Université Sultan Moulay Slimane Faculté Polydisciplinaire – Beni Mellal

Filière : LICENCE D'EXCELLENCE

Spécialité : IOT-R

Module : Python

Rapport de
Mini _ Projet _ Pandas _ Python

Réalisé par :

MOHAMED AMRAOUI

Encadré par :

Pr. MOUNCIF

Année universitaire : 2024-2025

I- Introduction :

Ce mini-projet vise à analyser un ensemble de données sur la dépression chez les étudiants en utilisant des techniques d'analyse de données et d'apprentissage automatique. Le projet suit un pipeline complet, allant de l'importation des données à l'évaluation d'un modèle prédictif.

1- Importation des Données :

a- Points Positifs :

- Bibliothèques utilisées : Les bibliothèques essentielles (pandas, numpy, matplotlib, seaborn, scikit-learn) sont correctement importées.
- Gestion des erreurs : Un bloc try/except est utilisé pour gérer les erreurs potentielles lors du chargement du fichier.
- Configuration initiale : Les options d'affichage sont bien configurées pour afficher toutes les colonnes et utiliser le style ggplot pour les visualisations.

b- Améliorations Possibles :

- Chemin du fichier : Le chemin absolu (C:\Users\PC\Desktop...) rend le code non portable. Une solution serait d'utiliser un chemin relatif ou de permettre à l'utilisateur de spécifier le chemin via une interface.
- Structure du fichier : Aucune information n'est fournie sur la structure attendue du fichier CSV (colonnes, format, etc.), ce qui pourrait poser des problèmes de compatibilité.

2- Exploration Initiale des Données :

a- Points Positifs :

- Exploration complète : Les méthodes info(), describe(), et la vérification des valeurs manquantes sont utilisées pour une analyse initiale.
- Inspection visuelle : Les premières lignes du dataset sont affichées pour une vérification rapide.

b- Améliorations Possibles :

- Erreur de syntaxe : La commande describe(include=all) manque de guillemets autour de all.
- Commentaires : Aucune observation ou interprétation n'est fournie pour les résultats de l'exploration, ce qui limite la compréhension des données.

3- Nettoyage des Données :

a- Points Positifs :

- Suppression des doublons : Les doublons sont supprimés systématiquement.
- Gestion des valeurs manquantes : Les valeurs manquantes sont traitées de manière appropriée (mode pour les variables catégorielles, médiane pour les variables numériques).

b- Améliorations Possibles :

- Faute de frappe : La colonne "Mork Pressure" semble être une erreur pour "Work Pressure".
- Indentation : L'indentation dans le bloc de gestion des valeurs manquantes est incorrecte.
- Valeurs aberrantes : Aucune analyse des valeurs aberrantes (outliers) n'est réalisée.

4- Analyse Exploratoire des Données (EDA) :

a- **Points Positifs :**

- Visualisation de la variable cible : Une tentative de visualisation de la distribution de la variable cible est faite.
- Matrice de corrélation : Une matrice de corrélation est générée pour les variables numériques, ce qui est utile pour identifier les relations entre les caractéristiques.

b- **Améliorations Possibles :**

- Erreur de syntaxe : "fgsize" au lieu de "figsize" dans les appels à plt.figure.
- Incohérence des noms : Le code suppose une colonne depression_level, alors que le dataset montre une colonne "Depression".
- Variables catégorielles : Aucune visualisation n'est proposée pour les variables catégorielles, ce qui limite l'analyse.

5- Prétraitement pour l'Apprentissage Automatique :

a- **Points Positifs :**

- Encodage des variables catégorielles : LabelEncoder est utilisé pour convertir les variables catégorielles en valeurs numériques.
- Séparation train/test : Les données sont correctement divisées en ensembles d'entraînement et de test.
- Normalisation : Les caractéristiques sont normalisées à l'aide de StandardScaler.

b- **Améliorations Possibles :**

- Encodage : LabelEncoder n'est pas idéal pour les variables nominales (non ordonnées). OneHotEncoder serait plus approprié.
- Équilibre des classes : Aucune vérification de l'équilibre des classes n'est effectuée avant la modélisation.

