Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное 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-84Б
Падалко К.Р.
подпись, дата

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

2025 г.

СОДЕРЖАНИЕ ОТЧЕТА

1. Цель лабораторной работы

Изучение способов предварительной обработки данных для дальнейшего формирования моделей.

2. Описание задания

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: **Most Popular YouTube 1000 videos** (Самые популярные видео с YouTube 1000)

Ссылка: https://www.kaggle.com/datasets/samithsachidanandan/most-popular-1000-youtube-videos

О датасетах

Этот набор данных содержит информацию о 1000 самых популярных видео на YouTube. Он включает в себя различные параметры, такие как количество просмотров, лайков, дизлайков, категорию видео и год публикации. Этот датасет предоставляет возможность исследовать факторы, которые влияют на популярность видео на YouTube, анализировать тенденции

в контенте и изучать взаимосвязи между различными характеристиками видео.

Структура данных:

- **rank: Ранг видео в списке (от 1 до 1000).**
- **Video: Название видео.**
- **Video views: Количество просмотров видео.**
- **Likes: Количество лайков видео.**
- **Dislikes: Количество дизлайков видео.**
- **Category: Категория, к которой относится видео (например, Music, Entertainment, Sports).**
- **published: Год публикации видео.**

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("Most popular 1000 Youtube videos.csv", encoding='UTF-8-SIG')
```

Выводим информацию.

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   rank         1000 non-null   int64 
 1   Video        1000 non-null   object
 2   Video views  1000 non-null   object
 3   Likes        1000 non-null   object
 4   Dislikes     527 non-null    object
 5   Category     982 non-null    object
 6   published    1000 non-null   int64 
dtypes: int64(2), object(5)
memory usage: 54.8+ KB
```

Выводим названия столбцов датасета.

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

[4]:
```
Index(['rank', 'Video', 'Video views', 'Likes', 'Dislikes', 'Category',
       'published'],
      dtype='object')
```

Выводим первые и последние 5 строк датасета.

[6]:
```
#возвращает первые 5 строк
display(df.head())
#возвращает последние 5 строк
display(df.tail())
```

	rank	Video	Video views	Likes	Dislikes	Category	published
0	1	Lil Nas X - Old Town Road (Official Movie) ft....	54,071,677	3,497,955	78,799	Music	2019
1	2	20 Tennis shots if they were not filmed, NOBOD...	3,471,237	19,023	859	NaN	2017
2	3	JoJo Siwa - Karma (Official Video)	34,206,747	293,563	NaN	Music	2024
3	4	David Kushner - Daylight (Official Music Video)	18,558,390	680,732	NaN	Music	2023
4	5	Wiz Khalifa - See You Again ft. Charlie Puth [...	6,547,981,039	44,428,537	NaN	Music	2015

	rank	Video	Video views	Likes	Dislikes	Category	published
995	996	New Champ Kayn/Rhaast Leak for LOL (Moobeat cr...	847,249	1,857	173	People & Blogs	2017
996	997	Ford Mustang Launch (street)	1,001,605	2,214	27	Autos & Vehicles	2008
997	998	Eminem is gay - The Interview	2,718,939	43,492	0	Entertainment	2014
998	999	Yakuza OST - Baka Mitai (ばかみたい) Kiryu full ver...	52,890,986	850,425	0	Gaming	2017
999	1000	What a Twist #memes #shorts #movie	11,637,337	938,043	NaN	Gaming	2024

Устраним и приводим к нижнему регистру.

