

Rapport Projet Science des données

1. Présentation de la problématique du projet et des données

A partir d'un ensemble d'images de visages de référence (gallery), on cherche à déterminer si une image requête correspond ou non à l'un des individus représentés dans cette base.

Dans ce projet, on dispose de deux gallery dont les images sont représentées en niveaux de gris, de taille 150x150 pixels et au format JPG. Les images ont été prétraitées de telle sorte qu'aucun processus de traitement d'images (autre que l'application de la méthode Eigenfaces) n'est attendu. Nous allons par la suite utiliser le dataset 1 contenant 7462 images de 373 individus.

Une image est formatée sous la forme X.Y.jpg avec :

- ❖ X : L'identifiant de l'identité de la personne sur l'image.
- ❖ Y : L'identifiant de l'image parmi toutes les images associées à cette identité.

2. Prétraitement des données

Dans le but de tester notre système d'authentification, nous avons besoin de requêtes potentiellement connues ou non par celui-ci. Notre démarche est donc de créer deux groupes de requêtes différents tirés aléatoirement :

- ❖ 100 requêtes de test qui ont des images de la même identité dans la gallery (cas : utilisateurs enregistrés dans le système).
- ❖ 100 requêtes de test dont on supprime toutes images de la même identité dans la gallery (cas : utilisateurs non enregistrés dans le système).

Les images encore disponibles de l'ensemble de données sont utilisées pour constituer la gallery.

Lorsque l'on crée une requête, on sauvegarde l'image tirée ainsi que le nom de l'image correspondant formaté sous la forme X.Y. La séparation des requêtes connues et inconnues, ainsi que leurs noms permettent alors de constituer la vérité-terrain qui sera utilisée pour évaluer les performances du système.

Dans un soucis de reproductibilité des résultats, nous avons décidé d'enregistrer un résultats de prétraitements dans trois fichiers différents :

- ❖ *data.npy* : contenant un tableau numpy de la gallery (de n = 5323 images) et des noms des images associées formatés sous la forme X.Y
- ❖ *unknown.npy* : contenant un tableau numpy de l'ensemble des requêtes tests inconnus et des noms des images associées formatés sous la forme X.Y
- ❖ *known.npy* : contenant un tableau numpy de l'ensemble des requêtes tests connus et des noms des images associées formatés sous la forme X.Y

Une image étant représentée par un tableau de taille (150, 150), il nous faut ensuite transformer les données pour qu'elles puissent être traitées par notre système. Nous allons donc linéariser les $n = 5323$ images de résolution 150x150 pour obtenir un dataset de taille $(n, d) = (5323, 22500)$.

3. Les systèmes mis en oeuvres

Nous avons mis en oeuvres deux systèmes d'authentifications principaux :

- ❖ Un système d'authentification par force brute
- ❖ Un système d'authentification avec Eigenfaces

Les deux systèmes sont basés sur une recherche par force brute des plus proches voisins, on utilise ici un radius search avec un rayon fixé à l'avance. A partir du moment où on retrouve autour une ou plusieurs images qui ont une distance euclidienne inférieure au rayon, alors on considérera que cette personne est autorisée (les images ayant une faible distance entre elles doivent en théorie appartenir à la même personne).

3.a Système d'authentification par force brute

Le système par force brute mesure la distance euclidienne entre chaque image, c'est à dire ici la somme des distances au carré entre chaque pixel des images (la racine carré n'influence pas la comparaison, elle est omise) :

On calcule donc $\sum_{i=1}^d (p_i - q_i)^2$, avec d le nombre de dimensions, ici le nombre total de pixels : 150 (largeur) x 150 (hauteur) = 22500, p_i un pixel du dataset et q_i un pixel de la requête.

C'est un système qui semble efficace car il calcule la distance entre la requête et toutes les images du dataset, il ne conservera que les images inférieures au rayon de la recherche. Le problème principal de cette méthode est la dimension de chaque image, 22500 variables à comparer avec notre résolution pour chaque individu de la gallery.

3.b Système d'authentification avec Eigenfaces

i. Description du système d'authentification avec Eigenfaces

Le deuxième système apporte une solution à ce problème de dimension, l'idée étant de réduire le nombre de variables à l'aide d'une ACP efficace. Les temps de calculs seront bien plus rapides avec cette méthode qu'avec une ACP classique tout en retournant le même résultat. On peut utiliser cette méthode car nous avons plus de dimensions dans notre matrice d'individus. Contrairement à la méthode par force brute, on centre les données après linéarisation pour réaliser l'ACP, ce qui revient à soustraire le visage d'un individu moyen à chaque individu.

A la place de rechercher les vecteurs propres de $Cov(D)$, on calcule les n vecteurs propres de $Cov(D^T)$ tel que : $Cov(D^T) v_i = \lambda_i v_i$ (qui sont moins nombreux puisque $n < d$). On peut ensuite retrouver les n vecteurs principaux de D avec : $w_i = D^T \cdot v_i$ (qui correspondent aux vecteurs principaux significatifs de D , les autres étant nulles) ainsi que les valeurs propres associées $\mu_i = \frac{d-1}{n-1} \lambda_i$ avec λ_i les valeurs propres de $Cov(D^T)$.

Pour sélectionner les k premiers vecteurs principaux nous avons utilisé trois méthodes : le critère de Kaiser, le taux d'inertie cumulé et l'éboulis des parts d'inertie.

- ❖ Avec le critère de Kaiser, on ne retient que les axes associés aux valeurs propres considérées comme les plus "informatives", donc supérieures à leur moyenne ($\frac{1}{d} \sum_{j=1}^n \mu_j$).

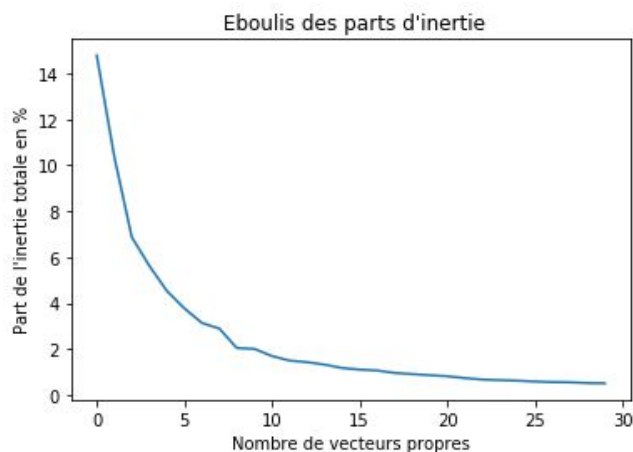
Dans notre cas, ce critère sélectionne les 521 premiers vecteurs principaux, ce qui correspond à un rapport d'inertie de 98.15%.

- ❖ Avec le taux d'inertie cumulé, on choisit ici le nombre de vecteurs principaux de façon à conserver un certain pourcentage de l'inertie totale $I = \sum_{j=1}^n \mu_j$, ce qui revient à calculer $\sum_{i=1}^k \frac{\mu_i}{I}$. En choisissant de fixer cette part

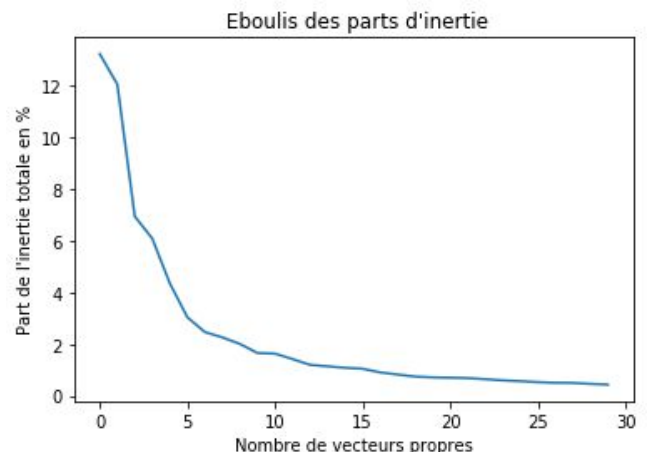
d'inertie à 80% minimum, cela correspond dans notre cas aux 46 premiers vecteurs principaux, avec un rapport d'inertie de 80.18%.

- ❖ L'éboulis des parts d'inertie : Initialement cette méthode utilise l'éboulis des valeurs propres, mais dans notre cas les écarts entre les valeurs étaient tels qu'il a fallu les normer pour pouvoir les comparer. Après calcul des valeurs propres, on trouve ceci :

Pour le dataset 1 :



Pour le dataset 2 :



On voit une fracture entre 8 et 10, pour conserver le maximum d'inertie, on choisit donc de conserver les 10 premiers vecteurs principaux pour les deux sets de données.

ii. Résultat obtenu :