6- Modélisation (Random Forest) :

a- **Points Positifs :**

- Choix du modèle : RandomForestClassifier est un choix adapté pour des données complexes.
- Évaluation complète : Le modèle est évalué à l'aide d'une matrice de confusion, d'un rapport de classification et de l'accuracy.
- Importance des caractéristiques : L'importance des caractéristiques est analysée et visualisée.

b- **Améliorations Possibles :**

- Erreur de syntaxe : La f-string dans print("\nPrécision globale : {accuracy_score(y_test, y_pred):.2f}") est incorrecte (accolades manquantes).
- Optimisation : Aucune optimisation des hyperparamètres n'est réalisée.
- Validation croisée : L'absence de validation croisée limite la robustesse des résultats.

7- Observations Générales :

a- **Problèmes Identifiés :**

- Syntaxe : Plusieurs erreurs mineures mais impactantes (fgsize, describe(include=all), f-string incorrecte).

- Incohérence des noms : La variable cible est tantôt appelée depression_level, tantôt "Depression".
- Portabilité : Le chemin absolu vers le fichier limite la réutilisabilité du code.
- Documentation : Aucun commentaire n'explique les choix méthodologiques ou les résultats intermédiaires.
- Profondeur d'analyse : L'EDA pourrait être enrichi par des analyses univariées/bivariées plus poussées.

II- Partie pratique :

1- Importation des données :

a- Points positifs :

- Bonne utilisation des bibliothèques essentielles (pandas, numpy, matplotlib, seaborn, scikit-learn)
- Gestion d'erreur appropriée avec try/except pour le chargement du fichier
- Configuration initiale utile (affichage complet des colonnes, style ggplot)

b- Améliorations possibles :

- Le chemin du fichier est absolu et spécifique à une machine (C:\Users\PC\Desktop...), ce qui rend le code non portable
- Aucune information sur la structure attendue du fichier CSV

 # Importation des bibliothèques nécessaires

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.preprocessing import StandardScaler
```

 # Configuration de l'affichage

```
pd.set_option('display.max_columns', None)
plt.style.use('ggplot')
```

 # 1. Chargement des données

```
try:
    # Essayez de lire le fichier (ajustez le chemin/nom si nécessaire)
    df = pd.read_csv(r"C:\Users\PC\Desktop\Licence
d'excellence\Python_\MiniProjetPython\student_depression_dataset.csv") # ou .xlsx pour Excel
    print("Dataset chargé avec succès. Voici les premières lignes :")
    print(df.head())
except FileNotFoundError:
    print("Fichier non trouvé. Veuillez vérifier le chemin ou le nom du fichier.")
    exit()
```

```
Chargement des données :
Dataset chargé avec succès. Voici les premières lignes :
   id  Gender   Age           City Profession  Academic Pressure  \
0   2    Male  33.0  Visakhapatnam    Student                5.0
1   8  Female  24.0      Bangalore    Student                2.0
2  26    Male  31.0       Srinagar    Student                3.0
3  30  Female  28.0       Varanasi    Student                3.0
4  32  Female  25.0         Jaipur    Student                4.0

   Work Pressure  CGPA  Study Satisfaction  Job Satisfaction  \
0            0.0  8.97                 2.0               0.0
1            0.0  5.90                 5.0               0.0
2            0.0  7.03                 5.0               0.0
3            0.0  5.59                 2.0               0.0
4            0.0  8.13                 3.0               0.0

        Sleep Duration Dietary Habits   Degree  \
0          '5-6 hours'        Healthy  B.Pharm
1          '5-6 hours'       Moderate      BSc
2  'Less than 5 hours'        Healthy       BA
3          '7-8 hours'       Moderate      BCA
4          '5-6 hours'       Moderate   M.Tech
```

```
        Sleep Duration Dietary Habits   Degree  \
0          '5-6 hours'        Healthy  B.Pharm
1          '5-6 hours'       Moderate      BSc
2  'Less than 5 hours'        Healthy       BA
3          '7-8 hours'       Moderate      BCA
4          '5-6 hours'       Moderate   M.Tech

