МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Систем обработки информации и управления»

ОТЧЕТ

**Лабораторная работа №\_\_2\_\_**

по дисциплине«Методы машинного обучения»

Тема: «Обработка признаков (часть 1)»

ИСПОЛНИТЕЛЬ: Лу Сяои

ФИО

группа ИУ5И-22М \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись

"1" Июнь 2023 г.

ПРЕПОДАВАТЕЛЬ: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ФИО

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись

"\_\_"\_\_\_\_\_\_\_\_\_2023 г.

Москва - 2023

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## **описание задания**

1. Выбрать набор данных (датасет), содержащий категориальные и числовые признаки и пропуски в данных. Для выполнения следующих пунктов можно использовать несколько различных наборов данных (один для обработки пропусков, другой для категориальных признаков и т.д.) Просьба не использовать датасет, на котором данная задача решалась в лекции.
2. Для выбранного датасета (датасетов) на основе материалов лекций решить следующие задачи:
3. устранение пропусков в данных;
4. кодирование категориальных признаков;
5. нормализация числовых признаков.

**текст программы и экранные формы с примерами выполнения**

Цель лабораторной работы: изучение продвинутых способов предварительной обработки данных для дальнейшего формирования моделей.

# подключение библиотек

import numpy as np # linear algebra

import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)

import random

import math as math

import seaborn as sns #

import matplotlib.pyplot as plt

import missingno as msno

import plotly.graph\_objs as go

import plotly.express as px #

plt.style.use('seaborn-dark')

plt.style.context('grayscale')

%matplotlib inline

import re

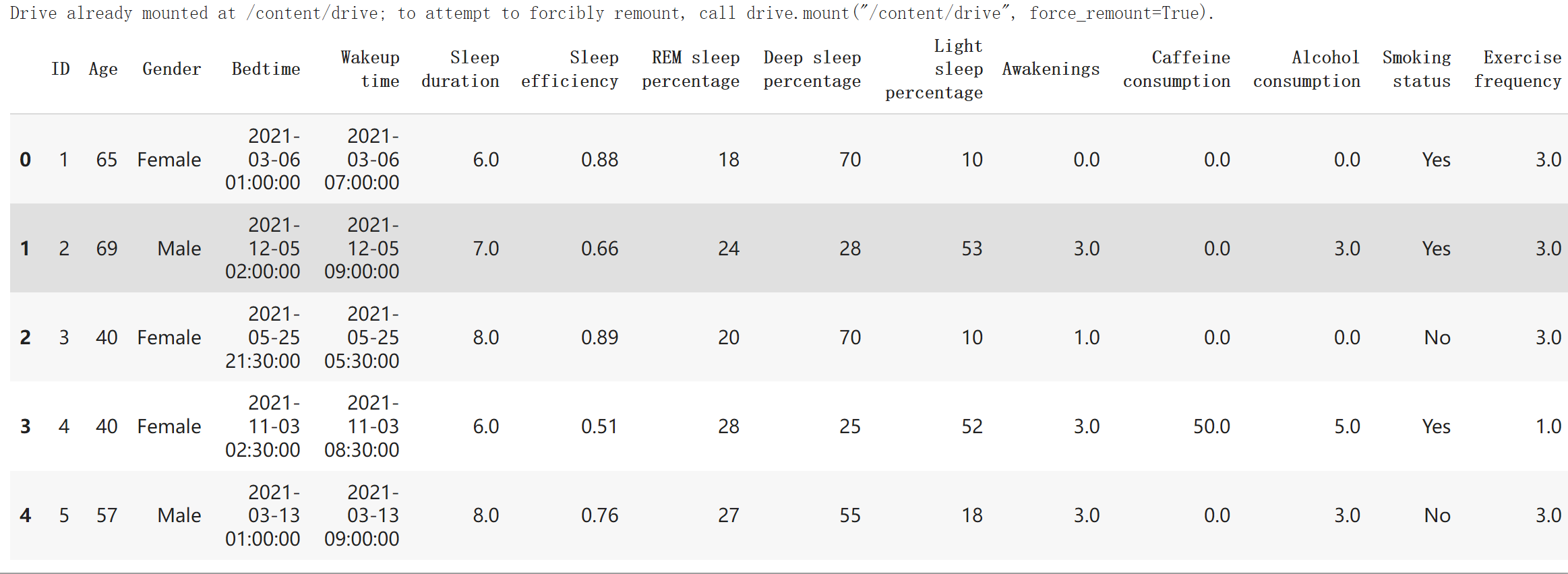
from wordcloud import WordCloud, STOPWORDS

from google.colab import drive

drive.mount('/content/drive')

df = pd.read\_csv('/content/drive/My Drive/Sleep\_Efficiency.csv')

df.head()



