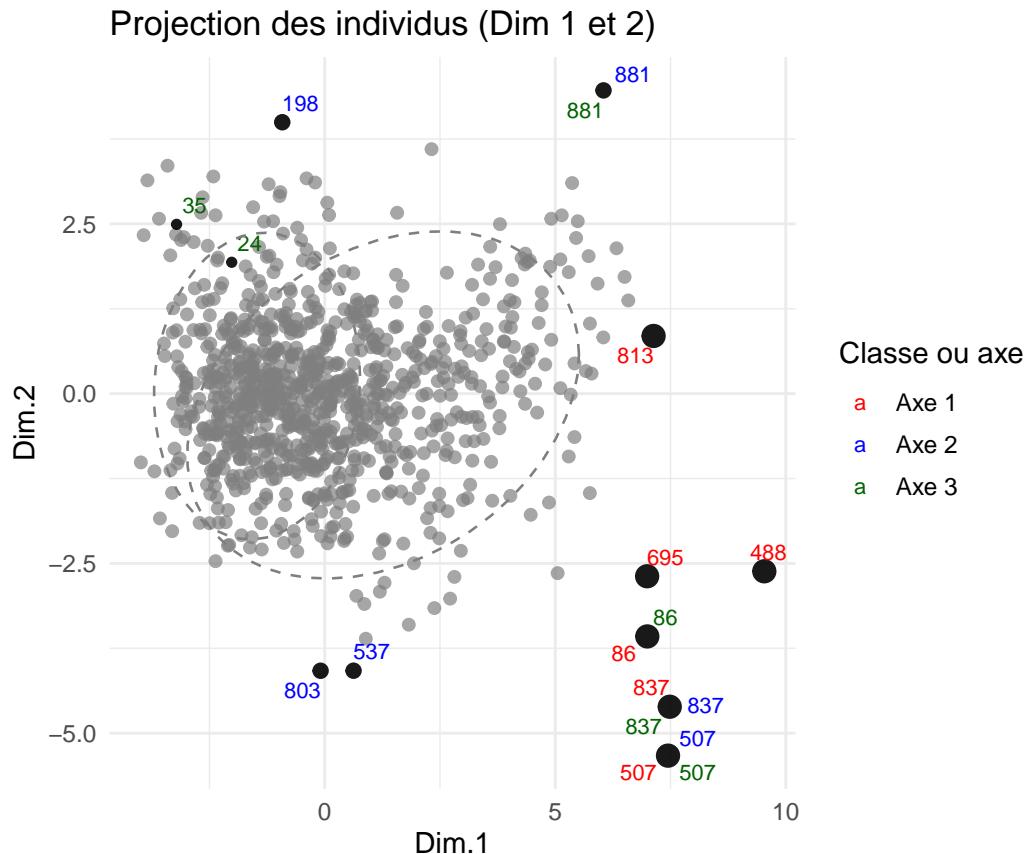


FIGURE 13 – Individus les plus contributifs sur chaque axe dans le plan principal

On remarque que les individus les plus contributifs sont aussi ceux qui sont assez éloignés du centre (*ce à quoi on s'attendait*) et que certains individus comme les 837 et 507 sont parmi les plus contributifs pour chaque axe.

### 1.3 Classification hiérarchique ascendante (CAH)

Cette section présente la classification hiérarchique ascendante (CAH) réalisée sur le jeu de données. L'objectif est de regrouper les observations en classes homogènes, en s'appuyant uniquement sur les variables quantitatives.

**Méthodologie :** La classification a été effectuée selon les étapes suivantes :

- Calcul de la matrice des distances euclidiennes entre individus ;
- Application de l'algorithme de classification hiérarchique agglomérative avec la méthode de liaison par défaut (`hcclus`) ;
- Analyse du dendrogramme pour déterminer le nombre optimal de groupes ;
- Évaluation de la qualité de la partition via l'indice de silhouette ;
- Comparaison avec la classification réelle des variétés de raisin.

**Choix du nombre de groupes.** Le diagramme des hauteurs (Figure 14) permet de visualiser les plus grands sauts dans le dendrogramme, correspondant à des fusions majeures entre groupes.

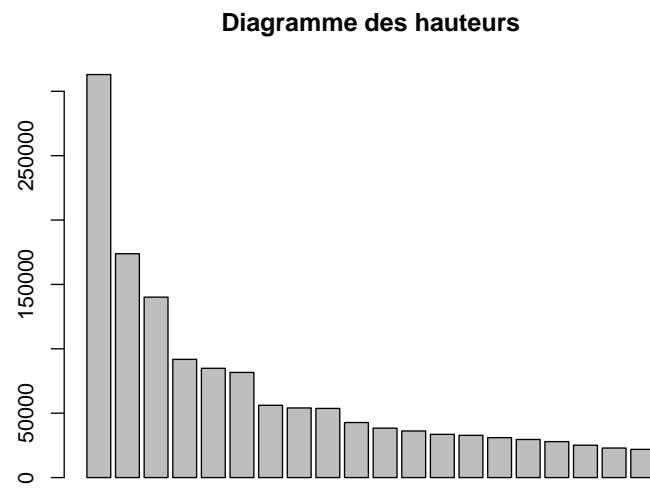


FIGURE 14 – Diagramme des 20 plus grandes hauteurs dans le dendrogramme

Pour affiner ce choix, nous avons calculé l'indice moyen de silhouette pour des partitions allant de 2 à 10 groupes (Figure 15). Le nombre optimal de groupes est celui maximisant cet indice.

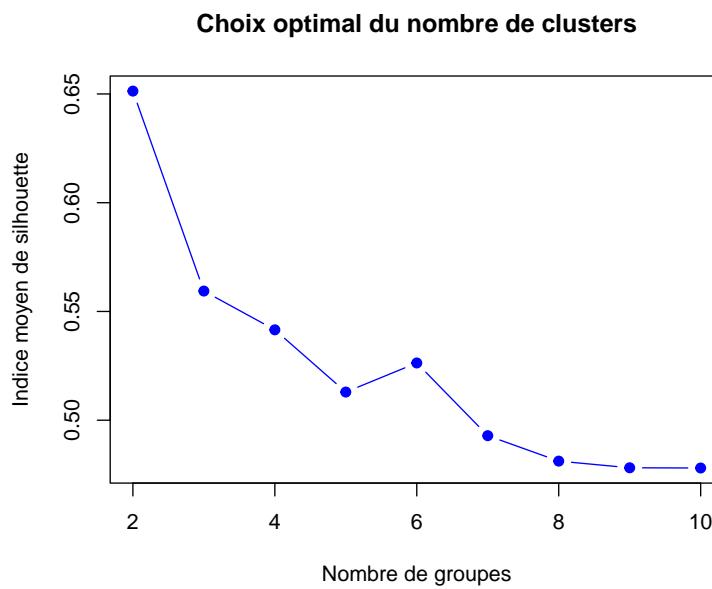


FIGURE 15 – Indice moyen de silhouette en fonction du nombre de groupes

On observe que l'indice de silhouette est maximal pour  $k = 2$ , ce qui correspond bien au nombre réel de variétés de raisin présentes dans le jeu de données.

**Résultat de la classification ( $k = 2$ ) :** La Figure 16 montre le dendrogramme complet avec la coupe à deux groupes.

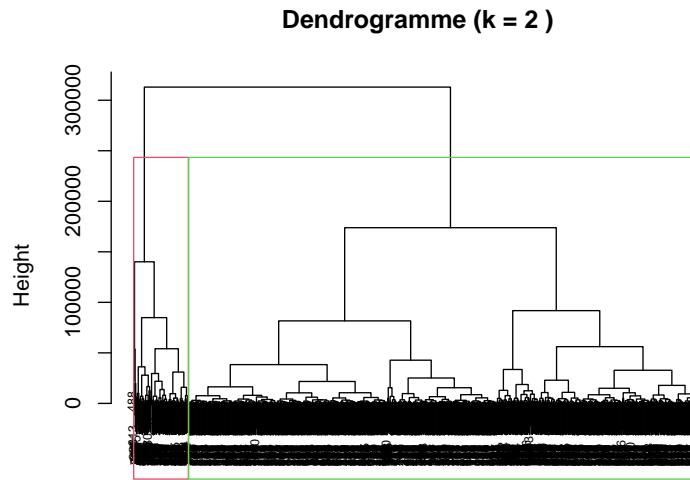


FIGURE 16 – Dendrogramme avec découpe en 2 groupes

La Figure 17 présente le diagramme de silhouette de la partition retenue. L'indice moyen de silhouette est de 0,651, ce qui indique une structure de groupe relativement bien définie.

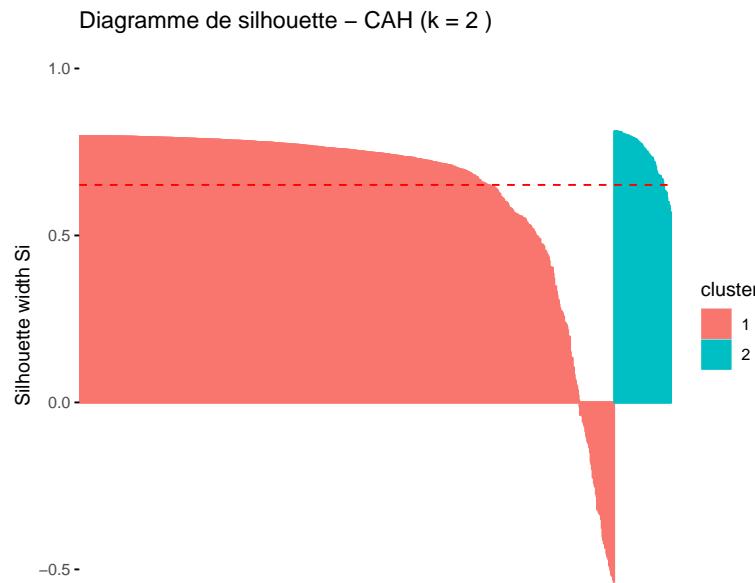


FIGURE 17 – Diagramme de silhouette (k = 2)

**Évaluation de la classification :** Nous comparons les clusters obtenus avec les vraies classes de raisins à l'aide d'une matrice de confusion :

TABLE 3 – Matrice de confusion entre classes réelles et clusters

Classe réelle	Cluster 1	Cluster 2
Besni	365	85
Kecimen	448	2

Le taux d'erreur global est donné par :

$$\text{Erreur} = 1 - \frac{365 + 2}{900} \approx 0,592$$

Ce résultat indique une capacité limitée de la CAH à retrouver la classification d'origine des variétés. En effet, bien que l'indice de silhouette soit relativement élevé, l'erreur de classification est supérieure à 59%.

**Influence de la normalisation.** Afin d'évaluer l'effet de la normalisation des variables, nous avons répété l'analyse en appliquant un centrage-réduction (avec correction biaisée). La variance de chaque variable normalisée a été vérifiée comme proche de 1, assurant une pondération équitable.

Après cette normalisation, une nouvelle CAH a été réalisée, toujours avec  $k = 2$ . Le taux d'erreur obtenu est de 0,627, soit légèrement supérieur à celui obtenu sans normalisation. Cela suggère que la normalisation n'améliore pas la classification dans ce cas, probablement car les variables initiales étaient déjà de nature comparable.

Ainsi, la classification hiérarchique ascendante détecte bien une structure à deux groupes, mais ne parvient pas à reproduire fidèlement les classes réelles des raisins, avec une erreur de classification notable. L'indice de silhouette élevé indique cependant une bonne séparation géométrique des groupes. La normalisation n'apporte pas d'amélioration significative sur ce jeu de données.

#### 1.4 Clustering hiérarchique sur les composantes principales.

Nous cherchons ici à améliorer la classification en combinant l'analyse en composantes principales (ACP) avec une classification hiérarchique ascendante (CAH). L'idée est de réduire la dimensionnalité du jeu de données pour ne conserver que les axes les plus informatifs avant d'appliquer la classification.

**Méthodologie :** Pour chaque valeur  $k \in \{1, \dots, 7\}$  :

- Nous extrayons les coordonnées des individus sur les  $k$  premières composantes principales issues de l'ACP réalisée précédemment ;
- Nous calculons les distances euclidiennes entre individus dans cet espace réduit ;
- Nous effectuons une classification hiérarchique ascendante sur ces données ;
- Nous découpons l'arbre en deux groupes (car il y a deux classes réelles de raisin) ;
- Nous comparons les clusters obtenus à la classification réelle à l'aide de la matrice de confusion et calculons le taux d'erreur.

**Résultats :** La Figure 18 présente l'évolution de l'erreur de classification en fonction du nombre de composantes principales utilisées.

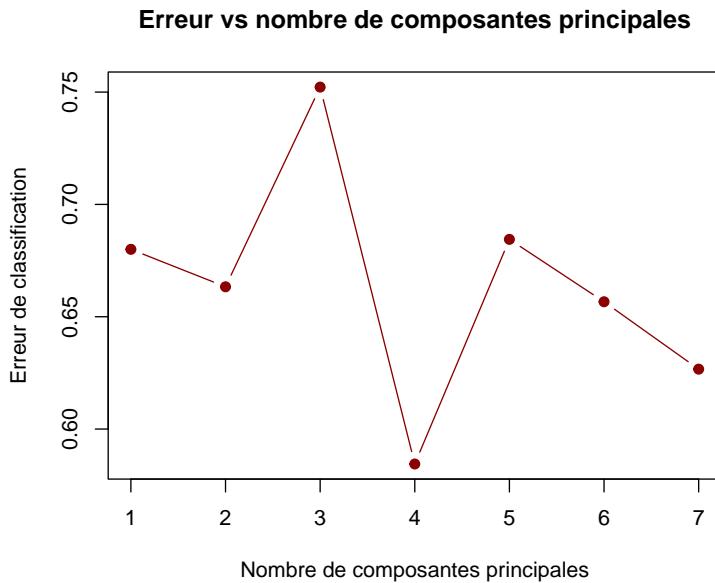


FIGURE 18 – Erreur de classification selon le nombre de composantes principales

On observe que l'erreur diminue initialement lorsque l'on augmente le nombre de composantes, puis se stabilise. L'erreur minimale est atteinte pour  $k = 4$  composantes principales, avec un taux d'erreur de 0,584.

**Interprétation :** Ce résultat montre qu'une représentation des données dans un espace réduit à 4 dimensions permet d'améliorer légèrement la qualité de la classification par rapport à l'espace original (*Même si avec notre première règle, on avait décidé de conserver pour l'étude globale que les trois premières composantes principales*). L'ACP contribue donc à mieux séparer les groupes en éliminant du bruit ou des redondances.