  Have you ever had suicidal thoughts ?  Work/Study Hours Financial Stress  \
0                                   Yes               3.0              1.0
1                                    No               3.0              2.0
2                                    No               9.0              1.0
3                                   Yes               4.0              5.0
4                                   Yes               1.0              1.0

  Family History of Mental Illness  Depression
0                               No           1
1                              Yes           0
2                              Yes           0
3                              Yes           1
4                               No           0
```

2- *Exploration initiale des données*

a- Points positifs :

- Exploration complète avec info(), describe() et vérification des valeurs manquantes
- Affichage des premières lignes pour une inspection visuelle

b- Améliorations possibles :

- La commande describe(include=all) devrait être describe(include='all') (guillemets manquants)
- Aucun commentaire sur les observations tirées de cette exploration

```
print("\nInformations sur le dataset :")
print(df.info())

print("\nStatistiques descriptives :")
print(df.describe(include='all'))

print("\nValeurs manquantes par colonne :")
print(df.isnull().sum())
```

```
Exploration initiale des données ;

Informations sur le dataset :
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27901 entries, 0 to 27900
Data columns (total 18 columns):
 #   Column                                 Non-Null Count  Dtype
---  ------                                 --------------  -----
---  ------                                 --------------  -----
 0   id                                     27901 non-null  int64
 1   Gender                                 27901 non-null  object
 2   Age                                    27901 non-null  float64
 3   City                                   27901 non-null  object
 4   Profession                             27901 non-null  object
 5   Academic Pressure                      27901 non-null  float64
 6   Work Pressure                          27901 non-null  float64
 7   CGPA                                   27901 non-null  float64
 8   Study Satisfaction                     27901 non-null  float64
 9   Job Satisfaction                       27901 non-null  float64
 10  Sleep Duration                         27901 non-null  object
 11  Dietary Habits                         27901 non-null  object
 12  Degree                                 27901 non-null  object
 13  Have you ever had suicidal thoughts ?  27901 non-null  object
 14  Work/Study Hours                       27901 non-null  float64
 15  Financial Stress                       27901 non-null  object
 16  Family History of Mental Illness       27901 non-null  object
 17  Depression                             27901 non-null  int64
dtypes: float64(7), int64(2), object(9)
memory usage: 3.8+ MB
None
```

Statistiques descriptives :

	id	Gender	Age	City	Profession
count	27901.000000	27901	27901.000000	27901	27901
unique	NaN	2	NaN	52	14
top	NaN	Male	NaN	Kalyan	Student
freq	NaN	15547	NaN	1570	27870
mean	70442.149421	NaN	25.822300	NaN	NaN
std	40641.175216	NaN	4.905687	NaN	NaN
min	2.000000	NaN	18.000000	NaN	NaN
25%	35039.000000	NaN	21.000000	NaN	NaN
50%	70684.000000	NaN	25.000000	NaN	NaN
75%	105818.000000	NaN	30.000000	NaN	NaN
max	140699.000000	NaN	59.000000	NaN	NaN

	Academic Pressure	Work Pressure	CGPA	Study Satisfaction
count	27901.000000	27901.000000	27901.000000	27901.000000
unique	NaN	NaN	NaN	NaN
top	NaN	NaN	NaN	NaN
freq	NaN	NaN	NaN	NaN
mean	3.141214	0.000430	7.656104	2.943837
std	1.381465	0.043992	1.470707	1.361148
min	0.000000	0.000000	0.000000	0.000000
25%	2.000000	0.000000	6.290000	2.000000
50%	3.000000	0.000000	7.770000	3.000000
75%	4.000000	0.000000	8.920000	4.000000
max	5.000000	5.000000	10.000000	5.000000

	Job Satisfaction	Sleep Duration	Dietary Habits	Degree
count	27901.000000	27901	27901	27901
unique	NaN	5	4	28
top	NaN	'Less than 5 hours'	Unhealthy	'Class 12'
freq	NaN	8310	10317	6080
mean	0.000681	NaN	NaN	NaN
std	0.044394	NaN	NaN	NaN
min	0.000000	NaN	NaN	NaN
25%	0.000000	NaN	NaN	NaN
50%	0.000000	NaN	NaN	NaN
75%	0.000000	NaN	NaN	NaN
max	4.000000	NaN	NaN	NaN

	Have you ever had suicidal thoughts ?	Work/Study Hours	\
count	27901	27901.000000	
unique	2	NaN	
top	Yes	NaN	
freq	17656	NaN	
mean	NaN	7.156984	
std	NaN	3.707642	
min	NaN	0.000000	
25%	NaN	4.000000	
50%	NaN	8.000000	
75%	NaN	10.000000	
max	NaN	12.000000	

	Financial Stress	Family History of Mental Illness	Depression
count	27901	27901	27901.000000
unique	6	2	NaN
top	5.0	No	NaN
freq	6715	14398	NaN
mean	NaN	NaN	0.585499
std	NaN	NaN	0.492645
min	NaN	NaN	0.000000
25%	NaN	NaN	0.000000
50%	NaN	NaN	1.000000
75%	NaN	NaN	1.000000
max	NaN	NaN	1.000000

