Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра программного обеспечения информационных технологий
Дисциплина: Методы машинного обучения

ОТЧЁТ
к лабораторной работе №2
на тему

EDA

Студент Маталыга Е.А.

Преподаватель Воронина А.М.

Минск 2025

1. Задания

1) Импортировать датасет (Исходные данные можно выбрать самостоятелно на kaggle.com, либо взять один из файлов, прикрепленных к заданию)

2) Проверить данные на пропуски и наличие аномальных значений, использовать для этого статистические методы и средства графической визуализации(графики распределения, диаграммы, графики типа BoxPlot, тепловые карты и т.п.). Заполнить пропуски и устранить выбросы. Проверить, при помощи графических средств, что пропуски и аномальные значения отсутствуют.

3) Сгенерировать новые признаки (не менее двух), а так же применить кодирование категориальных признаков (применить OHE-кодирование, бинаризацию признаков и др.)

4) Выполнить стандартизацию и/или нормализацию данных.

2. Выполнение

Ссылка на полный код.

2.1 Используемые библиотеки

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from scipy.stats import mode
```

2.2 Описание датасета

Датасет содержит данные о количестве принимаемого кофеина в день и некоторых показателей здоровья разных демографических групп.

- `ID` - Unique record ID (1–10000)
- `Age` - Age of participant (18–80 years)
- `Gender` - Male, Female
- `Country` - Country of residence (20 countries)
- `Coffee_Intake` - Daily coffee consumption in cups (0–10)
- `Caffeine_mg` - Estimated daily caffeine intake in mg (1 cup ≈ 95 mg)
- `Sleep_Hours` - Average hours of sleep per night (3–10 hours)