**Biais et résultat.** Il convient cependant de rester prudent quant à l'interprétation de ce résultat, en effet : l'ACP est une méthode non supervisée, c'est-à-dire qu'elle ne prend pas en compte les classes réelles dans sa construction. Cela garantit qu'il n'y a pas de biais direct lié aux labels. Cependant, en choisissant le nombre de composantes a posteriori, en minimisant l'erreur de classification, on introduit un biais de sélection. En pratique, cela revient à effectuer plusieurs tests et à ne retenir que le meilleur, ce qui peut surestimer la performance réelle. Pour éviter ce biais, il serait plus rigoureux d'utiliser une procédure de validation croisée ou de fixer à l'avance un critère pour choisir  $k$  (comme le pourcentage d'inertie expliquée).

**Conclusion :** L'ACP permet ici une légère amélioration de la classification en réduisant la dimension des données. Le meilleur résultat est obtenu avec 4 composantes principales, mais cette optimisation introduit un biais de sélection qu'il convient de garder à l'esprit.

## 2 Partie II : Choisir un modèle

### 2.1 Définition du modèle logistique et discussion sur le centrage-réduction

**Modèle logistique :** Le modèle logistique est un modèle de régression utilisé lorsque la variable à prédire est binaire. Dans notre cas, il s'agit de prédire la variété de raisin à partir des mesures physico-chimiques. On note  $Y \in \{0, 1\}$  la variable réponse (par exemple : 0 pour Besni et 1 pour Kecimen), et  $\mathbf{x} = (x_1, x_2, \dots, x_p)^\top$  les variables explicatives.

Le modèle logistique s'écrit :

$$\mathbb{P}(Y = 1 | \mathbf{x}) = \frac{1}{1 + \exp(-(\beta_0 + \mathbf{x}^\top \boldsymbol{\beta}))}$$

où :

- $\beta_0$  est l'ordonnée à l'origine (intercept),
- $\boldsymbol{\beta} = (\beta_1, \dots, \beta_p)$  est le vecteur des coefficients associés aux variables explicatives.

Ce modèle est estimé par maximisation de la vraisemblance (maximum likelihood).

Ensuite, on va centrer et réduire les variables explicatives, ce qui consiste à soustraire à chaque variable sa moyenne puis diviser chaque variable centrée par son écart-type. Cela revient à transformer chaque variable pour qu'elle ait une moyenne nulle et une variance unitaire.

#### Aspects théoriques

D'un point de vue théorique, le centrage-réduction n'est pas nécessaire pour l'estimation du modèle logistique en tant que tel : la vraisemblance est invariante par changement d'échelle des variables.

Cependant, plusieurs aspects justifient son utilisation :

- **Comparaison des coefficients** : Les coefficients  $\beta_j$  dépendent de l'échelle des variables. Si les variables ne sont pas réduites, il est difficile de comparer leur importance relative. Avec des variables standardisées, l'amplitude des coefficients permet une interprétation directe de l'impact relatif de chaque variable.
- **Stabilité numérique** : Si certaines variables ont des échelles très différentes (par exemple, des longueurs en mm et des densités sans unité), cela peut rendre l'optimisation numériquement instable.
- **Modèles avec régularisation** : Dans les modèles pénalisés, le centrage-réduction est indispensable. En effet, sans cela, la pénalisation affecterait différemment chaque variable selon son échelle, ce qui biaiserait la sélection automatique des variables.

#### Aspects pratiques

Dans notre cas, les variables explicatives proviennent de mesures hétérogènes (aires, périmètres, densités, etc.). Elles sont donc naturellement de nature et d'échelle différentes. Leur centrage et réduction est fortement recommandé pour :

- éviter que certaines variables dominent l'ajustement uniquement en raison de leur échelle,
- améliorer la lisibilité des coefficients estimés,
- préparer le jeu de données pour une éventuelle sélection de variables ou régularisation.

Même si le modèle logistique fonctionne sans centrage-réduction, il est judicieux, dans un contexte réel, de standardiser les variables explicatives. Cela améliore la stabilité numérique, la lisibilité des résultats, et constitue une étape indispensable pour des méthodes plus avancées de sélection automatique de variables.

## 2.2 Définition de l'échantillon d'apprentissage et projection sur le premier plan principal

**Définition de l'échantillon d'apprentissage.** Dans le but de construire un modèle prédictif robuste, nous divisons les données en deux sous-ensembles :

- un **échantillon d'apprentissage** (ou training set), utilisé pour ajuster les paramètres du modèle,
- un **échantillon de test**, utilisé pour évaluer la capacité de généralisation du modèle.

Nous avons défini l'échantillon d'apprentissage en tirant aléatoirement, pour chaque individu, une probabilité d'appartenir à l'apprentissage de  $2/3$  :

```
set.seed(1)
train <- sample(c(TRUE, FALSE), n, rep = TRUE, prob = c(2/3, 1/3))
```

Cela permet une répartition aléatoire et reproductible des données.

**Analyse en composantes principales (ACP) sur les données d'apprentissage.** Nous appliquons une ACP sur les données centrées et réduites issues uniquement de l'échantillon d'apprentissage, sans inclure la variable cible (variété de raisin). Cela permet d'identifier les directions principales de la variabilité des données explicatives.

Nous notons que l'ACP est réalisée avec un nombre maximal de composantes principales (i.e.,  $p - 1$ , où  $p$  est le nombre total de variables explicatives), mais nous nous intéressons principalement à la projection sur le premier plan principal, c'est-à-dire les deux premières composantes principales.

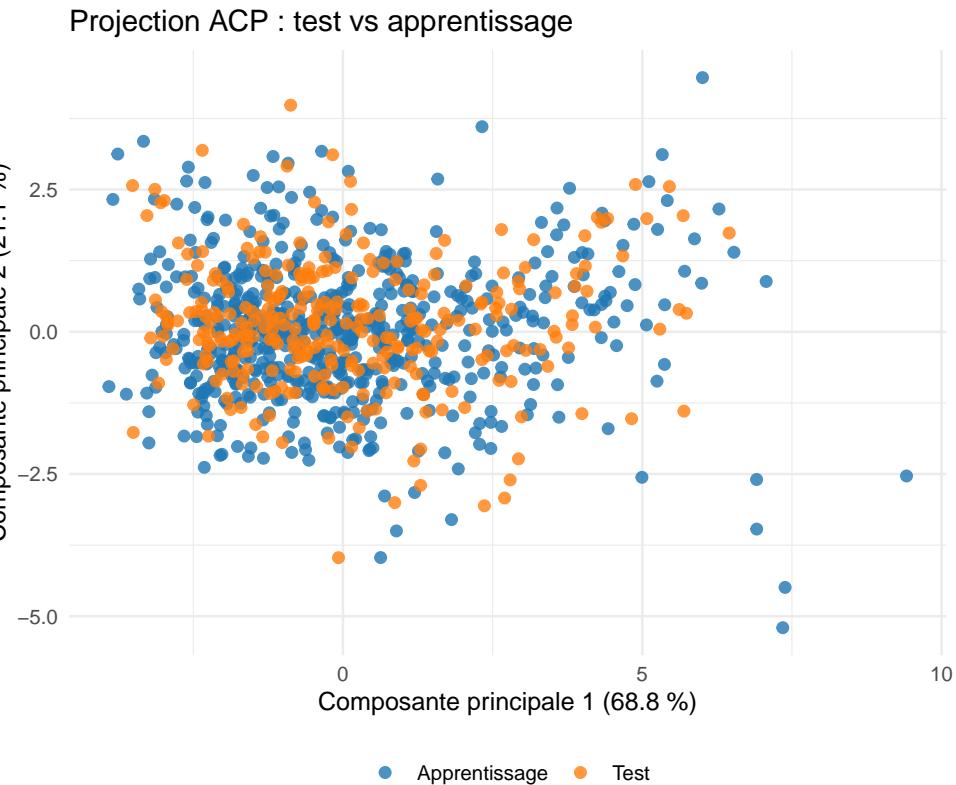
**Projection des données de test sur l'ACP d'apprentissage.** Afin de comparer les deux sous-populations (apprentissage et test) dans l'espace factoriel, nous projetons les observations du jeu de test sur les axes de l'ACP calculée à partir des données d'apprentissage.

Cette projection se fait via la commande `predict` de FactoMineR, qui permet d'obtenir pour chaque individu test :

- ses coordonnées sur les axes principaux (résultat dans `$coord`),
- ses contributions relatives aux dimensions (résultat dans `$cos2`),
- sa distance à l'origine du nuage (résultat dans `$dist`).

Par exemple, la première observation test est projetée avec une forte contribution sur la première composante ( $\cos^2$  de 0.95) et une distance relativement grande (2.63), ce qui témoigne de sa bonne représentation dans le plan principal.

**Visualisation des projections :** Nous représentons graphiquement la projection des deux sous-ensembles (apprentissage et test) sur le premier plan principal (Dim.1 vs Dim.2), ce qui permet de visualiser leur répartition respective dans l'espace factoriel.



Ce graphique permet d'analyser si les deux sous-groupes sont similaires du point de vue structurel. On observe que les individus test se projettent de manière cohérente avec ceux de l'apprentissage, confirmant la stabilité de l'ACP.

Cette étape est cruciale pour valider la compatibilité entre les échantillons d'apprentissage et de test. En projetant les données de test sur l'ACP construite à partir des données d'apprentissage, on s'assure que la structure principale de la variabilité est conservée, ce qui est essentiel pour la suite de la modélisation (régression logistique).

## 2.3 Estimation des modèles de régression logistique

Nous allons désormais estimer plusieurs modèles de régression logistique et les comparer :

- modèle complet, comprenant toutes les composantes
- uniquement les deux premières composantes principales
- sélection de variables avec critère AIC
- régression pénalisée lasso

### 2.3.1 Modèle complet

Nous commençons par estimer un modèle de régression logistique utilisant l'ensemble des variables explicatives standardisées. Ce modèle complet permet de modéliser directement la probabilité d'appartenance à la classe `Kecimen` en fonction des caractéristiques géométriques des grains de raisin.

L'équation du modèle estimé est la suivante :

$$\log \left( \frac{\mathbb{P}(y = \text{Kecimen})}{\mathbb{P}(y = \text{Besni})} \right) = \beta_0 + \beta_1 \cdot \text{Area} + \beta_2 \cdot \text{MajorAxisLength} + \dots + \beta_7 \cdot \text{Perimeter}$$

### Estimation des coefficients

- Parmi les variables les plus significatives au seuil de 1% figurent : `Area`, `MajorAxisLength`, `ConvexArea`, et `Perimeter`.
- La variable `MinorAxisLength` est significative au seuil de 5%.

- D'autres variables comme **Eccentricity**, **Extent** ne sont pas significatives individuellement, mais leur présence peut néanmoins contribuer à la qualité globale du modèle.

Les signes des coefficients permettent d'interpréter l'effet marginal sur la probabilité d'être de la classe **Kecimen** :

- Une augmentation de l'**Area** ou du **Perimeter** diminue la probabilité d'être **Kecimen**, ce qui correspond à un coefficient négatif.
- Une augmentation de la **MajorAxisLength** ou de la **MinorAxisLength** augmente cette probabilité, avec des coefficients positifs.

### Critères d'ajustement

- AIC : **425.85**
- BIC : **461.03**
- Déviance du modèle nul (sans covariables) : **831.45**
- Déviance résiduelle du modèle complet : **409.85**
- Gain de déviance : **421.60**, très significatif.

Ces résultats indiquent que le modèle complet permet une forte réduction de la déviance, traduisant une bonne capacité explicative.

**Analyse de déviance** L'analyse de déviance confirme la significativité globale de plusieurs variables, en particulier :

- **Area** ( $p < 2.2 \times 10^{-16}$ ),
- **MajorAxisLength** ( $p < 5 \times 10^{-14}$ ),
- **Perimeter** ( $p < 10^{-8}$ ).

Certaines variables comme **Eccentricity** ou **ConvexArea** n'apportent pas de réduction significative de la déviance.

**Visualisation sur le plan factoriel ACP** La figure suivante illustre la classification obtenue par le modèle sur l'échantillon d'apprentissage, projetée sur les deux premières composantes principales de l'ACP. Chaque point est colorié selon sa classe réelle, la forme indique la classe prédite, et la transparence représente la probabilité prédite d'être **Kecimen**.

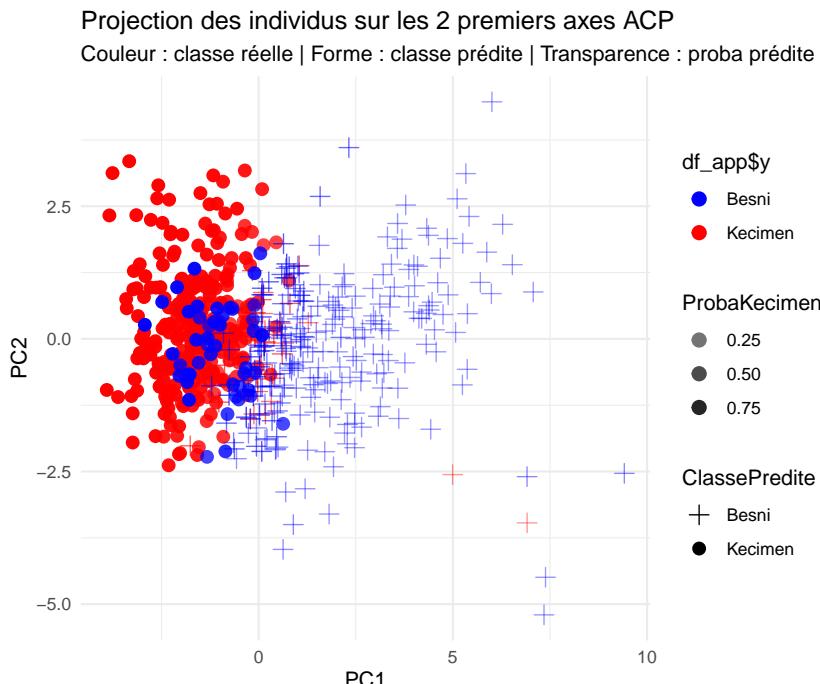


FIGURE 19 – Projection des individus sur le plan ACP — Modèle complet (train)

On observe que les observations sont globalement bien séparées dans l'espace réduit. La classification respecte la structure des données projetées, avec peu de confusions visibles.

Ainsi, le modèle complet offre un ajustement global satisfaisant, avec une réduction importante de la déviance et plusieurs variables significatives. Cependant, certaines variables pourraient être retirées sans perte notable de performance, comme nous allons voir dans la méthode suivante.

### 2.3.2 Modèle logistique sur les deux premières composantes principales

Dans cette approche, nous ne retenons que les deux premières composantes principales issues de l'ACP comme variables explicatives. Ce choix permet de travailler dans un espace de dimension réduite, tout en conservant l'essentiel de la variance des données. Cela constitue une méthode de régularisation par projection dans un sous-espace informatif.

