DETECTION OF MICROCALCIFICATIONS IN MAMMOGRAPHY

Coding Week
Rapport

Rapport :

Élèves :
TEMRI Othmane
ELYOUBI Asmae
KHALIL Mohammed Adam
YAAGOUBI Malak
CHERKAOUI Ismail
MIDINE Zakaria

Enseignant :
Kawtar ZERHOUNI

Table des matières

Abstract :

The project aims to detect breast cancer from radiological examinations using a mammography dataset, which contains microcalcifications indicating cancer and non-microcalcifications which are negative cases. For this, six features will be extracted from the segmented objects and two popular machine learning algorithms, namely Support Vector Machines and Random Forest, will be used. However, due to the large class imbalance in the dataset, with only 2% positive cases, cost-sensitive techniques will be implemented to address this issue. These techniques involve penalizing the model for misclassifications of one class more than the other, which will improve the performance of the model on the minority class. It is thus expected that the use of these cost-sensitive learning techniques significantly improves the predictive performance of the model for the minority class, which is essential for cancer detection.

Resumé :

Le projet a pour objectif de détecter le cancer du sein à partir d'examens radiologiques en utilisant un ensemble de données de mammographie, qui contient des microcalcifications indiquant un cancer et des non-microcalcifications qui sont des cas négatifs. Pour cela, six caractéristiques seront extraites des objets segmentés et deux algorithmes d'apprentissage automatique populaires, à savoir Support Vector Machines et Random Forest, seront utilisés. Toutefois, en raison du déséquilibre important des classes dans l'ensemble de données, avec seulement 2% de cas positifs, des techniques sensibles aux coûts seront mises en place pour résoudre ce problème. Ces techniques impliquent la pénalisation du modèle pour les erreurs de classification d'une classe plus que l'autre, ce qui permettra d'améliorer les performances du modèle sur la classe minoritaire. On s'attend ainsi à ce que l'utilisation de ces techniques d'apprentissage sensibles aux coûts améliore significativement les performances prédictives du modèle pour la classe minoritaire, ce qui est essentiel pour la détection du cancer.

1 Introduction

La détection précoce du cancer du sein est cruciale pour augmenter les chances de survie des patients. Les mammographies sont l'un des outils les plus couramment utilisés pour diagnostiquer cette maladie. Cependant, la classification des mammographies est un problème difficile en raison de l'importante inégalité entre les classes positives (cancer) et négatives (non-cancer).
Dans ce rapport, nous aborderons le problème de la détection du cancer du sein à partir de mammographies à l'aide de techniques de machine learning. Nous utiliserons le mammography dataset, un ensemble de données standard pour les problèmes de classification déséquilibrés, pour explorer l'utilisation de techniques telles que les machines à vecteurs de support et les forêts aléatoires. Nous nous concentrerons également sur l'utilisation de techniques de coût-sensibilité pour améliorer la performance de nos modèles sur les cas de cancer, qui sont d'une importance critiques.

2 Contextualisation

Les techniques d'imagerie médicale permettent d'acquérir des images de l'intérieur du corps humain en utilisant différents domaines physiques. Depuis leur introduction, ces techniques ont considérablement amélioré le diagnostic et le suivi thérapeutique en médecine. L'imagerie mammographique est la modalité médicale de référence pour l'analyse et le diagnostic du cancer du sein, et les experts en radiologie sont chargés d'interpréter les images mammographiques pour détecter les anomalies et déterminer les traitements appropriés.

Cependant, avec l'augmentation du nombre de mammographies réalisées en raison de l'augmentation de la fréquence d'apparition du cancer du sein, des systèmes d'aide à la décision (CAD) ont été développés pour fournir un deuxième avis aux radiologues et améliorer la précision de leur diagnostic. Ces systèmes ne visent pas à remplacer les radiologues, mais à fournir un soutien supplémentaire. Cependant, la qualité des images acquises et la maîtrise de la démarche d'interprétation suivie par le radiologue sont des facteurs clés pour la pertinence de ce deuxième avis.

Pour améliorer encore la précision des diagnostics, les chercheurs travaillent sur l'optimisation des systèmes CAD en utilisant des algorithmes d'apprentissage automatique pour la détection de lésions suspectes. L'objectif est de permettre aux systèmes CAD de reconnaître automatiquement les caractéristiques des lésions suspectes et de fournir des informations précises pour aider les radiologues à prendre des décisions éclairées. Ces avancées technologiques ont le potentiel d'améliorer considérablement la détection précoce du cancer du sein et, par conséquent, d'améliorer les résultats cliniques pour les patients

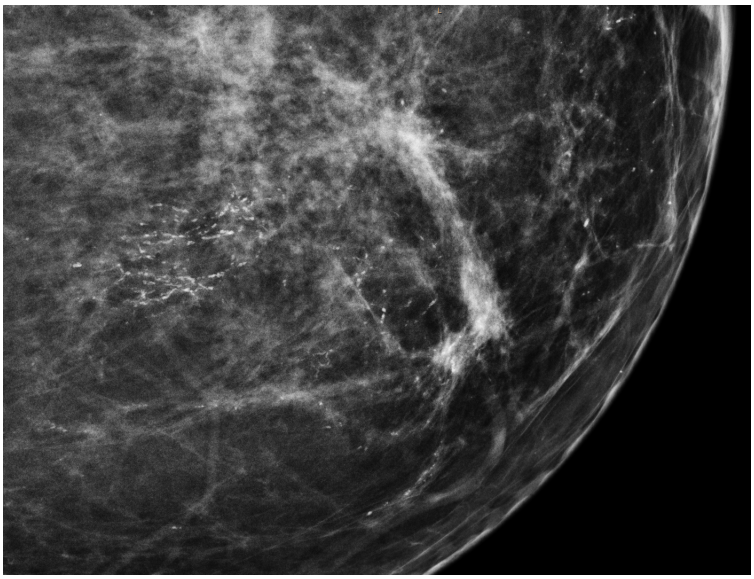


FIGURE 1 – Mammographie montrant la microlacification

3 Problematisation

Le dépistage précoce du cancer du sein est crucial pour améliorer considérablement les chances de guérison et la qualité de vie des patientes. Le cancer du sein est l'un des cancers les plus courants chez les femmes, avec environ 2,2 millions de nouveaux cas diagnostiqués chaque année dans le monde[1]. La mammographie est un outil couramment utilisé pour le dépistage précoce du cancer du sein, mais la détection des calcifications dans les images peut être difficile pour les radiologues et les médecins, car les calcifications peuvent être très petites et difficiles à distinguer du tissu mammaire normal. Cette situation peut entraîner un diagnostic tardif et une prise en charge inadéquate des maladies du sein, ce qui souligne la nécessité de développer des méthodes plus précises et efficaces pour la détection des calcifications dans les mammographies.

Les techniques de machine learning sont une solution prometteuse pour améliorer la détection des calcifications dans les mammographies. Ces techniques peuvent aider à détecter les anomalies dans les images et à distinguer plus facilement les calcifications du tissu mammaire normal. De plus, les algorithmes de machine learning peuvent apprendre à partir de grandes quantités de données et s'adapter pour améliorer continuellement leur précision. Cette approche pourrait permettre une détection précoce plus fiable et une prise en charge plus rapide et plus efficace des maladies du sein.

