МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
"ЛЭТИ" ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

КАФЕДРА ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

ОТЧЁТ
ПО ИНДИВИДУАЛЬНОМУ ДОМАШНЕМУ ЗАДАНИЮ
"СТАТИСТИЧЕСКАЯ МОДЕЛЬ ПРОИЗВОДИТЕЛЬНОСТИ НОУТБУКОВ (Python 3)"
"ПРЕДОБРАБОТКА"

"ОПТИМИЗАЦИЯ И МНОГОКРИТЕРИАЛЬНЫЙ ВЫБОР В ТЕХНИЧЕСКИХ СИСТЕМАХ"

Выполнил: _____ Лесниченко Александр Олегович (группа 0308)

Преподаватель: _____ Шумилов Лев Алексеевич

Санкт-Петербург

2022

ОГЛАВЛЕНИЕ

ЗАДАНИЕ

Воспроизвести на Python 3 статистическую модель линейной регрессии, которая описана в ВКР Тумановой Веры Дмитриевны.

РЕАЛИЗАЦИЯ

Подробно изучив ВКР Тумановой Веры Дмитриевны, было выделено три пункта реализации:

1. Предобработка данных.
2. Обучение модели линейной регрессии, получение значений.
3. Тестирование модели.

Для реализации был выбран язык программирования Python, как самый подходящий язык для работы с данными и обучения модели линейеной регрессии.

- Для работы с данными была использована библиотека Pandas.
- Для создания модели линейной регрессии была использована библиотека statsmodel.
- Для визуализации были использованы библиотеки matplotlib и seaborn.

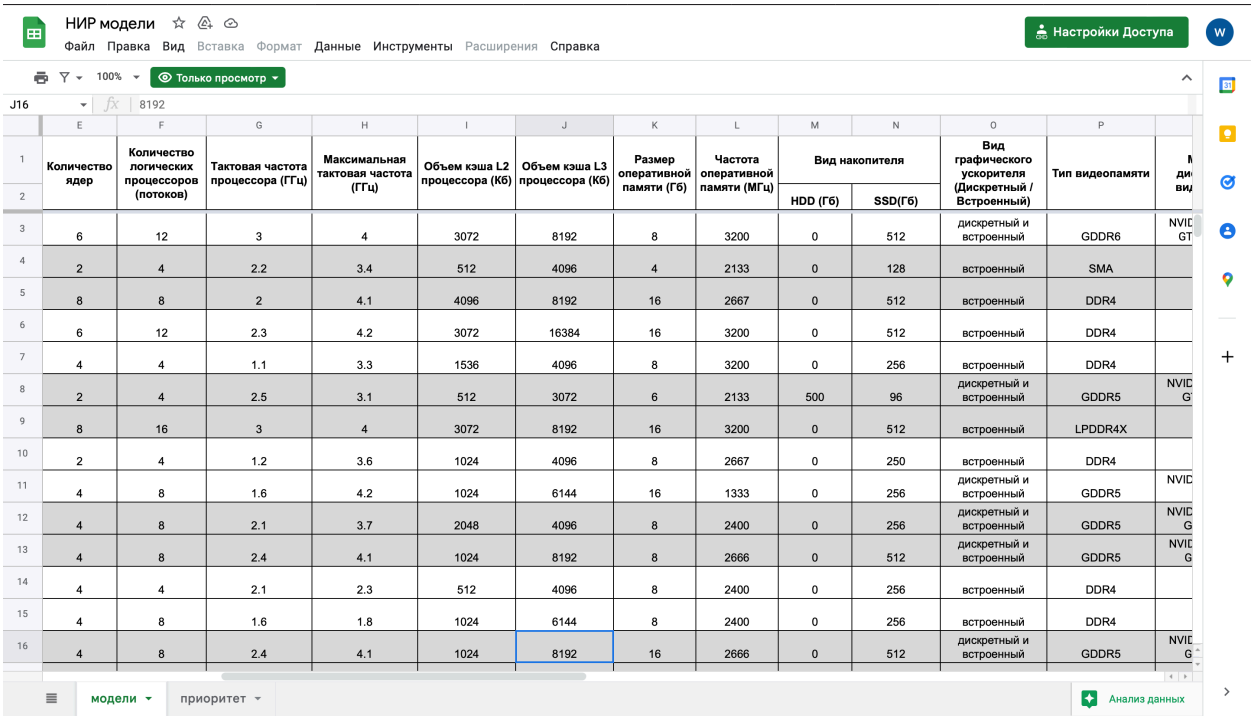
После получения набора данных, была необходимость предобработать данные:

1. Изменить таблицу для создания DataFrame в Pandas.
2. Удалить значения NaN.
3. Закодировать категориальные и номинальные значения.
4. Визуализировать данные:
    1. Построить матрицу корреляции.
    2. Построить «ящик с усами» для выявления выбросов в выборке.

ОБРАБОТКА EXCEL - ТАБЛИЦЫ

Так как данные были получены в формате онлайн - таблицы Excel, первым шагом были убраны столбцы, которые Pandas бы не смог прочитать.

Первоначальный вид таблицы:


НИР модели

Файл Правка Вид Вставка Формат Данные Инструменты Расширения Справка

Настройки Доступа

Только просмотр

J16 8192

Количество ядер	Количество логических процессоров (потоков)	Тактовая частота процессора (ГГц)	Максимальная тактовая частота (ГГц)	Объем кэша L2 процессора (Кб)	Объем кэша L3 процессора (Кб)	Размер оперативной памяти (Гб)	Частота оперативной памяти (МГц)	Вид накопителя		Вид графического ускорителя (Дискретный / Встроенный)	Тип видеопамяти	
								HDD (Гб)	SSD(Гб)			
6	12	3	4	3072	8192	8	3200	0	512	дискретный и встроенный	GDDR6	NVID GT
2	4	2.2	3.4	512	4096	4	2133	0	128	встроенный	SMA	
8	8	2	4.1	4096	8192	16	2667	0	512	встроенный	DDR4	
6	12	2.3	4.2	3072	16384	16	3200	0	512	встроенный	DDR4	
4	4	1.1	3.3	1536	4096	8	3200	0	256	встроенный	DDR4	
2	4	2.5	3.1	512	3072	6	2133	500	96	дискретный и встроенный	GDDR5	NVID GT
8	16	3	4	3072	8192	16	3200	0	512	встроенный	LPDDR4X	
2	4	1.2	3.6	1024	4096	8	2667	0	250	встроенный	DDR4	
4	8	1.6	4.2	1024	6144	16	1333	0	256	дискретный и встроенный	GDDR5	NVID
4	8	2.1	3.7	2048	4096	8	2400	0	256	дискретный и встроенный	GDDR5	NVID G
4	8	2.4	4.1	1024	8192	8	2666	0	512	дискретный и встроенный	GDDR5	NVID G
4	4	2.1	2.3	512	4096	8	2400	0	256	встроенный	DDR4	
4	8	1.6	1.8	1024	6144	8	2400	0	256	встроенный	DDR4	
4	8	2.4	4.1	1024	8192	16	2666	0	512	дискретный и встроенный	GDDR5	NVID G

модели приоритет Анализ данных

Рис. 1

Видно, что в столбце «Вид накопителя» есть две строки, подобных ситуаций необходимо избегать.

Также на первом шаге были устранены незначительные помарки (например, правильное наименование столбцов), которые в дальнейшем важны для работы.

Устранив вышеописанные проблемы, таблица была экспортирована в расширении .csv.