**Estimation du modèle** Le modèle logistique est estimé selon la spécification suivante :

$$\log \left( \frac{\mathbb{P}(y = \text{Kecimen})}{\mathbb{P}(y = \text{Besni})} \right) = \beta_0 + \beta_1 \cdot \text{Dim.1} + \beta_2 \cdot \text{Dim.2}$$

Les résultats de l'estimation sont les suivants :

— **Intercept** :  $-0,298$  ( $p = 0,019$ )

— **Dim.1** :  $-1,375$  ( $p < 2 \cdot 10^{-16}$ )

La première composante est fortement significative et négativement corrélée avec la probabilité d'appartenir à la classe **Kecimen**.

— **Dim.2** :  $+0,491$  ( $p < 10^{-4}$ ), également significative.

#### Critères d'ajustement

— AIC : **453.38**

— BIC : **466.57**

— Déviance du modèle nul : **831.45**

— Déviance résiduelle : **447.38**

— Gain de déviance : **384.07**

Ces résultats montrent que, bien que ce modèle utilise uniquement deux variables synthétiques, il conserve une capacité explicative importante, tout en étant plus parcimonieux que le modèle complet.

#### Analyse de déviance

— L'ajout de **Dim.1** réduit significativement la déviance ( $p < 2,2 \cdot 10^{-16}$ ).

— L'ajout de **Dim.2** améliore encore le modèle ( $p < 1,2 \cdot 10^{-5}$ ).

**Visualisation sur le plan factoriel ACP** Nous visualisons les résultats du modèle projetés sur les deux premières composantes principales (identiques aux variables du modèle). La figure suivante illustre la classification obtenue :

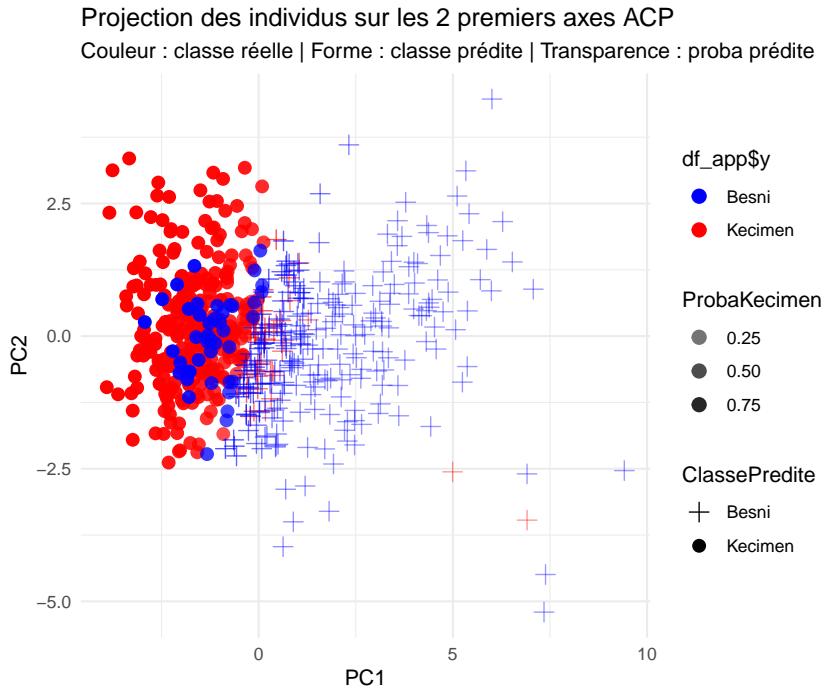


FIGURE 20 – Projection des individus sur le plan ACP — Modèle logistique avec PC1 et PC2 (train)

Comme précédemment :

- la couleur représente la classe réelle (**Besni** en bleu, **Kecimen** en rouge),
- la forme des points indique la classe prédictée,
- la transparence reflète la probabilité prédictée d'être **Kecimen**.

Ce graphique montre que la séparation entre les classes est bien capturée par les deux premières composantes principales, ce qui confirme leur pertinence pour la classification.

Ainsi, ce modèle réduit à deux composantes offre un compromis intéressant entre performance et simplicité. Malgré la perte d'information due à la réduction de dimension, le modèle reste très performant avec une bonne capacité de discrimination. Il constitue donc une alternative efficace au modèle complet, notamment pour des raisons d'interprétabilité et de robustesse.

### 2.3.3 Modèle logistique avec sélection de variables par critère AIC

Afin d'obtenir un modèle parcimonieux tout en conservant une bonne capacité prédictive, nous appliquons une procédure de sélection pas à pas (backward) fondée sur le critère AIC. Cette méthode supprime successivement les variables les moins contributives au modèle en termes d'information (*Les détails du processus de sélection sont retrouvables dans le code R*).

**Processus de sélection** À partir du modèle complet (incluant les 7 variables), la procédure a supprimé successivement les variables Eccentricity puis Extent, menant à un modèle final avec les 5 variables suivantes :

- Area
- MajorAxisLength
- MinorAxisLength
- ConvexArea
- Perimeter

**Résultats du modèle sélectionné** Le modèle final présente les coefficients suivants :

TABLE 4 – Coefficients du modèle sélectionné par AIC

Variable	Estimation	Erreur Std.	p-value
Constante	5.941	5.054	0.240
Area	-0.000572	0.0000935	<0.001
MajorAxisLength	0.0574	0.0183	0.00174
MinorAxisLength	0.0918	0.0251	0.000261
ConvexArea	0.000484	0.0000792	<0.001
Perimeter	-0.0416	0.00747	<0.001

Toutes les variables sélectionnées sont hautement significatives à l'exception de l'intercept, qui reste non significatif.

### Critères d'ajustement

- AIC : **422.76**, inférieur au modèle complet (425.85) et au modèle avec uniquement 2 composantes (453.38).
- BIC : **449.14**
- Déviance nulle : **831.45**
- Déviance résiduelle : **410.76**
- Gain de déviance : **420.69**

### Analyse de déviance

- L'ajout de la variable **Area** explique à elle seule une part substantielle de la déviance ( $p < 2,2 \cdot 10^{-16}$ ).
- Les variables **MajorAxisLength** et **Perimeter** apportent également des contributions significatives.
- Les variables **MinorAxisLength** et **ConvexArea** apparaissent marginales dans l'analyse de déviance, mais ont été conservées dans le modèle AIC pour leur contribution globale à l'information.

**Visualisation sur le plan ACP** Le graphique ci-dessous illustre les prédictions du modèle sur le plan formé par les deux premières composantes principales, bien qu'elles ne soient pas directement utilisées dans le modèle (elles servent ici de plan de projection pour la visualisation).

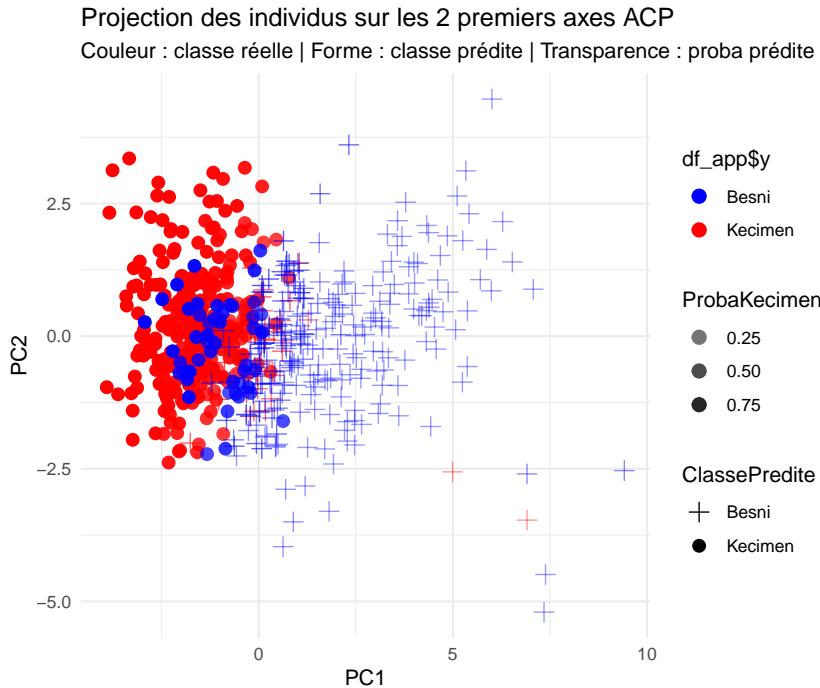


FIGURE 21 – Projection des individus sur le plan ACP — Modèle logistique après sélection AIC (train)

Le modèle obtenu par sélection AIC offre un très bon compromis entre complexité et performance. Il atteint une performance presque équivalente au modèle complet, tout en étant plus simple (5 variables au lieu de 7). C'est un bon candidat pour une utilisation en pratique, d'autant plus que les variables retenues sont toutes significatives et interprétables.

#### 2.3.4 Régression logistique pénalisée (Lasso)

Dans cette approche, nous cherchons à régulariser le modèle afin de réduire la variance et favoriser une sélection précise des variables explicatives. Nous utilisons pour cela la régression logistique pénalisée avec une pénalisation de type Lasso (norme  $\ell_1$ ), en optimisant le paramètre de régularisation par validation croisée.

Dans notre code, nous avons alors préparer les données via una matrice puis effectué une validation croisée pour le paramètre  $\lambda$ . Ensuite, nous avons regardé les différentes valeurs de  $\lambda$  (*Donnant une information entre la pondération entre l'erreur lié aux données et la norme  $\ell_1$* ) et gardé la meilleure (la plus petite), et on obtient alors les coefficients du modèle optimal.

Le paramètre de régularisation optimal obtenu par validation croisée est  $\lambda_{\min} = 5.90 \times 10^{-4}$ .  
Les variables sélectionnées (dont les coefficients estimés ne sont pas nuls) sont :

`Area, MajorAxisLength, MinorAxisLength, Eccentricity, Extent, Perimeter.`

On ajuste ensuite un modèle de régression logistique classique uniquement avec ces variables sélectionnées, en effet on va d'abord construire le modèle et procéder à une estimation à partir des variables sélectionnées.

Les coefficients estimés ainsi que leur significativité sont présentés dans le tableau suivant :

TABLE 5 – Coefficients du modèle logistique avec sélection Lasso

Variable	Estimation	Erreur Std.	z	p-value
Constante	6.872	8.397	0.818	0.413
Area	-0.000121	0.0000661	-1.83	0.067
MajorAxisLength	0.0420	0.0179	2.35	0.019
MinorAxisLength	0.0694	0.0306	2.27	0.023
Eccentricity	-3.69	6.58	-0.56	0.575
Extent	-2.98	3.18	-0.94	0.349
Perimeter	-0.0237	0.00604	-3.93	<0.001

Parmi les variables sélectionnées, certaines sont significatives au seuil 5% (**MajorAxisLength**, **MinorAxisLength** et **Perimeter**), tandis que d'autres (comme **Extent** ou **Eccentricity**) ne le sont pas.

Les critères d'information associés sont :

- AIC = 439.96
- BIC = 470.74

Le score de résidus (ou somme des carrés résiduels du modèle ajusté) est :

$$\text{SCR}_{\text{Lasso}} = 425.96, \quad \text{vs. } \text{SCR}_{\text{null}} = 831.45.$$

On représente ci-dessous la projection des individus sur les deux premiers axes de l'ACP, en indiquant la classe réelle (couleur), la classe prédictée par le modèle Lasso (forme), ainsi que la probabilité prédictive pour la classe **Kecimen** (transparence) :

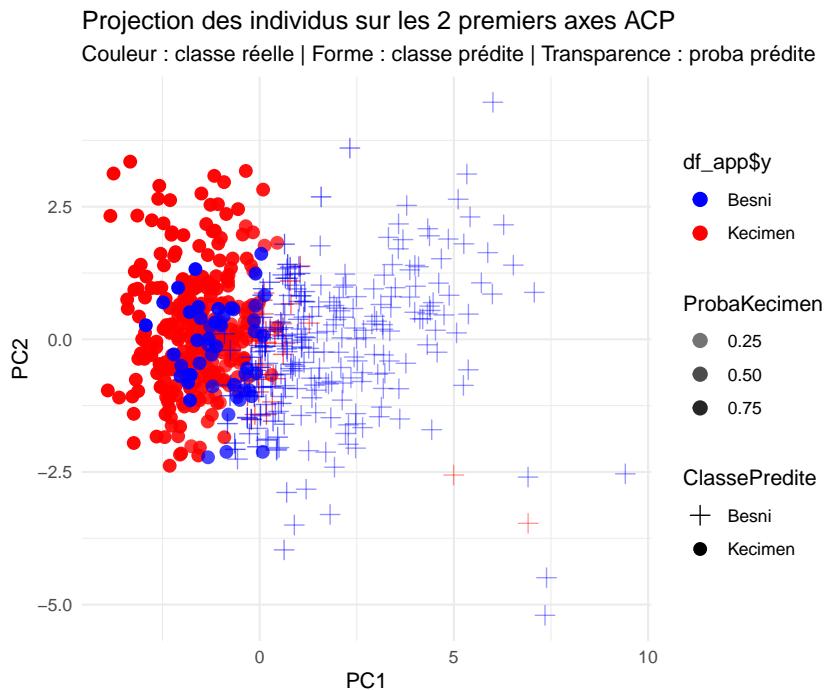


FIGURE 22 – Projection des individus selon les deux premiers axes ACP (modèle Lasso)

Ce modèle présente de bonnes performances avec un nombre réduit de variables, ce qui le rend intéressant dans un contexte d'interprétabilité. On notera cependant que toutes les variables sélectionnées ne sont pas significatives individuellement.,

### 2.3.5 Comparaison avec la régression Ridge

Une régression logistique avec pénalisation Ridge (norme  $\ell_2$ ) a également été ajustée pour comparaison. Elle sélectionne toutes les variables, mais avec des coefficients généralement plus petits. Le modèle Ridge obtient un AIC plus faible (425.85), ce qui suggère un meilleur compromis biais-variance, au prix d'une interprétabilité plus faible par absence de sélection stricte des variables.