Le présent rapport a pour objectif d'analyser les différents défis liés la détection des microcalcifications dans les mammographies pour [4] un dépistage plus précoce et une prise en charge plus efficace des maladies du sein.

4 AI Machine learning

L'intelligence Artificielle vise à créer des machines capables de réaliser des tâches nécessitant normalement l'intelligence humaine. c'est un domaine interdisciplinaire qui utilise des techniques de mathématiques, de statistiques, d'informatique et de neuros-

ciences pour développer des algorithmes et des modèles capables d'apprendre à partir de données et de s'adapter à de nouvelles situations.

Bien que l'IA soit déjà omniprésente dans notre vie quotidienne, elle soulève également des questions éthiques et sociales qui doivent être prises en compte pour façonner un avenir meilleur. l'intelligence artificielle et le Maching learning

L'IA (intelligence artificielle) et le ML (Maching learning) sont deux termes étroitement liés. En effet, le ML est une branche de l'IA qui utilise des algorithmes pour permettre à une machine d'apprendre et de s'améliorer à partir de données. En d'autres termes, le ML est un sous-ensemble de l'IA qui permet à une machine de devenir plus intelligente en apprenant à partir de ses expériences passées. Ainsi, le ML est une technique clé utilisée dans la construction de systèmes d'IA sophistiqués qui peuvent prendre des décisions complexes en utilisant des données et des modèles[2].

les techniques de ML (apprentissage automatique) qui permettent à une machine d'apprendre à partir de données et de s'améliorer au fil du temps. Les principales techniques de ML sont les suivantes :

L'apprentissage supervisé : Dans cette technique, la machine apprend à partir de données étiquetées, c'est-à-dire des données qui ont été préalablement classées ou catégorisées. La machine utilise ces données étiquetées pour créer un modèle qui peut prédire des résultats pour de nouvelles données non étiquetées.

L'apprentissage non supervisé : Cette technique consiste à apprendre à partir de données non étiquetées. La machine utilise des algorithmes pour trouver des modèles ou des relations cachées dans les données et créer des groupes ou des clusters qui permettent de mieux comprendre les données.

L'apprentissage par renforcement : Cette technique consiste à apprendre à partir de l'interaction avec l'environnement. La machine apprend en recevant des récompenses ou des punitions pour chacune de ses actions, ce qui la pousse à prendre les meilleures décisions possibles.

4.1 Deep learning

Le Deep Learning est une branche de l'intelligence artificielle qui s'inspire du fonctionnement du cerveau humain pour analyser et apprendre à partir de données. Cette technique consiste à utiliser des réseaux de neurones artificiels qui sont capables de traiter des quantités massives de données pour effectuer des tâches complexes telles que la reconnaissance vocale, la détection d'objets ou encore la traduction automatique.

Le Deep Learning est utilisé dans de nombreux domaines tels que la finance, la santé, la reconnaissance faciale, la sécurité, la publicité et bien d'autres. Cette technique est particulièrement utile pour traiter et balancer les données car elle permet de traiter des ensembles de données très volumineux et complexes en un temps record. Les algorithmes de Deep Learning sont capables d'extraire des caractéristiques importantes des données brutes, ce qui permet d'obtenir des résultats de meilleure qualité et plus précis.

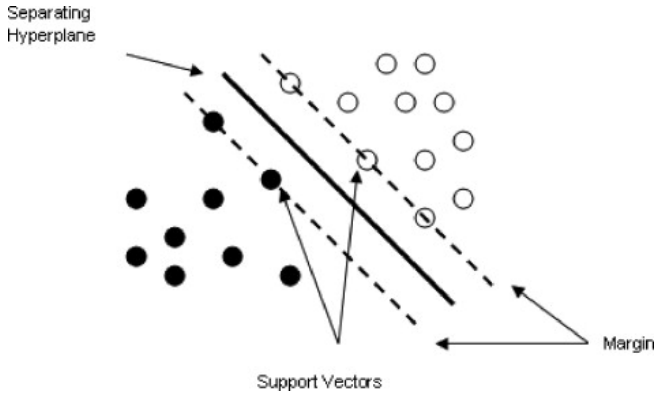
La classification est l'une des tâches les plus courantes pour lesquelles le Deep Learning est utilisé. La classification consiste à assigner une étiquette ou une catégorie à un objet en fonction de ses caractéristiques. Dans le Deep Learning, cette tâche est réalisée en utilisant des réseaux de neurones profonds qui sont entraînés à partir d'un ensemble de données d'apprentissage annotées. Le réseau apprend à reconnaître les motifs dans les données d'entrée et à associer ces motifs à des étiquettes de sortie.

Par exemple, dans le domaine de la reconnaissance d'image, les algorithmes de Deep Learning peuvent être utilisés pour identifier les caractéristiques clés d'une image, comme les contours, les textures et les couleurs, et ainsi permettre de classer une image en fonction de son contenu. Dans le domaine de la finance, le Deep Learning peut être utilisé pour prédire les tendances du marché, ce qui permet de prendre des décisions plus éclairées en matière d'investissement.

En somme, le Deep Learning est une technique très efficace pour la classification car elle permet de traiter des données complexes et d'apprendre des motifs dans les données d'entrée pour associer des étiquettes de sortie. Cette technique est utilisée dans de nombreux domaines pour résoudre des problèmes de classification complexes et améliorer les performances des systèmes de reconnaissance et de détection.

Support vector machines

Les SVMs sont une famille d'algorithmes d'apprentissage automatique qui permettent de résoudre des problèmes tant de classification que de régression ou de détection d'anomalie. Ils sont connus pour leurs solides garanties théoriques, leur grande flexibilité ainsi que leur simplicité d'utilisation même sans grande connaissance de data mining.



5 Etude théorique

5.1 Support Vector Machine

5.1.1 L'hyperplan

C'est un algorithme d'apprentissage automatique supervisé utilisé pour la classification et la régression. SVM est un algorithme de classification linéaire qui sépare les données en utilisant un hyperplan, en trouvant la frontière de décision optimale qui maximise la marge entre les deux classes. Lorsque les données ne sont pas linéairement séparables, le SVM utilise le "kernel trick" pour projeter les données dans un espace de dimension supérieure où elles peuvent être séparées linéairement. SVM est largement utilisé dans de nombreux domaines tels que la reconnaissance d'image.

5.1.2 Classifiers

$$h(x_i) = \begin{cases} +1 & w.x + b \geq 0 \\ -1 & if w.x + b0 \end{cases}$$

Le point au-dessus ou sur l'hyperplan sera classé comme classe +1, et le point en dessous de l'hyperplan sera classé comme classe -1.

Donc fondamentalement, l'objectif de l'algorithme d'apprentissage SVM est de trouver un hyperplan qui pourrait séparer les données avec précision. Il peut y avoir plusieurs hyperplans possibles. Et nous devons trouver le meilleur, souvent appelé[5] l'hyperplan optimal.

Dans un premier instant, considérons l'équation de l'hyperplan donnée par : w.x+b=0.

On sait qu'un point (x, y) appartient à l'hyperplan si : $w.x + b = 0$

Si le point (x, y) n'est pas dans l'hyperplan , la valeur de $w.x + b$ est soit positive soit négative. Pour tous les points de l'échantillon, Nous voulons connaître le point qui est le plus proche de l'hyperplan.