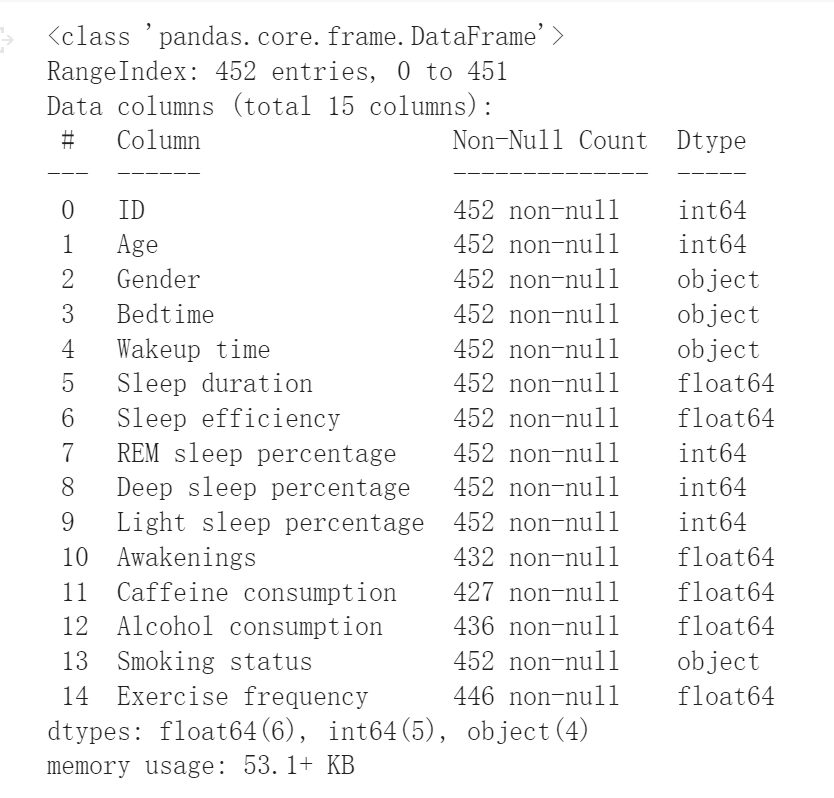
The "Sleep efficiency" feature is a measure of the proportion of time spent in bed that is actually spent asleep. Additionally, the dataset includes information about each subject's caffeine and alcohol consumption in the 24 hours prior to bedtime, their smoking status, and their exercise frequency.

Caffeine consumption:the amount of caffeine consumed in the 24 hours prior to bedtime (in mg)

Alcohol consumption:the amount of alcohol consumed in the 24 hours prior to bedtime (in oz)

Exercise frequency:the number of times the test subject exercises each week

df.info()



