Atelier Python pour Data Scientist

Séance 1: Lecture et affichage de données *via* Pandas et Matplotlib

Ali ZAINOUL
ali.zainoul.az@gmail.com

December 19, 2023

Windows

- Téléchargez Anaconda depuis
  `https://www.anaconda.com/products/distribution`
- Suivez les instructions d'installation.
- Ouvrez le PowerShell.
- Créez un environnement virtuel:
  `conda create --name mon_environnement python=3.8`
- Activez l'environnement:
  `conda activate mon_environnement`
- Installez Jupyter Notebook:
  `conda install jupyter`

MacOS / Linux

- Téléchargez Anaconda depuis `https://www.anaconda.com/products/distribution`
- Suivez les instructions d'installation.
- Ouvrez le terminal.
- Créez un environnement virtuel:
  `conda create --name mon_environnement python=3.8`
- Activez l'environnement:
  `conda activate mon_environnement`
- Installez Jupyter Notebook:
  `conda install jupyter`

Lecture d'un Fichier CSV et Visualisation avec Matplotlib

- Ouvrez Jupyter Notebook dans l'environnement virtuel.
- Créez un nouveau notebook.
- Importez Pandas: `import pandas as pd`.
- Importez Matplotlib: `import matplotlib.pyplot as plt`.
- Utilisez la commande pour lire le fichier CSV:
  `df = pd.read_csv("csvExamples/courses.csv")`
- Affichez les premières lignes du DataFrame:
  `df.head()`
- Set labels and title:
  - `plt.xlabel('Programming Language')`
  - `plt.ylabel('Days to Learn')`
  - `plt.title('Days to Learn Programming Languages')`
- Create and display a bar chart:
  - `plt.bar(df['Programming Language'], df['Learning Days'], color='blue')`
  - `plt.show()`

Explications (Activité 1)

- **Import Libraries:**
  - `pandas` est importé pour la manipulation et l'analyse des données.
  - `matplotlib.pyplot` est importé pour créer des visualisations.
- **Read CSV File:**
  - `pd.read_csv()` lit le fichier CSV dans un DataFrame pandas.
- **Inspect Data:**
  - `head()` affiche les premières lignes du DataFrame pour inspecter les données.
- **Set Labels and Title:**
  - `xlabel()`, `ylabel()`, et `title()` définissent les étiquettes et le titre du graphique.
- **Create Bar Chart:**
  - `plt.bar()` crée un graphique à barres avec des données x et y spécifiées, et définit la couleur des barres à bleu.
- **Display the Plot:**
  - `plt.show()` affiche le graphique créé.

Visualisation de Données d'Étudiants avec Matplotlib et Pandas - Étape 1

- Ouvrez Jupyter Notebook dans l'environnement virtuel.
- Créez un nouveau notebook.
- Importez Pandas: `import pandas as pd`.
- Importez Matplotlib: `import matplotlib.pyplot as plt`.
- Utilisez la commande pour lire le fichier CSV:
  `df = pd.read_csv("csvExamples/randomStudentData.csv")`
- Créez des sous-graphiques avec 1 ligne et 2 colonnes, ajustez la taille de la figure:
  `plt.figure(figsize=(12, 4))`
- Set the main title for the entire plot:
  `plt.suptitle('Student analytics')`

Visualisation de Données d'Étudiants avec Matplotlib et Pandas - Étape 2

- Create Subplot 1: Bar chart for "Marks":
  - `plt.subplot(1, 2, 1)`
  - `plt.bar(df['StudentID'], df['Marks'], color='blue')`
  - `plt.title('Student marks')`
  - `plt.xlabel('Student ids')`
  - `plt.ylabel('Student marks')`
  - `plt.xticks(rotation=45, ha='right')`

Visualisation de Données d'Étudiants avec Matplotlib et Pandas - Étape 3

- Create Subplot 2: Bar chart for "IQ":
  - `plt.subplot(1, 2, 2)`
  - `plt.bar(df['StudentID'], df['IQ'], color='orange')`
  - `plt.title('Student IQs')`
  - `plt.xlabel('Student ids')`
  - `plt.ylabel('Corresponding student IQ')`
  - `plt.xticks(rotation=45, ha='right')`
- Adjust subplot parameters for better layout:
  `plt.tight_layout()`
- Display the entire plot with subplots:
  `plt.show()`

Explications (Activité 2)

- **Import Libraries:**
  - `pandas` est importé pour la manipulation et l'analyse des données.
  - `matplotlib.pyplot` est importé pour créer des visualisations.
- **Read CSV File:**
  - `pd.read_csv()` lit le fichier CSV dans un DataFrame pandas.
- **Create Subplots:**
  - `plt.figure(figsize=(12, 4))` crée une figure avec une taille spécifiée.
  - `plt.suptitle()` définit le titre principal pour l'ensemble du graphique.
  - `plt.subplot()` crée des sous-graphiques avec 1 ligne et 2 colonnes.