On peut envisager le calcul de | β = w.x + b |

Formellement :

étant donné $D = \{(x, y) \mid x_i \in R^n, y_i \in \{-1, 1\}_{i=1}^m\}$. On calcule β pour tous les échantillons , et B est la valeur minimale β. On a :

$$\min_{i=1...m} \mid w.x + b \mid$$

$$H = \max_{i=1...s} \{h_i \mid B_i\}$$

On définit $\gamma = y(\frac{w}{||w||}.x + \frac{b}{||w||})$ étant donné un dataset D, on calcule γ pour chaque exemple d'entrainement et M est dite la valeur minimale M est appelée la "marge géométrique" de dataset.

5.1.3 Le problème d'optimisation pour le SVM

Pour trouver les valeurs de w et b caractérisant l'hyperplan optimal, nous sommes amenés à résoudre un problème d'optimisation dont les contraintes sont :

$$\max_{w,b} M$$
$$telque \ \gamma_i \geq M, i = 1...m$$

On sait que $M = \frac{F}{||w||}$, on peut réécrire le problème en dessus :

$$\max_{w,b} M$$
$$telque \ f_i \geq F, i = 1...m$$

If we rescale w and b, we are still maximizing M and the optimization result will not change. Let's rescale w and b and make F=1, the above problem can be rewritten as :

$$\max_{w,b} \frac{1}{|| \ w \ ||}$$
$$telque \ f_i \geq 1, i = 1...m$$

$\Longleftrightarrow$

$$\min_{w,b} || \ w \ ||$$
$$telque \ f_i \geq 1, i = 1...m$$

$\Longleftrightarrow$

$$\min_{w,b} \frac{1}{2} || \ w \ ||^2$$
$$telque \ y_i(w.x + b) - 1 \geq 0, i = 1 \dots m$$

5.1.4 Résolution de problème d'optimisation

On peut reformuler le problème d'optimisation de SVM en utilisant la méthode de multiplication de Lagrange.

$$\nabla f(x) - \alpha \nabla g(x) = 0$$

ou α est le multiplicateur de Lagrange.

Ici, $f(w) = \frac{1}{2} \mid\mid w \mid\mid^2$ $g(w,b) = y_i(w.x + b) - 1, i = 1 \ldots m$. La fonction Langragienne est donnée par : $\mathcal{L}(w,b,\alpha) = \frac{1}{2} \mid\mid w \mid\mid^2 - \sum_{i=1}^{m} \alpha_i [y_i(w.x+b) - 1]$

Pour résoudre analytiquement $\nabla \mathcal{L}(w,b,\alpha) = 0$, le nombre d'échantillon doit être petit. Donc, on va réécrire le problème d'optimisation en utilisant le principe de dualité.

De manière equivalente, on résout le problème de Lagrange primal

$$\min_{w,b} \max \mathcal{L}(w,b,\alpha)$$
$$telque \;\; \alpha_i \geq 0, i = 1...m$$

Pus précisément, α doit être égal aux multiplieurs de Karush-Kuhn-Tucker parce que on a affaire à des contraintes/inégalités.

Pour chaque exemple existe un certain α. On doit maximiser $\mathcal{L}(w,b,\alpha)$ pour tous les exemples, et il existe un (w,b) pour chaque hyperplan.

5.1.5 Le problème dual de Wolfe

$\mathcal{L}(w,b,\alpha) = \frac{1}{2} w.w - \sum_{i=1}^{m} \alpha_i [y_i(w.x+b) - 1]$

Pour le problème dual, on a :

$$\nabla_w \mathcal{L}(w,b,\alpha) = w - \sum_{i=1}^{m} \alpha_i y_i x_i = 0$$
$$\nabla_b \mathcal{L}(w,b,\alpha) = -\sum_{i=1}^{m} \alpha_i y_i = 0$$

Ces deux équations fournissent :

$w = \sum_{i=1}^{m} \alpha_i y_i x_i$ et $\sum_{i=1}^{m} \alpha_i y_i = 0$.

On les injecte dans la fonction Lagrangienne :

$$W(\alpha, b) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i \cdot x_j$$

Le problème dual est :

$$\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i \cdot x_j$$
$$telque \;\; \alpha_i \geq 0, i = 1...m, \sum_{i=1}^{m} a_i y_i = 0$$

L'avantage du problème de Wolfe, c'est que la fonction objective dépend maintenant seulement des multiplieurs de Lagrange,qui sont facile à résoudre analytiquement.

5.1.6 Calcul de w et b

A l'issu de la résolution du problème de Wolfe, on obtient un vecteur α contenant la valeur des multiplieurs pour chaque exemple. Ensuite, on procède au calcul de w et b pour déterminer l'hyperplan.

D'après l'équation en dessus :

$$w - \sum_{i=1}^{m} \alpha_i y_i x_i = 0$$

On obtient :

$$w - \sum_{i=1}^{m} \alpha_i y_i x_i = 0$$

$y_i(w \cdot x^\star + b) - 1 = 0$

On multiplie les deux membres par y_i et on sait que $y_i^2 = 1$.

Il vient :

$$b = y_i - w \cdot x^\star$$

Par conséquent :

$$b = \frac{1}{S} \sum_{i=1}^{S} (y_i - w \cdot x)$$

S est le nombres des vecteurs supports.

Nous pouvons résoudre le problème dual de Wolfe en utilisant un certain package ou une bibliothèque de manière analytique. Par exemple, nous pouvons utiliser un package Python appelé CVXOPT, qui est destiné à l'optimisation convexe. Ce package fournit un solveur QP pour les problèmes de programmation quadratique. Nous pouvons réorganiser notre problème dual, énoncer les paramètres requis, les fournir au solveur QP et obtenir la solution attendue. La solution contiendra les valeurs des multiplicateurs de Lagrange pour chaque exemple. Nous pouvons alors calculer w et b et obtenir le plan hyperoptimal.

5.1.7 Résolution du problème d'optimisation - Soft Margin

Le problème avec SVM à marge rigide (Hard Margin SVM) est qu'il ne fonctionne que pour des données linéairement séparables. Cependant, ce ne serait pas le cas dans le monde réel. Il est très probable que dans des cas pratiques, les données contiendront du bruit et pourraient ne pas être linéairement séparables. Nous verrons comment SVM à marge souple (Soft Margin SVM) traite ce problème.

Essentiellement, la méthode utilisée par SVM à marge souple est très simple, elle ajoute des variables d'écart aux contraintes du problème d'optimisation. Les contraintes deviennent alors :

$$y_i(w \cdot x_i + b) \geq 1 - \zeta_i, i = 1...m$$

En ajoutant les variables d'écart, lors de la minimisation de la fonction objectif, il est possible de satisfaire la contrainte même si l'exemple ne respecte pas la contrainte initiale. Le problème est que nous pouvons toujours choisir une valeur de ζ suffisamment grande pour que tous les exemples respectent les contraintes.

