M4101 Intelligence Artificielle

Analyse de données et réduction de dimensionalité

Patrick Félix, Bruno Mery, Grégoire Passault et Pierre Ramet, à partir de notes d'Akka Zemmari

LaBRI, Université de Bordeaux - CNRS

2020 - 2021

Outline

Enoncé du problème

On dispose d'un jeu de données, dont la représentation importe peu pour le moment, composé de données obtenues par imagerie médicale sur des patientes ayant des tumeurs du sein.

On dispose aussi d'un assignement de chacune de ces données à une classe ; on suppose qu'en tout on a deux classes (qu'on note classe 0 et classe 1). Donc chaque donnée est soit dans la classe 0 soit dans la classe 1. Dans notres TP, la classe 0 représente les tumeurs malignes et la classe 1 les tumeurs bénignes.

On veut mettre au point une méthode informatique, qui apprend, à partir de nos données, de sorte que si les données d'une nouvelle patiente lui sont fournies, elle sera capable de déterminer – on espère avec exactitude – si cette patiente a une tumeur maligne ou bénigne (i.e. elle assignera cette patiente soit à la classe 0 soit à la classe 1).

On s'attaque donc à un problème d'apprentissage supervisé, et plus précisément de classification binaire

Mesurer la qualité d'un classifieur : séparer les données

Supposons qu'on a une méthode de classification. On veut pouvoir tester qu'elle marche bien sur des données qu'elle n'a jamais vues , i.e. avec lesquelles elle n'a pas appris à classifier.

Pour cela, l'approche la plus simple consiste à séparer nos données initiales en deux groupes:

- des données d'apprentissage (en général 80% des données initiales),
- des données de test (les 20% restants).

On mettra au point le classifieur avec les données d'apprentissage et on le testera avec les données de test.

On note X nos données initiales, y leur classe, (X_{train}, y_{train}) la partie de ces données réservée à l'apprentissage, (X_{test}, y_{test}) la partie réservée au test.

Mesures de qualité d'un classifieur binaire (1)

On a mis au point notre classifieur avec nos données d'apprentissage.

On l'applique à nos données de test : pour chacune de ces données, on a donc la classe prédite par le classifieur et la vraie classe (connue au départ).

Pour une donnée de test elle est

- un Vrai Positif (TP) si la classe prédite et la vraie classe sont la classe 1,
- un Faux Positif (FP) si la classe prédite est 1 et la vraie classe est 0,
- un Vrai Négatif (TN) si la classe prédite et la vraie classe sont la classe 0,
- un Faux Négatif (FN) si la classe prédite est 0 et la vraie classe est 1.

Mesures de qualité d'un classifieur binaire (2)

La **Précision** du classifieur est définie par

$$Prec = \frac{TP}{TP + FP}$$

Le **Rappel (recall)** du classifieur est défini par

$$Rappel = \frac{TP}{TP + FN}$$

Le **Score** F_1 du classifieur est défini par

$$F_1 = 2\frac{Prec \times Rappel}{Prec + Rappel}$$

L'**Exactitude (accuracy)** du classifieur est définie par

$$Exactitude = \frac{TP + TN}{TP + FP + TN + FN}$$

Analyse préliminaire des données

Un principe capital. Avant toute analyse de données, il est nécessaire de regarder les données disponibles, d'en comprendre la nature (mathématique), et, souvent, de les prétraiter.

Dans un premier temps on ne s'intéresse pas à la classification, et on se contente de regarder X.

On passe donc à notre calepin Jupyter pour effectuer cette analyse préliminaire.

Description des données

Les données dont nous disposons représentent les résultats de l'analyse de 569 images de tumeurs (cancer du sein), obtenues de 569 patientes.

- Chaque patiente représente une donnée (point de donnée).
- Chaque point de donnée est décrit par 30 attributs (essentiellement géométriques) des cellules observées sur l'image : rayon, périmètre, aire, texture, régularité, compacité, concavité, points concaves, symétrie, dimension fractale. Pour chacun de ces attributs on a 3 valeurs : valeur moyenne, déviation standard, valeur maximum.

De plus, on a associé chaque point de donnée à une classe décrivant la nature de la tumeur bénigne (357), maligne (212).

Pour la suite, on se contentera de ne regarder que les valeurs moyennes, donc 10 attributs par point de donnée.