УДАЛЕНИЕ NaN ЗНАЧЕНИЙ

К сожалению, в некоторых ячейках таблицы присутствуют значения NaN. Это можно проверить следующим образом:

```
In [2]: import pandas as pd
        import numpy as np
        import seaborn as sns
        import matplotlib.pyplot as plt

In [3]: df = pd.read_csv(r"/Users/aleksandr/laptop-performance/data/laptop_info.csv")

In [4]: df.isnull().sum()

Out[4]: №                                                    0
        Модель (полное название)                             0
        Модель процессора                                    0
        Год выпуска процессора                               0
        Количество ядер                                      0
        Количество логических процессоров (потоков)          0
        Тактовая частота процессора (ГГц)                    0
        Максимальная тактовая частота (ГГц)                  0
        Объем кэша L2 процессора (Кб)                        0
        Объем кэша L3 процессора (Кб)                        0
        Размер оперативной памяти (Гб)                       0
        Частота оперативной памяти (МГц)                     0
        HDD (Гб)                                             0
        SSD(Гб)                                              0
        Вид графического ускорителя (Дискретный / Встроенный)  0
        Тип видеопамяти                                      0
        Модель дискретной видеокарты                         0
        Объем видеопамяти (Гб)                               0
        Модель встроенной видеокарты                         0
        Объем видеопамяти (Гб).1                             0
        Тип (марка) куллера материнской платы               22
        Браузер для тестов                                   0
        Учебный                                              0
        Развлекательный                                      0
        Эталон                                               0
        Исполнитель                                         11
        dtype: int64
```

Рис. 2

Видно, что в столбце «Тип (марка) куллера материнской платы» содержится 22 значения NaN. Удалим эти строки.

В ходе просмотра таблицы, было замечено, что ячейки столбцов «Объем видеопамяти (Гб) [дискретной]» и «Объем видеопамяти (Гб) [встроенной]» пустые. Ввиду того, что на некоторых компьютерах отсутствует та или иная видеопамять, было решено заменить пустые значения нулями. Также в некоторых столбцах тип данных числовых значений был типом Object, который нужно было убрать командой `pd.to_numeric()`:

```
# --- Обработка числовых значений ---
obj_df = df[num_features].copy()
obj_df['Объем видеопамяти (Гб) [дискретной]'] = pd.to_numeric(obj_df['Объем видеопамяти (Гб) [дискретной]'],
                                                              errors='coerce')
obj_df['Объем видеопамяти (Гб) [встроенной]'] = pd.to_numeric(obj_df['Объем видеопамяти (Гб) [встроенной]'],
                                                              errors='coerce')
obj_df = obj_df.fillna({"Объем видеопамяти (Гб) [встроенной]": 0,
                        "Объем видеопамяти (Гб) [дискретной]": 0})
df[num_features] = obj_df
```

Рис. 3

КОДИРОВАНИЕ КАТЕГОРИАЛЬНЫХ ЗНАЧЕНИЙ

Для кодирования категориальных значений был использован метод унитарного кодирования (One Hot Encoding). Основная стратегия состоит в том, чтобы преобразовать значение каждой категории в новый столбец и присвоить столбцу значение 1 или 1 (*Истина / Ложь*). Это дает преимущество в том, что значение не взвешивается неправильно, но имеет обратную сторону добавления дополнительных столбцов в набор данных.

Pandas поддерживает эту возможность с помощью `pd.get_dummies()`.
Эта функция названа так, потому что она создает фиктивные (dummy) индикаторные переменные (1 или 0):

```
# --- Вид графического ускорителя ---
graph_acc = pd.get_dummies(df['Вид графического ускорителя (Дискретный / Встроенный)'])
graph_acc.head()
graph_acc.columns = graph_acc.columns.str.strip()
graph_acc.loc[graph_acc['дискретный и встроенный'] == 1, ['встроенный', 'дискретный']] = 1
graph_acc = graph_acc.drop(['дискретный и встроенный', 'встроенный'], axis=1)
graph_acc = rename_cols('Наличие графического ускорителя', graph_acc)
graph_acc = graph_acc.fillna(0)

# --- Тип видеопамяти ---
video_mem = pd.get_dummies(df['Тип видеопамяти'])
video_mem = rename_cols('Тип видеопамяти', video_mem)
```