[7]:
```
#Приведение к нижнему регистру и удаление лишних пробелов в названиях столбцов
df.columns = df.columns.str.strip().str.lower()
```

5. Описательная статистика

[8]:
```
df.describe()
```

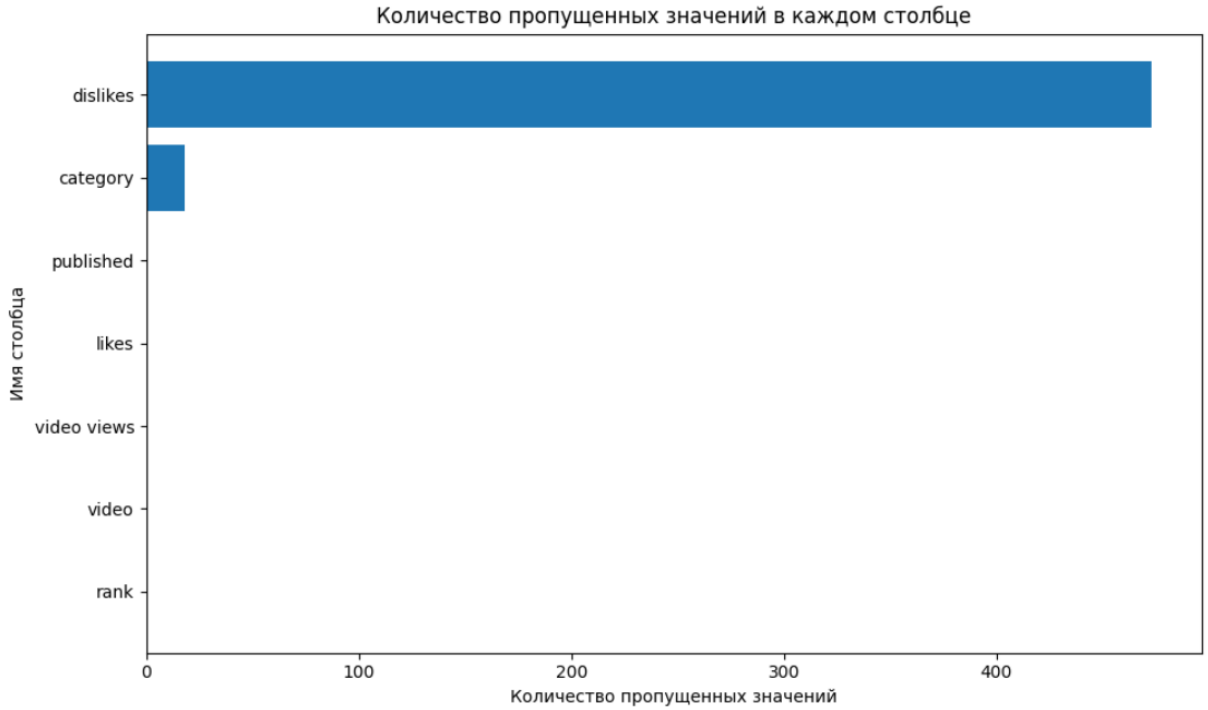
[8]:

	rank	published
count	1000.000000	1000.000000
mean	500.500000	2019.100000
std	288.819436	5.384328
min	1.000000	2005.000000
25%	250.750000	2017.000000
50%	500.500000	2021.000000
75%	750.250000	2024.000000
max	1000.000000	2025.000000

6. Предобработка данных

6.1. Пропуск значения

[9]:
```
# 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()
```



[10]:
```
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:
        category
        dislikes
```