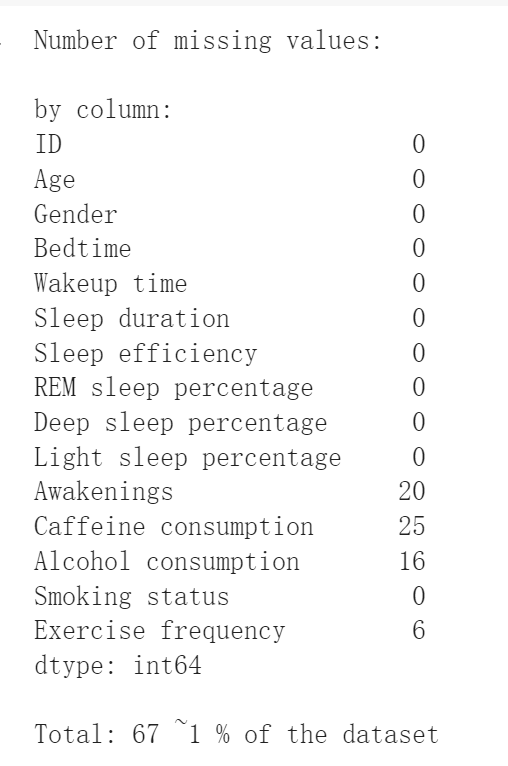
# shape of the data  
df.shape



# 1.устранение пропусков в данных;

check for missing values

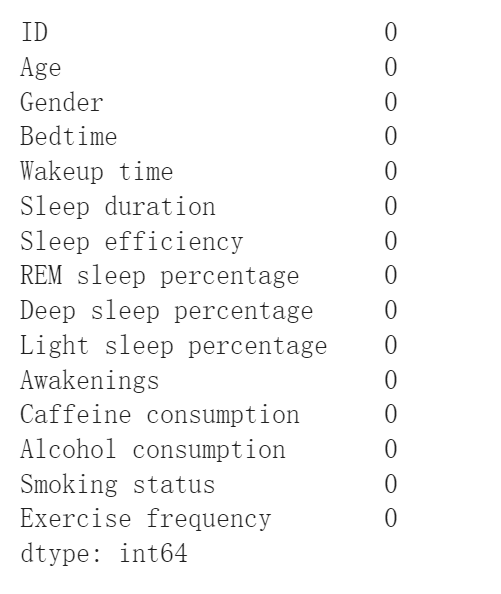
# number of missing values for each column  
print("Number of missing values:\n")  
print("by column:")  
nan\_val\_count = df.isnull().sum()  
print(nan\_val\_count, "\n")  
  
som = nan\_val\_count.sum()  
print("Total:", som, end= " ~")  
print(round(som / (df.shape[0] \* df.shape[1]) \* 100), "% of the dataset")



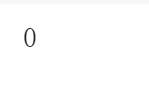
replacing missing values with mean

# replace null values with mean  
df['Awakenings'].fillna(df['Awakenings'].mean(), inplace=True)  
df['Caffeine consumption'].fillna(df['Caffeine consumption'].mean(), inplace=True)  
df['Alcohol consumption'].fillna(df['Alcohol consumption'].mean(), inplace=True)  
df['Exercise frequency'].fillna(df['Exercise frequency'].mean(), inplace=True)

# checking for null values  
df.isnull().sum()



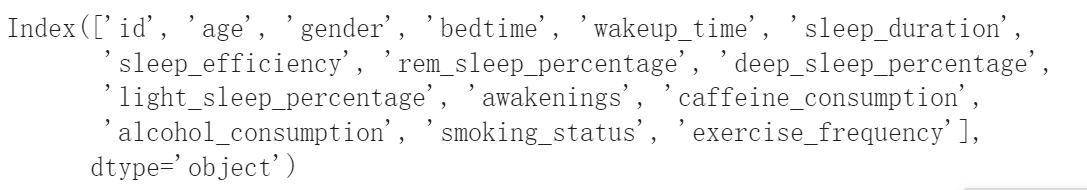
# cheching for duplicates  
df.duplicated().sum()



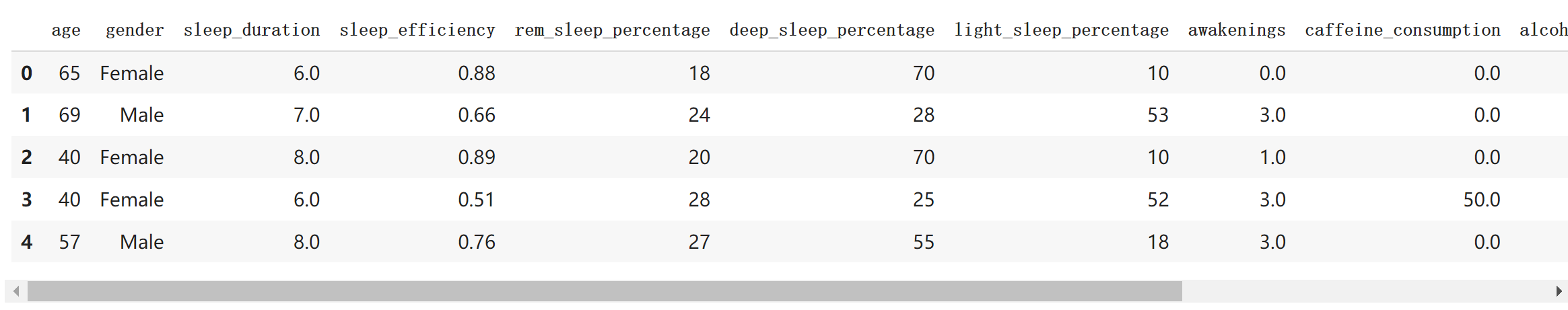
# 2.кодирование категориальных признаков;