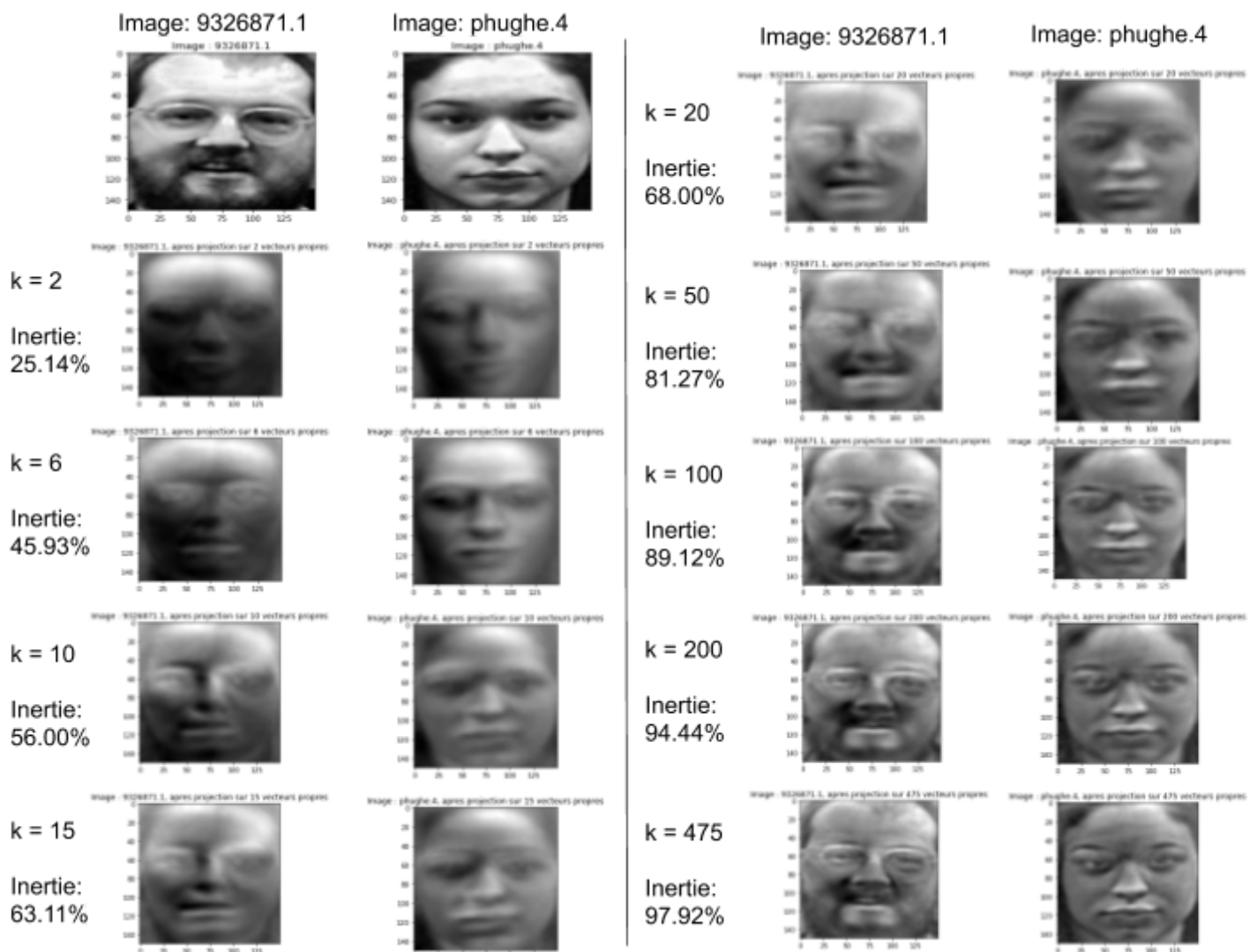
		Critère de Kaiser	Taux d'inertie cumulé	Éboulis des parts d'inertie
Dataset 1 n = 5323	Nombre de vecteurs principaux conservés	k = 521	k = 46	k = 10
	Part d'inertie conservées	98.15%	80.18%	56.00%
Dataset 2 n = 5207	Nombre de vecteurs principaux conservés	k = 797	k = 69	k = 10
	Part d'inertie conservées	96.03%	80.08%	54.04%

iii. Projection des données:

Après avoir sélectionné les k premiers vecteurs principaux, on projette les données sur le sous-espace vectoriel engendré par ces k vecteurs, réduisant ainsi la taille de la matrice des données à n x k.

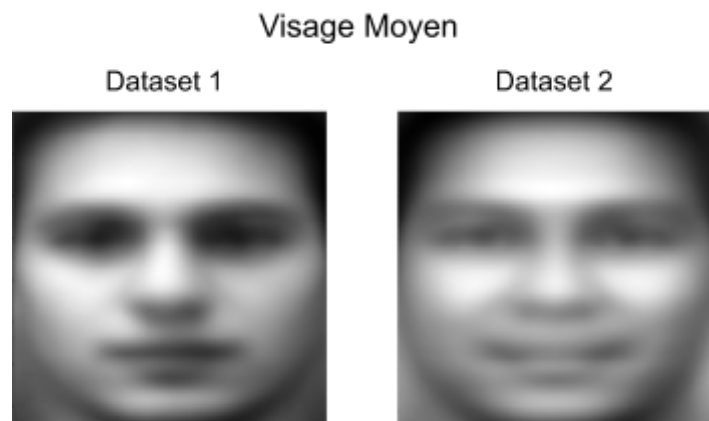
Enfin, on réalise une recherche par force brute des plus proches voisins comme précédemment, mais cette fois-ci à partir des données redimensionnées.

Pour mieux comprendre l'effet d'une ACP sur un objet comme un image, nous avons retracé les photographies de deux individus du dataset 1 pour des valeurs de k différentes :











On remarque bien que plus le nombre de vecteurs principaux k conservés augmente, plus le visage est détaillé et reconnaissable. L'évolution est fulgurante dans les premières valeurs de k et est de moins en moins visible dans les k élevés, ce qui correspond bien aux rapport d'inertie correspondant.

On peut également remarquer qu'un visage se dessine dès $k = 2$ alors que ça ne devrait pas être le cas puisque projeter une image sur deux vecteurs principaux reviendrait à ne conserver qu'une information entre 0 et 255 par individu. Cette silhouette s'explique par l'ajout de la moyenne, un "visage moyen" lors du processus pour retrouver les images. Cette étape est nécessaire pour que les images retrouvent leur cohérence car les images ont été centrées lors de l'ACP.



Pour avoir une idée de la qualité des modèles que l'on étudie, nous avons également représenté les photographies d'un membre du dataset 1 et d'un membre du dataset 2 en fonction des méthodes utilisées :

	Image initiale	Critère de Kaiser	Taux d'inertie cumulé	Éboulis des parts d'inertie
Dataset 1 n = 5323				
Dataset 2 n = 5207				

4. Evaluation du système

4.a Mesures sur les données

Lorsque le système fournit un résultat, il peut se trouver dans un des quatre cas suivants :

- ❖ **Vrai positif (TP)** : le système autorise l'accès à un utilisateur auquel l'accès doit être autorisé pour des motifs légitimes (le système a effectivement retrouvé une image de l'utilisateur dans sa base).
- ❖ **Faux positif (FP)** : le système autorise l'accès à tort (le système a autorisé l'accès à un utilisateur non-enregistré ou à un utilisateur enregistré par coïncidence, en retrouvant des images d'autres utilisateurs).
- ❖ **Vrai négatif (TN)** : le système refuse l'accès à un utilisateur auquel l'accès doit être refusé (le système n'a pas pu retrouver d'images de l'utilisateur car il n'en existe pas).
- ❖ **Faux négatif (FN)** : le système refuse l'accès à un utilisateur auquel l'accès doit être autorisé (le système n'a pas pu retrouver d'images de l'utilisateur dans sa base).

Puisque nous pouvons utiliser le critère de Kaiser, le critère d'inertie ou le critère d'éboullis, nous avons quatre situations différentes :

- ❖ Un système d'authentification par force brute.
- ❖ Un système d'authentification avec Eigenfaces et critère de Kaiser.
- ❖ Un système d'authentification avec Eigenfaces et critère d'inertie.
- ❖ Un système d'authentification avec Eigenfaces et critère d'éboullis.

Pour évaluer chaque système, notre démarche est la suivante :

On commence par tester chacune des requêtes connues puis inconnues pour obtenir les indices des plus proches voisins trouvés par le système considéré (on rappelle que les requêtes sont centrées dans les situations avec Eigenfaces).

On compare ensuite chaque requête avec la vérité-terrain correspondante, plus précisément en comparant les identifiants d'identité des images aux indices trouvés par le système avec celui de la vérité-terrain pour les requêtes connues, puis on incrémente les valeurs selon chaque cas.

4.b Métriques utilisées

Avec les valeurs de TP, FP, TN, FN, on peut calculer plusieurs métriques :

- ❖ **Exactitude** : $A = \frac{TP + TN}{TP + FP + TN + FN}$ qui mesure le taux de bonnes réponses produites par le système.
- ❖ **Précision** : $P = \frac{TP}{TP + FP}$ qui mesure le taux de personnes correctement autorisées parmi l'ensemble des personnes autorisées par le système.

- ❖ **Sensibilité (ou rappel) :** $Sen = \frac{TP}{TP+FN}$ qui mesure le taux de personnes correctement autorisées parmi l'ensemble des personnes qui devraient être autorisées par le système.
- ❖ **Spécificité :** $Spe = \frac{TN}{TN+FP}$ qui mesure le taux de personnes correctement refusées parmi l'ensemble des personnes refusées par le système.

Un exemple d'utilisation de ces métriques est la courbe de ROC, une mesure de la performance d'un système qui a pour objectif de catégoriser des éléments en deux groupes distincts sur la base d'une ou plusieurs des caractéristiques de chacun de ces éléments. Elle représente le taux de vrais positifs en fonction du taux de faux positifs.

On peut alors remarquer que le point de coordonnées (0,1) correspond à une sensibilité et une spécificité de 100%, le classificateur est alors parfait, c'est un objectif à atteindre. En opposition, la droite diagonale ou ligne de non-discrimination correspond à un classificateur de prédiction aléatoire, c'est à objectif à ne pas atteindre.

En conclusion, la droite diagonale divise l'espace en deux : Les points au-dessus de la diagonale représentent une bonne classification (meilleure que l'aléatoire) alors que les points en dessous représentent une mauvaise classification (pire que l'aléatoire).

Pour réaliser la recherche des plus proches voisins, on définit un rayon R de manière empirique. On cherche à analyser l'influence de ce méta paramètre sur les performances de chaque système.

On décide donc d'analyser les performances de manière itérative en répétant le calcul des métriques pour des R allant de 0 à 10^8 avec un pas à 10^6 , puis on représente graphiquement l'évolution de chaque métrique par rapport à R pour les différents systèmes étudiés (voir annexes).