```
Valeurs manquantes par colonne :
id                                       0
Gender                                   0
Age                                      0
City                                     0
Profession                               0
Academic Pressure                        0
Work Pressure                            0
CGPA                                     0
Study Satisfaction                       0
Job Satisfaction                         0
Sleep Duration                           0
Dietary Habits                           0
Degree                                   0
Have you ever had suicidal thoughts ?    0
Work/Study Hours                         0
Financial Stress                         0
Family History of Mental Illness         0
Depression                               0
dtype: int64
```

3- *Nettoyage des données :*

a- Points positifs :

- Suppression des doublons systématique

- Bonne approche pour le traitement des valeurs manquantes (mode pour les catégorielles, médiane pour les numériques)

b- **Améliorations possibles :**

- La colonne "Mork Pressure" semble être une faute de frappe pour "Work Pressure" (visible dans les sorties)
- Indentation incorrecte dans le bloc de gestion des valeurs manquantes
- Aucune analyse des valeurs aberrantes (outliers)

```
# Suppression des doublons
df = df.drop_duplicates()

# Gestion des valeurs manquantes (à adapter selon votre dataset)
for col in df.columns:
    if df[col].dtype == 'object':
        df[col].fillna(df[col].mode()[0], inplace=True)
    else:
        df[col].fillna(df[col].median(), inplace=True)

# Vérification après nettoyage
print("\nValeurs manquantes après nettoyage :")
print(df.isnull().sum())
```

```
Nettoyage des données :
Suppression des doublons :
Gestion des valeurs manquantes (à adapter selon votre dataset) :
Vérification après nettoyage :

