Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»

Отчет по лабораторной работе №2

«Обработка пропусков в данных, кодирование категориальных признаков, масштабирование данных»

по дисциплине «Технологии машинного обучения»

Выполнил:
студент группы ИУ5Ц-83Б
Пронин В.К.
подпись, дата

Проверил:
к.т.н., доц., Гапанюк Ю.Е.
подпись, дата

2026 г.

СОДЕРЖАНИЕ ОТЧЕТА

1. Цель лабораторной работы

Изучение способов предварительной обработки данных для дальнейшего формирования моделей.

2. Задание

1. Выбрать набор данных (датасет), содержащий категориальные 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.)
2. Для выбранного датасета (датасетов) на основе материалов лекции решить следующие задачи:
   - обработку пропусков в данных;
   - кодирование категориальных признаков;
   - масштабирование данных.
   -

3. Основные характеристики датасета

Название датасета: Netflix Userbase Dataset
Ссылка: https://www.kaggle.com/datasets/arnavsmayan/netflix-userbase-dataset

О датасетах

Набор данных предоставляет снимок примерной пользовательской базы Netflix, демонстрирующий различные аспекты пользовательских подписок, доходов, сведений об учетной записи и активности. Каждая строка представляет уникального пользователя, идентифицируемого по его идентификатору пользователя. Набор данных включает такую информацию, как тип подписки пользователя (базовая, Стандартная или Премиум), ежемесячный доход, получаемый от подписки, дата, когда он присоединился к Netflix (Join Date), дата его последнего платежа (Last Payment Date) и страна, в которой он находится.

Добавлены дополнительные столбцы, позволяющие получить представление о поведении и предпочтениях пользователя. В эти столбцы входят тип устройства (например, Smart TV, мобильный, настольный компьютер, планшет) и состояние учетной записи (активна учетная запись или нет). Набор данных служит синтетическим представлением и не отражает фактические данные о пользователях Netflix. Его можно использовать для анализа и моделирования, чтобы понять тенденции пользователей, предпочтения и получение дохода в рамках гипотетической базы пользователей Netflix.

Структура данных

User ID: Идентификатор пользователя
Subscription Type: Тип подписки
Monthly Revenue: Ежемесячный доход
Join Date: Дата присоединения
Last Payment Date: Дата последнего платежа

Country: Страна
Age: Возраст
Gender: Пол
Device: Устройство
Plan Duration: Срок действия тарифного плана

4. Изучение данных

Подключаем необходимые библиотеки.

[1]:
```
# Загружаем датасет и подключаем библиотеки
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
import numpy as np
import math as mth
import matplotlib.patches as patches
from scipy import stats as st
plt.rcParams.update({'figure.max_open_warning': 0})

from plotly.offline import init_notebook_mode, iplot
import plotly
import plotly.graph_objs as go
import textwrap
```

[2]:
```
df = pd.read_csv('NetflixUserbase.csv')
```

Выводим информацию.