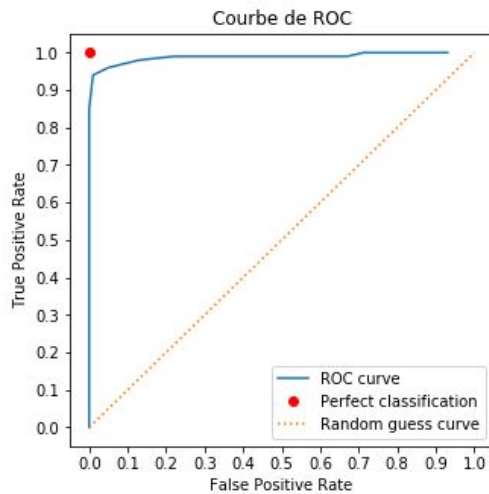
5. Analyse de la performance des systèmes

5.a Système d'authentification par force brute

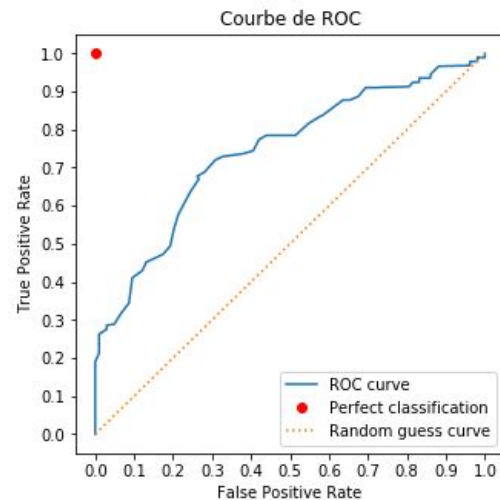
L'authentification par force brute sans ACP est la méthode qui aura la meilleure représentation de la réalité, puisqu'elle mesurera la distance réelle entre les individus, sans réduction des données.

Dans le cas du Dataset 1, cette recherche est presque parfaite, on peut voir sur la courbe de ROC que notre courbe frôle le point de classification parfaite, par contre pour le Dataset 2, la courbe de ROC est bien moins bonne. Cela s'explique par le nombre de photographies par personne. Dans le dataset 1, il y en avait à peu près 20/personnes, alors que dans le dataset 2 il n'y en a plus que 5/personne. Cette différence dans les données est responsable du manque de précision visible sur la deuxième courbe de ROC.

Dataset 1

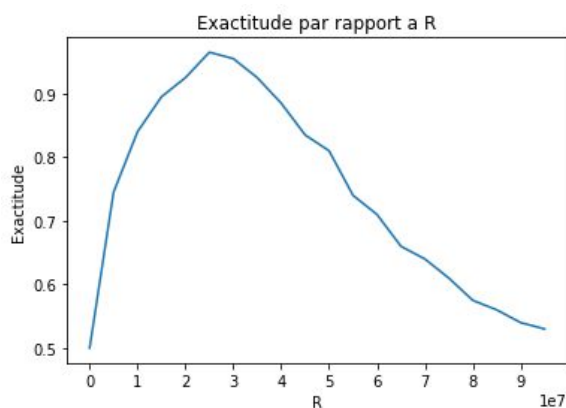


Dataset 2

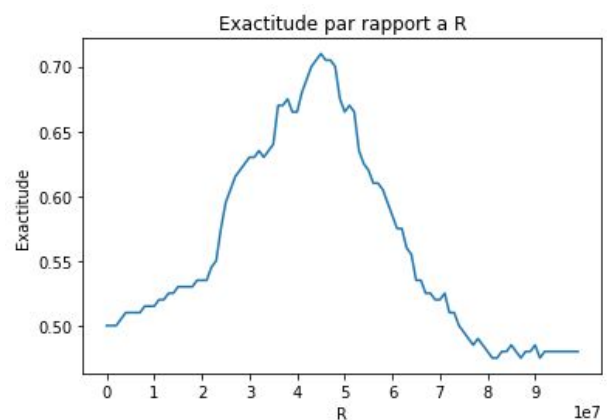


En partant du principe que l'on cherche à maximiser l'exactitude, le rayon de recherche optimal sera $R = 26 \times 10^6$ pour le Dataset 1, ce qui permettrait d'avoir une exactitude de 0.97. Et $R = 45 \times 10^6$, avec une exactitude maximale de 0.71 dans le Dataset 2.

Dataset 1



Dataset 2

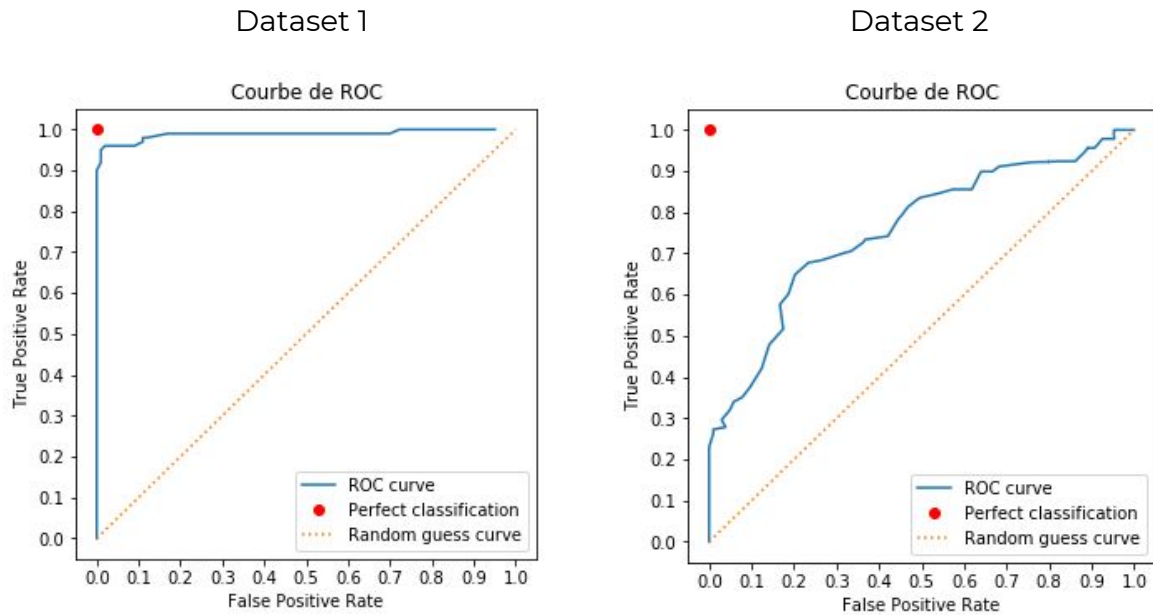


Néanmoins, ces chiffres sont à prendre avec des pincettes puisque ce rayon optimal a été mesuré pour nos 200 requêtes, il pourrait très bien être différent pour un autre ensemble de requêtes.

5.b Système d'authentification avec Eigenfaces et critère de Kaiser

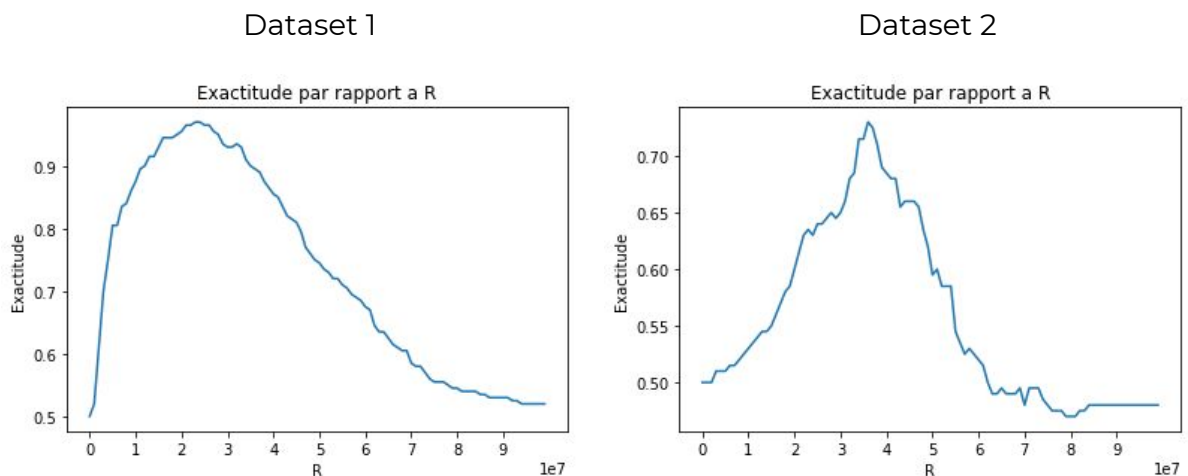
Cette méthode est la méthode qui se rapproche le plus des performances de la méthode sans ACP, puisque comme nous l'avons vu dans la présentation des systèmes, c'est la partie qui conserve le plus de vecteur principaux.

Cela se remarque directement sur les courbes de ROC, il est difficile de faire la différence entre la recherche par force brute sur les données réduites avec le critère de Kaiser et les données initiales.



Pour ce qui est de l'exactitude en fonction de R , l'allure est également très similaire à ce que l'on a trouvé précédemment sur les données brutes.

Ici, on trouve un R optimal $R = 23 \times 10^6$ correspondant à une exactitude de 0.97 pour le Dataset 1 et $R = 36 \times 10^6$ pour une exactitude de 0.73 dans le 2.