Une technique pour traiter cela est d'utiliser une régularisation. Par exemple, nous pourrions utiliser une régularisation L1 pour pénaliser les grandes valeurs de ζ. Le problème d'optimisation régularisé devient alors :

$$\min_{w,b,\zeta} \frac{1}{2} \mid\mid w \mid\mid^2 + \sum_{i=1}^{n} \zeta_i$$
$$tel\ que\ \ y_i(w.x + b) \geq 1 - \zeta_i, i = 1...m$$

Aussi, nous voulons nous assurer que nous ne minimisons pas la fonction objectif en choisissant des valeurs négatives de ζ. Nous ajoutons les contraintes $\zeta \geq 0$. Nous ajoutons également un paramètre de régularisation C pour déterminer l'importance de ζ, c'est-à-dire à quel point nous voulons éviter de mal classer chaque exemple d'entraînement. Le problème d'optimisation régularisé devient :

$$\min_{w,b,\zeta} \frac{1}{2} \mid\mid w \mid\mid^2 + C \sum_{i=1}^{n} \zeta_i$$
$$tel\ que\ \ 0 \leq \alpha_i \leq C, i = 1...m, \sum_{i=1}^{m} a_i y_i = 0$$

5.1.8 Régularisation du paramètre C

Alors, que fait le paramètre de régularisation C ? Comme nous l'avons dit, il détermine l'importance de ζ . Un C plus petit met l'accent sur l'importance deζ , tandis qu'un C plus grand diminue l'importance deζ .

Une autre façon de penser à C est qu'il vous donne un contrôle sur la façon dont SVM gérera les erreurs. Si nous définissons C à une valeur positive infinie, nous obtiendrons le même résultat que le SVM de marge dure. En revanche, si nous définissons C à 0, il n'y aura plus de contrainte et nous obtiendrons un hyperplan qui ne classera rien du tout. Les règles empiriques sont les suivantes : de petites valeurs de C donneront une marge plus large, au prix de quelques mauvaises classifications ; de grandes valeurs de C vous donneront le classificateur de marge dure et toléreront une violation de contrainte nulle. Nous devons trouver une valeur de C qui ne sera pas impactée par les données bruyantes.

5.1.9 Astuce du noyau

Maintenant, le SVM de marge douce peut gérer les données non linéairement séparables causées par des données bruyantes. Et si la non-séparabilité linéaire n'est pas causée par le bruit ? Et si les données sont caractéristiquement non-linéairement séparables ? Pouvons-nous toujours séparer les données en utilisant SVM ? La réponse est bien sûr oui. Et nous allons parler d'une technique appelée astuce du noyau pour y remédier.

Imaginez que vous avez un ensemble de données non-linéairement séparable à deux dimensions que vous souhaitez classifier à l'aide de SVM. Il semble impossible de le faire car les données ne sont pas linéairement séparables. Cependant, si nous transformons les données bidimensionnelles en une dimension supérieure, disons trois dimensions ou même dix dimensions, nous serions en mesure de trouver un hyperplan pour séparer les données.

Le problème est que si nous avons un grand ensemble de données contenant, disons, des millions d'exemples, la transformation prendra beaucoup de temps pour s'exécuter, sans parler des calculs dans le problème d'optimisation ultérieur. Revisitons le problème dual de Wolfe :

$$\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i \cdot x_j$$

$$telque \alpha_i \geq 0, i = 1...m, \sum_{i=1}^{m} a_i y_i = 0$$

Pour résoudre ce problème, nous nous intéressons en réalité uniquement au résultat du produit point xi.xj. S'il existe une fonction qui pourrait calculer le produit point et que le résultat est le même que lorsque nous transformons les données en une dimension supérieure, ce serait fantastique. Cette fonction s'appelle une fonction de noyau.

Ainsi, l'astuce du noyau consiste à définir une fonction de noyau $K(xi, xj) = xi.xj$. Nous réécrivons alors le problème dual de Wolfe :

$$\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j K(x_i \cdot x_j)$$

$$subjecttoo \alpha_i \geq 0, i = 1...m, \sum_{i=1}^{m} a_i y_i = 0$$

C'est un petit changement, mais c'est en réalité une astuce puissante. Ce qu'elle fait, c'est de calculer le résultat d'un produit point effectué dans un autre espace. Nous avons maintenant la capacité de changer la fonction de noyau pour classifier les données non-linéairement séparables.

Il existe plusieurs types de noyaux que nous pourrions utiliser pour classer les données. Certains des plus populaires sont le noyau linéaire, le noyau polynomial et le noyau RBF.

Le noyau linéaire est défini comme suit :

$K(xi, xj) = xi.xj$.

This is the same as the one we used in the above discussion. In practice, you should know that a linear kernel works well for text classification.

The polynomial kernel is defined as :

C'est le même que celui que nous avons utilisé dans la discussion précédente. En pratique, il convient de savoir qu'un noyau linéaire fonctionne bien pour la classification de texte.

Le noyau polynomial est défini comme suit : $K(x_i, x_j) = (x_i \cdot x_j + c)^d$

Ce noyau contient deux paramètres : une constante c et un degré de liberté d. Une valeur d égale à 1 correspond simplement au noyau linéaire. Une valeur de d plus grande rendra la frontière de décision plus complexe et pourrait entraîner un surapprentissage.

Le noyau RBF est défini comme suit : $K(x_i, x_j) = exp(-\gamma||x_i - x_j||^2)$

Le noyau RBF (Fonction de Base Radiale), également appelé noyau gaussien, donne lieu à une frontière de décision plus complexe. Le noyau RBF contient un paramètre γ. Une faible valeur de γ fera que le modèle se comportera comme un SVM linéaire. Une grande valeur de γaura un impact important sur les vecteurs de support.

En pratique, il est recommandé d'essayer d'abord le noyau RBF car il fonctionne généralement bien.

5.2 Random Forest

5.2.1 Definition

Les forêts aléatoires (ou random forests en anglais) sont une méthode d'apprentissage automatique supervisé pour la classification, la régression et autres tâches de prédiction. Cette méthode combine les prédictions de plusieurs arbres de décision individuels pour produire une prédiction globale plus précise et robuste. L'idée principale derrière les forêts aléatoires est de construire de nombreux arbres de décision aléatoires et d'agréger leurs prédictions. Chaque arbre de décision est entraîné sur un échantillon aléatoire de données et utilise un sous-ensemble aléatoire de variables (ou de caractéristiques) pour effectuer la division des nœuds. Cette variabilité réduit le risque de surajustement (overfitting) et rend la méthode plus robuste. Une forêt aléatoire est un classificateur constitué d'une collection de classificateurs arborescents h(x,θk), k=1,... ,où les k sont indépendants à l'identique vecteurs aléatoires distribués et chaque arbre émet un vote unitaire pour la classe la plus populaire à entrée x.