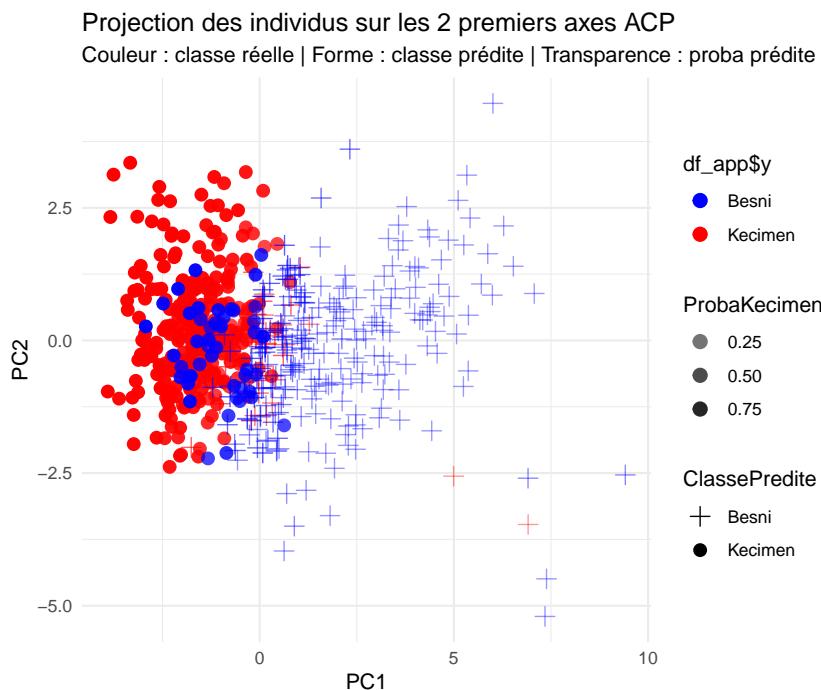


FIGURE 23 – Projection des individus selon les deux premiers axes ACP (modèle Ridge)

## 2.4 SVM linéaire et noyau polynomial

Dans cette question, nous avons estimé deux modèles de machine learning supervisé : un SVM à noyau linéaire, et un SVM avec noyau polynomial. L'objectif était de classifier les observations en deux classes (**Besni** et **Kecimen**) en optimisant les performances du modèle par validation croisée.

**Prétraitement des données :** Avant l'entraînement, les variables explicatives ont été standardisées car la fonction `tune` de la librairie `e1071` ne le fait pas automatiquement. Nous avons également conservé les coordonnées des deux premières composantes principales (ACP) pour visualiser les résultats dans un espace bidimensionnel.

**SVM à noyau linéaire :** Nous avons utilisé une validation croisée à 10 plis pour sélectionner le paramètre de régularisation `cost` parmi les valeurs suivantes : 0.01, 0.1, 1, 5, 10 et 100. Le meilleur paramètre sélectionné est `cost = 100` (*Ce qui est bizarre étant donné qu'en cours, un plus petit cost étant plus adapté pour minimiser l'erreur dans le modèle linéaire*), correspondant à une erreur moyenne de validation croisée de **13.5%**.

Paramètre <code>cost</code>	Erreur CV moyenne
0.01	14.3%
0.1	14.2%
1	13.8%
5	14.2%
10	14.0%
<b>100</b>	<b>13.5%</b>

Le modèle optimal utilise 216 vecteurs de support, répartis équitablement entre les deux classes. La frontière de décision ainsi que les marges ont été projetées dans le plan formé par les deux premières composantes principales de l'ACP. La figure 24 présente cette visualisation.

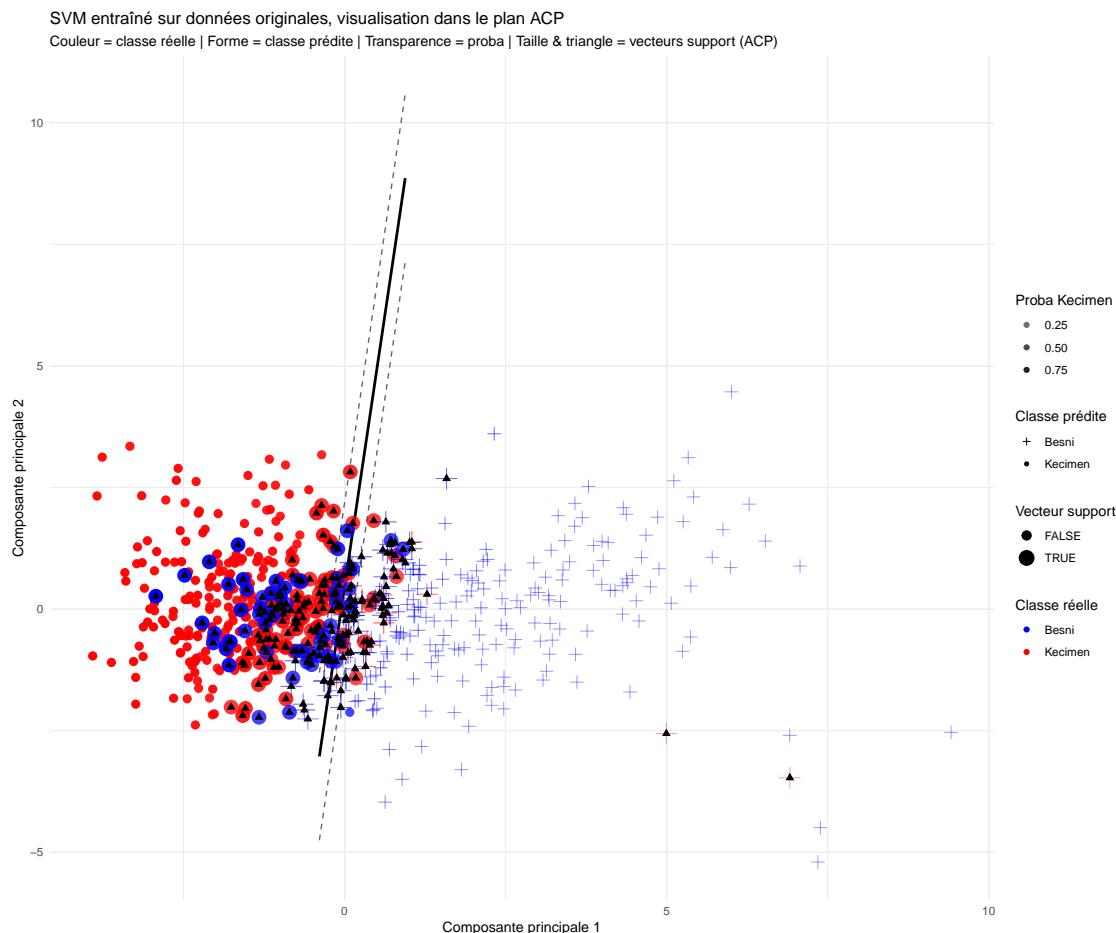


FIGURE 24 – SVM linéaire : classification projetée sur le plan ACP.

Couleur = classe réelle, Forme = classe prédictive, Transparence = probabilité, Taille & triangle = vecteurs support.

**SVM à noyau polynomial** Nous avons ensuite estimé un modèle SVM avec noyau polynomial (degré par défaut = 3). Le paramètre **cost** a de nouveau été sélectionné par validation croisée à 10 plis. Le meilleur coût obtenu est également **cost = 100**, avec une erreur moyenne de **15.2%**, légèrement supérieure au modèle linéaire.

Paramètre cost	Erreur CV moyenne
0.01	33.2%
0.1	23.3%
1	16.7%
5	16.8%
10	15.8%
<b>100</b>	<b>15.2%</b>

Le modèle polynomial optimal utilise 261 vecteurs de support, soit davantage que le modèle linéaire. La frontière de décision non linéaire a été estimée sur une grille dans le plan ACP et tracée à l'aide de courbes de niveau, comme illustré à la figure 25.

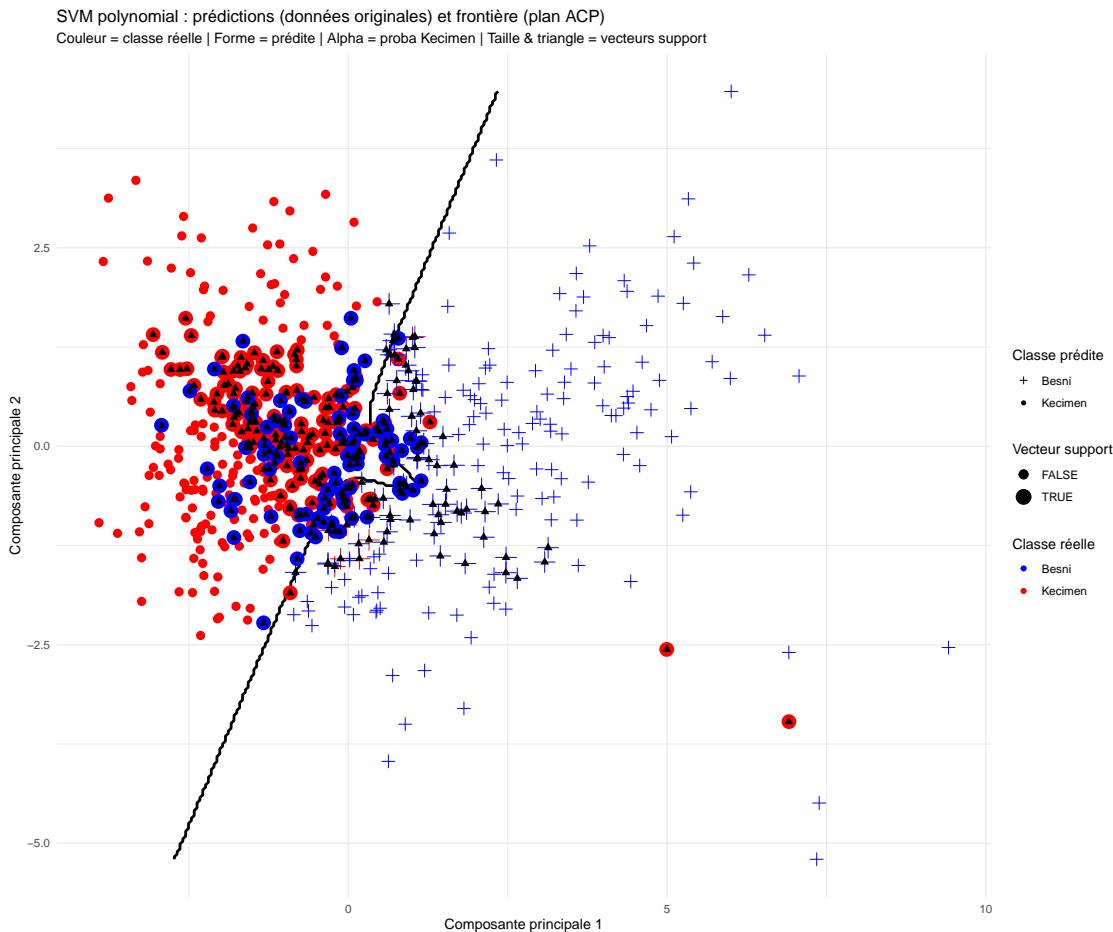


FIGURE 25 – SVM polynomial : classification projetée sur le plan ACP.

Couleur = classe réelle, Forme = classe prédite, Alpha = probabilité, Taille & triangle = vecteurs support.

**Comparaison :** Les deux modèles SVM donnent de bonnes performances de classification, mais le modèle linéaire obtient une légère meilleure performance moyenne en validation croisée (13.5% contre 15.2%). Le modèle polynomial, bien que plus flexible, tend à utiliser davantage de vecteurs de support et à présenter un risque de surapprentissage si les paramètres ne sont pas soigneusement choisis.

Ces résultats suggèrent que le modèle linéaire suffit pour séparer les classes dans cet espace, et qu'une complexité supplémentaire apportée par un noyau polynomial n'améliore pas significativement la classification.

## 2.5 Analyse des courbes ROC et comparaison des modèles

Afin d'évaluer et de comparer la performance des différents modèles de classification, nous avons tracé leurs **courbes ROC** (*Receiver Operating Characteristic*) ainsi que calculé l'**aire sous la courbe ROC (AUC)**, à la fois sur l'échantillon d'apprentissage (pour le modèle complet) et sur l'échantillon de test (pour tous les modèles). Ces analyses permettent de visualiser le compromis entre la sensibilité (rappel) et la spécificité des modèles pour différents seuils de classification.

**Méthodologie de construction des courbes ROC :** Pour chaque modèle, les étapes suivantes ont été réalisées :

1. **Calcul des probabilités prédites** pour la classe Kecimen, soit via `predict(..., type = "response")` ou bien via des méthodes spécifiques pour les SVM par exemple.
2. **Création d'un vecteur de seuils** (de 0 à 1, avec 1000 valeurs), pour générer les prédictions binaires à chaque seuil.

3. Pour chaque seuil, **calcul des métriques de classification** :
  - Taux de faux positifs (1 - spécificité),
  - Taux de vrais positifs (sensibilité),
  - Nombre d'erreurs,
  - Seuil minimisant l'erreur de classification,
  - Seuil garantissant une sensibilité supérieure à 95%.
4. Construction de la **courbe ROC**, c'est-à-dire le tracé de la sensibilité en fonction du taux de faux positifs, avec superposition de la diagonale correspondant à une règle de classification aléatoire (ligne en pointillés).
5. Calcul de **l'AUC** à l'aide du package **ROCR**, ce qui permet une quantification globale de la qualité de la séparation opérée par le modèle : plus l'AUC est proche de 1, meilleure est la performance.
6. Pour le **modèle complet**, la courbe ROC a été tracée à la fois sur l'analyse à la main et sur l'analyse automatique par la librairie fourni directement par R.
7. Chaque modèle possède ensuite le même type de graphique que le suivant, qui sont affichés puis enregistrés lors de la compilation du code.