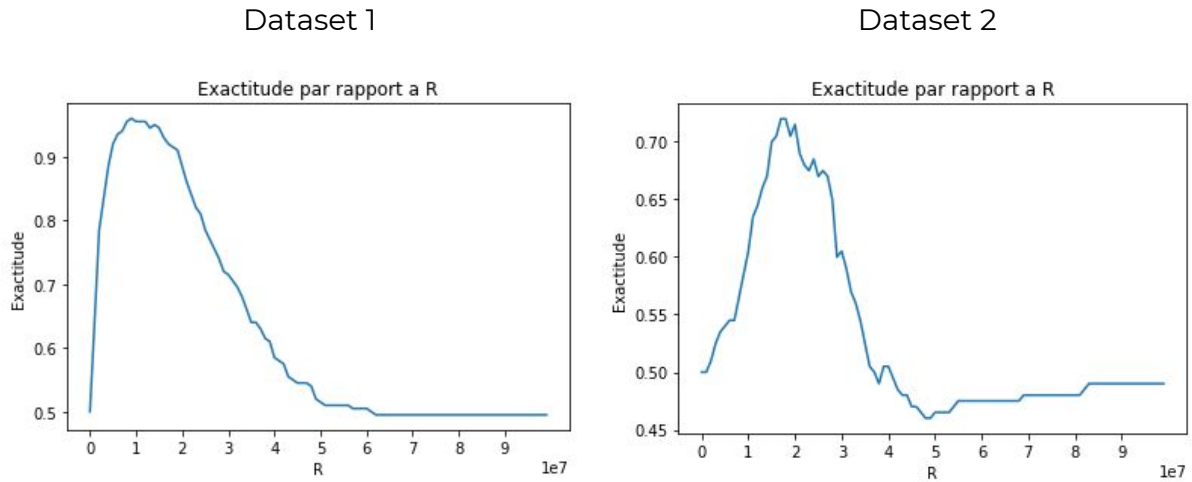


5.c Système d'authentification avec Eigenfaces et critère d'inertie

En prenant le taux d'inertie cumulé comme critère plutôt que le critère de kaiser, on réduit considérablement notre dimension, déjà fortement réduite avec Kaiser. Pour se donner une idée, on conserve approximativement 11 fois moins de vecteurs principaux avec cette méthode plutôt qu'avec Kaiser et plus de 75 fois moins qu'avec les données brutes.

Et pourtant, avec nos 80% d'inertie conservés, la courbe de ROC est quasi identique à celle des données brutes et du critère de Kaiser, elles sont visibles en annexe 3.a et 3.b.

Pour ce qui est de l'exactitude, on remarque que la plage en fonction de R permettant d'obtenir une exactitude maximale se réduit, mais est toujours assez large pour permettre d'obtenir de bon résultat. Cette méthode permet donc de conserver une bonne exactitude dans les deux datasets si on choisit bien R .

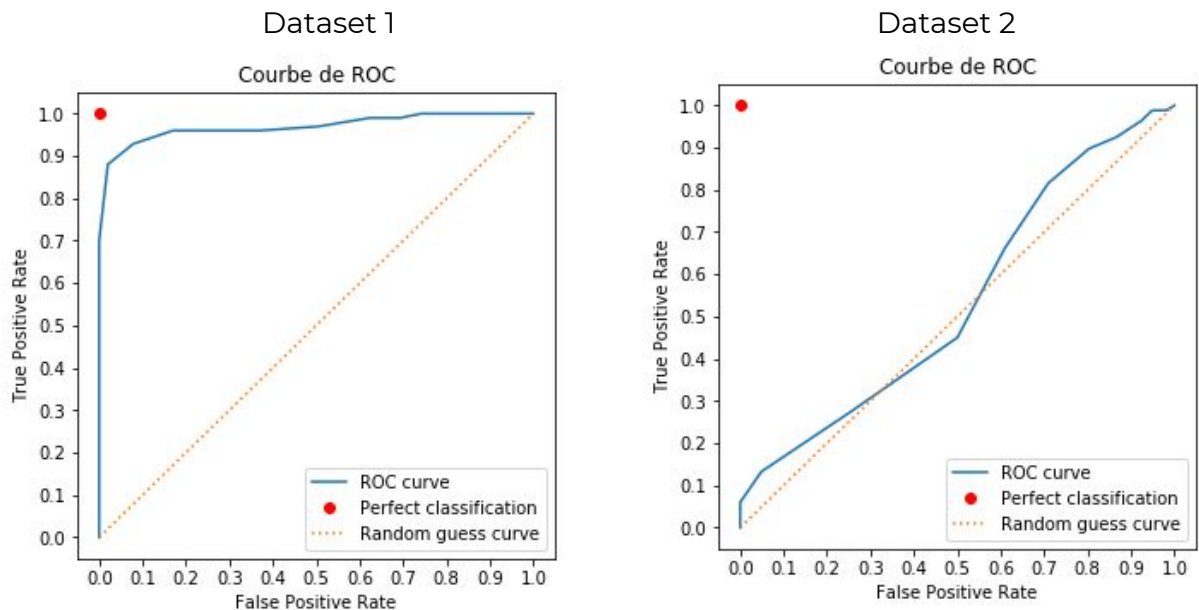


Avec ce système, on trouve un rayon optimal $R = 9 \times 10^6$ correspondant à une exactitude de 0.96 pour le dataset 1. Et pour le dataset 2, $R = 17 \times 10^6$, pour une exactitude maximale de 0.72.

5.d Système d'authentification avec Eigenfaces et critère d'ébouilis

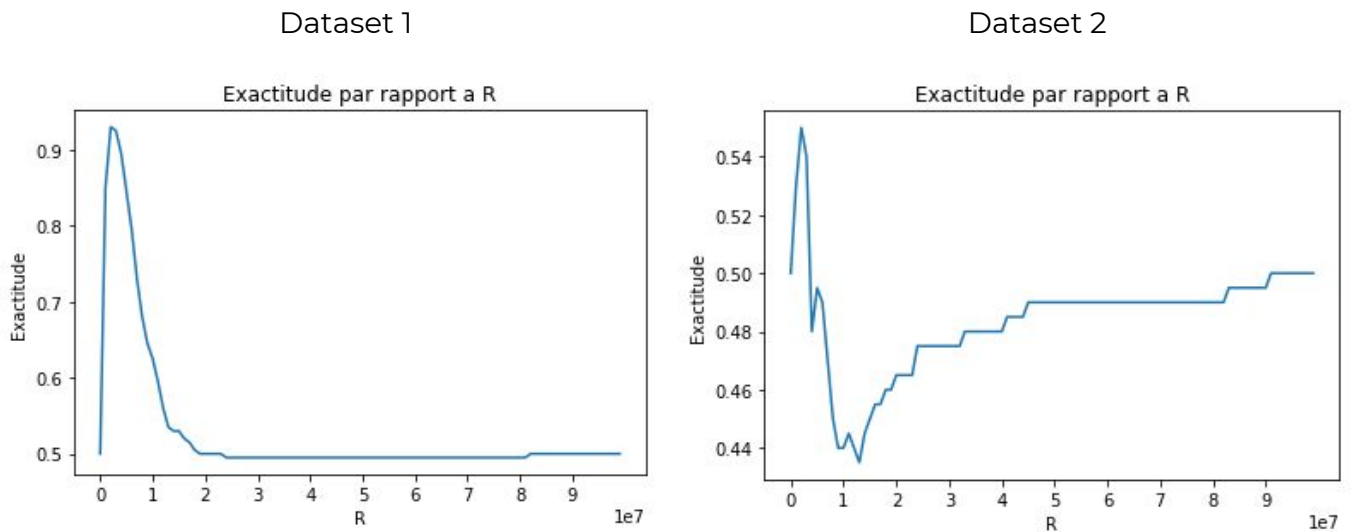
L'authentification avec ACP en utilisant l'ébouilis des parts d'inertie comme critère est intéressant dans le sens où c'est notre cas extrême. On ne conserve ici que 10 dimensions sur les 5500 et quelques dimensions significatives initiales.

Assez étonnamment, même avec seulement 56% d'inertie conservée on trouve d'excellents résultats pour le dataset 1, la courbe de ROC est presque aussi bonne que pour les méthodes précédentes. Le gros point faible de cette méthode réside dans le dataset 2. Dès que l'on manque de redondance dans les données nous n'arrivons plus à trouver les plus proches voisins, peu importe le rayon de recherche. Sur la courbe de ROC du dataset 2, on peut voir que la courbe est proche de la diagonale, ce qui se résume par une classification aléatoire.



Pour ce qui est de l'exactitude, on voit que cette méthode est viable pour le dataset 1, mais pour le 2 elle semble totalement instable et aucune valeur n'est exploitable.

On trouve donc un R optimal $R = 2 \times 10^6$ correspondant à une exactitude de 0.93 pour le dataset 1, et rien pour le 2.



6. Analyse temporelle des systèmes

Après s'être intéressé aux performances de nos systèmes, une seconde manière de les comparer est de mesurer leur temps d'indexation et de recherche.

Pour le temps d'indexation, la recherche par brute force sans ACP est évidemment la plus rapide puisqu'elle n'en a pas, mais on peut également noter une différence significative entre le Dataset 1 et le 2, pourtant c'est le set 1 qui comprend le plus de photographies. On peut donc sérieusement se demander si la redondance des images n'aurait pas un impact sur la vitesse d'indexation.

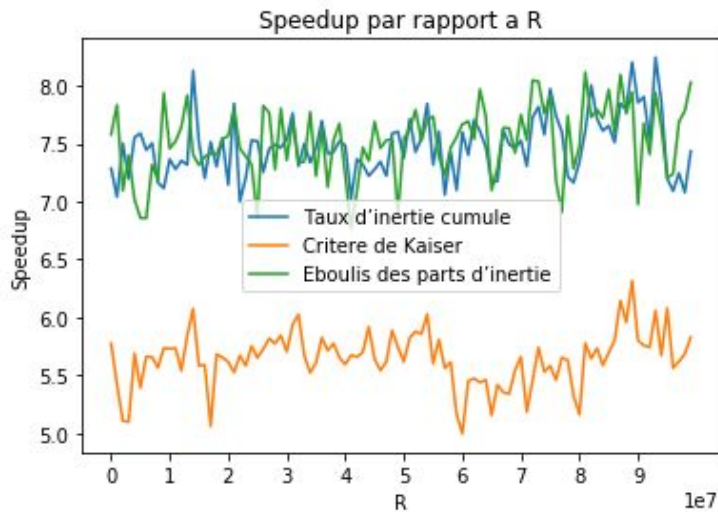
Pour ce qui est de comparer les méthodes par ACP entre eux, les écarts ne sont pas assez significatifs pour justifier une préférence, il pourrait très bien s'agir d'erreurs de mesure.