Рис. 4

Для кодирования размера оперативной памяти предложено бинарное кодирование:

```
# --- Размер оперативной памяти ---
RAM_size = (df['Размер оперативной памяти (Гб)'] // 8 > 0).astype(int)

df = df.drop(cat_features, axis=1)
add_cols(df, graph_acc)
add_cols(df, video_mem)
df['Оперативная память > 8'] = RAM_size
```

Рис. 5

ВИЗУАЛИЗАЦИЯ ДАННЫХ

КОРРЕЛЯЦИОННЫЙ АНАЛИЗ

После того, как мы сделали предобработку данных, необходимо понять, как данные зависят друг от друга. Для этого используем функцию `corr()` из библиотеки pandas. С помощью неё мы сможем понять как признаки коррелируют между собой:

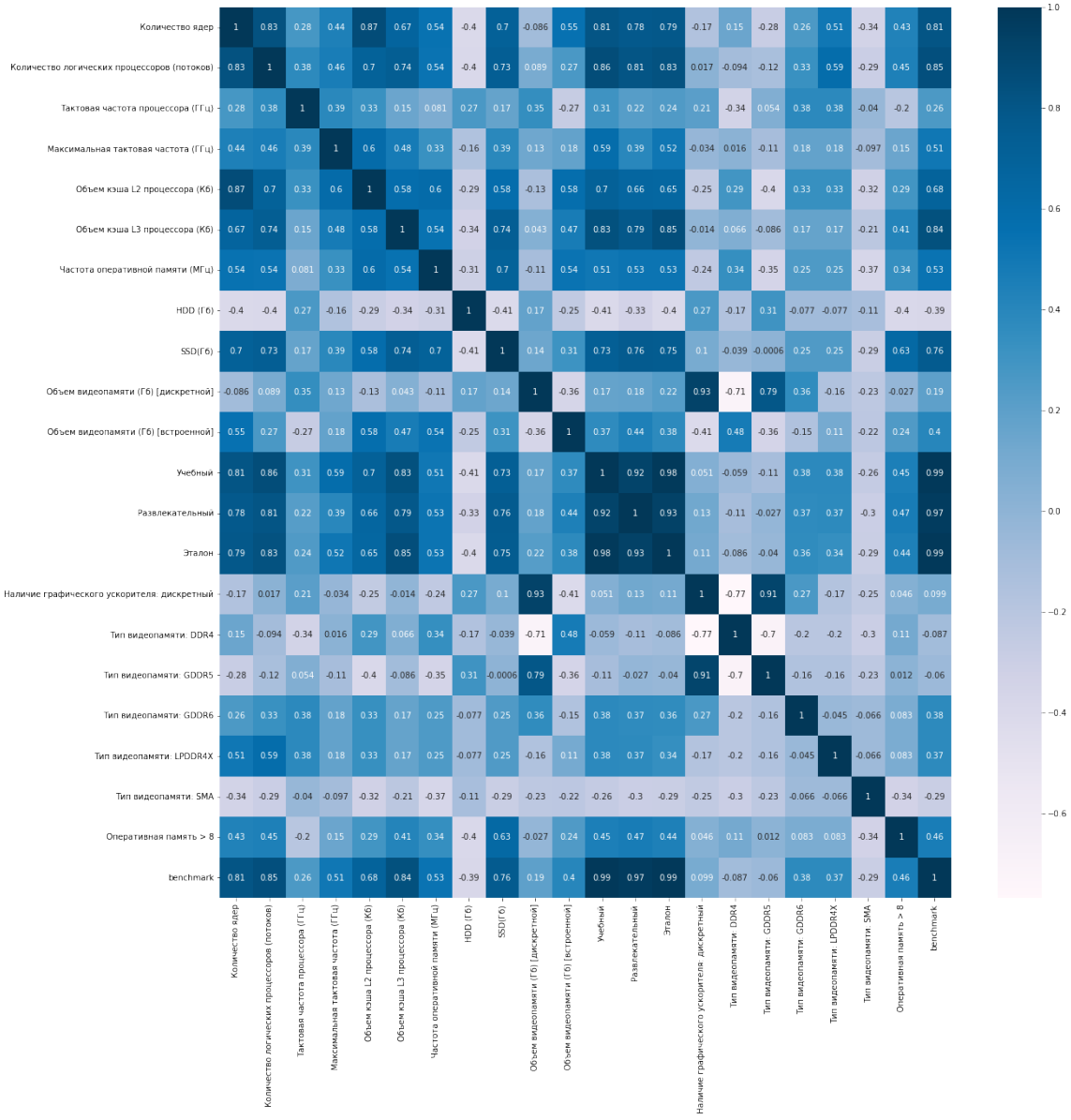
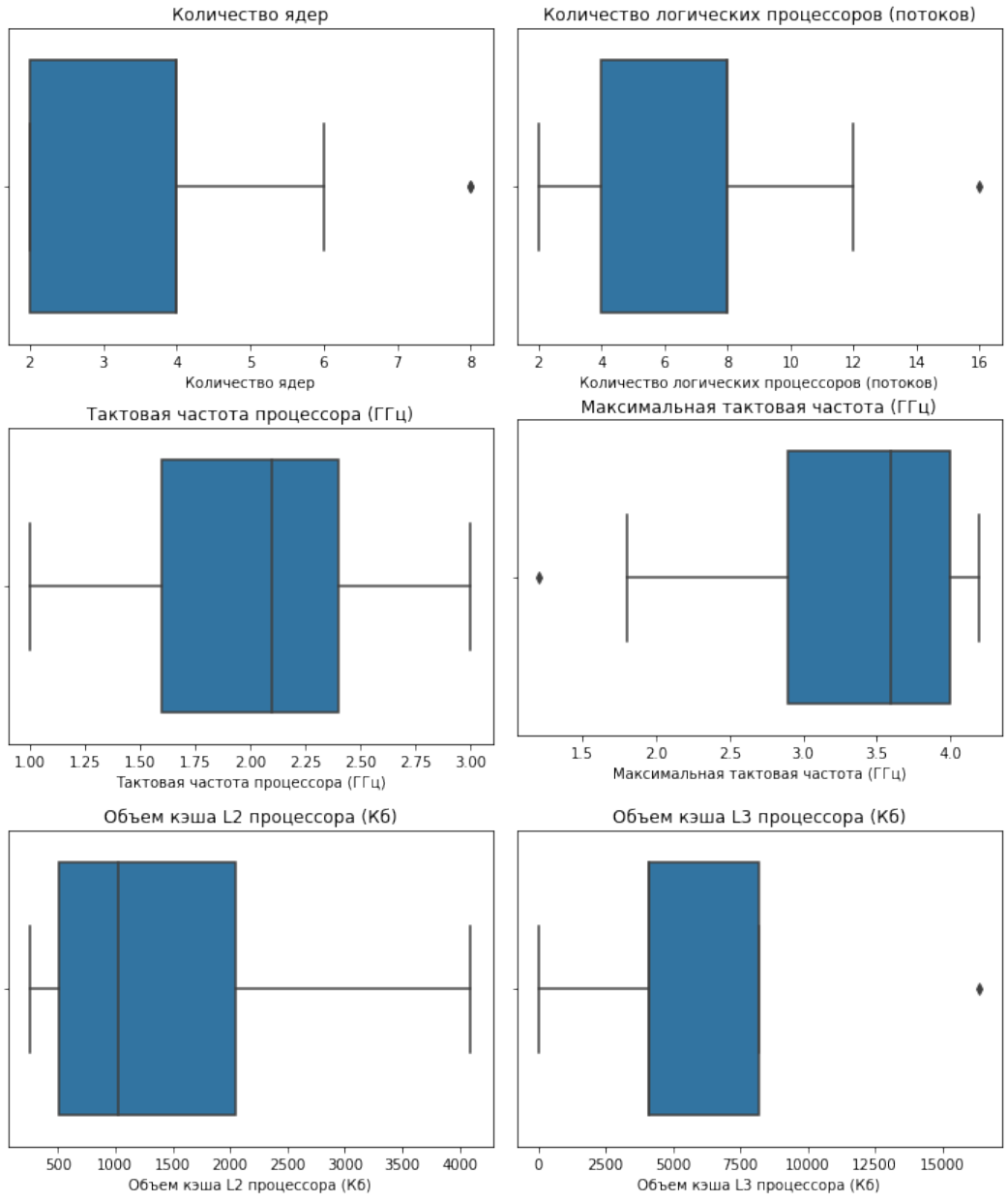
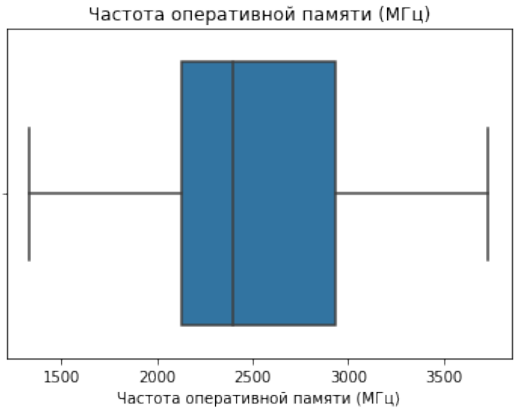