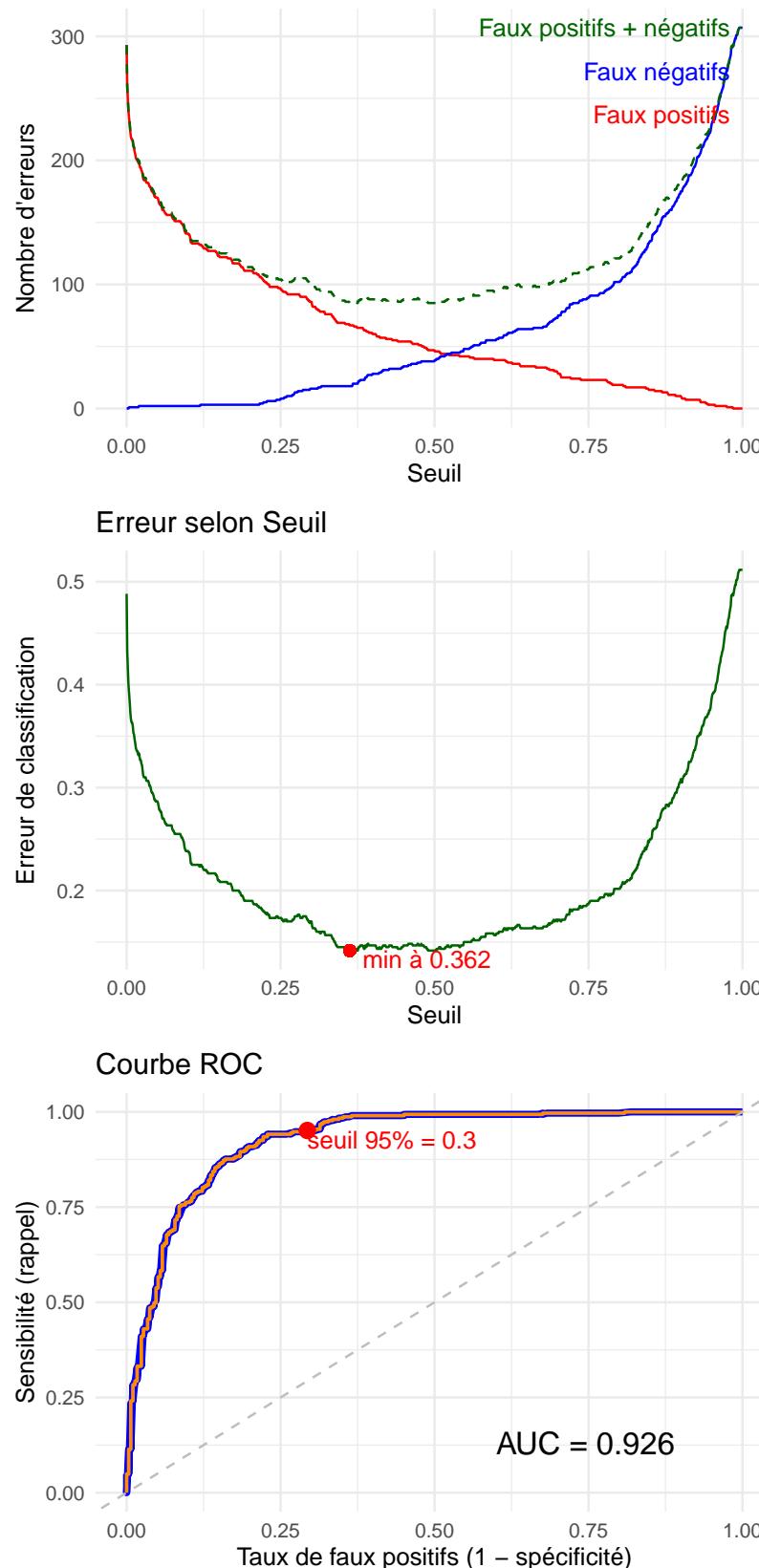


FIGURE 26 – Courbe ROC du modèle complet (échantillon d'apprentissage)

**Résultats obtenus :** Sur l'échantillon d'apprentissage, le modèle complet obtient une AUC de **0.926**, ce qui traduit une excellente capacité de discrimination. Le **seuil optimal** minimisant l'erreur est de **0.36**, et un seuil de **0.30** permet

d'obtenir une sensibilité supérieure à 95%.

Pour les autres modèles, voici le résumé des performances :

TABLE 6 – Performances comparées des modèles selon l'analyse ROC

Modèle	Seuil optimal	Seuil 95% sensibilité	AUC test
Régression logistique complète	0.362	0.297	0.926
Régression sur 2 CP	0.489	0.307	0.920
Régression avec AIC	0.442	0.288	0.925
Régression Lasso	0.464	0.368	0.933
Régression Ridge	0.362	0.297	0.926
SVM linéaire	0.549	0.453	0.933
SVM polynomial	0.531	0.521	<b>0.941</b>

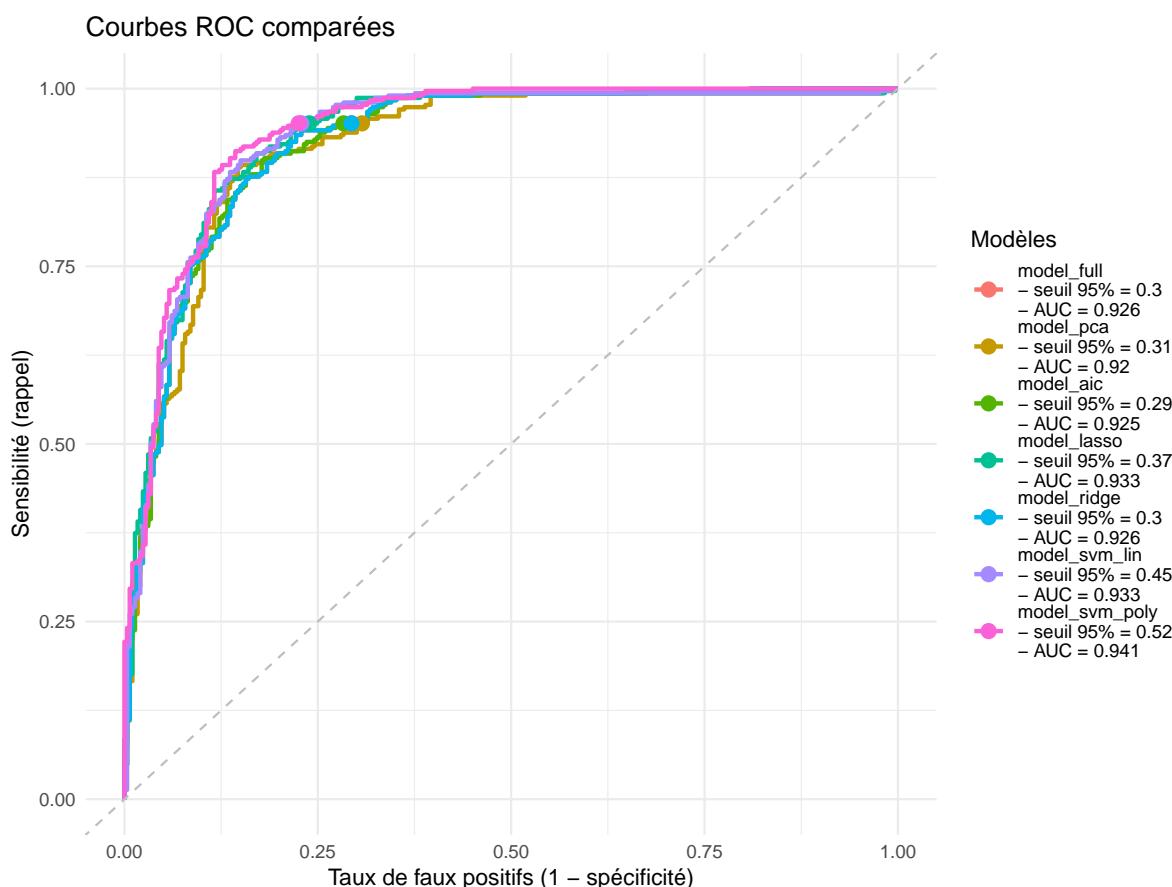


FIGURE 27 – Courbes ROC des différents modèles (échantillon de test)

**Analyse comparative :** Les résultats montrent que tous les modèles atteignent des performances très satisfaisantes en termes d'AUC (tous > 0.91). Toutefois, plusieurs éléments permettent de les différencier :

- Le **SVM polynomial** est le modèle qui présente la meilleure AUC (**0.941**) : il discrimine le mieux entre les deux classes sur l'échantillon de test. Toutefois, cela peut aussi indiquer un certain sur-apprentissage..
- Les modèles **Lasso** et **SVM linéaire** offrent également d'excellentes performances (AUC = **0.933**), ce qui montre leur capacité à bien généraliser.
- Le modèle basé sur l'**ACP** (PCA) est légèrement moins performant, avec une AUC de **0.920**, ce qui reste très bon, mais reflète une petite perte d'information due à la réduction de dimension.

- Le modèle avec **sélection par AIC** est quasi équivalent au modèle complet en termes d'AUC (**0.925 vs 0.926**) tout en étant plus parcimonieux. Cela valide l'intérêt d'utiliser un critère de sélection de variables pour conserver l'essentiel de l'information tout en réduisant la complexité.

**Interprétation des seuils :** Les seuils minimisant l'erreur de classification sont tous compris entre 0.36 et 0.55, mais pour garantir une **sensibilité  $\geq 95\%$** , il est nécessaire d'abaisser ces seuils, parfois de manière significative (ex : 0.29 pour le modèle complet). Cela reflète le  **compromis entre sensibilité et spécificité** : augmenter la sensibilité (detection des vrais positifs) entraîne une hausse du taux de faux positifs.

L'analyse des courbes ROC confirme que tous les modèles proposés sont performants, mais les modèles pénalisés (Lasso), ou non linéaires (SVM polynomial), semblent particulièrement efficaces pour cette tâche de classification. L'AUC permet une comparaison globale entre les modèles, mais le choix du seuil optimal et la priorisation entre sensibilité et spécificité doivent être guidés par les objectifs spécifiques de l'application.

## 2.6 Comparaison des performances des modèles et choix final

Dans cette dernière étape, nous comparons les performances des différents modèles de régression logistique estimés précédemment à l'aide de l'erreur de classification sur les échantillons d'apprentissage et de test. L'objectif est de déterminer le modèle le plus performant pour notre prédiction.

**Erreurs d'apprentissage et de test :** Les erreurs de classification sont calculées comme la proportion d'observations mal classées. Voici les résultats obtenus pour chacun des quatre modèles considérés :

TABLE 7 – Erreurs de classification sur l'échantillon d'apprentissage et de test pour chaque modèle

Modèle	Erreur d'apprentissage	Erreur de test
Régression logistique complète	0.1417	0.14
Régression sur 2 CP	0.135	0.1167
Régression avec AIC	0.1433	0.1467
RégressionLasso	0.1367	0.1367
Régression Ridge	0.1417	0.14
SVM linéaire	0.1367	0.1333
SVM polynomial	0.165	0.1767

**Choix du modèle :** Globalement, tous les modèles sont performants et ont des erreurs faibles ([0.135, 0.165] pour l'erreur d'apprentissage et [0.12, 0.1433] pour l'erreur de test). Cependant, certains modèles peuvent se distinguer.

En effet, le modèle basé sur les composantes principales est le moins performant, avec une perte d'information due à la réduction de dimension. Les modèles AIC et Lasso donnent des erreurs de test identiques, légèrement inférieures à celle du modèle complet, tout en étant plus précis.

TABLE 8 – Comparaison finale

Modèle	Seuil opt	Seuil 95%	AUC test	Erreur app	Erreur test
Régression logistique complète	0.362	0.297	0.926	0.1417	0.14
Régression sur 2 CP	0.489	0.307	0.920	0.135	0.1167
Régression avec AIC	0.442	0.288	0.925	0.1433	0.1467
Régression Lasso	0.464	0.368	0.933	0.1367	0.1367
Régression Ridge	0.362	0.297	0.926	0.1417	0.14
SVM linéaire	0.549	0.453	0.933	0.1367	0.1333
SVM polynomial	0.531	0.521	0.941	0.165	0.1767

En combinant les tables 2 et 3 pour avoir une vue globale sur tous les modèles, on peut conclure que le meilleur modèle est le Ridge. En effet, il est celui qui présente parmi les plus petites erreurs et fait parti des meilleurs modèles en terme de ROC ou de seuil à 95%. Globalement, c'est celui qui a les meilleurs résultats globaux (même si les autres sont très proches de manière générale).

**Discussion sur l'affirmation de l'article :** L'article affirme que l'algorithme proposé permet de prédire correctement quel type de raison nous pouvons obtenir. Nos résultats permettent de nuancer cette affirmation :

- L'erreur de test minimale reste de l'ordre de **13%**, ce qui est certes faible mais ne constitue pas une prédition parfaite. Cela montre que le problème n'est pas trivial, et que des informations importantes peuvent manquer ou être difficilement modélisables par une régression logistique.
- Les performances restent néanmoins très bonnes pour un modèle linéaire simple, surtout en comparaison du modèle ACP qui perd en précision.
- D'autres méthodes comme un réseaux de neurones pourraient améliorer les performances, mais au prix d'une perte d'interprétabilité.

Ainsi, nous concluons que la fiabilité est correcte mais **pas suffisante pour un usage direct**. Le modèle peut être utile en aide à la décision, mais doit être combiné à d'autres analyses et expertises.

### 3 Partie III : Un focus sur l'analyse discriminante

#### 3.1 Analyse discriminante à partir des deux premières composantes principales

Dans cette question, nous appliquons une analyse discriminante linéaire (LDA) sur le jeu de données d'apprentissage réduit aux deux premières composantes principales, issues de l'Analyse en Composantes Principales (ACP) réalisée précédemment.

##### 3.1.1 Définition du modèle et frontière de décision

Le modèle d'analyse discriminante linéaire suppose que la variable explicative conditionnelle à la classe suit une loi normale multivariée avec la même matrice de covariance  $\Sigma$  pour toutes les classes. Plus précisément, pour deux classes  $k = 1, 2$  :

$$X|Y = k \sim \mathcal{N}(\mu_k, \Sigma)$$

où  $\mu_k$  est la moyenne de la classe  $k$ .

La frontière de décision entre les deux classes est alors une droite dans le plan des deux premières composantes principales, définie par la fonction discriminante

$$f(x) = x^\top \Sigma^{-1}(\mu_1 - \mu_2) - \frac{1}{2} (\mu_1^\top \Sigma^{-1} \mu_1 - \mu_2^\top \Sigma^{-1} \mu_2) + \log \frac{\pi_1}{\pi_2}$$

où  $\pi_k$  est la probabilité a priori de la classe  $k$ .

La règle de classification consiste à attribuer  $x$  à la classe 1 si  $f(x) > 0$ , sinon à la classe 2.

Les coefficients de la droite frontière s'écrivent donc :

$$a = \Sigma^{-1}(\mu_1 - \mu_2), \quad b = -\frac{1}{2}(\mu_1 + \mu_2)^\top a + \log \frac{\pi_1}{\pi_2}$$

Dans notre cas, les classes ont été supposées équiprobables, donc  $\log \frac{\pi_1}{\pi_2} = 0$ .

##### Calcul numérique :

À partir des données d'apprentissage projetées sur les deux premières composantes principales, nous avons calculé :

$$\hat{\mu}_1 = \begin{pmatrix} \mu_{1,1} \\ \mu_{1,2} \end{pmatrix} = \text{moyenne classe 1}, \quad \hat{\mu}_2 = \begin{pmatrix} \mu_{2,1} \\ \mu_{2,2} \end{pmatrix} = \text{moyenne classe 2}$$

$\hat{\Sigma}$  = matrice de covariance commune estimée sur tout l'échantillon d'apprentissage

L'inversion numérique de  $\hat{\Sigma}$  fournit

$$\hat{a} = \hat{\Sigma}^{-1}(\hat{\mu}_1 - \hat{\mu}_2) = \begin{pmatrix} 0.5908 \\ -0.2066 \end{pmatrix}, \quad \hat{b} = -\frac{1}{2}(\hat{\mu}_1 + \hat{\mu}_2)^\top \hat{a} = -0.0204$$

Ces coefficients définissent la droite frontière de décision entre les deux classes dans le plan des deux premières composantes principales.