Valeurs manquantes après nettoyage :
id                                       0
Gender                                   0
Age                                      0
City                                     0
Profession                               0
Academic Pressure                        0
Work Pressure                            0
CGPA                                     0
Study Satisfaction                       0
Job Satisfaction                         0
Sleep Duration                           0
Dietary Habits                           0
Degree                                   0
Have you ever had suicidal thoughts ?    0
Work/Study Hours                         0
Financial Stress                         0
Family History of Mental Illness         0
Depression                               0
dtype: int64
```

4- *Analyse exploratoire des données (EDA) :*

a- Points positifs :

- Tentative de visualisation de la variable cible
- Matrice de corrélation utile pour les variables numériques

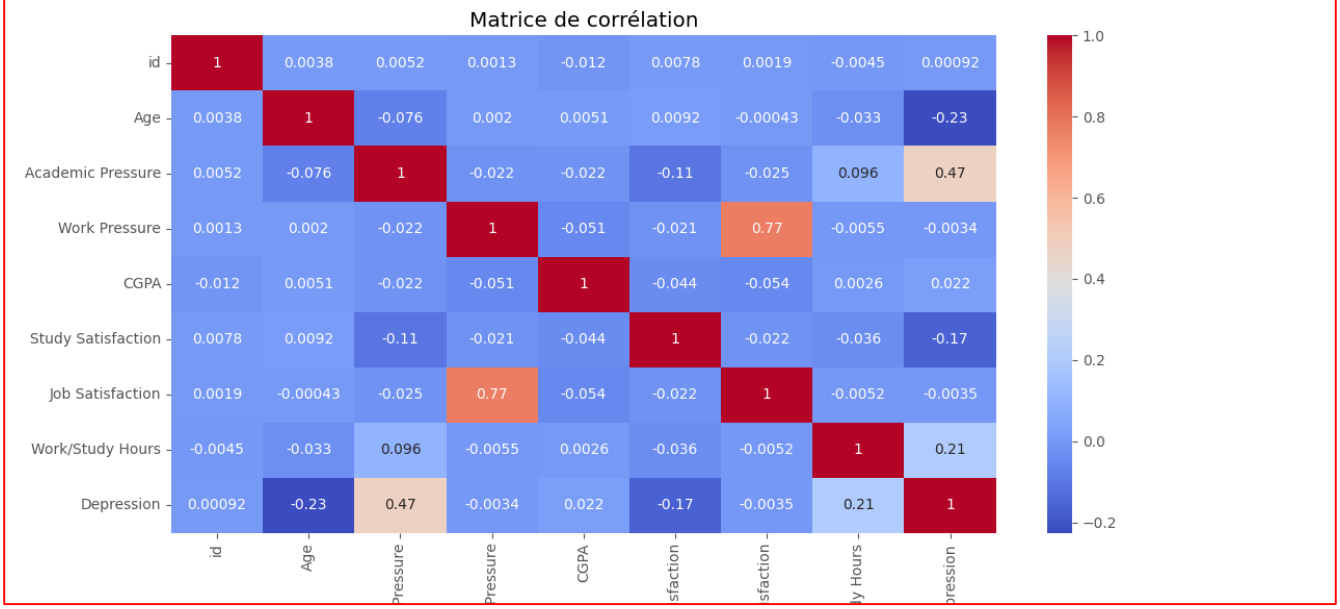
b- Améliorations possibles :

- "fgsize" au lieu de "figsize" dans le premier plot (erreur de syntaxe)
- Le code suppose une colonne 'depression_level' qui n'apparaît pas dans les données affichées (la colonne semble s'appeler "Depression")
- Aucune visualisation pour les variables catégorielles

```
# Distribution de la variable cible (supposons qu'elle s'appelle 'depression_level')
if 'depression_level' in df.columns:
    plt.figure(figsize=(8, 6))
    sns.countplot(x='depression_level', data=df)
    plt.title('Distribution des niveaux de dépression')
    plt.show()
```

```
Analyse exploratoire des données (EDA) :
Distribution de la variable cible (supposons qu'elle s'appelle 'depression_level') :
```

```
# Corrélations entre variables numériques
numeric_cols = df.select_dtypes(include=[np.number]).columns
if len(numeric_cols) > 0:
    plt.figure(figsize=(12, 8))
    sns.heatmap(df[numeric_cols].corr(), annot=True, cmap='coolwarm')
    plt.title('Matrice de corrélation')
    plt.show()
```



5- *Prétraitement pour l'apprentissage automatique :*

a- Points positifs :

- Encodage correct des variables catégorielles avec LabelEncoder
- Bonne pratique de séparation train/test et normalisation

b- **Améliorations possibles :**

- L'encodage LabelEncoder n'est pas toujours idéal pour les variables catégorielles nominales (OneHotEncoder serait parfois préférable)
- Aucune vérification de l'équilibre des classes avant la modélisation

```
# Encodage des variables catégorielles
le = LabelEncoder()
categorical_cols = df.select_dtypes(include=['object']).columns
for col in categorical_cols:
    df[col] = le.fit_transform(df[col])
```

```
Prétraitement pour l'apprentissage automatique :
Encodage des variables catégorielles :
```

```
# Séparation des caractéristiques et de la cible
# Supposons que 'depression_level' est la variable cible
if 'depression_level' in df.columns:
    X = df.drop('depression_level', axis=1)
    y = df['depression_level']

