TELECOM ParisTech **SD204**
2016/2017 **Joseph Salmon, François Portier**

TP noté : SVD-PCA

Pour ce travail vous devez déposer un **unique** fichier au format `nom_prenom.ipynb` sur le site pédagogique du cours, partie validation.

Vous deposerez votre fichier sur Éole (SD204 > TP), avant le dimanche 22/01/2017 23h59.

La note totale est sur **20** points répartis comme suit :

— qualité des réponses aux questions : **15** pts,

— qualité de rédaction, de présentation et d'orthographe : **2** pts,

— indentation, style PEP8, commentaires adaptés, etc. : **2** pts,

— absence de bug : **1** pt.

Les personnes qui n'auront pas rendu leur devoir avant la limite obtiendront **zéro**.

Exercice 1. (Retour sur l'algèbre linéaire)

On rappelle l'*astuce du noyau* : pour tout $X \in \mathbb{R}^{n\times p}$ et $\mathbf{y} \in \mathbb{R}^n$ l'équation suivante est vraie :

$$X^\top (XX^\top + \lambda \operatorname{Id}_n)^{-1}\mathbf{y} = (X^\top X + \lambda \operatorname{Id}_p)^{-1} X^\top \mathbf{y} \tag{1}$$

1) Vérifier cette propriété numériquement pour $\lambda = 10^{-5}$ sans inverser de matrice[1], pour une matrice X dont les entrées sont *i.i.d.* et tirées selon une loi gaussienne centrée et de variance 4, et $\mathbf{y}$ un vecteur dont les entrées sont tirés selon une loi uniforme sur $[-1, 1]$
   (a) avec $n = 100$ et $p = 2000$,
   (b) avec $n = 2000$ et $p = 100$.
2) Pour les même X et $\mathbf{y}$ que ci-dessus, proposez une étude numérique et/ou graphique qui compare selon la taille de n et de p il est plus intéressant d'utiliser l'une ou l'autre des méthodes de calcul dans l'équation (1). Commentez votre choix.

Exercice 2. (Spectre de matrice aléatoire)

3) Choisissez deux lois de probabilités, non-gaussiennes, de moyenne nulle et de variance 1, et construire des matrices $X \in \mathbb{R}^{n\times p}$ dont les entrées sont tirées suivant ces lois.
4) Afficher sur un même graphique les valeurs singulières X pour $n = 1000$, pour $p = 200, 500, 1000, 2000$.
5) Afficher sur un même graphique le spectre (*i.e.,* les valeurs propres) de $X^\top X/n$ pour $n = 1000$, pour $p = 200, 500, 1000, 2000$.

Exercice 3. (Méthode de la puissance itérée)

On considère de nouveau la matrice $X \in \mathbb{R}^{p\times n}$ générée à l'exercice 1, question 1a.

1. on pourra utiliser la fonction `allclose` de `numpy`

6) Écrire une fonction qui code l'algorithme 1.

Algorithm 1: Méthode de la puissance itérée

Input : X matrice ; n_{iter} le nombre maximal d'itérations ; u, v vecteurs initiaux

for $j = 0, \ldots, n_{\text{iter}}$ **do**

  $u \leftarrow Xv$ // u update

  $v \leftarrow X^\top u$ // v update

  $v \leftarrow v/\|v\|$; $u \leftarrow u/\|u\|$ // Normalisation

Output: u, v

7) Illustrer visuellement la convergence de l'algorithme. Identifier que les vecteurs u, v obtenus approchent les vecteurs singuliers associés à la plus grande valeur singulière de X.

8) Appliquer l'algorithme précédent pour diverses valeurs d'initialisation pour u et v. L'algorithme converge-t-il encore ? si oui est-ce vers les même solutions ?

q9 undone 9) Proposer un algorithme qui approche la seconde valeur singulière la plus grande de X (sans utiliser la fonction SVD !). On pourra notamment utiliser deux fois l'algorithme précédent.

Exercice 4. (PCA)

10) Importer avec `Pandas` la base de données `defra_consumption.csv` disponible sur Éole.

11) Centrer et réduire les données en utilisant le module `preprocessing` de `sklearn`. On notera $X \in \mathbb{R}^{n \times p}$ une telle matrice (avec n le nombre d'observations, et p le nombre de variables).

12) Afficher un *scatter plot* des points après projection sur les deux premiers axes principaux.

13) Faire de même en gardant cette fois trois axes principaux [2].

14) Comparez les graphiques 2D et 3D précédents avec ceux obtenus de la manière suivante :

  (a) Calculer la matrice $X^\top X$, et la diagonaliser. Projeter sur les vecteurs associés aux 2 valeurs propres les plus grandes (puis aux trois plus grandes).

  (b) Calculer la SVD de X. Projeter sur les vecteurs associés aux 2 valeurs singulières les plus grandes (puis aux trois plus grandes).

14ab逆向问题

14c undone

  (c) Évaluer les différences de temps de calcul pour ces différentes méthodes.