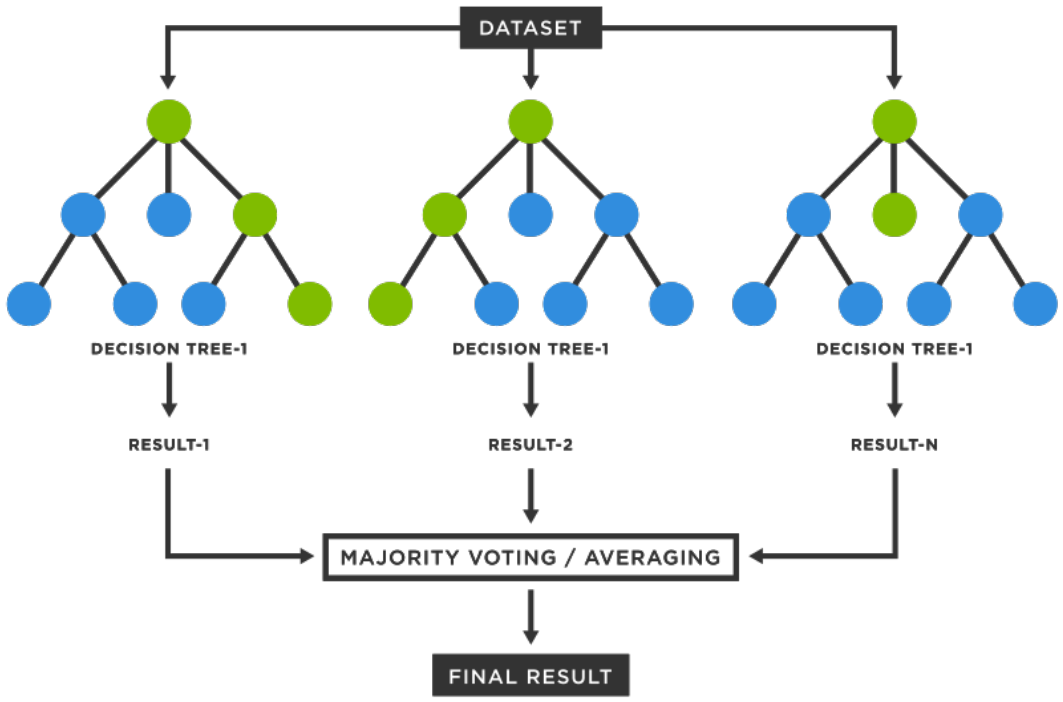


Figure1 : decision trees[6]

5.2.2 Characterizing the Accuracy of Random Forests

Random Forests Converge Étant donné un ensemble de classificateurs h1(x),h2(x), ... ,hk (x), et avec l'ensemble d'apprentissage tirée au hasard de la distribution du vecteur aléatoire Y,X, définir la fonction de marge comme :

$$\boxed{mg(X,Y) = \frac{1}{a}\sum_{k=1}^{a} I(h_k(X) = Y) - \max_{j \neq Y} \frac{1}{a}\sum_{k=1}^{a} I(h_k(X) = j)}$$

où I(•) est la fonction indicatrice. La marge mesure dans quelle mesure le nombre moyen de votes à X,Y pour la bonne classe dépasse le vote moyen pour toute autre classe. Plus la marge est grande, plus la confiance dans le classification. L'erreur de généralisation est donnée par :

$$PE* = P\max_{X,Y}(mg(X,Y) < 0)$$

où les indices X,Y indiquent que la probabilité est sur l'espace X,Y. Dans les forêts aléatoires, $hk(X) = h(X,k)$. Pour un grand nombre d'arbres, il résulte de la loi forte des grands nombres et l'arborescence qui :

5.2.3 Théorème

Lorsque le nombre d'arbres augmente, pour presque sûrement toutes les séquences 1 ,... PE* converge vers :

$$PX,Y(P_\Theta(h(X,\Theta) = Y) - \max_{j \neq Y} P_\Theta(h(X,\Theta) = j) < 0)$$

5.2.4 Strength and Correlation

Pour les forêts aléatoires,[3] une limite supérieure peut être dérivée pour l'erreur de généralisation en termes de deux paramètres qui sont des mesures de la précision de l'individu classificateurs sont et de la dépendance entre eux. L'interaction entre ces deux donne la base pour comprendre le fonctionnement du hasard les forêts. Nous nous appuyons sur l'analyse d'Amit et Geman [1997].

5.2.5 margin function

La fonction de marge d'une forêt aléatoire est :

$$\text{mr(X,Y)}=PX,Y(P_\Theta(h(X,\Theta) = Y) - \max_{j \neq Y} P_\Theta(h(X,\Theta) = j) = 0)$$

La force de l'ensemble des classificateurs $\{h(x,\Theta)\}$ is $s = E_{X,Y}[mr(X,Y)]$. En supposant $s \geq 0$, l'inégalité de Chebychev donne $P(E^*) \leq \frac{\text{var}(mr)}{s^2}$.

Une expression plus révélatrice de la variance de mr est dérivée dans ce qui suit :

Let $\hat{j}(X,Y) = \arg\max_{j \neq Y} P_\Theta(h(X,\Theta) = j)$, so

$$mr(X,Y) = P_\Theta(h(X,\Theta) = Y) - P_\Theta(h(X,\Theta) = \hat{j}(X,Y)) = E_\Theta[I(h(X,\Theta) = Y) - I(h(X,\Theta) = \hat{j}(X,Y))]$$

5.2.6 raw margin function

La fonction de marge brute est.

$$rmg(\Theta,X,Y) = I(h(X,\Theta) = Y) - I(h(X,\Theta) = \hat{j}(X,Y))$$

Ainsi, $mr(X,Y)$ est l'espérance de $rmg(\Theta,X,Y)$ par rapport à Θ. Pour toute fonction f, l'identité

$$[E_\Theta f(\Theta)]^2 = E_{\Theta,\Theta'}[f(\Theta)f(\Theta')]$$

est vrai, où Θ et Θ' sont indépendants avec la même distribution, ce qui implique que

$$mr(X,Y)^2 = E_{\Theta,\Theta'}[rmg(ta,X,Y)rmg(ta',X,Y)]$$

L'utilisation de cela donne

$$\text{var}(mr) = E_{\Theta,\Theta'}[\text{cov}_{X,Y}(rmg(\Theta,X,Y),rmg(\Theta',X,Y))] = E_{\Theta,\Theta'}[\rho(\Theta,\Theta')\text{sd}(\Theta)\text{sd}(\Theta')]$$

Soit $\rho(\Theta,\Theta')$ la corrélation entre $rmg(\Theta,X,Y)$ et $rmg(\Theta',X,Y)$ contenant Θ et Θ' fixe, et soit $sd(\Theta)$ l'écart type de $rmg(\Theta,X,Y)$ en maintenant Θ fixe. Alors,

$$var(mr) = \rho(E_\Theta sd(\Theta))^2 \qquad \leq \rho E_\Theta var(\Theta) \qquad (1)$$

où ρ est la valeur moyenne de la corrélation, donnée par $\boxed{\rho = \frac{E_{\Theta,\Theta'}(\rho(\Theta,\Theta')sd(\Theta)sd(\Theta'))}{E_{\Theta,\Theta'}(sd(\Theta)sd(\Theta'))}(2)}$