- `Sleep_Quality` - Poor, Fair, Good, Excellent (based on sleep hours)
- `BMI` - Body Mass Index (15–40)
- `Heart_Rate` - Resting heart rate (50–110 bpm)
- `Stress_Level` - Low, Medium, High (based on sleep hours and lifestyle)
- `Physical_Activity_Hours` - Weekly physical activity (0–15 hours)

- `Health_Issues` - None, Mild, Moderate, Severe (based on age, BMI, and sleep)
- `Occupation` - Office, Healthcare, Student, Service, Other
- `Smoking` - 0 = No, 1 = Yes
- `Alcohol_Consumption` - 0 = No, 1 = Yes

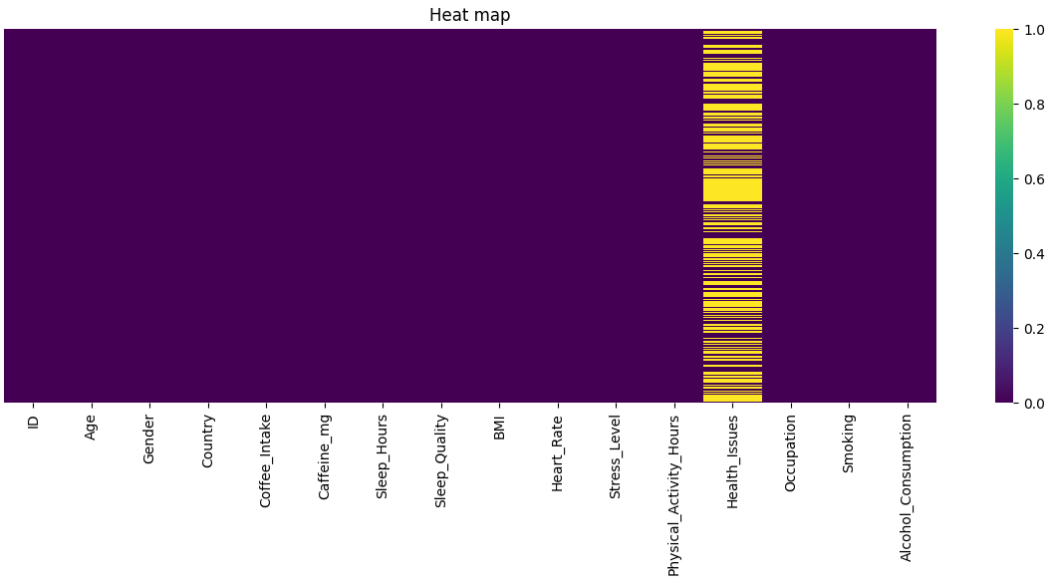
2.3 Поиск и обработка пропусков

Поиск пропусков:

- Таблица: `missing_stats = df.isnull().sum()`
- Тепловая карта: `sns.heatmap(df.isnull(), cbar=True, yticklabels=False, cmap='viridis')`

Результаты:

ID	0
Age	0
Gender	0
Country	0
Coffee_Intake	0
Caffeine_mg	0
Sleep_Hours	0
Sleep_Quality	0
BMI	0
Heart_Rate	0
Stress_Level	0
Physical_Activity_Hours	0
Health_Issues	5941
Occupation	0
Smoking	0
Alcohol_Consumption	0

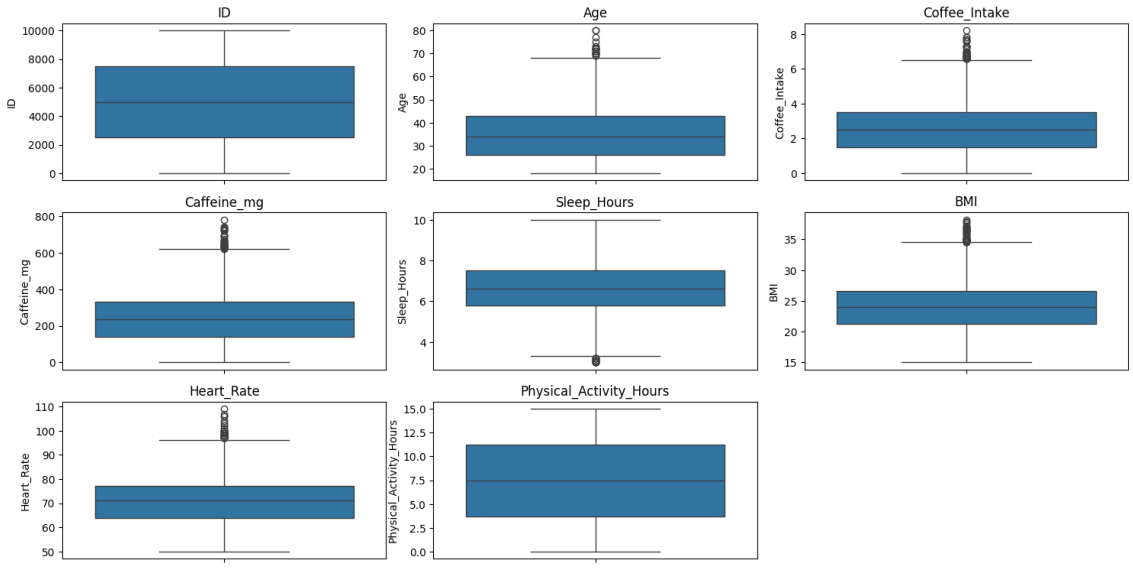


Поскольку пропусков в признаке “Health_Issues” больше, чем известных значений, то убрать все строки с пропусками или закодировать их средним не получусится, потому что сильно повлияет на результат. Поэтому обработка пропусков в Health_Issues осуществлялась при помощи библиотечного алгоритма случайного леса.

2.4 Поиск и обработка выбросов в количественных данных

Графически пропуски отрисованы при помощи BoxPlot.

```
plt.figure(figsize = (15, 10))
for i, col in enumerate(numeric_cols, 1):
  plt.subplot(4,3,i)
  sns.boxplot(y = df[col])
  plt.title(f"{col}")
plt.tight_layout()
plt.show()
```



Также процент выбросов для каждого признака вычислен при помощи формул BoxPlot.

