

# Exercices « Pandas »

Les données, réponses et supports se trouvent sur la page GitHub :  
<https://github.com/pbellot/FormationPANDAS>

Les données se trouvent dans le dossier Datasets.

## Manipulation de Dataframes sur les données Titanic

L'objectif est d'obtenir un certain nombre d'analyses des données relatives aux passagers ayant embarqué sur le Titanic.

- 1) Récupérer le fichier et étudier son contenu notamment afin de connaître son format (paramètres pour lire le CSV)
- 2) Avec la méthode `read_csv` Pandas, ouvrir le fichier et transférer son contenu dans un `DataFrame`
- 3) Etudier le contenu du `Dataframe` avec les méthodes `info`, `head`, `describe`
- 4) S'entraîner à n'afficher que certaines colonnes ou lignes, à accéder à une valeur précise
- 5) Intervertir deux colonnes dans le dataframe
- 6) Déterminer les différentes valeurs d'âge contenues dans le dataframe et les ordonner (utiliser la méthode `sort` du Numpy et la méthode `unique()` de Pandas)
- 7) Afficher les lignes correspondantes aux 3 passagers les plus jeunes, puis aux 3 passagers les plus âgés (utiliser les méthodes `nsmallest()` et `nlargest()`)
- 8) S'entraîner à utiliser la méthode `query()` de Pandas de manière à ne sélectionner que les lignes qui répondent à certains critères. Par exemple déterminer quels sont les passagers de 3<sup>è</sup> classe qui ont survécu. Utiliser la méthode `len()` sur le résultat de manière à en connaître le nombre.
- 9) Déterminer le prix moyen du billet pour les personnes de 3<sup>è</sup> classe qui ont survécu et pour celles qui n'ont pas survécu
- 10) Trier le dataframe par ordre alphabétique selon les noms du passagers grâce à la méthode `sort_values()`
- 11) Remplacer les valeurs C, Q et S de la variable `Embarked` par Cherbourg, Queenstown, Southampton en utilisant la méthode `map()`
- 12) A l'aide d'une boucle `for`, itérer sur le dataframe et afficher chaque ligne, une à une (utilisation de la méthode `iterrows()`). Eventuellement utiliser un `break` une fois arrivé à la ligne 10
- 13) Afficher les lignes pour lesquelles l'âge n'est pas connu (utiliser la méthode `isnull()`)
- 14) Déterminer le nombre de personnes dans chaque classe (`Pclass`) en utilisant :
  - a. les méthodes `groupby` et `count`
  - b. la méthode `value_counts` seule
- 15) Affichage avec Seaborn :
  - a. Afficher un histogramme du nombre de personnes dans chaque classe, sans utiliser `groupby` ni `count`, mais seulement la méthode `countplot` de la bibliothèque Seaborn
  - b. On s'intéresse à connaître le nombre de survivants dans chaque classe. Subdiviser chaque classe en deux en utilisant le paramètre `hue` de `countplot` sur la variable `Survived`.
  - c. On souhaite maintenant connaître le nombre de survivants pour différentes classes d'âge. Définir 4 catégories d'âge selon les différentes valeurs de la variable `Age` puis utiliser la méthode `cut()` de Pandas pour rajouter une colonne « `Age_categorie` » avant d'afficher le résultat avec `sns.countplot()`
- 16) Filtrer le dataframe de manière à ne garder que des lignes pour lesquelles la classe et l'âge sont connus (méthode `dropna` avec paramètre `subset`)
- 17) Afficher les corrélations entre variables avec la méthode `corr()` de Pandas puis avec la méthode `heatmap()` de Seaborn
- 18) Expérimenter la méthode `get_dummies` pour transformer les variables catégorielles en différentes nouvelles variables binaires
- 19) Appliquer une analyse en composantes principales sur le dataframe `titanic`

## Données « évolution des hospitalisations Covid-19 »

L'objectif est d'afficher l'évolution de l'incidence du Covid-19 dans chaque région de France puis de partitionner (approche de type *clustering* K-moyennes) les régions de telle sorte que dans chaque classe apparaissent des régions où l'évolution de l'incidence du covid-19 est comparable.

Les données se trouvent dans un fichier CSV : covid-hospit-incid-reg-2022-05-30-19h00.csv (15258 lignes)

- 1) Récupérer le fichier et étudier son contenu notamment afin de connaître son format (paramètres pour lire le CSV)
  - 2) Avec Pandas, ouvrir le fichier et transférer son contenu dans un DataFrame
  - 3) Afficher sur un graphique l'évolution temporelle du taux d'incidence en réanimation : une courbe par région (aide : utiliser la fonction pivot de Pandas)
  - 4) La réalisation de la question 3) a dû passer par la création d'un DataFrame où les valeurs de l'index correspondent aux noms de régions et les colonnes aux jours des relevés. Les valeurs des lignes seront vues comme des vecteurs associés aux régions. Utiliser ces vecteur pour appliquer la méthode `sklearn.cluster.KMeans` (<https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>). Créer un modèle (KMeans) puis obtenir les groupes avec la méthode `predict` appliquée au modèle.
  - 5) Afficher un graphique où chaque point correspond à une région en utilisant une couleur différente par classe. NB: les points étant dans un espace de dimension  $> 3$ , une réduction de la dimension est nécessaire.
  - 6) Faire varier le nombre d'itérations (condition d'arrêt des kMeans) et le nombre de classes (clusters) et estimer l'évolution de la cohérence des groupes grâce à l'estimation d'un indice de Rand
- ```
from sklearn import metrics
metrics.adjusted_rand_score(_le résultat de la prédiction_, _labels_de_référence_ou_précédents)
```

## Données « rythme cardiaque / Apple Watch »

L'objectif est d'afficher sur un graphique l'évolution temporelle d'un rythme cardiaque tel que capté par une Apple Watch 5 puis de déterminer les moments de forte variation.

Les données se trouvent dans un fichier XML : exportExtract.xml.gz (77497 lignes)

- 1) Récupérer le fichier et étudier son contenu : en-tête XML, structure, format...
  - 2) Avec Pandas, ouvrir le fichier en ne conservant dans un DataFrame que les instants de captation *endDate* (heures/min/secondes) et les valeurs du rythme cardiaque *value*
  - 3) Afficher sur un graphique l'évolution temporelle du rythme cardiaque
  - 4) Rajouter une colonne dans le DataFrame où l'on indique l'écart du rythme cardiaque entre deux mesures.
  - 5) Calculer un delta moyen toutes les 30 mesures puis indiquer les 5 minutes où les changements de rythme cardiaque ont été les plus forts.
- Aide : utiliser la méthode *rolling*