		Sans ACP	Avec ACP		
			Critère de Kaiser	Taux d'inertie cumulé	Éboulis des parts d'inertie
Temps d'indexation	Dataset 1	-	76.67 sec	78.58 sec	68.53 sec
	Dataset 2	-	102.31 sec	106.44 sec	119.55 sec

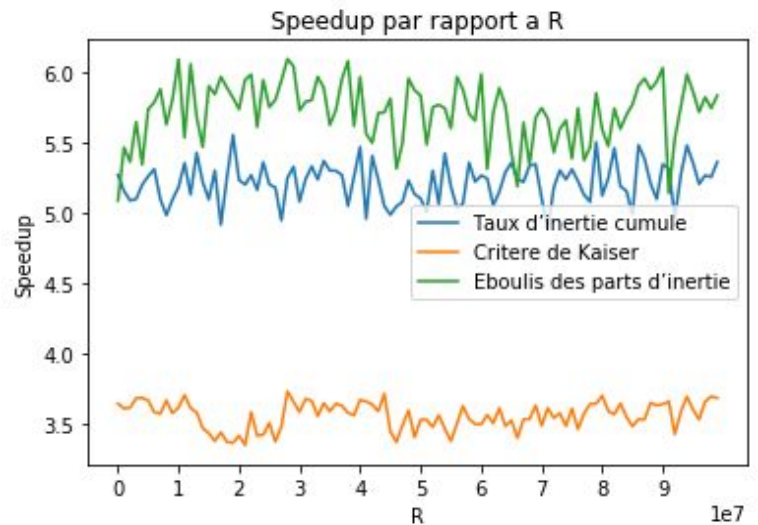
Tout l'intérêt des méthodes de recherches par ACP est de gagner du temps sur le temps de recherche. Pour représenter ce gain de temps, nous avons tracé le speedup, c'est à dire le rapport : $\frac{\text{temps de recherche de la baseline}}{\text{temps de recherche du modèle}}$ pour chacun des modèles.

Les temps d'exécution qui ont servi pour tracer ces courbes sont en annexes.

Dataset 1



Dataset 2



Premièrement, on peut remarquer que contrairement au temps d'indexation, ici les recherches sur le dataset 2 sont plus rapides que sur le dataset 1. Cela semble plus logique car comme dit précédemment, le dataset 1 contient plus d'images que le 2. Une recherche par force brute prend donc plus de temps.

Deuxièmement, on remarque directement que la vitesse des méthodes est indexée sur leurs nombres de vecteurs principaux. Ici aussi cela est logique, une méthode qui doit calculer des distances sur plus de dimensions mettra plus de temps à parcourir la totalité des données.

Les méthodes les plus rapides sont le taux d'inertie cumulé et l'éboulis des parts d'inertie, le problème avec cette dernière étant que dans le dataset 2 la qualité des résultats est extrêmement mauvaise.

7. Conclusion et analyse critique des résultats obtenus

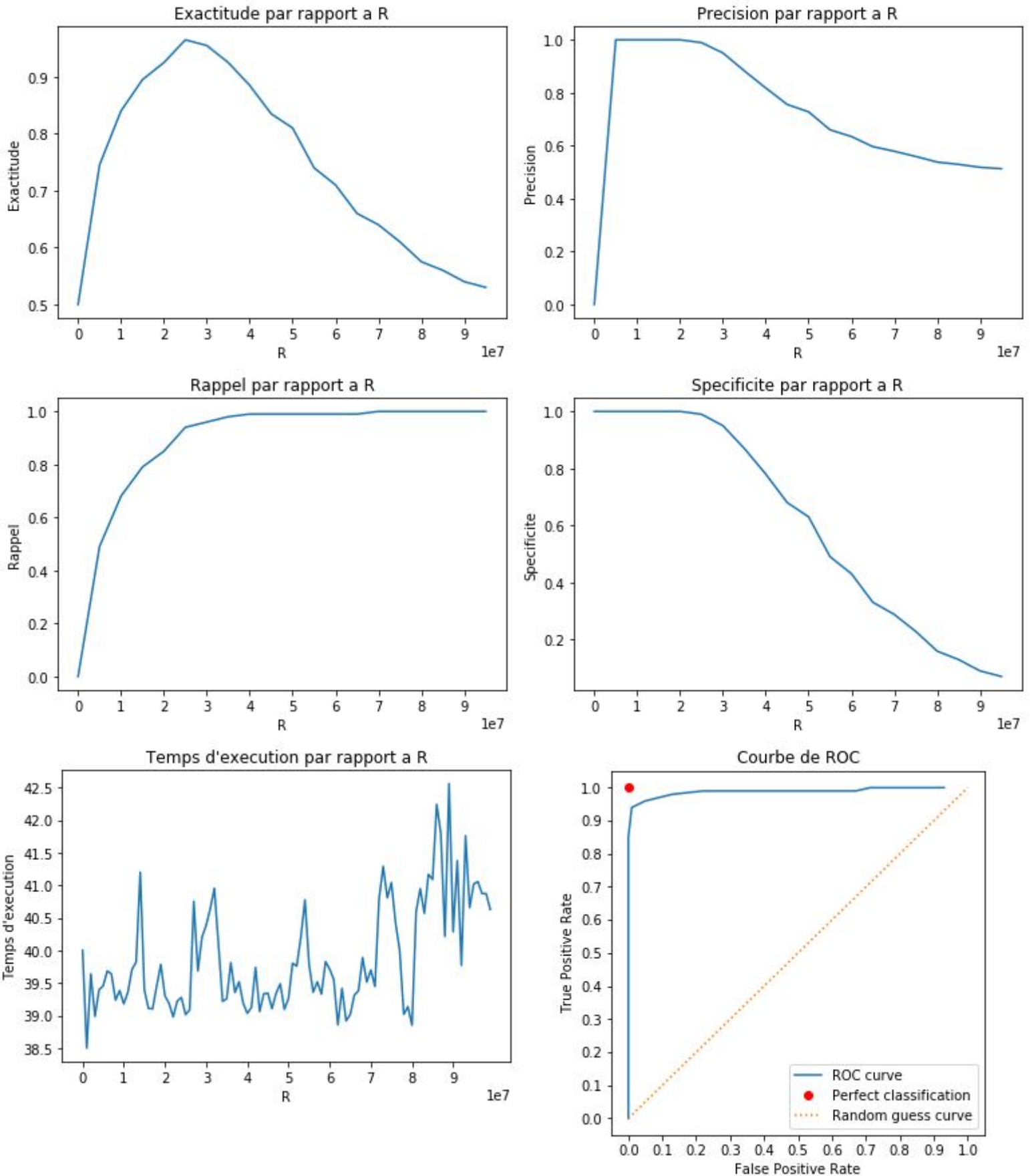
Le défaut de l'authentification par force brute est son temps d'exécution. En effet, on effectue une recherche calculant une distance euclidienne sur tous les individus et sur une dimension de 22500, ce qui prend beaucoup de temps. Pour se donner un ordre d'idée, 200 requêtes prennent environ 40 secondes avec cette méthode, ce qui représente 0.2 secondes par requête. C'est un temps qui peut être gagné avec une réduction des dimensions par ACP.

Ne serait-ce qu'avec le critère de Kaiser, la recherche se fait déjà entre 3.5 et 5.5 fois plus vite, alors que comme nous l'avons vu précédemment, les performances entre la recherche par force brute sans ACP et avec ACP en utilisant le critère de Kaiser sont strictement identiques. La seule raison logique de préférer la méthode force brute sans ACP serait d'éviter un temps d'indexation trop long, mais dans notre cas il n'excède jamais 2 minutes. Sachant que c'est une opération unique, cela ne constitue pas un obstacle.