[3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2500 entries, 0 to 2499
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   User ID            2500 non-null   int64 
 1   Subscription Type  2500 non-null   object
 2   Monthly Revenue    2500 non-null   int64 
 3   Join Date          2500 non-null   object
 4   Last Payment Date  2500 non-null   object
 5   Country            2500 non-null   object
 6   Age                2500 non-null   int64 
 7   Gender             2500 non-null   object
 8   Device             2500 non-null   object
 9   Plan Duration      2500 non-null   object
dtypes: int64(3), object(7)
memory usage: 195.4+ KB
```

Выводим названия столбцов датасета.

[4]:
```
df.columns
```

[4]:
```
Index(['User ID', 'Subscription Type', 'Monthly Revenue', 'Join Date',
       'Last Payment Date', 'Country', 'Age', 'Gender', 'Device',
       'Plan Duration'],
      dtype='object')
```

Устраним и приводим их к нижнему регистру.

[6]:
```
#Приведение к нижнему регистру и удаление лишних пробелов в названиях столбцов
df.columns = df.columns.str.strip().str.lower()
```

5. Описательная статистика

[7]:
```
df.describe()
```

[7]:

	user id	monthly revenue	age
count	2500.00000	2500.000000	2500.000000
mean	1250.50000	12.508400	38.795600
std	721.83216	1.686851	7.171778
min	1.00000	10.000000	26.000000
25%	625.75000	11.000000	32.000000
50%	1250.50000	12.000000	39.000000
75%	1875.25000	14.000000	45.000000
max	2500.00000	15.000000	51.000000

6. Предобработка данных

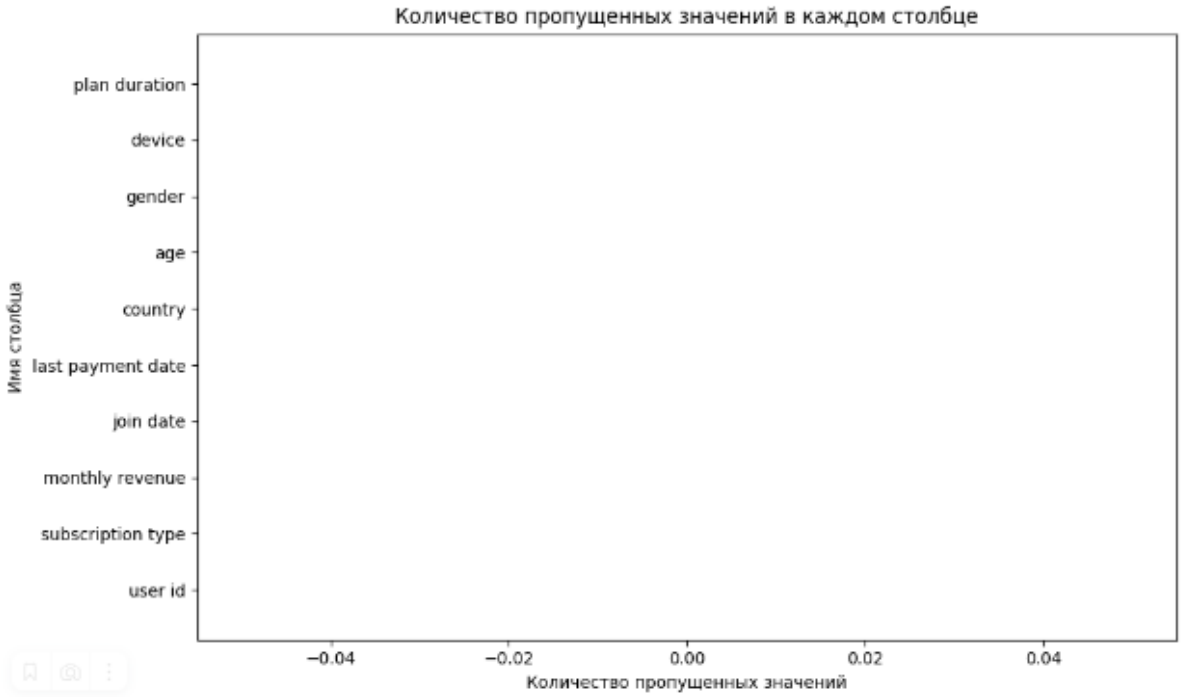
6.1.Пропуск значения

[8]:
```python
# Создаем список с именами столбцов и количеством пропущенных значений
missing_counts = [df[column].isnull().sum() for column in df.columns]

# Сортируем столбцы в порядке убывания количества пропущенных значений
sorted_columns, sorted_missing_counts = zip(*sorted(zip(df.columns, missing_counts), key=lambda x: x[1], reverse=False))

# Создаем горизонтальную столбчатую диаграмму
plt.figure(figsize=(10, 6))
# Используем barh для горизонтальных столбцов
plt.barh(sorted_columns, sorted_missing_counts)
plt.xlabel('Количество пропущенных значений')
plt.ylabel('Имя столбца')
plt.title('Количество пропущенных значений в каждом столбце')
plt.tight_layout()

# Отображаем график
plt.show()
```



[9]:
```python
columns_isnull = [col for col, count in zip(sorted_columns, sorted_missing_counts) if count > 0]
print(f'Названий столбцов, у которых пропуски:')
for col in columns_isnull:
    print('\t' + col)
```

```
Названий столбцов, у которых пропуски:
```

[10]:
```python
# Проверим наличие пустых значений
# Цикл по колонкам датасета
for col in df.columns:
    # Количество пустых значений - все значения заполнены
    temp_null_count = df[df[col].isnull()].shape[0]
    print('{} - {}'.format(col, temp_null_count))
```