    # Division en ensembles d'entraînement et de test
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

    # Normalisation des données
    scaler = StandardScaler()
    X_train = scaler.fit_transform(X_train)
    X_test = scaler.transform(X_test)
```

```
Séparation des caractéristiques et de la cible :
Supposons que 'depression_level' est la variable cible :
```

6- *Modélisation (exemple avec Random Forest) :*

a- **Points positifs :**

- Choix approprié de RandomForest pour des données potentiellement complexes
- Évaluation complète avec matrice de confusion, rapport de classification et précision
- Analyse d'importance des caractéristiques utile

b- **Améliorations possibles :**

- Erreur de syntaxe dans le print de la précision globale (accolades manquantes pour f-string)
- Aucune optimisation des hyperparamètres du modèle
- Pas de validation croisée
- Le modèle est appliqué sans savoir si 'depression_level' existe vraiment (le dataset montre une colonne "Depression" binaire)

```
    model = RandomForestClassifier(n_estimators=100, random_state=42)
    model.fit(X_train, y_train)
```

```
    # Prédictions
    y_pred = model.predict(X_test)

    # 7. Évaluation du modèle
    print("\nRapport de classification :")
    print(classification_report(y_test, y_pred))

    print("\nMatrice de confusion :")
    print(confusion_matrix(y_test, y_pred))

    print(f"\nPrécision globale : {accuracy_score(y_test, y_pred):.2f}")

    # Importance des caractéristiques
    feature_importance = pd.DataFrame({
        'Feature': X.columns,
        'Importance': model.feature_importances_
    }).sort_values('Importance', ascending=False)