```
def analyze_outliers_percentage(df, numeric_columns):
    outlier_analysis = {}

    for col in numeric_columns:
        Q1 = df[col].quantile(0.25)
        Q3 = df[col].quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR

        outliers = df[(df[col] < lower_bound) | (df[col] >
upper_bound)]
        outlier_count = len(outliers)
        outlier_percentage = (outlier_count / len(df)) * 100

        outlier_analysis[col] = {
            'count': outlier_count,
            'percentage': outlier_percentage,
            'bounds': [lower_bound, upper_bound]
        }

    return outlier_analysis
```

Результаты процентного подсчёта выбросов:

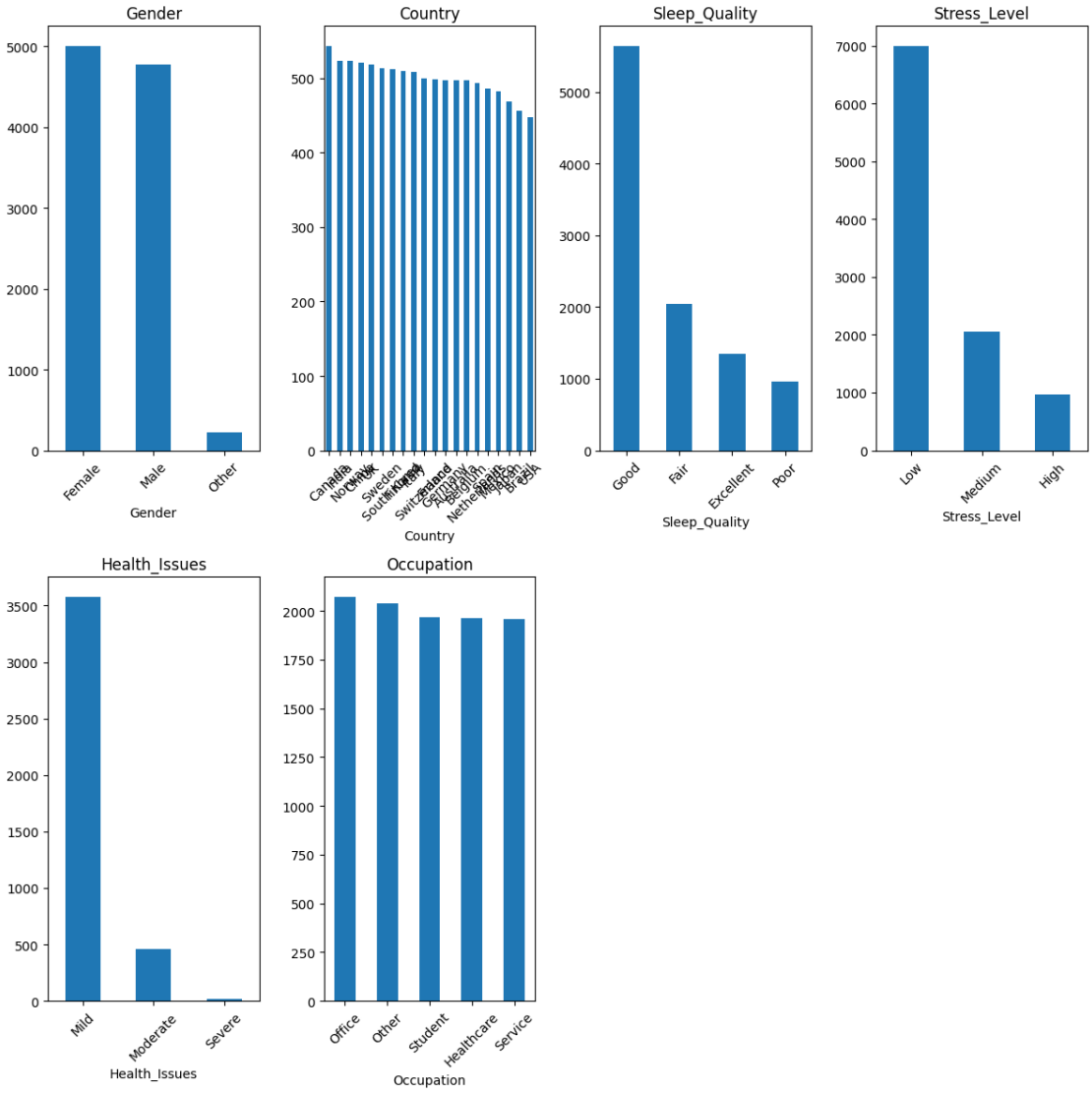
Column outlier anaysis:

Column	Outlier	%
ID	0	0.0
Age	25	0.2
Coffee_Intake	39	0.4
Caffeine_mg	39	0.4
Sleep_Hours	26	0.3
BMI	40	0.4
Heart_Rate	50	0.5
Physical_Activity_Hours	0	0.0

Поскольку параметры 'Age', 'Sleep_Hours', 'BMI', 'Heart_Rate', 'Physical_Activity_Hours' лежат в диапазоне биологических значений и не являются ошибками измерений, нет особых причин исключать их. Другие параметры, такие как 'Coffee_Intake' и 'Caffeine_mg' были обработаны от выбросов путём их замены на крайние значения при помощи библиотечной функции winsorize.

2.5 Поиск и обработка выбросов в качественных данных.

Исходя из диаграммы были обработаны выбросы только для одного признака 'Gender'. Выбросы были заменены на один из двух возможных значений признака, выбранных случайным образом.



Код обработки выбросов:

```
    gender_distribution = df[df['Gender'].isin(['Male', 'Female'])]['Gender'].value_counts(normalize=True)
    other_mask = df['Gender'] == 'Other'
    other_count = other_mask.sum()
    random_genders = np.random.choice(
        ['Male', 'Female'],
        size=other_count,
        p=[gender_distribution.get('Male', 0.5), gender_distribution.get('Female', 0.5)]
    )
    df.loc[other_mask, 'Gender'] = random_genders
```

2.6 Генерация новых признаков

Health_Risk_Score.

Components:
- BMI/25: deviation from ideal BMI (25)
- Stress: stress score (1-3 points)
- Age/80: relative age

The higher the value, the higher the overall health risk

Sleep_Productivity_Ratio.

Componentss:
- Sleep Hours
- Coffee Intake

High values: Lots of sleep, little coffee (natural energy)
Low values: Little sleep, lots of coffee (stimulant dependence)

2.6 Кодирование категориальных признаков

«Stress_Level», «Sleep_Quality» и «Health_Issues» были закодированы через Ordinal Encoding.

```
sleep_quality_map = {
    'Poor': 0,
    'Fair': 1,
    'Good': 2,
    'Excellent': 3
}

stress_level_map = {
    'Low': 0,
    'Medium': 1,
    'High': 2
}

health_issues_map = {
    'None': 0,
    'Mild': 1,
    'Moderate': 2,
    'Severe': 3
}

df_cat_enc['Sleep_Quality'] =
df_cat_enc['Sleep_Quality'].map(sleep_quality_map)
df_cat_enc['Stress_Level'] =
df_cat_enc['Stress_Level'].map(stress_level_map)
df_cat_enc['Health_Issues'] =
df_cat_enc['Health_Issues'].map(health_issues_map)
```

Признак «Occupation» был закодирован через Count Encoding поскольку в признаке есть много уникальных значений.

```
    count_enc = CountEncoder()
    df_cat_enc["Occupation"] = count_enc.fit_transform(df_cat_enc["Occupation"])
```

Признак «Gender» был закодирован значениями 0 и 1, поскольку признак принимает только 2 значения.

2.7 Масштабирование данных было выполнено при помощи библиотечного класса StandartScaler().

```
    scaler = StandardScaler()
    df[numerical_features] = scaler.fit_transform(df[numerical_features])
```