```
user id - 0
subscription type - 0
monthly revenue - 0
join date - 0
last payment date - 0
country - 0
age - 0
gender - 0
device - 0
plan duration - 0
```

В нашем датасете нет пропусков значений, это говорится о том, что датасет

сделали идеальным образом.

6.2.Дубликаты

```
[11]: # Количество дублирующих значений
      df.duplicated().sum()
```

```
[11]: 0
```

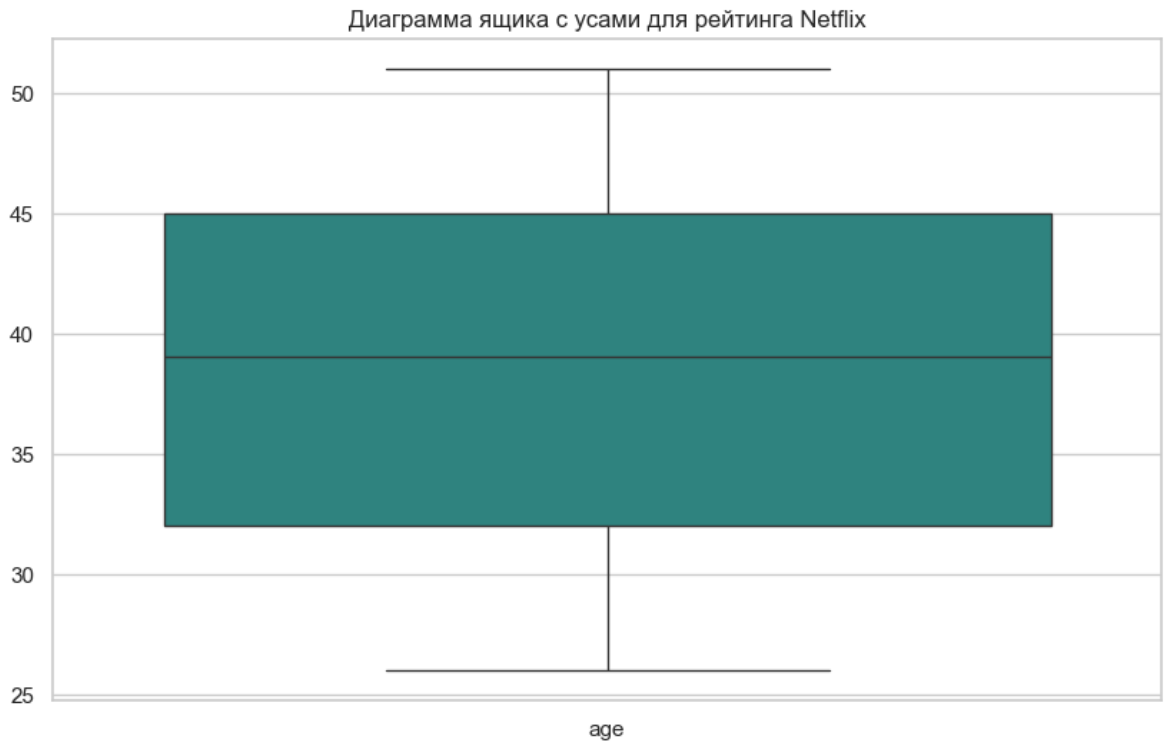
Дубликатов тоже нет.

6.3.Выбросы (Ящик с усами)

```
[12]: sb.set(style="whitegrid")

      plt.figure(figsize=(10, 6))
      sb.boxplot(data=df[['age']], palette='viridis')

      plt.title('Диаграмма ящика с усами для рейтинга Netflix')
      plt.show()
```



Часто просматриваемые это возраст от 32 до 45 лет

6.4.Преобразование категорий в числа

[13]:
```python
# Создаем словарь для соответствия категорий и их кодов
type_subscription = {
    'Basic': 1,
    'Standard': 2,
    'Premium': 3
}

# Присваиваем числовые коды
df['hotelrating_encoded'] = df['subscription type'].map(type_subscription)
```

7. Итог

7.1.Предобработка данных

- Пропусков нет
- Дубликатов нет
- Количество выбросов отсутствует
- В ходе предобработки данных было вявлено, что есть колонка, которая дает информацию типов подписок, но однако информация носит HTML формата, что на парсинг уходит много времени и необходимо сопровождать кода. Для тщательного исследования будет полезно.

8. Вывод

В ходе выполнения лабораторной работы изучили способы предварительной обработки данных для дальнейшего формирования моделей.