    plt.figure(figsize=(10, 6))
    sns.barplot(x='Importance', y='Feature', data=feature_importance)
    plt.title('Importance des caractéristiques')
    plt.show()
else:
    print("\nLa colonne 'depression_level' n'a pas été trouvée dans le dataset.")
```

7- Observations générales :

Problèmes de syntaxe : Plusieurs erreurs mineures mais impactantes (fgsize, describe(include=all), f-string incorrect)

Incohérence des noms : La variable cible est parfois appelée 'depression_level' alors que les données montrent "Depression"

Manque de portabilité : Chemin d'accès absolu au fichier

Documentation absente : Aucun commentaire expliquant les choix méthodologiques

Profondeur d'analyse : L'EDA pourrait être plus complet (analyse univariée/bivariée plus poussée)

8- Recommandations :

- Corriger les erreurs de syntaxe : Utiliser figsize au lieu de fgsize, ajouter les guillemets manquants, et corriger les f-strings.
- Standardiser les noms de colonnes : S'assurer que le nom de la variable cible est cohérent dans tout le code.
- Rendre le code portable : Utiliser des chemins relatifs ou permettre à l'utilisateur de spécifier le chemin du fichier.
- Améliorer la documentation : Ajouter des commentaires pour expliquer les étapes clés et les résultats.
- Approfondir l'analyse : Explorer davantage les variables catégorielles et les valeurs aberrantes, et implémenter une validation croisée.

Conclusion :

Ce mini-projet sur l'analyse des données de dépression chez les étudiants démontre une compréhension globale des étapes clés du processus d'analyse de données et de modélisation. Le pipeline complet, allant de l'importation des données à l'évaluation d'un modèle prédictif, a été suivi avec une approche méthodique. Les points forts incluent :

Utilisation appropriée des outils *:* Les bibliothèques essentielles (pandas, numpy, matplotlib, seaborn, scikit-learn) ont été correctement employées pour l'analyse et la visualisation des données.

Gestion des erreurs *:* La mise en place d'un bloc try/except pour le chargement des données montre une bonne pratique de programmation.

Analyse multidimensionnelle : L'exploration des données, le nettoyage, et l'analyse exploratoire (EDA) ont été abordés de manière structurée.

Modélisation pertinente : Le choix de RandomForest comme algorithme de classification était adapté

Finalement, ce projet constitue une base solide pour l'analyse de données en Python, mais son potentiel pourrait être pleinement exploité en adressant les points faibles identifiés. Les compétences acquises ici sont transférables à de nombreux autres projets d'analyse de données et d'apprentissage automatique.

Annexe 1 :

	A	B	C	D	E	F	G	H
1	id,Gender,Age,City,Profession,Academic Pressure,Work Pressure,CGPA,Study Satisfaction,Job Satisfaction,Slee							
2	2,Male,33.0,Visakhapatnam,Student,5.0,0.0,8.97,2.0,0.0,'5-6 hours',Healthy,B.Pharm,Yes,3.0,1.0,No,1							
3	8,Female,24.0,Bangalore,Student,2.0,0.0,5.9,5.0,0.0,'5-6 hours',Moderate,BSc,No,3.0,2.0,Yes,0							
4	26,Male,31.0,Srinagar,Student,3.0,0.0,7.03,5.0,0.0,'Less than 5 hours',Healthy,BA,No,9.0,1.0,Yes,0							
5	30,Female,28.0,Varanasi,Student,3.0,0.0,5.59,2.0,0.0,'7-8 hours',Moderate,BCA,Yes,4.0,5.0,Yes,1							
6	32,Female,25.0,Jaipur,Student,4.0,0.0,8.13,3.0,0.0,'5-6 hours',Moderate,M.Tech,Yes,1.0,1.0,No,0							
7	33,Male,29.0,Pune,Student,2.0,0.0,5.7,3.0,0.0,'Less than 5 hours',Healthy,PhD,No,4.0,1.0,No,0							
8	52,Male,30.0,Thane,Student,3.0,0.0,9.54,4.0,0.0,'7-8 hours',Healthy,BSc,No,1.0,2.0,No,0							
9	56,Female,30.0,Chennai,Student,2.0,0.0,8.04,4.0,0.0,'Less than 5 hours',Unhealthy,'Class 12',No,0.0,1.0,Yes,0							
10	59,Male,28.0,Nagpur,Student,3.0,0.0,9.79,1.0,0.0,'7-8 hours',Moderate,B.Ed,Yes,12.0,3.0,No,1							
11	62,Male,31.0,Nashik,Student,2.0,0.0,8.38,3.0,0.0,'Less than 5 hours',Moderate,LLB,Yes,2.0,5.0,No,1							
12	83,Male,24.0,Nagpur,Student,3.0,0.0,6.1,3.0,0.0,'5-6 hours',Moderate,'Class 12',Yes,11.0,1.0,Yes,1							
13	91,Male,33.0,Vadodara,Student,3.0,0.0,7.03,4.0,0.0,'Less than 5 hours',Healthy,BE,Yes,10.0,2.0,Yes,0							
14	94,Male,27.0,Kalyan,Student,5.0,0.0,7.04,1.0,0.0,'Less than 5 hours',Moderate,M.Tech,No,10.0,1.0,Yes,1							
15	100,Female,19.0,Rajkot,Student,2.0,0.0,8.52,4.0,0.0,'Less than 5 hours',Unhealthy,'Class 12',No,6.0,2.0,Yes,0							
16	103,Female,19.0,Kalyan,Student,5.0,0.0,5.64,5.0,0.0,'Less than 5 hours',Moderate,'Class 12',Yes,4.0,5.0,Yes,1							
17	106,Male,29.0,Srinagar,Student,3.0,0.0,8.58,3.0,0.0,'More than 8 hours',Moderate,M.Tech,Yes,10.0,2.0,Yes,1							
18	120,Male,25.0,Nashik,Student,5.0,0.0,6.51,2.0,0.0,'Less than 5 hours',Unhealthy,M.Ed,Yes,2.0,5.0,Yes,1							
19	132,Female,20.0,Ahmedabad,Student,5.0,0.0,7.25,3.0,0.0,'5-6 hours',Healthy,'Class 12',Yes,10.0,3.0,No,1							
20	139,Male,19.0,Chennai,Student,2.0,0.0,7.83,2.0,0.0,'7-8 hours',Unhealthy,'Class 12',No,6.0,3.0,No,0							
21	145,Male,25.0,Kalyan,Student,3.0,0.0,9.93,3.0,0.0,'5-6 hours',Moderate,B.Ed,No,8.0,3.0,Yes,1							
22	161,Male,29.0,Kolkata,Student,3.0,0.0,8.74,4.0,0.0,'5-6 hours',Moderate,B.Ed,Yes,1.0,1.0,No,0							

Annexe 2 :