Apartée statistique : moyenne, déviation standard, erreur standard

Supposons que pour une patiente, on observe N cellules dans l'image, de rayons respectifs $r_1, \ldots, r_N$.

Le rayon moyen μ, la déviation standard σ et l'erreur standard σ' sont définis par

$$\mu = \left(\sum_{i=1}^{N} r_i \right) / N, \ \sigma = \sqrt{\left(\sum_{i=1}^{N} (r_i - \mu)^2 \right) / N}, \ \sigma' = \sigma / \sqrt{N}.$$

La déviation standard est une quantité statistique représentant la moyenne de la déviation d'un rayon par rapport au rayon moyen.

Exemple. $N = 4, r_1 = 1, r_2 = 4, r_3 = 4, r_4 = 1$. On a donc $\mu = 2.5$, $\sigma = 1,837$, $\sigma' = 0,918$.

Si on avait $r_1 = 2.25, r_2 = 2.75, r_3 = 2.3, r_4 = 2.8$, $\mu = 2.5$, $\sigma = 0,276$ et $\sigma' = 0,138$.

Représentation mathématiques des données

On peut représenter ces données par deux objects mathématiques.

- Une matrice X à 569 lignes et 10 colonnes. Chaque ligne représente un point de donnée et chaque colonne un attribut : $X[i, j]$ représente la valeur de l'attribut en colonne j pour le point de donnée en ligne i. Chaque point de données est donc un vecteur de dimension (longueur) 10.
- Un vecteur binaire y de 569 entrées : $y[i] = 0$ indique que la patiente représentée par la ligne i de X a une tumeur maligne, et $y[i] = 1$ indique que la patiente représentée par la ligne i de X a une tumeur bénigne.

Dimension/taille. On a donc un jeu de données de taille 569 et de dimension 10.

Que faire avec ces données, que regarder avant de les traiter?

Questions.

- En quelles unitées sont-elles exprimées?
- Quel intervalle de valeurs prend chaque attribut?
- Quelle est la moyenne, déviation standard, ... de chaque attribut?
- Plus précis, quelles est la distribution des valeurs de chaque attribut.

Observation. Les données ont des ordres de grandeur différents, la surface moyenne ayant un effet écrasant. C'est une situation qui généralement conduit à une diminution des performances des méthodes de classification qui vont avoir tendance à accorder une importance démesurée à cet attribut.

Normalisation et standardisation

Normalisation. Pour un attribut A, on normalise comme suit:

$$A_{norm} = \frac{A - A_{min}}{A_{max} - A_{min}}.$$

Standardisation. Pour un attribut A, on standardise comme suit:

$$A_{std} = \frac{A - \mu_A}{\sigma_A}.$$

On utilise la standardisation quand les attributs ont une distribution normale (en forme de cloche), de manière à ce que les attributs standardisés suivent une distribution de moyenne 0 et de déviation standard 1.
Dans notre cas, nous allons standardiser et obtenir une matrice X_{std}.

Le signal de classification est-il aisément observable?

Maintenant que nos données ont une forme qui nous convient, on peut se pencher sur la classification. La première question qui vient à l'esprit est la suivante : a-t'on vraiment besoin d'utiliser des méthodes de classification sophistiquées pour être capable de prédire, pour une nouvelle patiente, la nature de sa tumeur?

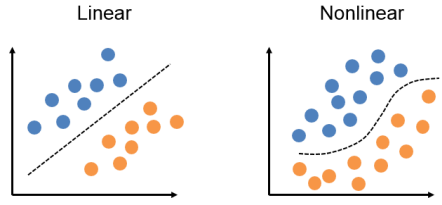
Comment peut-on explorer (idéalement de manière visuelle) cette question?

Peut-on classifier en utilisant un ou deux attributs?

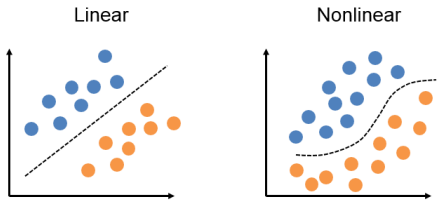
En d'autres termes, peut-on ignorer la plupart des attributs et se contenter d'un ou deux, bien choisis, pour classifier la nature de la tumeur d'une nouvelle patiente?