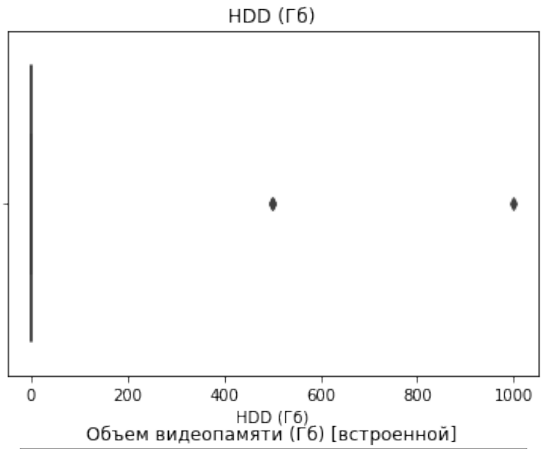
Рис. 6

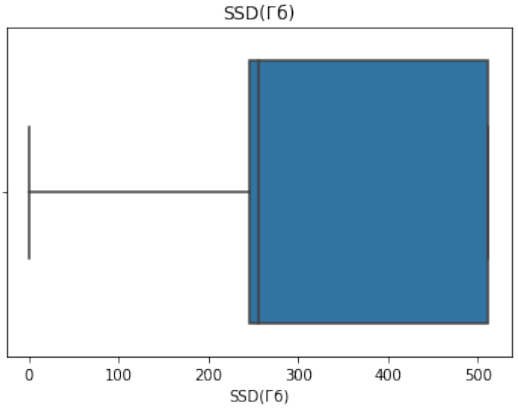
ДИАГРАММА РАССЕИВАНИЯ

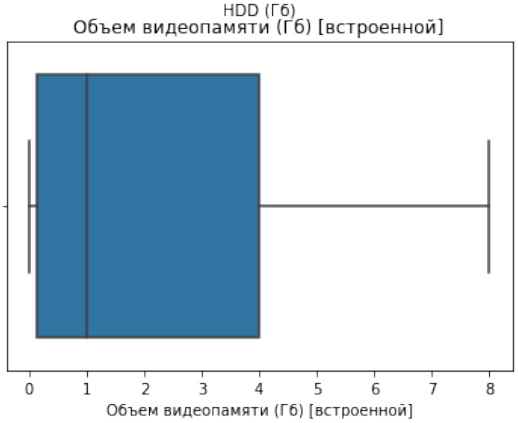
Для выявления «выбросов» выборки используем функцию `boxplot()` из библиотеки seaborn (рис. 7 - 20):