We can make the data a little more organized by clearing the spaces in the column names and converting them all to lowercase.

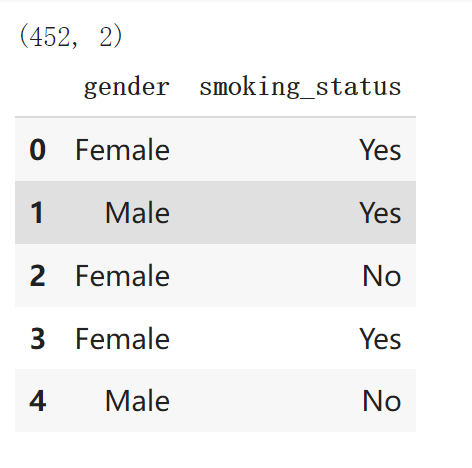
df.columns = [col.replace(' ', '\_').lower() for col in df.columns]  
print(df.columns)



# drop the ID column  
df = df.drop(['id', 'bedtime', 'wakeup\_time'], axis='columns')  
df.head()



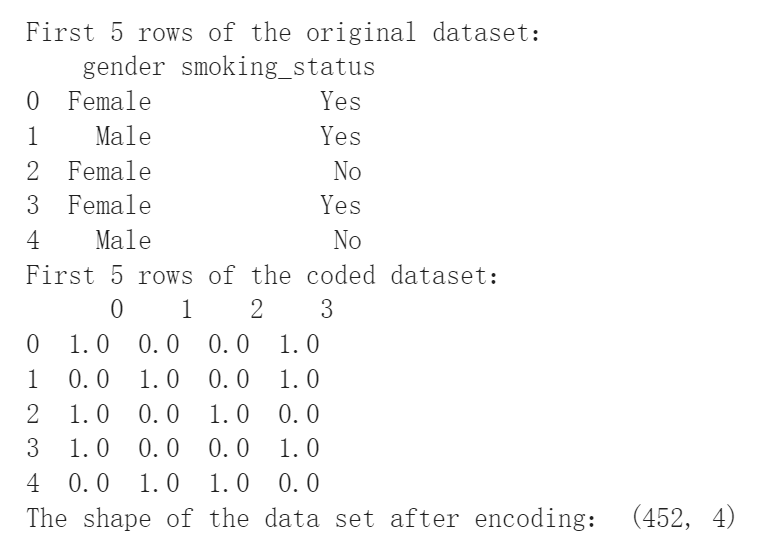
# split the data into features and target  
# 从 DataFrame 对象中提取特征矩阵 X  
X\_cate = df.loc[:, ['gender', 'smoking\_status']]  
  
# 查看特征矩阵的维度大小  
print(X\_cate.shape)  
  
X\_cate.head()



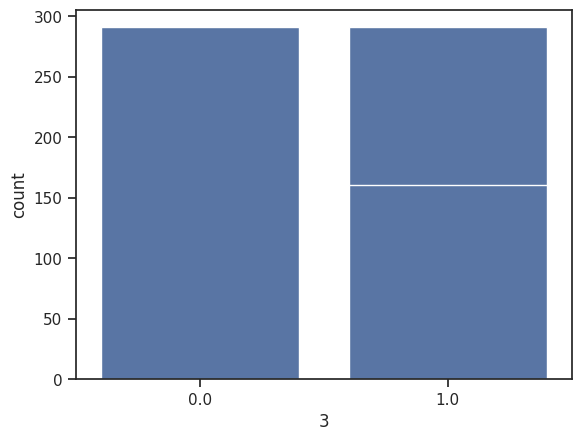
# Instantiate the OneHotEncoder class and encode the data