[11]:
```
# 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))
```

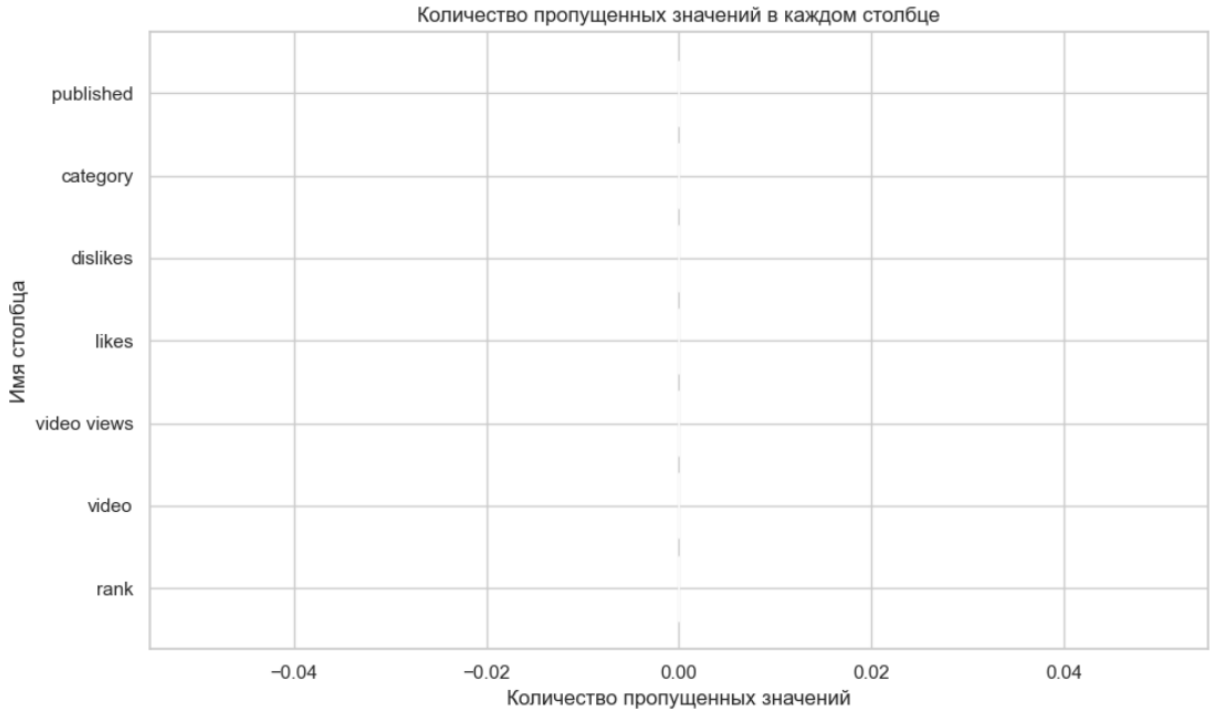
```
rank - 0
video - 0
video views - 0
likes - 0
dislikes - 473
category - 18
published - 0
```

Для заполнения пропусков dislikes и category выполним команды:

[ ]:
```
#заполнение пропусков в категориях на any (любая)
df['category'] = df['category'].fillna('any')
#заполнение пропусков в дизлайках на 0
df['dislikes'] = df['dislikes'].fillna(0)
```

Еще раз проверим наличие пропусков.

[36]:
```
# Проверка
# 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()
```



[37]:
```
# 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))
```

```
rank - 0
video - 0
video views - 0
likes - 0
dislikes - 0
category - 0
published - 0
```