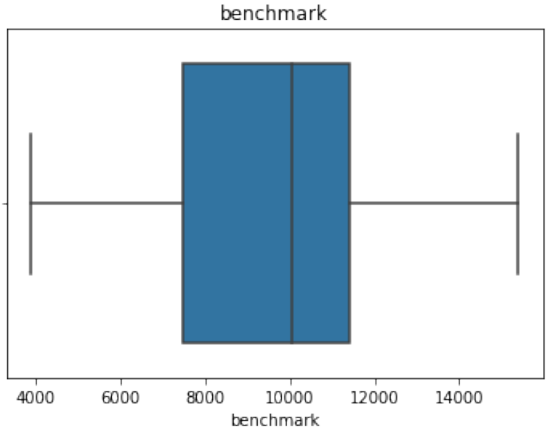


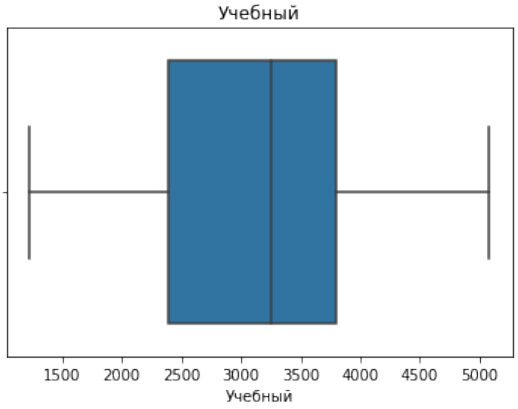

Частота оперативной памяти (МГц)
1500
2000
2500
3000
3500
Частота оперативной памяти (МГц)

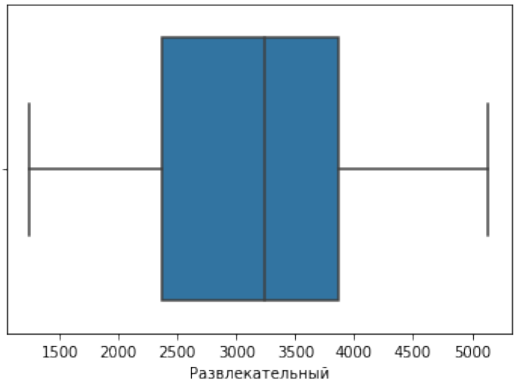

HDD (Гб)
0
200
400
600
800
1000
HDD (Гб)

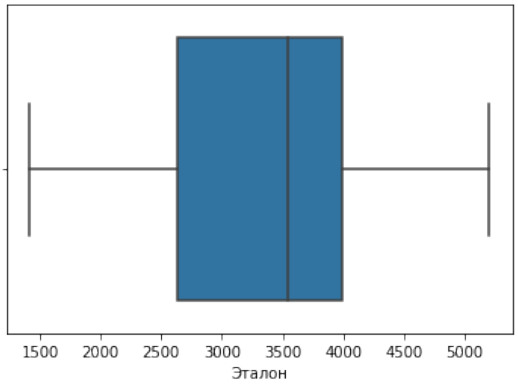

SSD(Гб)
0
100
200
300
400
500
SSD(Гб)