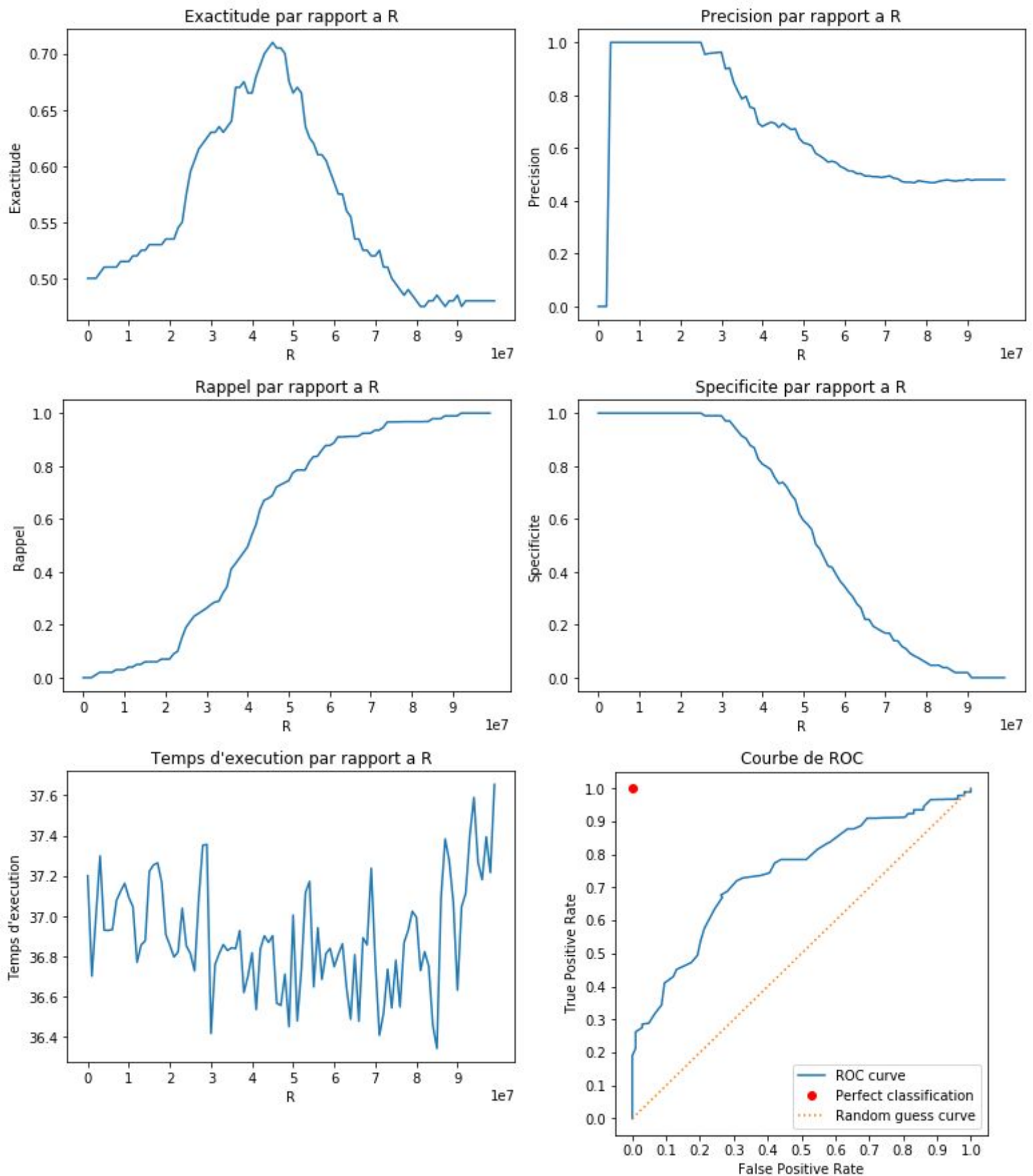
Ensuite, pour délibérer de la meilleure méthode d'ACP, on peut écarter l'éboullis des parts d'inertie, qui fonctionne très bien mais uniquement dans le Dataset 1 et qui n'est absolument pas performant dans le cas où l'on a moins de photographies par individus. Pour le taux d'inertie cumulé avec un ratio d'inertie fixé à 0.8, les performances sont similaires au critère de Kaiser dans les deux Dataset. Or, puisque la vitesse de recherche est plus rapide avec cette méthode, c'est celle que l'on privilégiera dans notre cas, et la meilleure des quatre que nous avons analysées.

Le seul paramètre qui peut varier avec cette méthode est le rayon R de la recherche. Nous avons donné des valeurs de R idéaux dans la partie : "Analyse de la performance des systèmes", mais ces valeurs ne sont valides que si l'on cherche à maximiser l'exactitude. Dans un système strict où l'on veut minimiser au maximum les risques de faux positifs, alors on cherchera plutôt à maximiser la spécificité, ce qui reviendrait à prendre un R inférieur au R optimal que l'on a calculé. A contrario, si on cherche à minimiser le nombre de faux négatifs, alors il faudra maximiser la précision, ce qui revient à prendre un R supérieur au R optimal d'exactitude.