6.2. Дубликаты

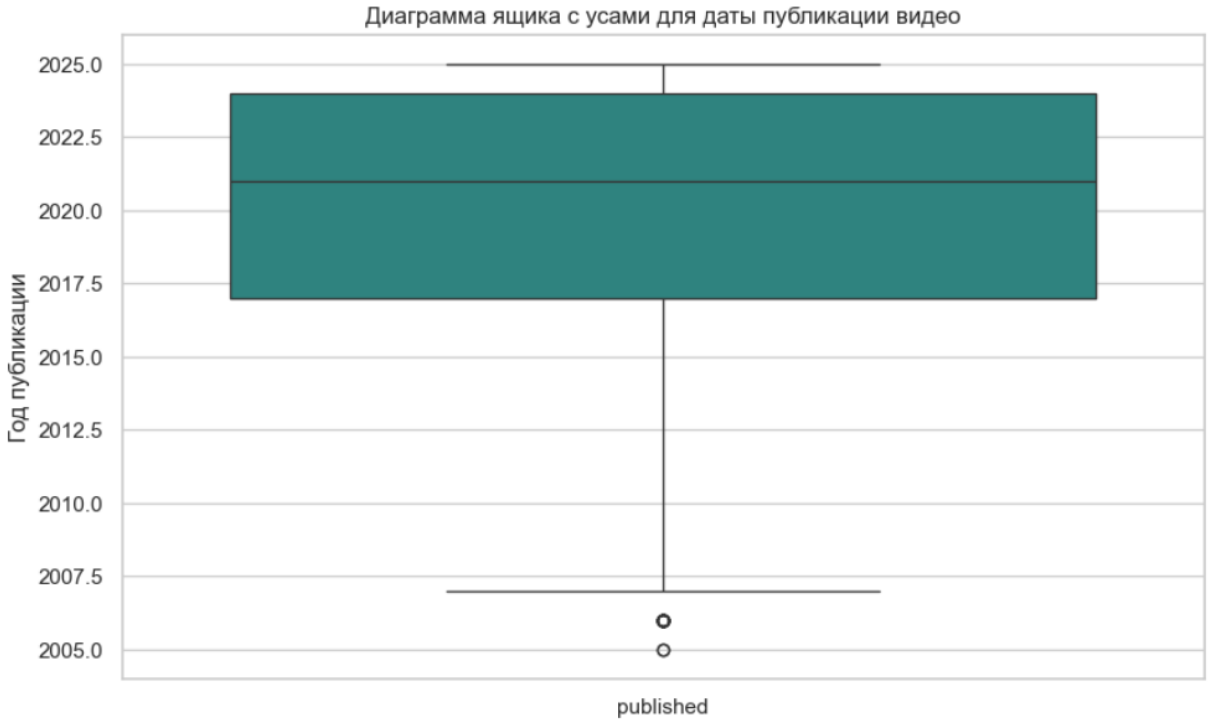
[38]:
```
# Количество дублирующих значений
df.duplicated().sum()
```

[38]:
```
np.int64(0)
```

Дубликатов нет.

6.3. Выбросы (Ящик с усами)

```
[39]: # Диаграмма ящика с усами для "Video views"
sb.set(style="whitegrid")
plt.figure(figsize=(10, 6))
sb.boxplot(data=df[['published']], palette='viridis')
plt.title('Диаграмма ящика с усами для даты публикации видео')
plt.ylabel('Год публикации')
plt.show()
```



Часто видео публиковали с 2017 по 2024.

6.4. Преобразование категорий в числа

Получаем какие есть категории.

```
[40]: # Получить уникальные категории
unique_categories = df['category'].unique()

print("Уникальные категории:", unique_categories)
```

```
Уникальные категории: ['Music' 'any' 'Entertainment' 'People & Blogs' 'News & Politics'
 'Pets & Animals' 'Sports' 'Travel & Events' 'Education' 'Gaming' 'Comedy'
 'Film & Animation' 'Autos & Vehicles' 'Howto & Style'
 'Science & Technology' 'Nonprofits & Activism']
```