from sklearn.preprocessing import OneHotEncoder  
  
# 实例化 OneHotEncoder 类并对数据进行编码  
encoder = OneHotEncoder()  
X\_cate\_encoded = encoder.fit\_transform(X\_cate)

# 将编码后的数据集转换成 Pandas DataFrame 对象  
X\_cate\_encoded\_df = pd.DataFrame(X\_cate\_encoded.toarray())  
  
# 查看原数据集的前 5 行  
print('First 5 rows of the original dataset：\n', X\_cate.head())  
  
# 查看编码后的数据集的前 5 行  
print('First 5 rows of the coded dataset：\n', X\_cate\_encoded\_df.head())  
  
# 查看编码后的数据集的形状  
print('The shape of the data set after encoding：', X\_cate\_encoded.shape)

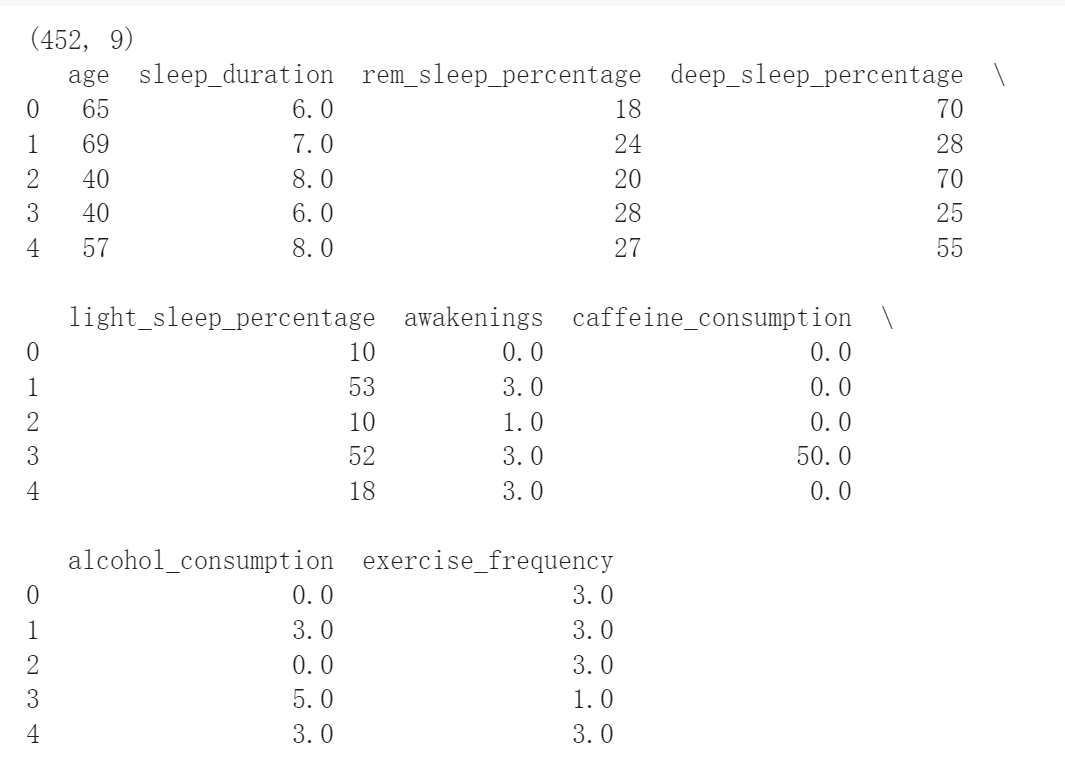


import seaborn as sns  
  
# 绘制编码后的数据集中，每个二进制虚拟变量的计数情况  
sns.set(style="ticks")  
sns.color\_palette("bright")  
for feature in range(X\_cate\_encoded\_df.shape[1]):  
    sns.countplot(x=X\_cate\_encoded\_df.iloc[:, feature], color='b')