5.2.7 Rapport Ratio

Le rapport c/s^2 pour une forêt aléatoire est défini comme
$c \;/\; s^2 = \rho \;/\; s^2$
Il y a des simplifications dans la situation à deux classes. La fonction de marge est $mr(X,Y) = 2P\theta\Theta(h(X,\Theta) = Y) - 1$. L'exigence que la force soit positive (voir (4)) devient similaire à la condition d'apprentissage faible familière $EX,Y\ P\theta\ (h(X,\Theta) = Y) > 0{,}5$. La fonction de marge brute est $2I(h(X,\Theta) = Y) - 1$ et la corrélation ρ est entre $I(h(X,\Theta) = Y)$ et $I(h(X,\Theta') = Y)$. En particulier, si les valeurs de Y sont prises à +1 et -1, alors
$\rho = E_{\Theta,\Theta'}[\rho(h(\cdot,\Theta),h(\cdot,\Theta')]$ de sorte que ρ est la corrélation entre deux membres différents de la forêt moyenné sur la distribution Θ,Θ', nous avons :
$\rho = \frac{1}{|\mathcal{D}|^2}\sum_{(X,Y),(X',Y')\in\mathcal{D}} \text{corr}(h(X,\Theta),h(X',\Theta'))$
Pour plus de deux classes, la mesure de la force définie dans (3) dépend de la forêt ainsi que des arbres individuels puisque c'est la forêt qui détermine $\hat{j}(X,Y)$. Une autre approche est possible. Écrire :
$PE^* = PX,Y(P\Theta(h(X,\Theta) = Y) - \max_{j\neq Y} P_\Theta(h(X,\Theta) = j) < 0) \leq PX,Y(P\Theta(h(X,\Theta) = Y) - P_\Theta(h(X,\Theta) = j) < 0) \leq \frac{1}{\sum_j \text{var}(P_\Theta(h(X,\Theta)=j))}\sum_j \text{var}(P_\Theta(h(X,\Theta) = Y) - P_\Theta(h(X,\Theta) = j))/s_j^2$
Définir $s_j = EX,Y(P\Theta(h(X,\Theta) = Y) - P_\Theta(h(X,\Theta) = j))$ pour être la force de l'ensemble de classificateurs $h(x,\Theta)$ par rapport à la classe j. Remarquez que cette définition de force ne dépend pas de la forêt. En utilisant l'inégalité de Chebyshev, en supposant que tous les $s_j > 0$ conduisent à : $PE^* \leq \frac{1}{\sum_j \text{var}(P_\Theta(h(X,\Theta)=j))}\sum_j \text{var}(P_\Theta(h(X,\Theta) = Y) - P_\Theta(h(X,\Theta) = j))/s_j^2$

5.2.8 Utilisation des fonctions aléatoires

Certaines forêts aléatoires rapportées dans la littérature ont systématiquement des erreur de généralisation que d'autres. Par exemple, la sélection aléatoire fractionnée (Dieterrich [1998]) fait mieux que l'ensachage. L'introduction par Breiman du bruit aléatoire dans les sorties (Breiman [1998c]) font aussi mieux. Mais aucune de ces trois forêts faire

aussi bien qu'Adaboost (Freund et Schapire [1996]) ou d'autres algorithmes qui travail par repondération adaptative (arcing) de l'ensemble d'apprentissage (voir Breiman [1998b], Dieterrich [1998], Bauer et Kohavi [1999]). Pour améliorer la précision, le caractère aléatoire injecté doit minimiser la corrélation tout en conservant sa force. Les forêts étudiées ici consistent à utiliser au hasard entrées sélectionnées ou combinaisons d'entrées à chaque nœud pour développer chaque arbre. Le les forêts résultantes donnent une précision qui se compare favorablement à Adaboost. Cette classe des procédures a des caractéristiques souhaitables :
i) Sa précision est aussi bonne qu'Adaboost et parfois meilleure.
ii) Il est relativement robuste aux valeurs aberrantes et au bruit.
iii) C'est plus rapide que d'ensacher ou de booster.
iv) Il donne des estimations internes utiles de l'erreur, de la force, de la corrélation et d'importance variable.
v) C'est simple et facilement parallélisé.

En somme ,Les algorithmes de Random Forest et de Support Vector Machine sont deux techniques d'apprentissage automatique largement utilisées dans divers domaines tels que la reconnaissance de formes, la classification, la prédiction et l'analyse de données. Dans leur étude théorique, nous pouvons conclure que ces deux algorithmes ont des avantages et des inconvénients spécifiques.

6 La métrique en ML

Le recall permet de savoir le pourcentage de positifs bien prédit par notre modèle. En d'autres termes c'est le nombre de positifs bien prédit (Vrai Positif) divisé par l'ensemble des positifs (Vrai Positif + Faux Négatif).

$$\boxed{recall = \frac{\text{Vrai positif}}{\text{Vrai positif + Faux négatif}}}$$

La precision est assez similaire au recall, il faut donc bien comprendre leur différence.

Elle permet de connaître le nombre de prédictions positifs bien effectuées.

En d'autres termes c'est le nombre de positifs bien prédit (Vrai Positif) divisé par l'ensemble des positifs prédit (Vrai Positif + Faux Positif).

$$\boxed{precision = \frac{\text{Vrai positif}}{\text{Vrai positif + Faux positif}}}$$

Bien qu'ils soient utiles, ni la précision ni le recall ne permettent d'évaluer entièrement un modèle de Machine Learning.

Séparément c'est deux métrique sont inutiles :

si le modèle prédit tout le temps positif, le recall sera élevé au contraire, si le modèle ne prédit jamais positif , la precision sera élevée On aura donc des métriques qui nous indique que notre modèle est efficace alors qu'il sera au contraire plus naïf qu'intelligent.

Heureusement pour nous, une métrique permettant de combiner la precision et le recall existe : le F1 Score.

Le F1 Score permet d'effectuer une bonne évaluation de la performance de notre modèle.

$$\boxed{Score\ f1 = 2x\frac{recall*precision}{recall+precision}}$$

7 Solution implémentée

La détection du cancer est un exemple courant de problème de classification déséquilibré car il y a souvent beaucoup plus de cas de non-cancer que de cancer réel. Un ensemble de données de classification déséquilibré standard est l'ensemble de données de mammographie qui consiste à détecter le cancer du sein à partir de scans radiologiques, en particulier la présence de groupes de microcalcifications qui apparaissent lumineux sur une mammographie. Cet ensemble de données a été construit en scannant les images, en les segmentant en objets candidats et en utilisant des techniques de vision par ordinateur pour décrire chaque objet candidat.

C'est un ensemble de données populaire pour la classification déséquilibrée en raison du déséquilibre de classe important, notamment où 98 % des microcalcifications candidates ne sont pas cancéreuses et seulement 2% ont été étiquetées comme cancer par un radiographe expérimenté.

Donc on a implémenté un code source en utilisant les mméthodes de classifications **Random Forest** et **Support Vector Machine** et **la technique cost-sensitive** pour améliorer la précision.