Si c'est le cas, en choisissant bien disons deux attributs, et en représentant chaque point de données par un point dans le plan ayant pour coordonnées les valeurs pour ces deux attributs, on observerait une claire séparation entres les points de données représentant des tumeurs bénignes et malignes (e.g. bleu pour tumeur bénigne et orange pour tumeur maligne).

Exemple de deux classifieurs possibles.



Peut-on classifier en utilisant un ou deux attributs?



Etant donnée une nouvelle patiente, on extrait les valeurs observées pour les deux attributs choisis, on graphe le point correspondant et, en fonction de quelle région du plan il est contenu, on classifie la tumeur.

Retournons à notre calepin Jupyter.

Reduction de dimensionalité : Motivation

Notre analyse préliminaire suggère que l'on peut obtenir un bon classifieur en se basant sur le signal encodé dans deux attributs (i.e. en réduisant nos données à deux attributs). Mais peut-on faire mieux, tout en restant en deux dimensions?

Si on peut cela implique qu'il existe un meilleur signal dans nos donnée pour classifier une tumeur en bénigne ou maligne, ce signal est encodé par une combinaison non triviale de plusieurs attributs.

Notre but est de découvrir une telle combinaison (si elle existe) avant de l'utiliser pour mettre au point un algorithme de classification.

Nature mathématique de nos données

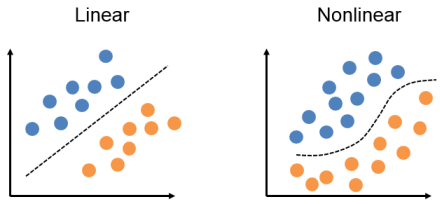
Notation. On note nos points de données $X_1, \ldots, X_N$ ($N = 569$ pour nous), où X_i est la ligne i de la matrice X_{std}.

Rappel. Un point de donnée X_i est un point dans un espace (euclidien) de dimension $D = 10$: $X_i = (X_{i,1}, \ldots, X_{i,10})$ est un vecteur de 10 coordonnées.

On peut donc voir nos données comme un objet géométrique (points dans un cube unitaire en 10 dimensions) ou algébrique (matrice). Nous allons naviguer entre ces deux points de vue.

Réduction de dimensionalité (1)

Supposons qu'il existe deux fonctions f_1 et f_2, de $\mathbf{R}^{10}$ dans $\mathbf{R}$ telles que si on représente chaque donnée X_i dans le plan par un point de coordonnées $(f_1(X_i), f_2(X_i))$, on obtient la figure suivante.



Alors, on a notre classifieur, qui est un classifieur de dimension $d = 2$.

Réduction de dimensionalité (2)

La situation décrite dans le transparent précédent est idéale dans le sens où on a pu réduire la dimension de notre problème de classification de $D = 10$ à $d = 2$, et on a donc un classifieur de dimension 2, i.e. visualisable dans le plan.

Il est possible qu'en fait, pour bien classifier, il faille considérer plus de deux fonctions, $f_1, \ldots, f_d$ $(d > 2)$. On perd la propriété d'un classifieur visualisable dans le plan, mais tant que $d < D$, on est satisfait, car on a réduit la dimensionalité de notre problème de classification.

Réduction de dimensionalité et IA

But. Nous voulons déterminer si il existe d (idéalement d est petit) et un vecteur $F = (f_1, \ldots, f_d)$ de fonctions tels que la transformation de notre matrice $N \times D$ de données X_{std} en une matrice $N \times d$ dénotée $F(X)$ (la ligne i de $F(X)$ est $F(X_i) = (f_1(X_i), \ldots, f_d(X_i))$) avec laquelle on peut bien séparer les tumeurs malignes des tumeurs bégnines.

Pourquoi est-ce de l'IA? On veut découvrir, avec le moins d'a priori possible, un signal non trivial, de classification, encodé par nos données défini comme une combinaison du signal encodé dans X_{std}.

Méthodes de réduction de dimensionalité

Il existe de nombreuses méthodes de réduction de dimensionalité, car il s'agit d'une technique d'analyse de données en grande dimension très puissante. Les trois approches les plus utilisées sont les suivantes.

- **MultiDimensional Scaling (MDS).**
- **T-distributed Stochastic Neighbor Embedding (t-SNE).**
- **Analyse en Composantes Principales (ACP).**

Elles sont bien entendu implémentées dans Scikit-learn et nous allons les tester dans notre calepin Jupyter.