Exercice 5. (Reconnaissance de visages par régression logistique)

15) Charger les données grâce au code suivant :

```
from sklearn.datasets import fetch_lfw_people
# Download the data, if not already on disk and load it as numpy arrays
lfw_people = fetch_lfw_people(min_faces_per_person=70, resize=0.4)
# introspect the images arrays to find the shapes (for plotting)
n_samples, h, w = lfw_people.images.shape
X = lfw_people.data
n_features = X.shape[1]
# the label to predict is the id of the person
y = lfw_people.target
target_names = lfw_people.target_names
n_classes = target_names.shape[0]
print("Total dataset size:")
print("n_samples: %d" % n_samples)
print("n_features: %d" % n_features)
print("n_classes: %d" % n_classes)
```

2. on affichera donc un graphique en 3D

16) Que représentent les variables explicatives ici ?

17) Pour éviter de lancer la régression logistique sur un trop grand nombre de variables explicatives, il faut réduire ce nombre avant de l'appliquer (ici, sans pénalisation). Vous comparerez deux approches :

17ab undone

(a) utiliser l'ACP et le pourcentage d'inertie expliquée pour sélectionner avant régression logistique le nombre d'attributs ;

(b) utiliser la validation croisée pour sélectionner le nombre de directions principales (vecteurs propres) trouvées par l'ACP qui seront utilisées comme nouvelle base pour apprendre la régression logistique. Vous pourrez notamment utiliser la fonction `pipeline` de `scikit-learn`.

Exercice 6. (Analyse du jeu de données `auto-mpg`)

On travaille maintenant sur le fichier `auto-mpg.data`. On cherche à régresser linéairement la consommation des voitures sur leurs caractéristiques : nombre de cylindres, cylindrées (*engine displacement* en anglais), puissance, poids, accélération, année, pays d'origine et le nom de la voiture. Le vecteur contenant la consommation des voitures (plus précisément la distance parcourue, en miles, pour un gallon, ou mpg) est noté $\mathbf{y}$; les colonnes de X sont les régresseurs quantitatifs, donc pour le moment on laisse de côté les variables `origin` et `car name`.

18) Importer avec `Pandas` la base de données disponible ici https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data-original. On ajoutera le nom des colonnes en consultant l'adresse : https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.names avec l'attribut `'name'` de `import_csv`. On pourra regarder l'intérêt de l'option `sep=r"\s\+"` si besoin. Quelle est le marqueur utilisé pour les données manquantes dans le fichier utilisé ? Enlever les lignes possédant des valeurs manquantes dans la base de données si besoin.

19) Calculer l'estimateur des moindres carrés $\hat{\boldsymbol{\theta}}$ et sa prédiction $\hat{\mathbf{y}}$ sur une sous partie de la base obtenue en gardant les 9 premières lignes. Que constatez-vous pour les variables `cylinders` et `model year` ?

20) Calculer $\hat{\boldsymbol{\theta}}$ et $\hat{\mathbf{y}}$ cette fois sur l'intégralité des données, après les avoir centrées et réduites. Quelles sont les deux variables qui expliquent le plus la consommation d'un véhicule ?

21) Calculer $\|\mathbf{r}\|^2$ (le carré de la norme du vecteur des résidus), puis $\|\mathbf{r}\|^2/(n-p)$. Vérifier numériquement que :

$$\|\mathbf{y} - \bar{y}_n \mathbf{1}_n\|^2 = \|\mathbf{r}\|^2 + \|\hat{\mathbf{y}} - \bar{y}_n \mathbf{1}_n\|^2.$$

22) Supposons que l'on vous fournisse les caractéristiques suivantes d'un nouveau véhicule :

cylinders	displacement	horsepower	weight	acceleration	year
6	225	100	3233	15.4	76

Prédire sa consommation [3].

23) Utiliser la transformation `PolynomialFeatures` de `sklearn` sur les données brutes, pour ajuster un modèle d'ordre deux (avec les termes d'interactions : `interaction_only=False`). On normalisera et recentrera après avoir créé les nouvelles variables explicatives.

24) Proposer une manière de gérer la variable `origin`, par exemple avec `pd.get_dummies`. On ajustera un modèle linéaire sans constante dans ce cas. Déterminer laquelle des trois origines est la plus efficace en terme de consommation [4].

3. A titre d'information, la consommation effectivement mesurée sur cet exemple était de 22 mpg.
4. Pour info, 1 = usa ; 2 = europe ; 3 = japan

25) Procéder de même cette fois en fonction de la marque de la voiture. On pourra utiliser `str.split` et `str.replace` pour créer une nouvelle variable `'brand'`.

26) Reprendre la matrice X obtenue sans les variables catégorielles. Obtenez numériquement la SVD (partielle) de $X = USV^\top$ (par exemple en considérant l'option `full_matrices=False`). La diagonale cette matrice H, forme le vecteur des "leviers", qu'on ajoutera comme nouvelle variable. Trier la base de données en fonction de cette nouvelle variable, et expliquer en quoi les voitures ayant les trois valeurs de "levier" maximales sont atypiques.

26 undone

- Lien pour aller plus loin : -

⋆ ⋆ ⋆ http://www.astro.washington.edu/users/vanderplas/