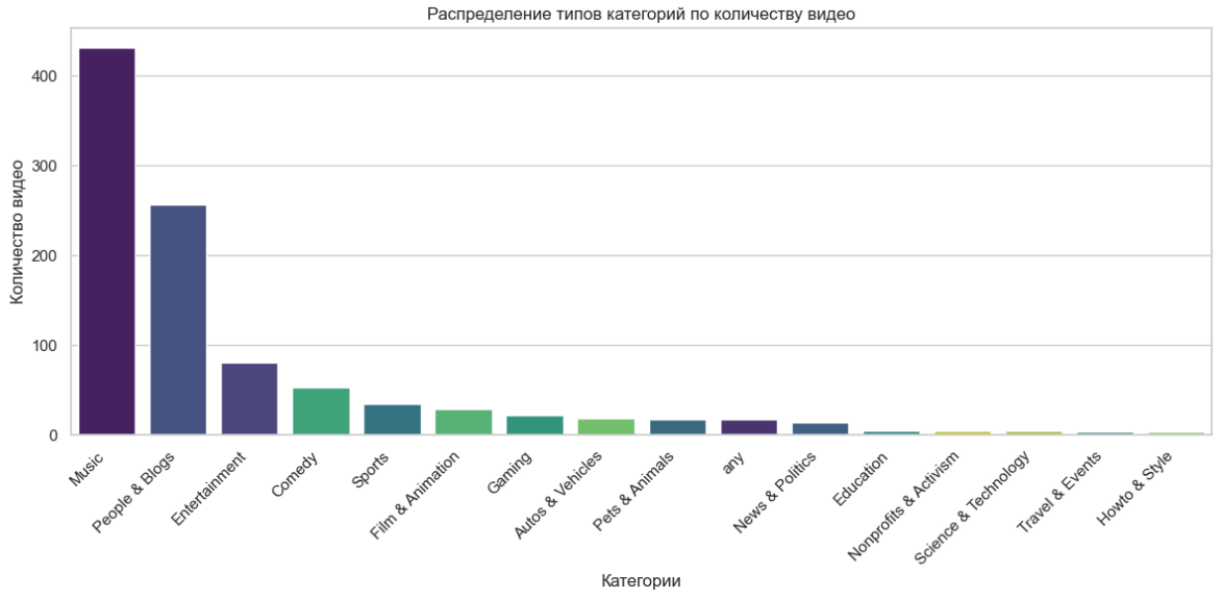
Преобразуем категории в числа.

[45]:
```python
# Создаем словарь для соответствия категорий и их кодов
type_category = {
    'Music': 1,
    'Entertainment': 2,
    'People & Blogs': 3,
    'News & Politics': 4,
    'Pets & Animals': 5,
    'Sports': 6,
    'Travel & Events': 7,
    'Education': 8,
    'Gaming': 9,
    'Comedy': 10,
    'Science & Technology': 11,
    'Nonprofits & Activism': 12,
    'any': 13
}

# Присваиваем числовые коды
df['rating'] = df['category'].map(type_category)
```

Проведем исследование на распределение категорий по количеству видео.

[46]:
```python
# Подсчет количества типов категорий и сортировка по убыванию
top_category = df['category'].value_counts().nlargest(20).index

# Фильтрация данных
df_top_category = df[df['category'].isin(top_category)]
```

[49]:
```python
# Установка стиля
sb.set(style="whitegrid")

# Создание диаграммы
plt.figure(figsize=(12, 6))
sb.countplot(x='category', data=df_top_category, order=top_category, hue='category', palette='viridis', legend=False)

# Настройки графика
plt.title('Распределение типов категорий по количеству видео')
plt.xlabel('Категории')
plt.ylabel('Количество видео')
plt.xticks(rotation=45, ha='right')  # Поворот меток X для удобства чтения

# Оптимизация отображения
plt.tight_layout()
plt.show()
```



```
[50]: df['category'].value_counts()
```

```
[50]: category
      Music                   431
      People & Blogs          257
      Entertainment            81
      Comedy                   53
      Sports                   35
      Film & Animation         29
      Gaming                   22
      Autos & Vehicles         19
      Pets & Animals           18
      any                      18
      News & Politics          14
      Education                 5
      Nonprofits & Activism     5
      Science & Technology      5
      Travel & Events           4
      Howto & Style             4
      Name: count, dtype: int64
```

Результат: Категория «Music» имеет самое большое количество - более 400 видео. Эта категория является самой популярной категорией в YouTube среди 1000 видео.

7. Итог

- Пропусков нет
- Дубликатов нет
- Количество выбросов отсутствует
- В ходе предобработки данных было выявлено, что есть колонка, которая дает информацию типов категорий, 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.