Explications (Activité 2 - Suite)

- **Create Bar Charts:**
  - `plt.bar()` crée des graphiques à barres avec des données spécifiées.
  - Les paramètres tels que la couleur, le titre, les étiquettes d'axe sont définis pour chaque sous-graphique.
- **Adjust Layout:**
  - `plt.tight_layout()` ajuste automatiquement les paramètres du sous-graphique pour une meilleure mise en page.
- **Display the Plot:**
  - `plt.show()` affiche l'ensemble du graphique avec les sous-graphiques.

Régression Linéaire: Marks vs IQ

- Ouvrez Jupyter Notebook dans l'environnement virtuel.
- Créez un nouveau notebook.
- Importez Pandas: `import pandas as pd`.
- Importez Matplotlib: `import matplotlib.pyplot as plt`.
- Importez la régression linéaire de Scikit-Learn: `from sklearn.linear_model import LinearRegression`.
- Utilisez la commande pour lire le fichier CSV:
  `df = pd.read_csv("csvExamples/randomStudentData.csv")`
- Extrait la variable indépendante (IQ) et la variable dépendante (Marks):
  - `X = df[['IQ']]`
  - `y = df['Marks']`

Régression Linéaire: Marks vs IQ (Suite)

- Créez un modèle de régression linéaire:
  `model = LinearRegression()`
- Adaptez le modèle aux données:
  `model.fit(X, y)`
- Faites des prédictions en utilisant le modèle:
  `y_pred = model.predict(X)`
- Plottez les points de données originaux:
  - `plt.scatter(X, y, color='blue', label='Original Data')`
- Plottez la ligne de régression:
  - `plt.plot(X, y_pred, color='red', linewidth=2, label='Linear Regression')`
- Set labels and title:
  - `plt.xlabel('IQ')`
  - `plt.ylabel('Marks')`
  - `plt.title('Linear Regression:  Marks vs IQ')`
- Show the legend:
  `plt.legend()`
- Display the plot: `plt.show()`

Explications (Activité 3)

- **Import Libraries:**
  - `pandas` est importé pour la manipulation et l'analyse des données.
  - `matplotlib.pyplot` est importé pour créer des visualisations.
  - `LinearRegression` est importé de `sklearn.linear_model` pour la régression linéaire.
- **Read CSV File:**
  - `pd.read_csv()` lit le fichier CSV dans un DataFrame pandas.
- **Prepare Data:**
  - Les variables indépendantes (IQ) et dépendantes (Marks) sont extraites du DataFrame.
- **Create Linear Regression Model:**
  - `LinearRegression()` crée un modèle de régression linéaire.
- **Fit Model to Data:**
  - `fit(X, y)` ajuste le modèle aux données d'entraînement.
- **Make Predictions:**
  - `predict(X)` fait des prédictions en utilisant le modèle.
- **Plot Original Data Points:**
  - `plt.scatter()` crée un nuage de points pour les données originales.

Explications (Activité 3 - Suite)

- **Plot Regression Line:**
  - `plt.plot()` trace la ligne de régression à partir des prédictions.
- **Set Labels and Title:**
  - `xlabel()`, `ylabel()`, et `title()` définissent les étiquettes et le titre du graphique.
- **Show Legend:**
  - `plt.legend()` affiche la légende sur le graphique.
- **Display the Plot:**
  - `plt.show()` affiche le graphique résultant.

Lectures recommandées

- Documentation Scikit-Learn: `https://scikit-learn.org/stable/`
- Documentation Pandas: `https://pandas.pydata.org`
- Documentation NumPy: `https://numpy.org`
- Documentation TensorFlow: `https://www.tensorflow.org`
- Teachable Machine avec Google: `https://teachablemachine.withgoogle.com`

Plateformes conseillées

- 360 Learning: `https://www.360learning.com`
- Lien avec exemples: `https://github.com/AliZainoul/DSEPSIBDX`