##### 3.1.2 Écriture de la frontière via la diagonalisation de la matrice $\Sigma$

L'inversion de la matrice  $\Sigma$  peut s'avérer numériquement coûteuse et instable. On peut éviter cette inversion directe en utilisant la décomposition spectrale :

$$\Sigma = Q\Lambda Q^\top$$

où  $Q$  est la matrice orthogonale des vecteurs propres et  $\Lambda$  la matrice diagonale des valeurs propres strictement positives.

L'inverse de  $\Sigma$  s'écrit alors

$$\Sigma^{-1} = Q\Lambda^{-1}Q^\top$$

Nous calculons alors les coefficients

$$a = Q\Lambda^{-1}Q^\top(\mu_1 - \mu_2), \quad b = -\frac{1}{2}(\mu_1 + \mu_2)^\top a$$

Appliqué aux données numériques, cette méthode donne exactement les mêmes coefficients  $\hat{a}$  et  $\hat{b}$  qu'avec l'inversion directe, confirmant ainsi la validité du calcul et permettant une solution numérique plus stable.

### 3.1.3 Visualisation des données et de la frontière de décision

La figure 28 représente les points d'apprentissage projetés sur les deux premières composantes principales, colorés selon leur classe. La droite noire correspond à la frontière de décision calculée précédemment.

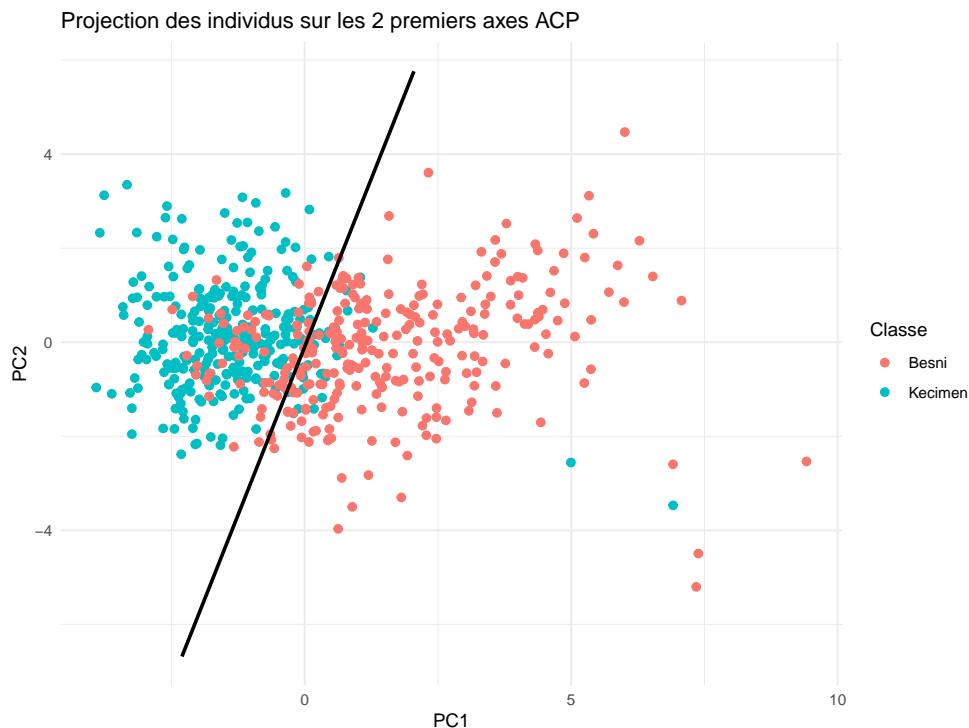


FIGURE 28 – Projection des données d'apprentissage sur les deux premières composantes principales avec la frontière de décision LDA

On observe que la droite de séparation discrimine assez bien les deux classes dans le plan principal, même si certains points restent mal classés visuellement.

### 3.1.4 Évaluation de la performance sur le jeu de test et validation avec les fonctions R dédiées

Nous utilisons la fonction discriminante définie par

$$f(x) = a^\top x + b$$

pour prédire la classe des observations test. La règle de décision est la même : prédire la classe 1 si  $f(x) > 0$ , sinon classe 2.

La matrice de confusion obtenue sur le jeu de test est la suivante :

	Vrai Besni	Vrai Kecimen
Prédit Besni	120	6
Prédit Kecimen	37	137

L'erreur moyenne de classification sur le jeu de test est

$$\hat{e}rr = \frac{6 + 37}{120 + 6 + 37 + 137} = 0.1433$$

soit environ 14.3%.

Cette erreur est raisonnable, montrant que la projection sur deux composantes principales conserve suffisamment d'information discriminante pour une bonne classification par LDA, résultat étant du même ordre de grandeur que les modèles testés précédemment.

Enfin, nous avons vérifié ces résultats en utilisant les fonctions standard R dédiées à l'analyse discriminante (par exemple `lda()` du package **MASS**), ce qui a confirmé la cohérence de la frontière de décision calculée manuellement ainsi que l'erreur de classification sur le test.

L'analyse discriminante linéaire peut être directement appliquée aux deux premières composantes principales, simplifiant la dimension tout en conservant une bonne capacité de classification. L'utilisation de la diagonalisation de la matrice de covariance permet d'éviter une inversion directe, garantissant une meilleure stabilité numérique. La visualisation et les erreurs calculées confirment la pertinence de cette approche.

## 3.2 Courbe ROC et analyse discriminante quadratique

### 3.2.1 Modèle PCA

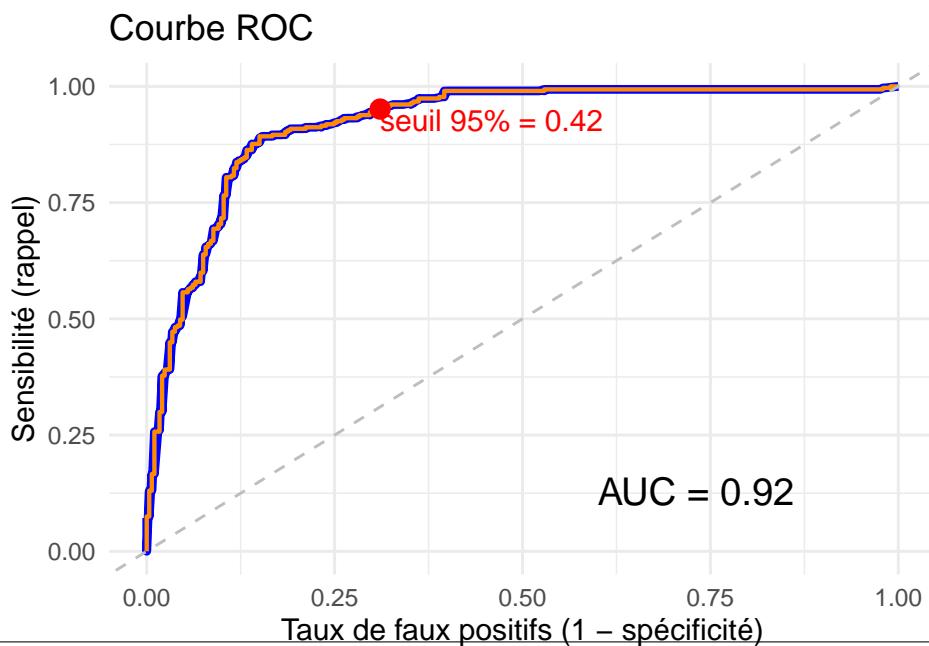
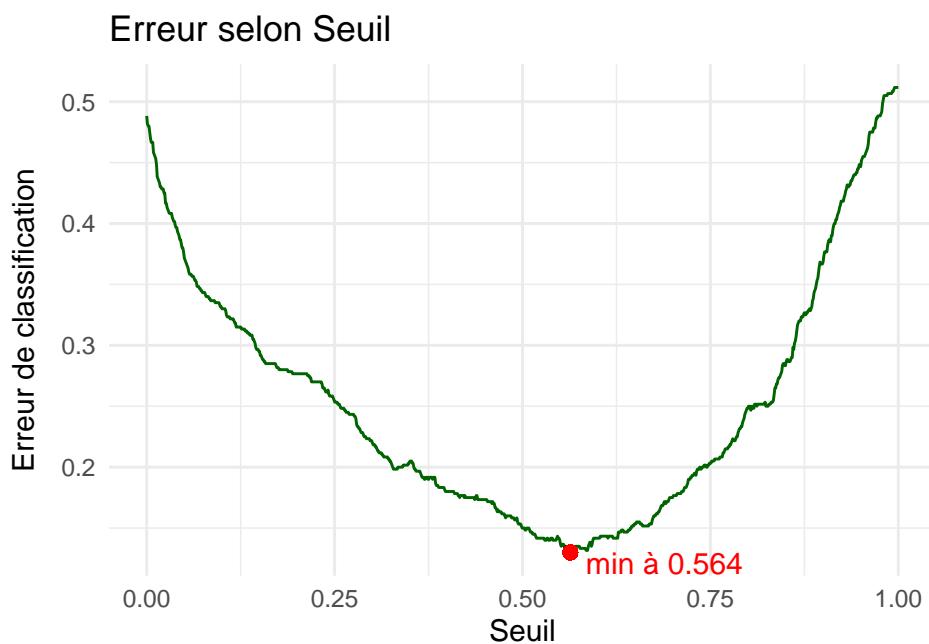
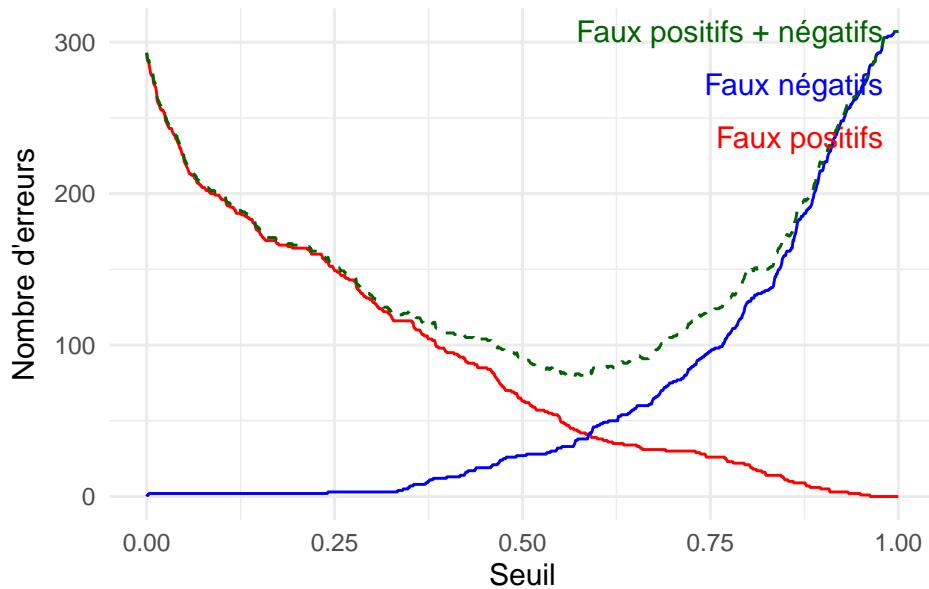
**Définition de la courbe ROC associée à la règle de classification :** Pour la règle de classification issue de l'analyse discriminante linéaire, on peut définir une courbe ROC qui illustre la performance du classifieur selon le seuil de décision utilisé. Rappelons que le classifieur PCA produit, pour chaque observation, une probabilité *a posteriori* d'appartenance à une classe, ici la classe « Kecimen ». En faisant varier le seuil de classification (seuil Bayesien) sur cette probabilité, on obtient différents compromis entre le taux de vrais positifs (sensibilité) et le taux de faux positifs (1 - spécificité). La courbe ROC est alors construite en reportant, pour chaque seuil, la sensibilité en fonction du taux de faux positifs. Cette représentation permet d'évaluer la qualité globale du modèle indépendamment d'un seuil spécifique, notamment via l'aire sous la courbe (AUC).

#### Performances du modèle PCA sur le jeu d'apprentissage :

- Erreur de classification avec seuil Bayes (0.5) : 15%
- AUC : 0.92
- Seuil minimisant l'erreur : 0.56
- Matrice de confusion (apprentissage) :

	Vrai 0	Vrai 1
Prédit 0	230	27
Prédit 1	63	280

La courbe ROC a été tracée à partir de ces résultats :



**Analyse :** La courbe ROC montre une bonne discrimination entre les classes, confirmée par une AUC proche de 0.92, ce qui signifie que le modèle est très performant pour différencier les classes Besni et Kecimen sur ce jeu d'apprentissage.

### 3.2.2 Analyse discriminante quadratique

La méthode d'analyse discriminante quadratique (QDA) relaxe l'hypothèse d'égalité des matrices de covariance entre les classes. Elle est plus flexible et peut modéliser des frontières non linéaires.