Méthode ACP (PCA)

On part de notre matrice X_{std} (de taille $N \times D$) et on veut trouver une matrice P de taille $D \times D$ (nous verrons comment et pourquoi). Cette matrice, appelée la matrice des composantes principales (CP) va nous permettre de transformer nos données dans un espace de dimension $d \leq D$.

Dans notre cas, nous voulons transformer nos données en deux dimensions, i.e. $d = 2$. Pour cela, notons P^i la i^{eme} colonne de P. On définit la fonction

$$f_i(X) = X \times P^i.$$

On transforme donc X_{std} en une matrice $X_{pca,d}$ de taille $N \times d$ dont les d colonnes sont respectivement $f_1(X), \ldots, f_d(X)$, ou de manière équivalente par le produit de matrice

$$X_{pca,d} = X \times [P^1 \ldots P^d]$$

et on peut ensuite utiliser cette matrice $X_{pca,d}$ pour notre problème de classification.

Méthodes t-SNE et MDS

Ces deux méthodes fonctionnent sur un principe similaire :

- On choisit d, la dimension de l'espace dans lequel on veut représenter nos données.
- L'algorithme (t-SNE ou MDS) calcule, à partir de X une matrice de taille $N \times d$ (notée $X_{tsne,d}$ pour $t - SNE$ ou $X_{mds,d}$ pour MDS).
- Le principe de construction de ces matrices est d'essayer de grouper (dans le plan si $d = 2$) les points qui sont proches dans l'espace initial à D dimensions et de séparer les points qui sont éloignés dans l'espace initial à D dimensions.
- La méthode t-SNE se base sur un modèle de probabilité pour définir la notion de points proches, alors que MDS se base sur la distance euclidienne.

Utilisation des méthodes de réduction de dimensionalité (1)

ACP produit un élément important, la matrice P, qui permet de transformer n'importe quelle matrice de données en D dimensions en une matrice de données réduite à d dimensions.

Nous allons l'utiliser comme suit:

- On sépare notre matrice X en deux matrices de dimensions D : la première, X_{train} comporte 80% des données initiales et la seconde, X_{test} les 20% restants des données.
- On apprend un classifieur en utilisant uniquement X_{train}; dans notre cas, on réduit d'abord X_{train} en une matrice à d dimensions en utilisant l'ACP, $X_{train,pca,d}$, et on apprend un classifieur à partir de cette matrice. On note P_{train} la matrice des CP obtenue avec l'ACP.
- On évalue la qualité de notre classifieur en utilisant la matrice $X_{test,pca,d} = X_{test} \times P_{train}$, i.e. la réduction à d dimensions de la matrice X_{test} : on évalue donc notre classifieur sur des données non vues durant l'entraînement.

Utilisation des méthodes de réduction de dimensionalité (2)

Essayons l'approche décrite dans le transparent précédent, avec un classifieur de type k-nearest neighbours (kNN, k plus proches voisins, nous verrons d'ici peu comment il marche). On observe que ce classifieur marche très bien.

Si nous essayons de reproduire cette expérience avec les méthodes t-SNE et MDS, il manque un élément capital : un algorithme de transformation, appris sur les données d'apprentissage X_{train} pour réduire la dimensionalité des données de test X_{test}.

On peut bien entendu réduire X_{test} en utilisant directement la méthode t-SNE ou MDS, mais essayons cette approche dans notre calepin et on verra que ça ne marche pas du tout. t-SNE et MDS servent à reduire la dimensionalité des données essentiellement pour les visualiser ou faire de la classification non supervisée mais pas pour apprendre un classifieur sur des données d'apprentissage.

Conclusion

Nous avons travaillé sur un problème classique d'IA appliquée à l'analyse de données, la classification binaire supervisée.

Nous avons vu comment prétraiter des données et essayer de déterminer essentiellement visuellement si on observe un bon signal de classification.

Nous avons vu comment apprendre un classifieur en utilisant un jeu de données d'apprentissage et comment le tester avec un jeu de test. Nous avons découvert des métriques d'évaluation, exactitude et score F1.

Nous avons exploré une technique de classification binaire : réduction de dimensionalité par ACP et classification par kNN, sans encore rentrer dans les détails techniques de ces méthodes classiques.

Nous avons brièvement vu deux autres méthodes de récuction de dimensionalité, t-SNE et MDS, et pourquoi elles se prêtent mal au problème de classification supervisée.