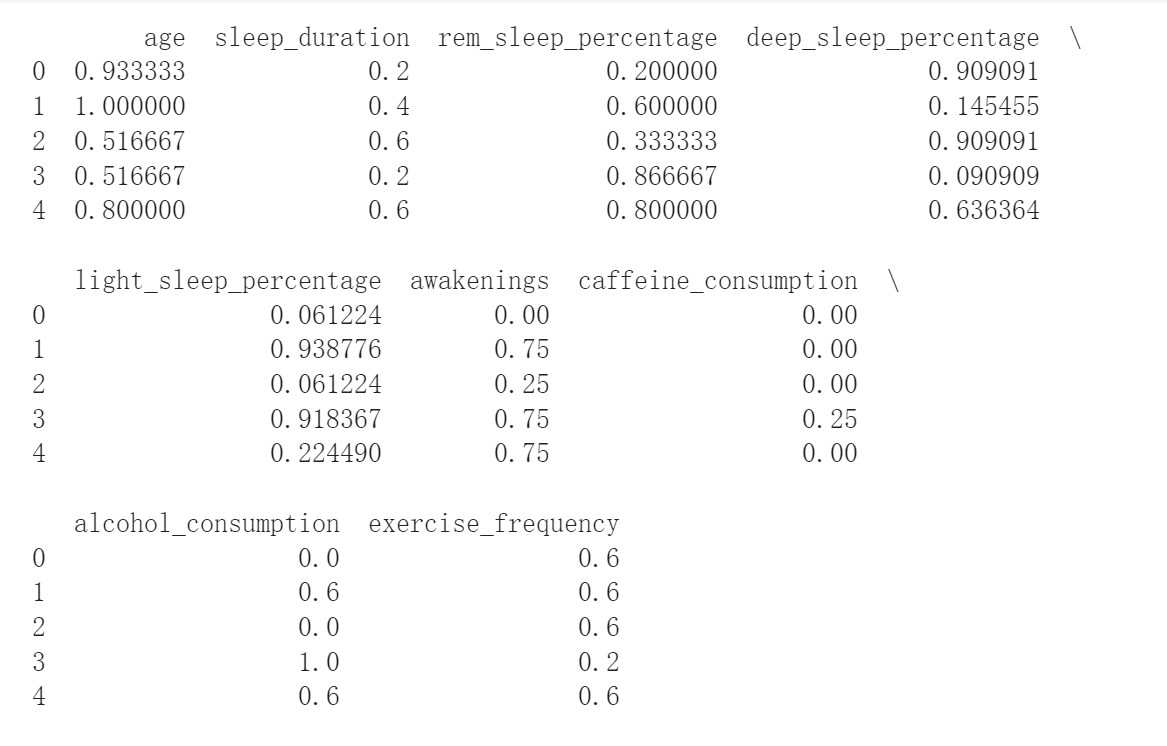
# 3.нормализация числовых признаков.

X\_num = df.loc[:, ['age','sleep\_duration','rem\_sleep\_percentage', 'deep\_sleep\_percentage',  
       'light\_sleep\_percentage', 'awakenings', 'caffeine\_consumption',  
       'alcohol\_consumption','exercise\_frequency']]  
print(X\_num.shape)  
print(X\_num.head())



from sklearn.preprocessing import MinMaxScaler  
scaler = MinMaxScaler()  
X\_num[['age','sleep\_duration','rem\_sleep\_percentage', 'deep\_sleep\_percentage',  
       'light\_sleep\_percentage', 'awakenings', 'caffeine\_consumption',  
       'alcohol\_consumption','exercise\_frequency']] = scaler.fit\_transform(X\_num[['age','sleep\_duration','rem\_sleep\_percentage', 'deep\_sleep\_percentage',  
       'light\_sleep\_percentage', 'awakenings', 'caffeine\_consumption',  
       'alcohol\_consumption','exercise\_frequency']])

print(X\_num.head())



import matplotlib.pyplot as plt  
  
# 查看规范化后的某一列数据的分布情况  
plt.hist(X\_num['age'], bins=20)  
plt.show()  
  
# 查看规范化后的两列数据的关系情况  
plt.scatter(X\_num['age'], X\_num['deep\_sleep\_percentage'])  
plt.xlabel('age')  
plt.ylabel('deep\_sleep\_percentage')  
plt.show()