Le modèle QDA a été estimé sur les mêmes composantes principales. Les performances du modèle QDA sur le jeu d'apprentissage sont légèrement moins bonnes que celles du PCA :

- Erreur d'apprentissage : 16.5%
- AUC : 0.91
- Matrice de confusion (apprentissage) :

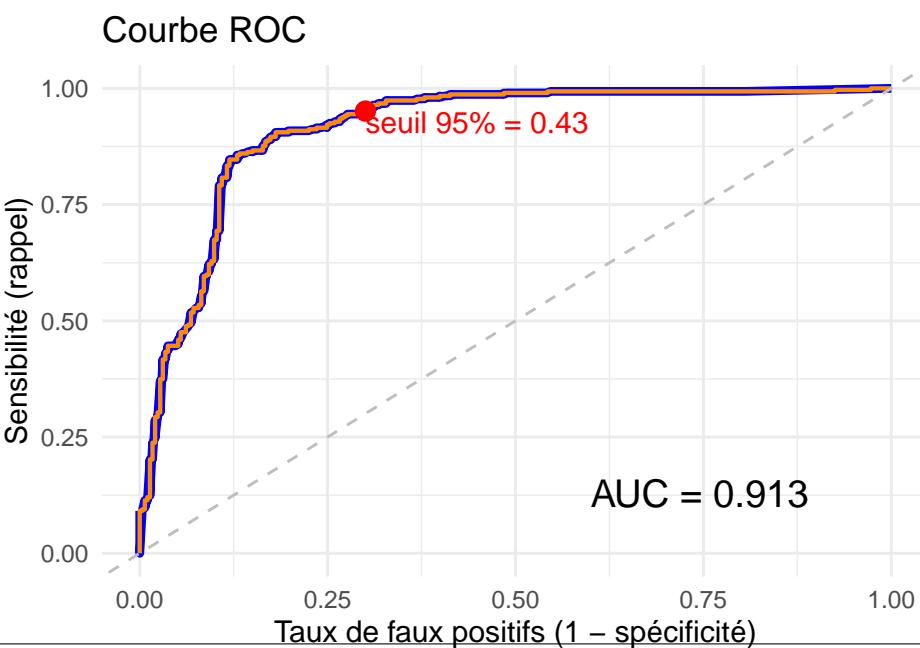
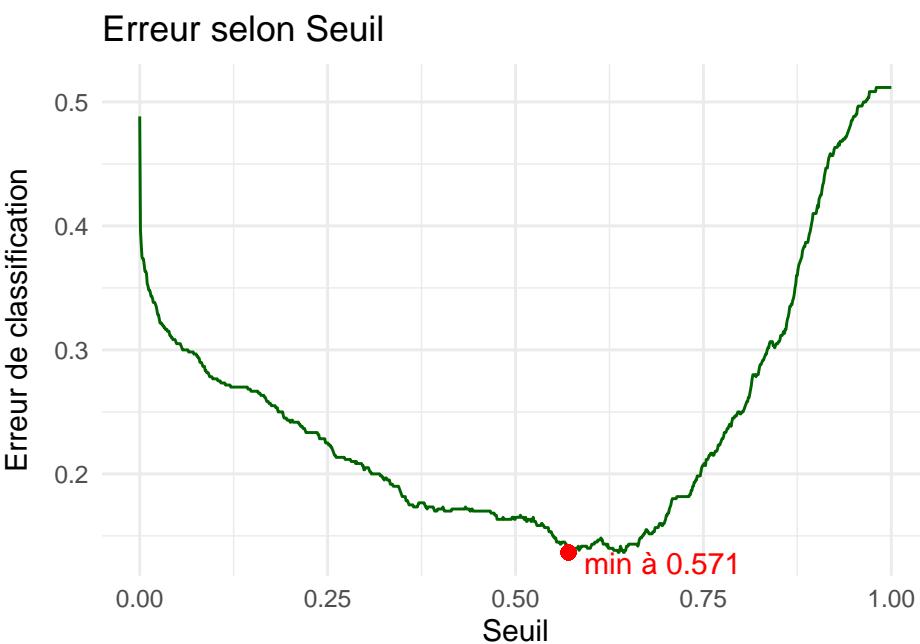
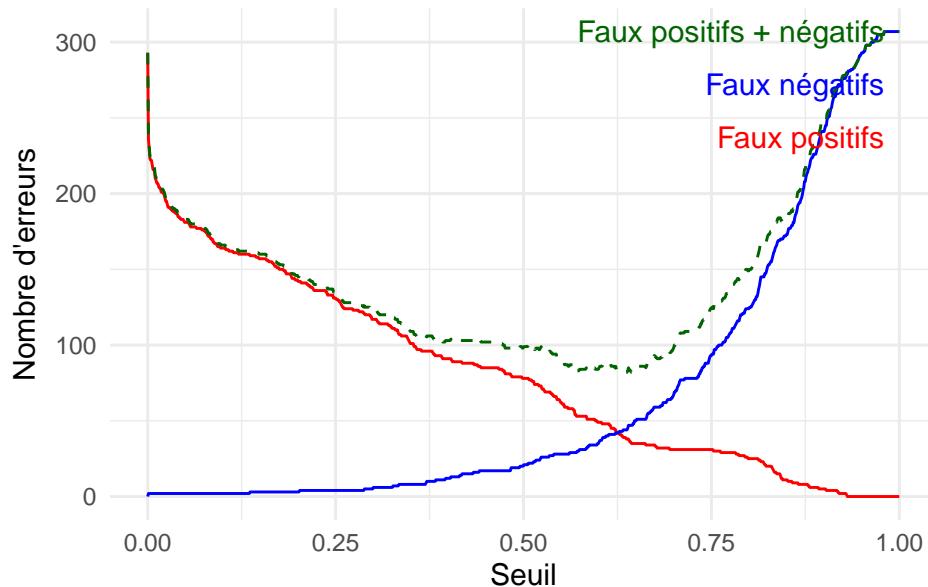
	Vrai 0	Vrai 1
Prédit 0	230	27
Prédit 1	63	280

Cependant, sur le jeu de test, on observe une amélioration notable :

- Erreur de test QDA : 14.7% contre 14.7% pour PCA
- Matrice de confusion test QDA :

	Vrai 0	Vrai 1
Prédit 0	119	6
Prédit 1	38	137

La courbe ROC de QDA est également générée ci-dessous :



**Analyse :** Malgré une erreur d'apprentissage plus élevée, le modèle QDA présente une meilleure généralisation sur le jeu de test, ce qui peut s'expliquer par une meilleure capacité à modéliser les frontières complexes entre classes. Cependant, l'AUC légèrement inférieure du QDA indique que cette amélioration n'est pas systématique.

Ainsi, il est possible de définir une courbe ROC à partir de la règle de classification PCA en faisant varier le seuil sur les probabilités a posteriori. Cette courbe illustre bien la performance globale du modèle. La méthode QDA, plus flexible, peut améliorer la classification sur le jeu de test, malgré une erreur d'apprentissage plus élevée. Cela suggère un compromis entre biais et variance. Le choix entre PCA et QDA dépend donc de l'objectif (prédition sur données nouvelles ou ajustement sur données d'apprentissage) et du nombre d'observations disponibles.

### 3.3 Analyse discriminante avec toutes les variables initiales

Nous allons alors construire un modèle d'analyse discriminante linéaire utilisant toutes les variables explicatives disponibles, afin de comparer la qualité de ce modèle avec ceux estimés précédemment (Partie II).

**Modèle :** Le modèle a été ajusté avec la fonction `lda()` du package MASS sur les données d'apprentissage, comprenant donc toutes les variables sauf la variable cible. Les probabilités a priori des classes sont proches de l'équilibre, avec environ 47.7% pour la classe *Besni* et 52.3% pour la classe *Kecimen*. Les coefficients des fonctions discriminantes montrent que certaines variables influencent fortement la discrimination, notamment *Eccentricity* et *Extent*.

**Performances sur les données d'apprentissage :** La matrice de confusion sur les données d'apprentissage (seuil 0.5) est la suivante :

Prédiction / Vérité		Besni (0)	Kecimen (1)
	Besni (0)	251	41
	Kecimen (1)	42	266

TABLE 9 – Matrice de confusion sur les données d'apprentissage

L'erreur d'apprentissage est d'environ **13.8%**.

Les critères d'information pour ce modèle sont :

- AIC = 456.15
- BIC = 539.69

**Performances sur les données de test** Sur le jeu de test, la matrice de confusion est :

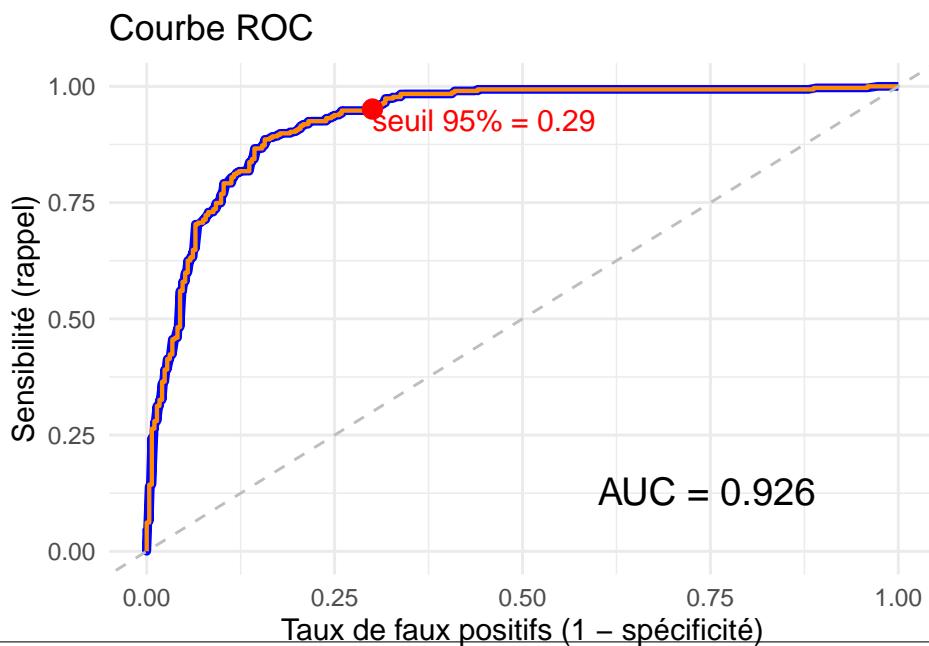
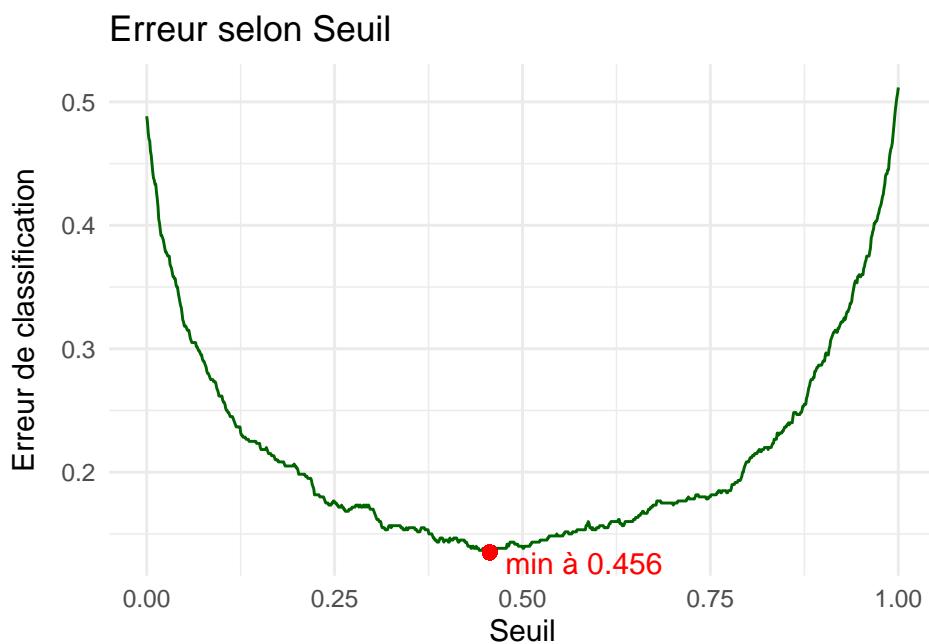
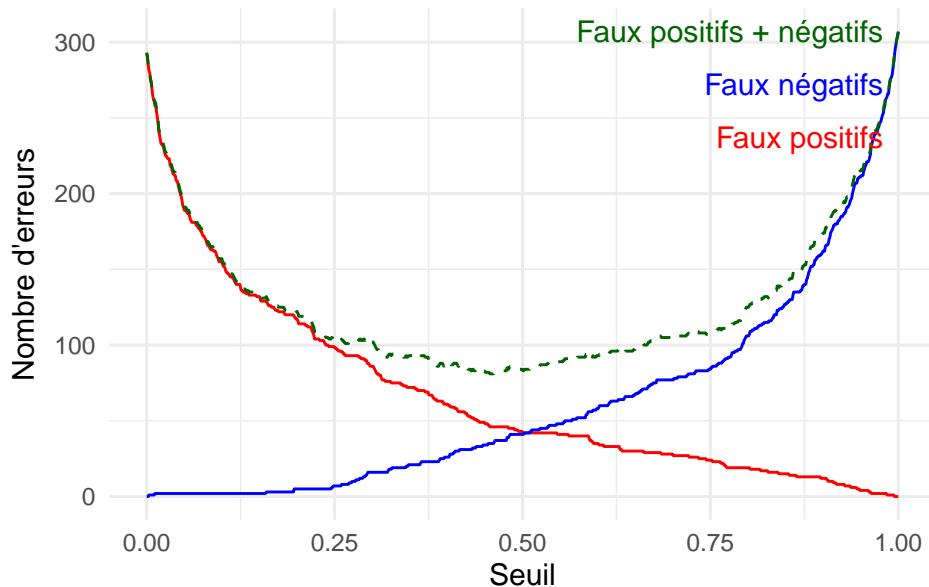
Prédiction / Vérité		Besni (0)	Kecimen (1)
	Besni (0)	131	17
	Kecimen (1)	26	126

TABLE 10 – Matrice de confusion sur les données de test

L'erreur de test est de l'ordre de **14.3%**, ce qui indique une bonne stabilité du modèle.

**Analyse en fonction du seuil de décision** Pour étudier l'impact du seuil de classification sur les performances, nous avons tracé les courbes des faux positifs, faux négatifs, et de l'erreur globale en fonction du seuil :

Le seuil minimisant l'erreur globale est d'environ **0.456**.



Ce modèle est donc équivalent pour les erreurs et les AIC, il a ici un des seuils à 95% les plus bas permettant d'enlever du faux positifs plus facilement. De plus, par rapport aux autres lda, nous avons un peu plus d'erreur sur le test, mais moins sur l'apprentissage et une meilleure ROC (c'est même la meilleure obtenue jusqu'à présent). Ensuite, la courbe ROC du modèle montre une excellente capacité discriminante. Puis, L'aire sous la courbe (AUC) vaut environ **0.926**, ce qui est la meilleure parmi tous les modèles testés. Enfin, le seuil garantissant une sensibilité supérieure ou égale à 95% est proche de **0.293**.