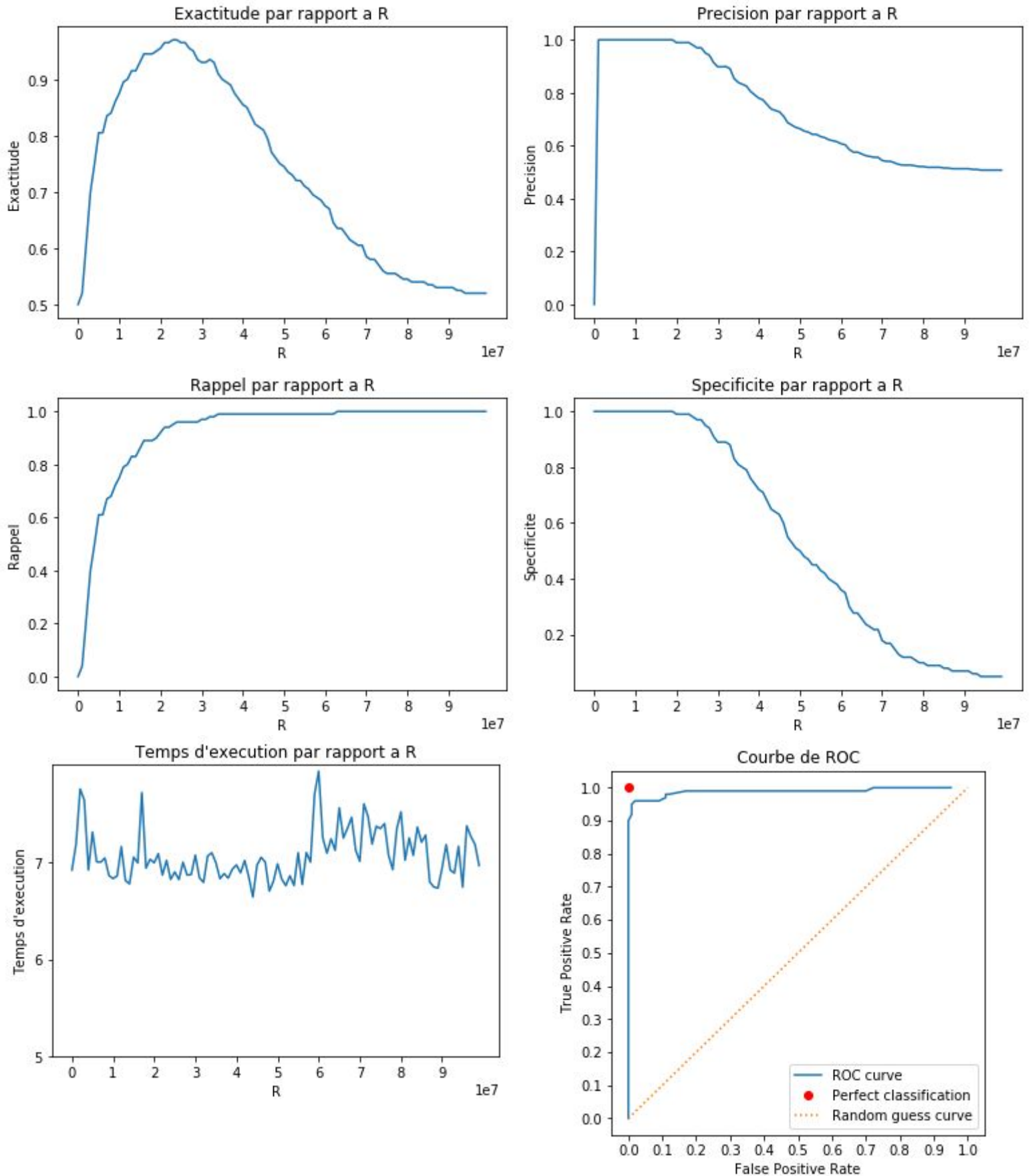
Annexe 1.a : Brute_force sans ACP set 1



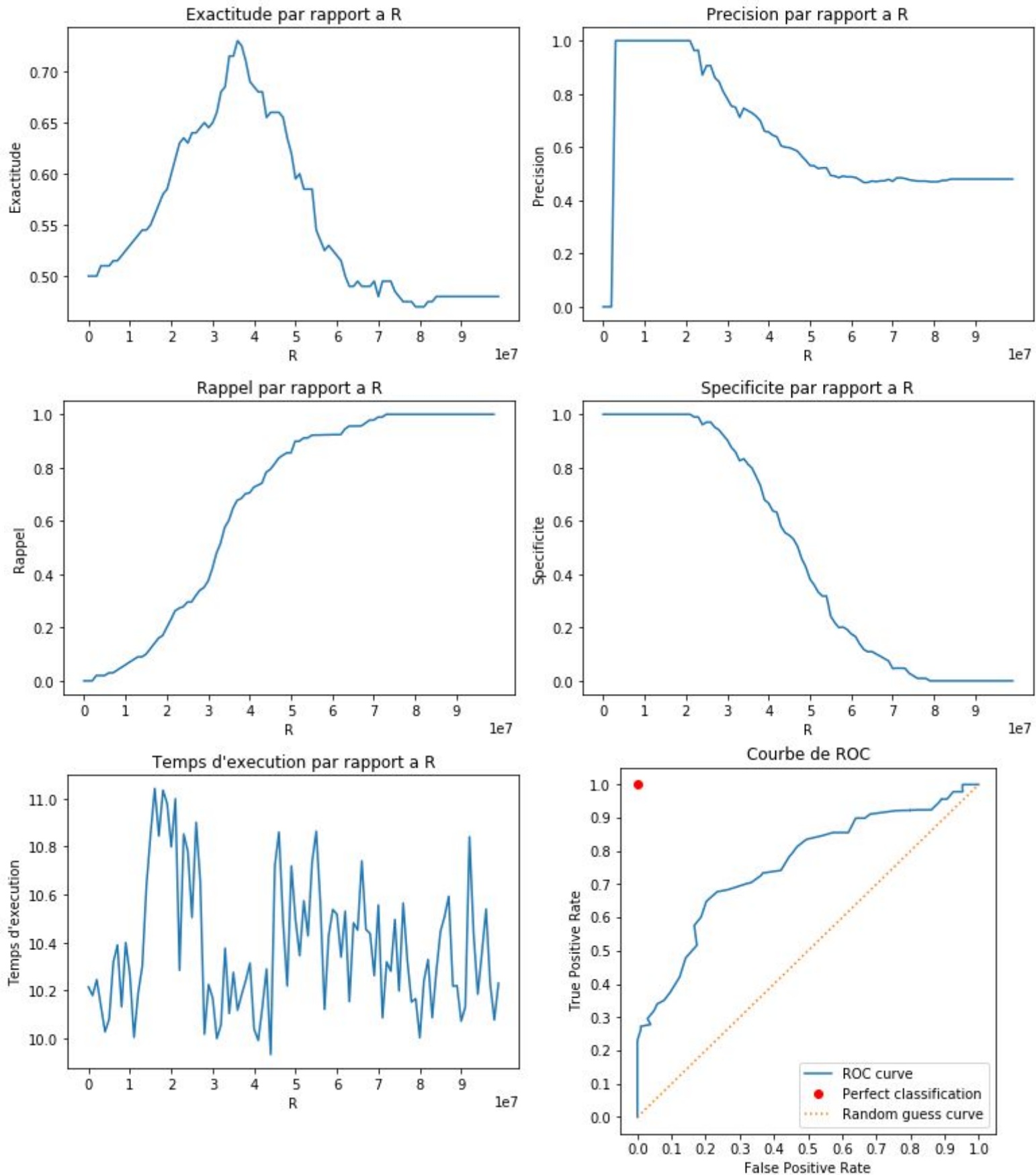
Annexe 1.b : Brute_force sans ACP set 2



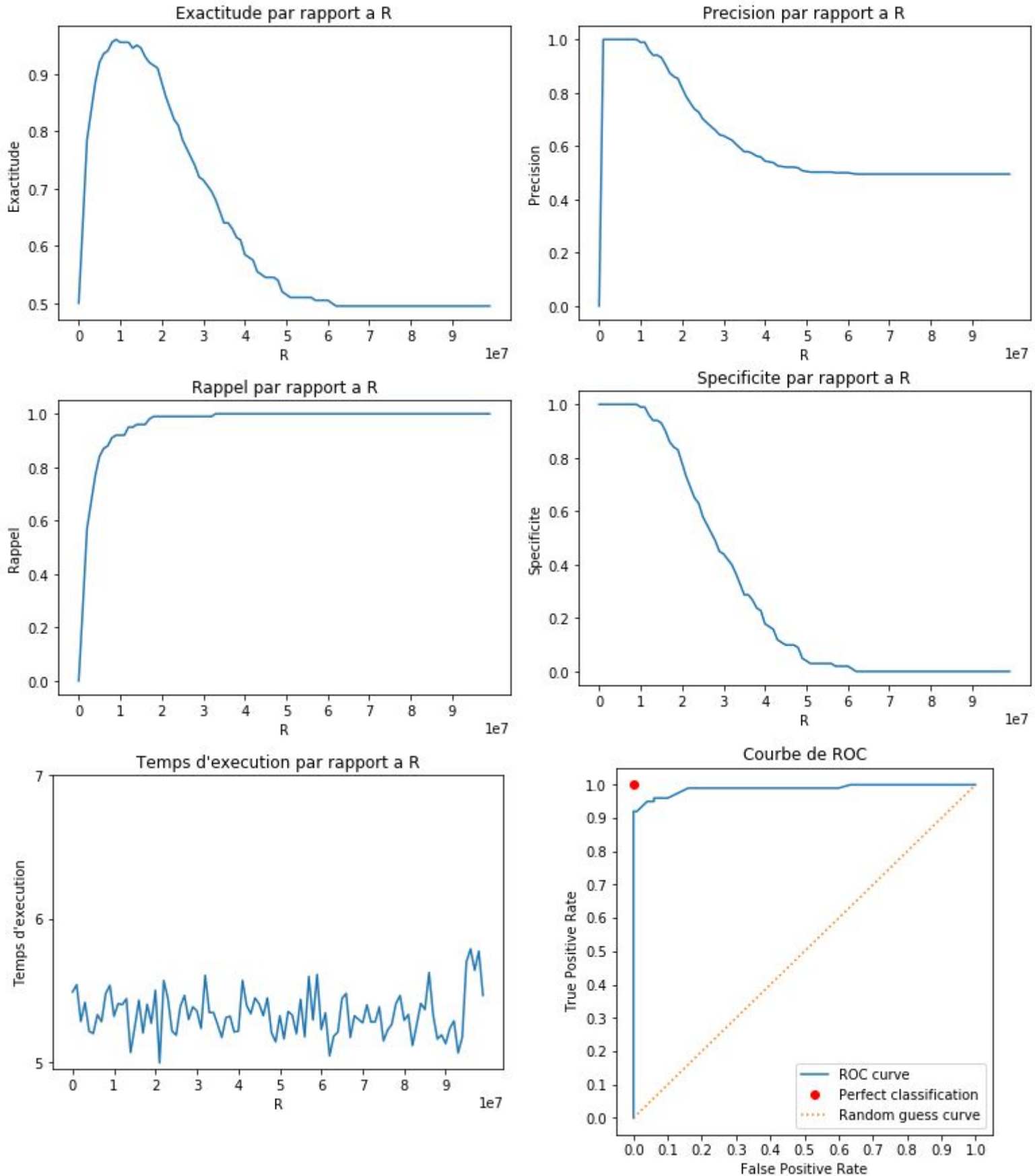
Annexe 2.a : Brute_force avec ACP Kaiser set 1



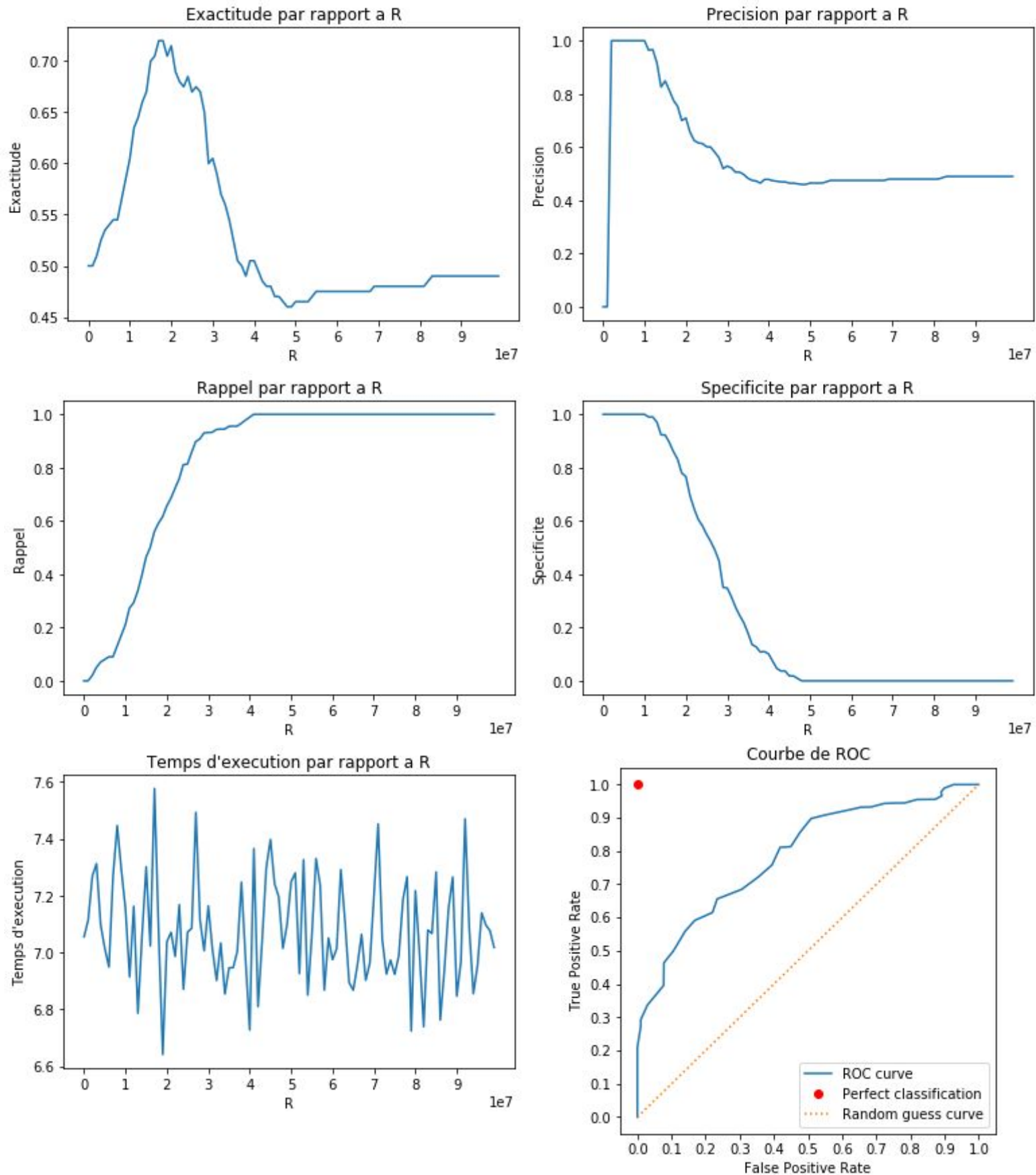
Annexe 2.b : Brute_force avec ACP Kaiser set 2