```
# Importation des bibliothèques nécessaires
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.preprocessing import StandardScaler

# Configuration de l'affichage
pd.set_option('display.max_columns', None)
plt.style.use('ggplot')

# 1. Chargement des données
try:
    # Essayez de lire le fichier (ajustez le chemin/nom si nécessaire)
```

```
    df = pd.read_csv(r"C:\Users\PC\Desktop\Licence
d'excellence\Python_\MiniProjetPython\student_depression_dataset.csv") # ou .xlsx
pour Excel
    print("Dataset chargé avec succès. Voici les premières lignes :")
    print(df.head())
except FileNotFoundError:
    print("Fichier non trouvé. Veuillez vérifier le chemin ou le nom du fichier.")
    exit()

# 2. Exploration initiale des données
print("\nInformations sur le dataset :")
print(df.info())

print("\nStatistiques descriptives :")
print(df.describe(include='all'))

print("\nValeurs manquantes par colonne :")
print(df.isnull().sum())

# 3. Nettoyage des données
# Suppression des doublons
df = df.drop_duplicates()

# Gestion des valeurs manquantes (à adapter selon votre dataset)
for col in df.columns:
    if df[col].dtype == 'object':
        df[col].fillna(df[col].mode()[0], inplace=True)
    else:
        df[col].fillna(df[col].median(), inplace=True)

# Vérification après nettoyage
print("\nValeurs manquantes après nettoyage :")
print(df.isnull().sum())

# 4. Analyse exploratoire des données (EDA)
# Distribution de la variable cible (supposons qu'elle s'appelle 'depression_level')
if 'depression_level' in df.columns:
    plt.figure(figsize=(8, 6))
    sns.countplot(x='depression_level', data=df)
    plt.title('Distribution des niveaux de dépression')
    plt.show()
```

```
# Corrélations entre variables numériques
numeric_cols = df.select_dtypes(include=[np.number]).columns
if len(numeric_cols) > 0:
    plt.figure(figsize=(12, 8))
    sns.heatmap(df[numeric_cols].corr(), annot=True, cmap='coolwarm')
    plt.title('Matrice de corrélation')
    plt.show()

# 5. Prétraitement pour l'apprentissage automatique
# Encodage des variables catégorielles
le = LabelEncoder()
categorical_cols = df.select_dtypes(include=['object']).columns
for col in categorical_cols:
    df[col] = le.fit_transform(df[col])

# Séparation des caractéristiques et de la cible
# Supposons que 'depression_level' est la variable cible
if 'depression_level' in df.columns:
    X = df.drop('depression_level', axis=1)
    y = df['depression_level']

    # Division en ensembles d'entraînement et de test
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

    # Normalisation des données
    scaler = StandardScaler()
    X_train = scaler.fit_transform(X_train)
    X_test = scaler.transform(X_test)

    # 6. Modélisation (exemple avec Random Forest)
    model = RandomForestClassifier(n_estimators=100, random_state=42)
    model.fit(X_train, y_train)

    # Prédictions
    y_pred = model.predict(X_test)

    # 7. Évaluation du modèle
    print("\nRapport de classification :")
    print(classification_report(y_test, y_pred))

    print("\nMatrice de confusion :")
```

```
    print(confusion_matrix(y_test, y_pred))

    print(f"\nPrécision globale : {accuracy_score(y_test, y_pred):.2f}")

    # Importance des caractéristiques
    feature_importance = pd.DataFrame({
        'Feature': X.columns,
        'Importance': model.feature_importances_
    }).sort_values('Importance', ascending=False)

    plt.figure(figsize=(10, 6))
    sns.barplot(x='Importance', y='Feature', data=feature_importance)
    plt.title('Importance des caractéristiques')
    plt.show()
else:
    print("\nLa colonne 'depression_level' n'a pas été trouvée dans le dataset.")
```