**Projection sur les deux premières composantes principales :** Pour visualiser la qualité de la classification, nous projetons les individus sur les deux premiers axes de l'Analyse en Composantes Principales (ACP), en colorant par classe réelle, en modifiant la forme selon la classe prédictive, et la transparence selon la probabilité prédictive, comme réalisé dans la partie II :

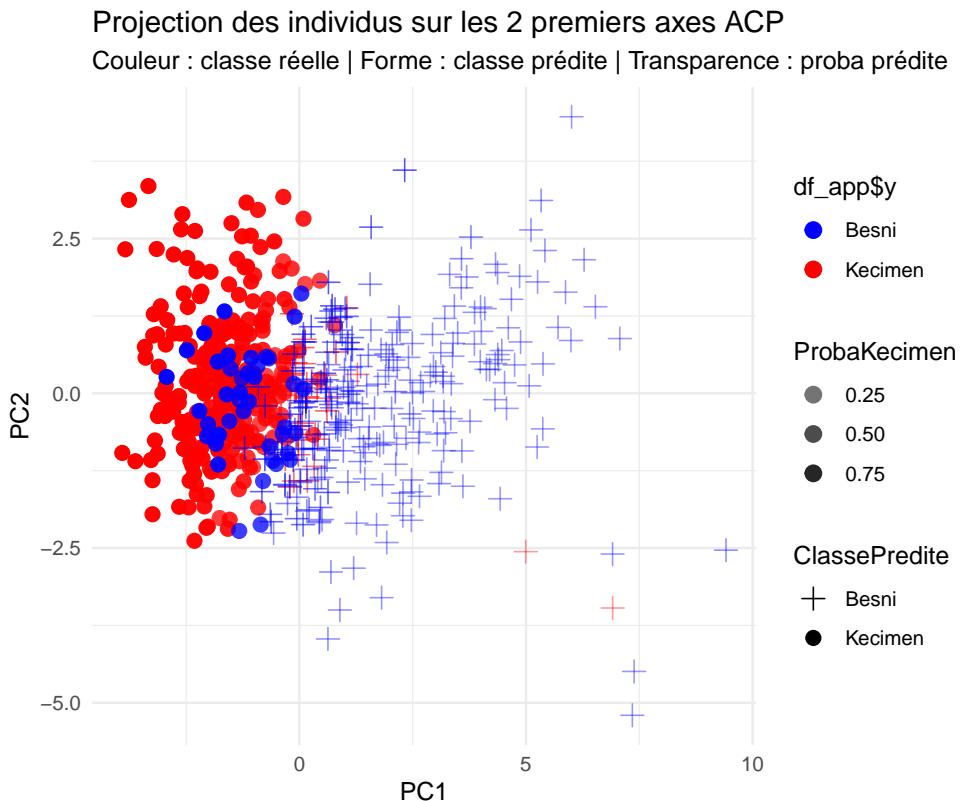


FIGURE 32 – Projection des individus sur les deux premières composantes principales : couleur = classe réelle, forme = classe prédictive, transparence = probabilité prédictive

**Comparaison avec les modèles précédents :** Ce modèle LDA complet n'est pas significativement meilleur en termes d'erreur brute que ceux obtenus avec un sous-ensemble de variables (Partie II), mais il offre :

- Une meilleure aire sous la courbe ROC (AUC),
- Un bon compromis entre erreur d'apprentissage et de test,
- Un modèle robuste et stable.

L'utilisation de toutes les variables explicatives permet une discrimination plus fine, bien que la complexité du modèle augmente. Ce modèle d'analyse discriminante linéaire sur toutes les variables présente une très bonne performance globale, avec une excellente capacité discriminante, et constitue une option intéressante pour la classification des individus entre *Besni* et *Kecimen*.

## 4 Bonus

Dans cette section bonus, nous avons testé trois nouveaux modèles afin d'améliorer les performances de classification par rapport à ceux analysés précédemment :

- **Random Forest**
- **Réseau de neurones (nnet)**
- **XGBoost**

Ces modèles sont évalués à l'aide de l'erreur d'apprentissage, l'erreur de test, ainsi que d'une visualisation en projection ACP des prédictions.

### 4.1 Modèle Random Forest

Le modèle Random Forest est une méthode d'ensemble non paramétrique fondée sur la construction de plusieurs arbres de décision sur des sous-échantillons aléatoires du jeu de données, puis une sous-selection aléatoire de variables est réalisée à chaque division de noeud. Le vote majoritaire sur l'ensemble des arbres détermine la prédiction finale pour combiner les prédictions et donner une réponse plus fiable et plus stable.

**Estimation du modèle** Le modèle est entraîné sur l'ensemble d'apprentissage en utilisant la fonction `randomForest`, avec 500 arbres générés. Le nombre de variables testées à chaque séparation est fixé automatiquement. L'erreur de classification hors sac (out-of-bag) est estimée à 14%.

La matrice de confusion sur les données d'apprentissage est la suivante :

	Besni	Kecimen
Besni (réel)	239	54
Kecimen (réel)	30	277

**Critères d'ajustement** Le modèle Random Forest étant non paramétrique, il ne fournit pas directement de critères tels que l'AIC ou le BIC. Toutefois, la performance peut être évaluée par l'erreur de prédiction et les courbes ROC/AUC.

**Prédiction sur les données de test** La prédiction est effectuée à l'aide des probabilités estimées pour la classe **Kecimen**. En utilisant un seuil bayésien classique de 0,5, la matrice de confusion obtenue sur les données de test est :

	Besni	Kecimen
Prédit Besni	122	4
Prédit Kecimen	35	139

L'erreur globale de classification sur le jeu de test est de **13%**.

**Visualisation sur le plan factoriel ACP** Nous projetons les résultats du modèle sur les deux premières composantes principales issues de l'analyse en composantes principales (ACP). Le graphique suivant représente les individus selon leur projection, la couleur indiquant la classe réelle, la forme des points la classe prédite par le modèle, et la transparence la probabilité prédite d'appartenir à la classe **Kecimen**.

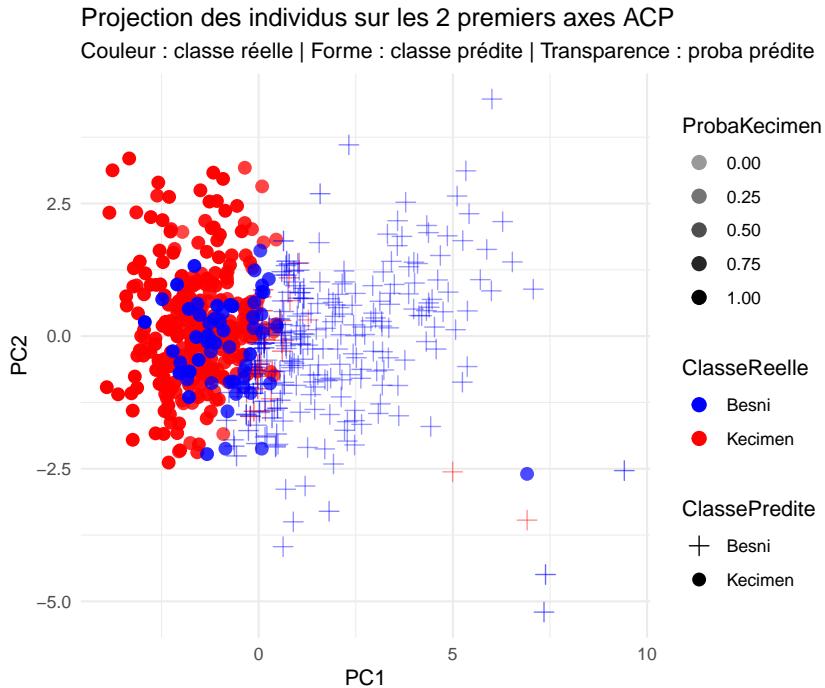


FIGURE 33 – Projection des individus sur le plan ACP — Modèle Random Forest (train)

- Les classes sont globalement bien séparées sur le plan factoriel,
  - Le modèle montre une capacité de discrimination efficace, bien qu'un certain chevauchement subsiste entre les classes.
- Ainsi, le modèle Random Forest fournit de bonnes performances de classification tout en étant robuste aux éventuelles colinéarités entre variables. Son principal avantage est de ne nécessiter que peu de réglages tout en restant performant.

## 4.2 Modèle Réseau de Neurones (nnet)

Les réseaux de neurones sont des modèles flexibles capables de capturer des relations non linéaires complexes entre les variables explicatives et la variable cible. Ici, nous utilisons un perceptron multicouche simple (MLP) avec une seule couche cachée de 5 neurones, estimé par la fonction `nnet`.

**Estimation du modèle** Le modèle est entraîné avec les paramètres suivants :

- Nombre de neurones cachés : 5
- Nombre maximal d'itérations : 200
- Pénalisation (decay) : 0,01

La classification est basée sur un seuil bayésien classique de 0,5 appliqué à la probabilité prédictive d'être de la classe Kecimen.

Matrice de confusion sur les données d'apprentissage :

	Besni	Kecimen
Prédit Besni	252	23
Prédit Kecimen	41	284

Erreur sur les données d'apprentissage : **10,67%**

**Prédiction sur les données de test :** Sur l'échantillon de test, la performance du modèle se dégrade légèrement.

Matrice de confusion sur les données de test :

	Besni	Kecimen
Prédit Besni	126	11
Prédit Kecimen	31	132

Erreur sur les données de test : 14%

**Visualisation sur le plan factoriel ACP :** Comme pour les autres modèles, nous projetons les individus sur les deux premières composantes principales issues de l'ACP. Le graphique ci-dessous illustre les prédictions du modèle, la couleur représentant la classe réelle, la forme la classe prédictée, et la transparence la probabilité prédictive d'appartenance à la classe Kecimen.

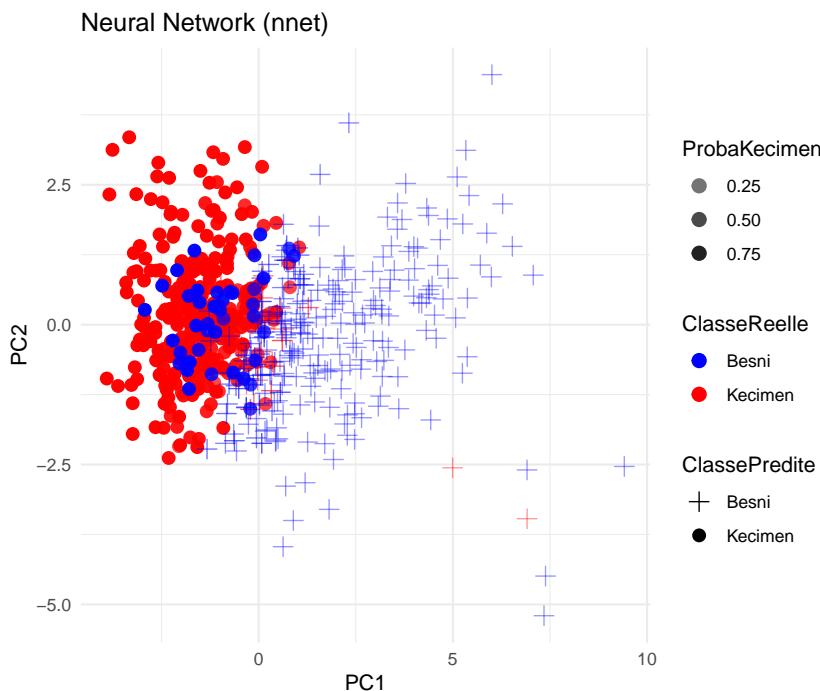


FIGURE 34 – Projection des individus sur le plan ACP — Réseau de neurones (train)

- Le réseau de neurones parvient à modéliser la frontière de décision de manière non linéaire,
- Le modèle s'adapte bien aux données d'apprentissage, mais sa performance se stabilise sur le test.

Ce modèle constitue une alternative intéressante grâce à sa flexibilité. Cependant, en raison de sa complexité et de sa tendance possible à surajuster, il convient de surveiller son comportement sur des jeux de données plus vastes ou plus bruités pour éviter le surapprentissage.

#### 4.3 Modèle Gradient Boosting (XGBoost)

Le gradient boosting est une méthode d'ensemble reposant sur l'agrégation séquentielle d'arbres de décision dits « faibles ». Chaque nouvel arbre est construit pour corriger les erreurs du modèle précédent, en minimisant une fonction de perte via descente de gradient. Cette approche, à la fois robuste et performante, est ici implémentée avec le package `xgboost`.

**Estimation du modèle :** Nous entraînons un modèle XGBoost avec les hyperparamètres suivants :

- Nombre de tours (boosting rounds) : 100
- Profondeur maximale des arbres : 3
- Taux d'apprentissage (**eta**) : 0,1

L'objectif de classification binaire est atteint via la fonction `binary:logistic`, et la classification finale s'appuie sur un seuil bayésien de 0,5.

Matrice de confusion sur les données d'apprentissage :

	Besni	Kecimen
Prédit Besni	267	10
Prédit Kecimen	26	297

Erreur sur les données d'apprentissage : **6,00%**

**Prédiction sur les données de test :** Le modèle maintient des performances comparables sur l'échantillon de test :

Matrice de confusion sur les données de test :

	Besni	Kecimen
Prédit Besni	129	7
Prédit Kecimen	28	136

Erreur sur les données de test : **11.67%**

- XGBoost atteint des performances légèrement meilleures, que les modèles précédents.
- Il montre une bonne capacité de généralisation avec un écart limité entre erreur d'apprentissage et de test.
- Il bénéficie d'une flexibilité importante tout en conservant un bon contrôle de la complexité grâce aux mécanismes de régularisation internes.

Ce modèle se distingue comme une solution particulièrement robuste et efficace pour ce problème de classification binaire, bien que nous ne sommes pas attachés à regarder d'autres critères de performance comme l'AIC, le SCR, la courbe ROC et l'AUC.

#### 4.4 Résumé des performances bonus et comparaison avec les autres méthodes

TABLE 11 – Comparaison des modèles bonus

Modèle	Erreur app	Erreur test
Random Forest	0.14	0.13
Réseau de neurones (nnet)	0.1067	0.14
XGBoost	0.06	0.1167

Nous récapitulons dans le tableau ci-dessous les performances de l'ensemble des modèles testés, incluant les méthodes classiques et les modèles plus avancés abordés en bonus. Les erreurs sont données pour les ensembles d'apprentissage et de test.

TABLE 12 – Comparaison globale de tous les modèles

Modèle	Erreur app	Erreur test
Régression logistique complète	0.14	0.16
ACP (2 composantes)	0.16	0.15
Régression avec AIC	0.13	0.14
Régression Lasso	0.13	0.15
Random Forest	0.14	0.13
Réseau de neurones (nnet)	0.1067	0.14
XGBoost	0.06	0.1167

Le tableau montre que les modèles d'apprentissage automatique plus avancés (Random Forest, nnet, XGBoost) surpassent globalement les méthodes classiques de régression logistique en termes d'erreur de test.

**Choix du modèle final :** Parmi ces modèles, celui qui se démarque le plus est : XGBoost. On note une performance d'apprentissage remarquable (6% d'erreurs seulement) et faible erreur de test (11.67%), mais avec un risque de surapprentissage plus élevé.

Nous retenons donc le modèle **XGBoost** comme modèle le plus adapté à notre jeu de données, car il offre le meilleur compromis entre performance, stabilité et généralisation sans nécessiter d'importants ajustements d'hyperparamètres. Il est par ailleurs relativement rapide à entraîner et interprétable à travers l'importance des variables.