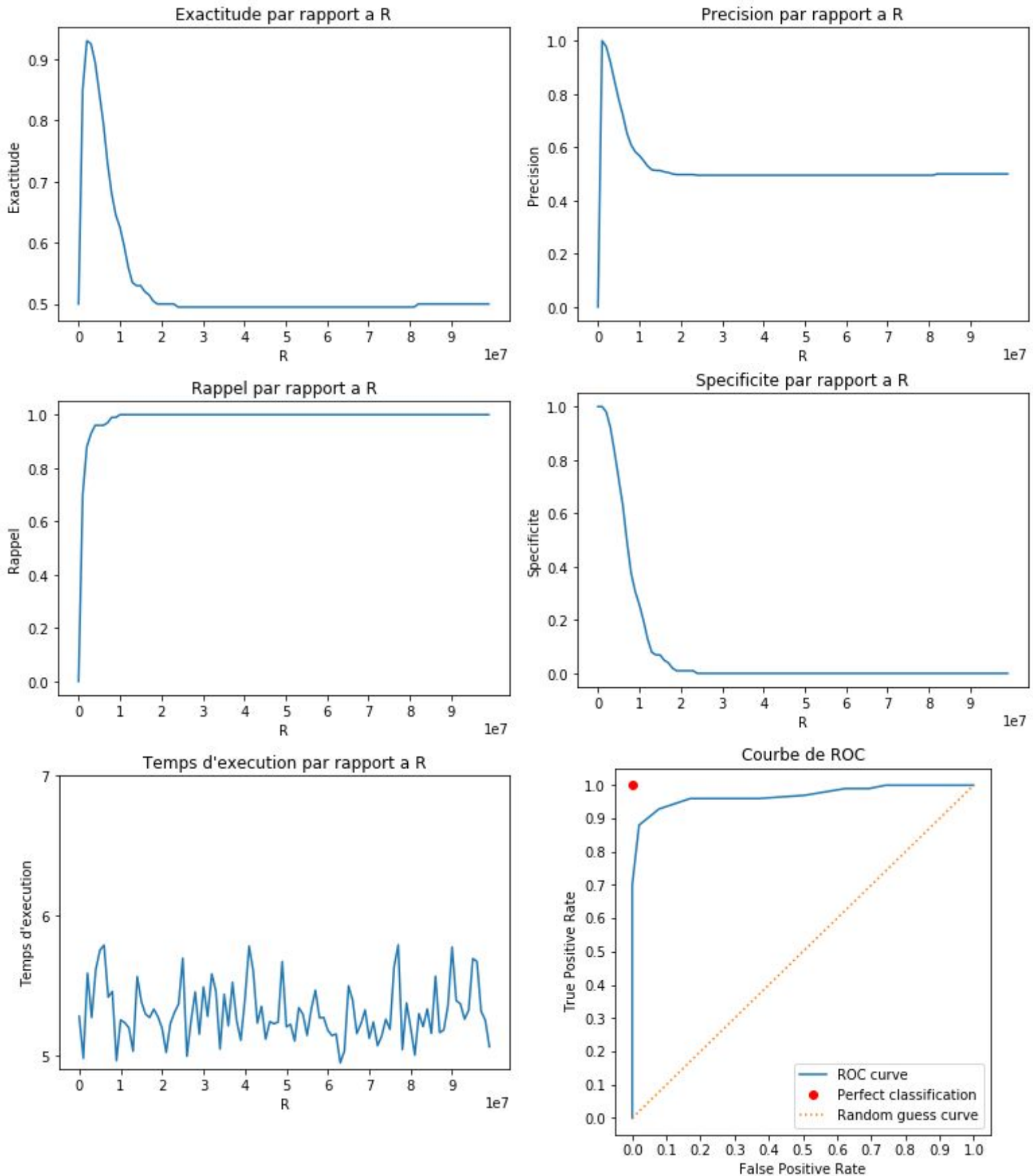
Annexe 3.a : Brute_force avec ACP Inertie set 1



Annexe 3.b : Brute_force avec ACP Inertie set 2

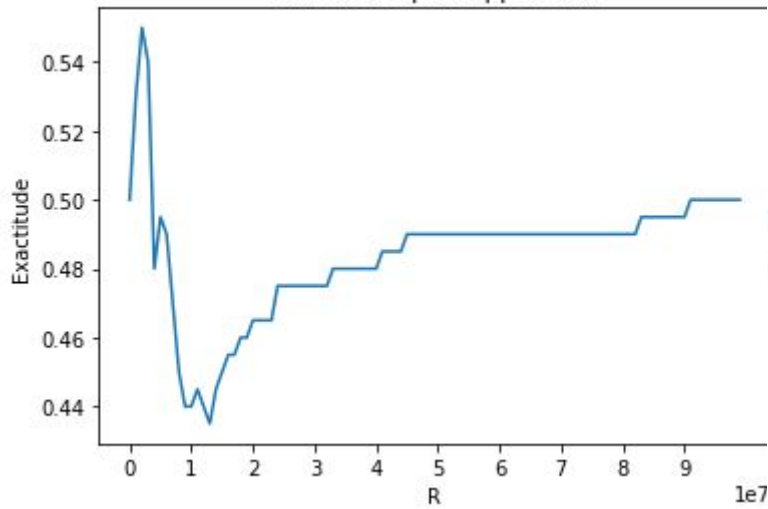


Annexe 4.a : Brute_force avec ACP Éboulis set 1

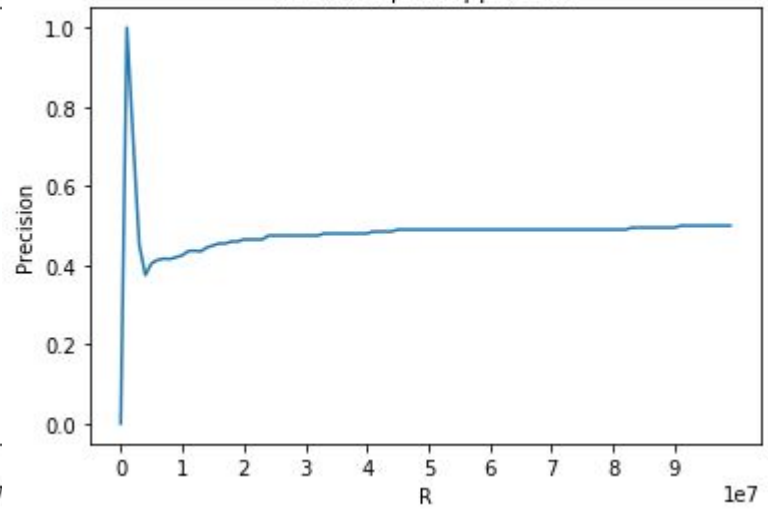


Annexe 4.b : Brute_force avec ACP Éboulis set 2

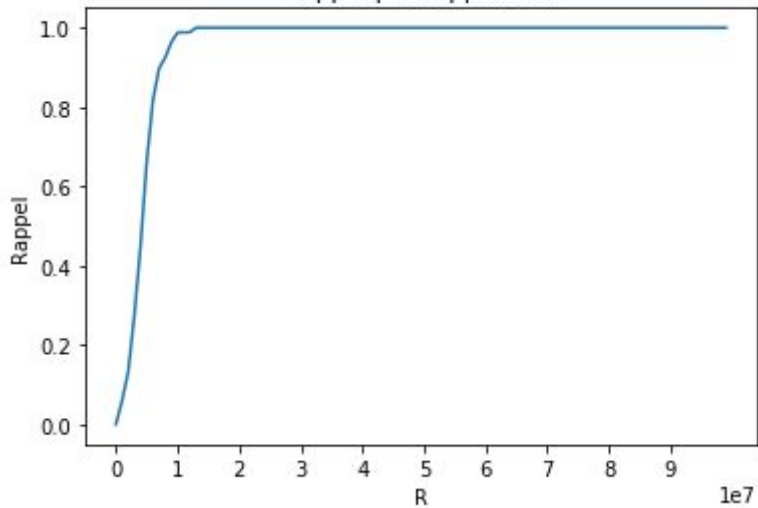
Exactitude par rapport a R



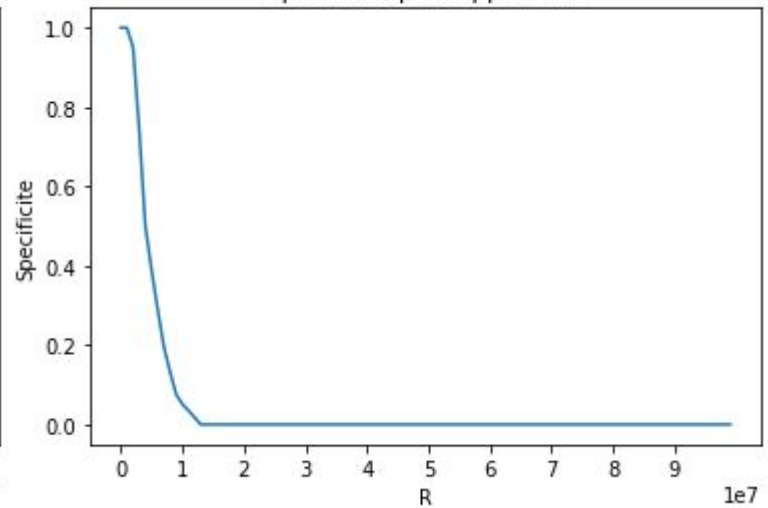
Precision par rapport a R



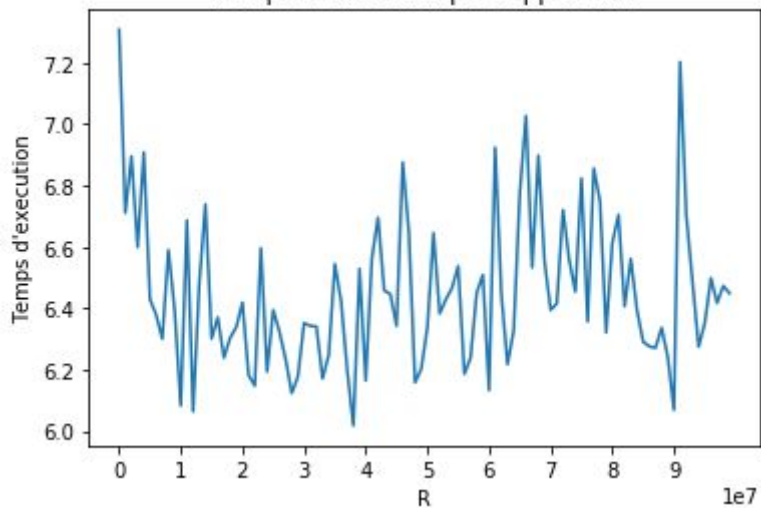
Rappel par rapport a R



Specificite par rapport a R



Temps d'exécution par rapport a R



Courbe de ROC