Le code présenté est une implémentation de différentes techniques de classification pour le jeu de données **"mammography.csv"**. Tout d'abord, les bibliothèques nécessaires sont importées et le jeu de données est chargé à partir du lecteur de disque.

```
import pandas as pd # pour lire les données a partir fichier excel mammography.cvs
import numpy as np #dans notre cas ona importer numpy as np pour génerer un nombre aléatoire (classes=np.unique(y_train))
from sklearn.model_selection import train_test_split #sklearn une bibliothèque pour langage de programmation Python pour la
from sklearn.utils import class_weight #class_weight` est utilisé pour ajuster les poids des classes dans un problème de cl
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score #Créez un rapport texte montrant les pr
from sklearn.svm import SVC #Classification des vecteurs de support C
import seaborn as sns # library for data visualization
from sklearn.preprocessing import LabelEncoder #Library that helps convert categorical data into numerical data.
from sklearn.ensemble import RandomForestClassifier #un méta estimateur qui adapte un certain nombre de classificateurs d'a
import matplotlib.pyplot as plt #pyplot` est une sous-bibliothèque de `matplotlib` qui fournit une interface de haut niveau
```

Ensuite, les fonctionnalités et la variable cible sont séparées et le jeu de données est divisé en ensembles d'apprentissage et de test. Pour tenir compte de la distribution des classes déséquilibrées,

```
X = df.drop('class', axis=1)
y = df['class']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.9, random_state=42)
```

des poids de classe sont calculés pour être utilisés dans la technique d'apprentissage coûteuse.

```
class_weights = class_weight.compute_class_weight('balanced', classes=np.unique(y_train), y=y_train)
class_weights = dict(zip(np.unique(y_train), class_weights))
```

Les graphiques sont utilisés pour visualiser la distribution des classes et la relation entre les fonctionnalités.

```
encoder=LabelEncoder()
df['class']=encoder.fit_transform(df['class'])
x=df['class'].value_counts().to_list()
labels=['Non-Microcalcification','Microcalcification']
plt.pie(x,labels=labels,autopct='%1.2f%%',startangle=0);
select= ['Area', 'AVG_grey', 'Grad_strength', 'Rootmean','Contrast','ordermoment','class']
sns.pairplot(df[select],hue='class')
```

Deux modèles de classification différents sont entraînés avec et sans l'utilisation de l'apprentissage coûteux. Les modèles sont évalués en utilisant différentes métriques telles que la précision, le rappel et le score F1.

```
print('SVM Classifier without cost-sensitive learning:')
print('Accuracy:', accuracy_score(y_test, svc_pred1))
print('Precision:', precision_score(y_test, svc_pred1,pos_label="'1'"))
print('Recall:', recall_score(y_test, svc_pred1,pos_label="'1'"))
print('F1 score:', f1_score(y_test, svc_pred1,pos_label="'1'"))
results_df = results_df.append({'Classifier': 'SVM', 'Accuracy':accuracy_score(y_test, svc_pred1), 'Precision': precision_score(y_test, svc_pred1,pos_
# Comparing the models
print('SVM Classifier with cost-sensitive learning:')
print('Accuracy:', accuracy_score(y_test, svc_pred))
print('Precision:', precision_score(y_test, svc_pred,pos_label="'1'"))
print('Recall:', recall_score(y_test, svc_pred,pos_label="'1'"))
print('F1 score:', f1_score(y_test, svc_pred,pos_label="'1'"))
results_df = results_df.append({'Classifier': 'SVM', 'Accuracy':accuracy_score(y_test, svc_pred), 'Precision': precision_score(y_test, svc_pred,pos_la
# Evaluating the models for Random Forest
print('Random Forest Classifier with cost-sensitive learning:')
print('Accuracy:', accuracy_score(y_test, rfc_pred))
print('Precision:', precision_score(y_test, rfc_pred,pos_label="'1'"))
print('Recall:', recall_score(y_test, rfc_pred,pos_label="'1'"))
print('F1 score:', f1_score(y_test, rfc_pred,pos_label="'1'"))
results_df = results_df.append({'Classifier': 'Random Forest ', 'Accuracy':accuracy_score(y_test, rfc_pred), 'Precision': precision_score(y_test, rfc_
print('Random Forest Classifier without cost-sensitive learning:')
print('Accuracy:', accuracy_score(y_test, rfc_pred1))
```

Une fois que les différentes métriques ont été évaluées pour chaque classificateur et stockées dans le DataFrame "$results_df$",

```
results_df = pd.DataFrame(columns=['Classifier', 'Accuracy', 'Precision', 'Recall', 'F1 Score', 'Cost-Sensitive'])
```

il est courant de visualiser ces résultats sous forme de graphiques pour faciliter la comparaison des performances des différents modèles.

Le but de notre étude est de déterminer quel algorithme de classification est le plus performant. Pour ce faire, nous avons utilisé un graphique en barres pour représenter les scores de précision, de rappel et de F1 pour chaque classificateur. Les barres peuvent être colorées en fonction du classificateur, ce qui permet de visualiser rapidement les performances relatives de chaque modèle. Enfin, la distribution de classe est à nouveau visualisée à l'aide d'un diagramme à barres.

En résumé, cette implémentation utilise plusieurs techniques de classification pour traiter un jeu de données déséquilibré et évalue l'efficacité de ces techniques à l'aide de différentes métriques.

8 Résultats et discussion

A l'issu de l'éxecution de notre code on obtient les résultats suivants :

SVM Classifier without cost-sensitive learning :

— Accuracy : 0.9832074721780604
— Precision : 0.8061224489795918
— Recall : 0.34497816593886466

SVM Classifier with cost-sensitive learning :

— Accuracy : 0.8610890302066773
— Precision : 0.12363168061815841
— Recall : 0.8384279475982532
— F1 score : 0.21548821548821548

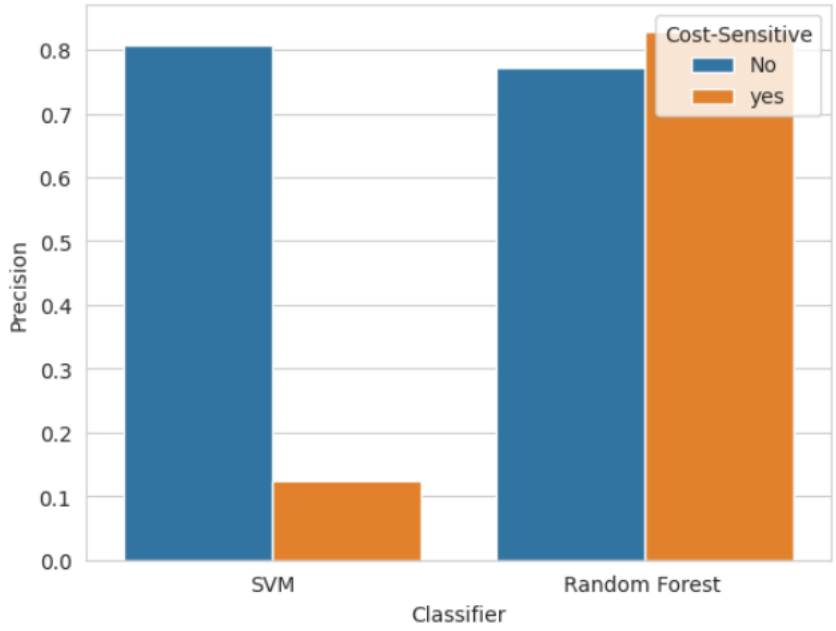
Random Forest Classifier with cost-sensitive learning :

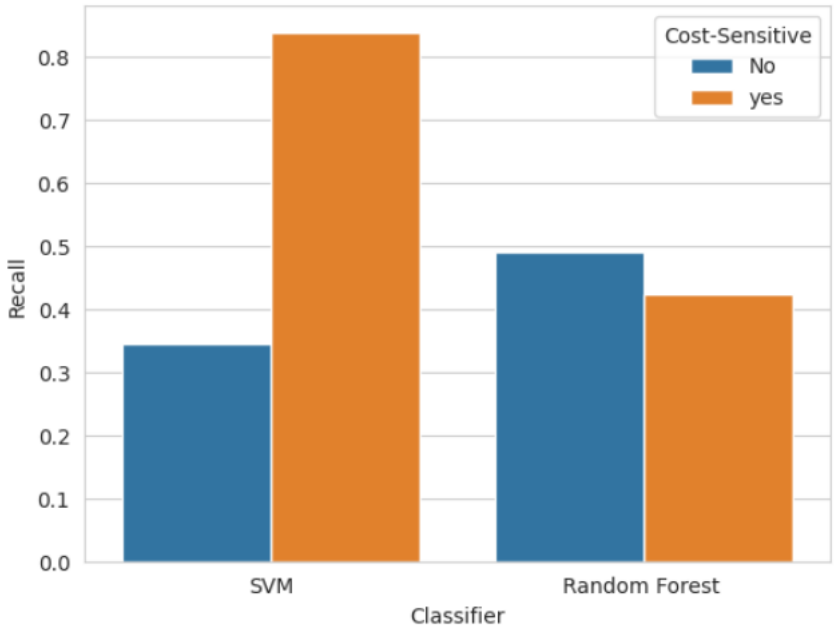
— Accuracy : 0.9848966613672496
— Precision : 0.8290598290598291
— Recall : 0.42358078602620086
— F1 score : 0.5606936416184971

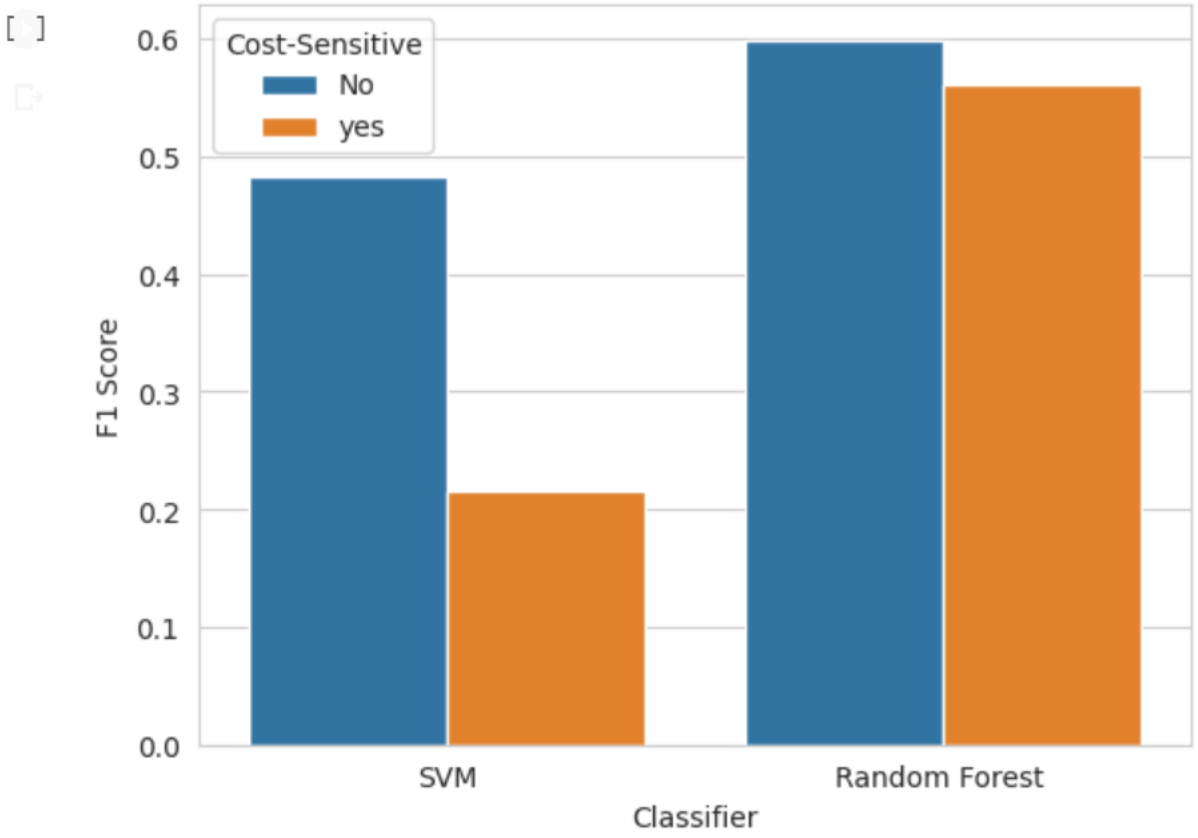
Random Forest Classifier without cost-sensitive learning :

— Accuracy : 0.9850953895071543
— Precision : 0.7724137931034483

— Recall : 0.4890829694323144
— F1 score : 0.5989304812834224







Dans un premier instant, on remarque que l'algorithme SVM without cost sensitive learning a enregistré une précision de 0.80 alors qu'avec le cost-sensitive learning la précision est de 0.12. Cela est dû au fait que l'algorithme est arrivé à prendre en considération les erreurs de calcul qui en découlent. Ceci est logique car on a affaire à des données déséquilbrées. D'ailleurs, les cas positives constituent 2% du nombre des cas enregistrés.

De même, on remarque également que la précision dans dans le Random Forest Classifier with cost-sensitive learning est de 0.82 meilleur que celle de sans cost-sensitive learning

Cela veut dire que ce dernier algorithme est mieux adapté à la prédiction des cas positives.

9 Conclusion et prespectives

En conclusion, l'ensemble de données de mammographie est un problème de classification déséquilibré où la classe minoritaire (microcalcifications) est de la plus haute importance. L'objectif était de faire la distinction entre les microcalcifications et les non-microcalcifications en utilisant six caractéristiques pertinentes. Cependant, le grave déséquilibre des classes a posé un défi aux algorithmes d'apprentissage automatique standard. Par conséquent, nous avons exploré l'utilisation de techniques sensibles aux coûts pour résoudre le problème de déséquilibre. Nous avons évalué deux classificateurs populaires, Support Vector Machines et Random Forest, et avons constaté que l'approche sensible au coût améliorait considérablement les performances des deux modèles, SVM atteignant la précision la plus élevée . Ces résultats démontrent l'efficacité des techniques sensibles aux coûts pour résoudre les problèmes de classification déséquilibrée.

Références

[1] *Cancer du sein*. https://www.who.int/fr/news-room/fact-sheets/detail/breast-cancer. (Accessed on 03/30/2023).

[2] *Intelligence Artificielle, Machine Learning, Deep Learning | Oracle France*. https://www.oracle.com/fr/artificial-intelligence/deep-learning-machine-learning-intelligence-artificielle.html. (Accessed on 03/30/2023).

[3] *randomforest2001.pdf*. https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf. (Accessed on 03/30/2023).

[4] *Techniques for Imbalanced Classification Problems | Kaggle*. https://www.kaggle.com/code/sudhanshu2198/techniques-for-imbalanced-classification-problems. (Accessed on 03/30/2023).

[5] *Understanding the mathematics behind Support Vector Machines*. https://shuzhanfan.github.io/2018/05/understanding-mathematics-behind-support-vector-machines/. (Accessed on 03/30/2023).

[6] *What is a Random Forest ? | TIBCO Software*. https://www.tibco.com/reference-center/what-is-a-random-forest. (Accessed on 03/30/2023).