Объем видеопамяти (Гб) [встроенной]
0
1
2
3
4
5
6
7
8
Объем видеопамяти (Гб) [встроенной]


benchmark
4000
6000
8000
10000
12000
14000
benchmark


Учебный
1500
2000
2500
3000
3500
4000
4500
5000
Учебный

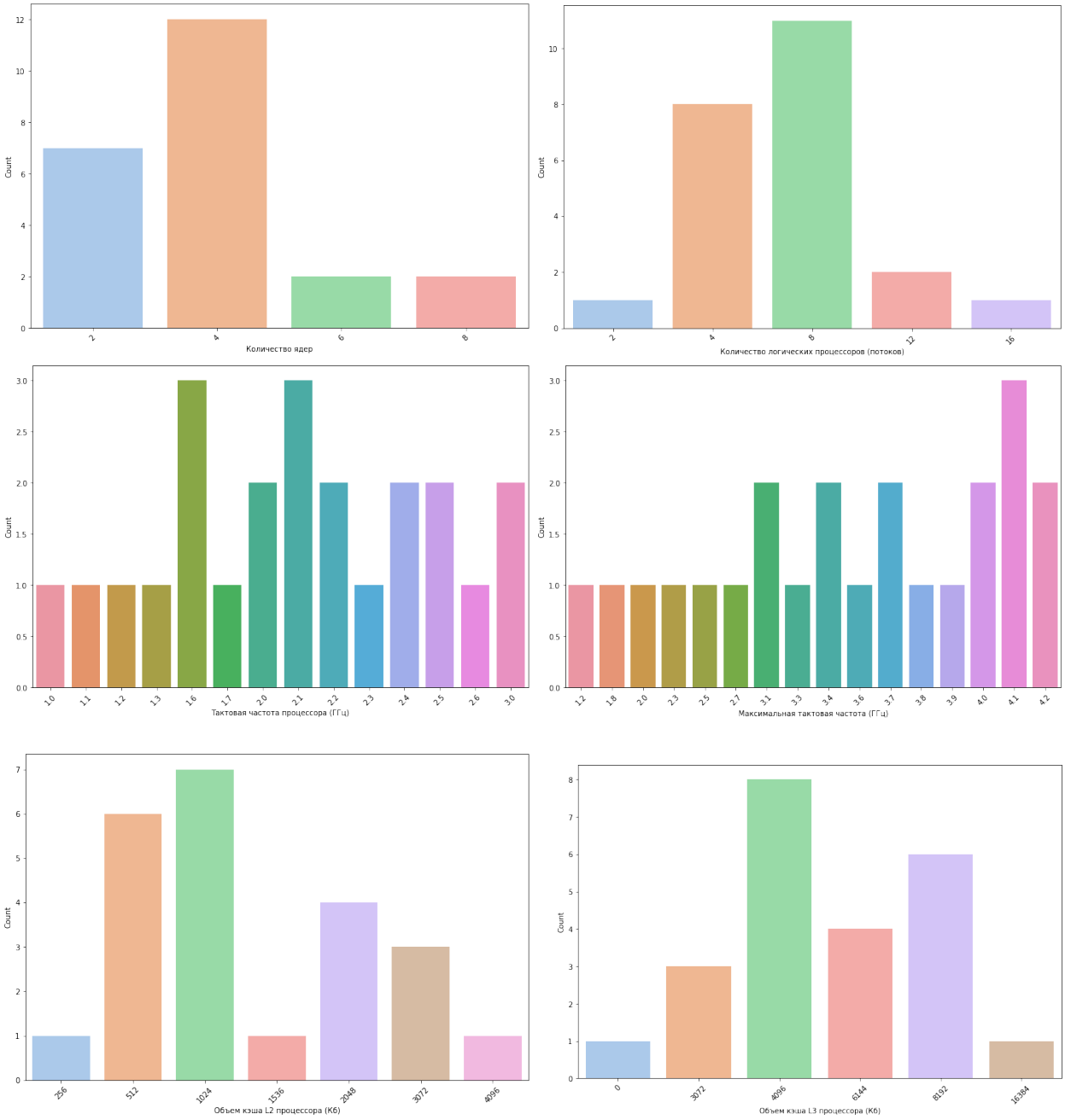

Развлекательный
1500
2000
2500
3000
3500
4000
4500
5000
Развлекательный

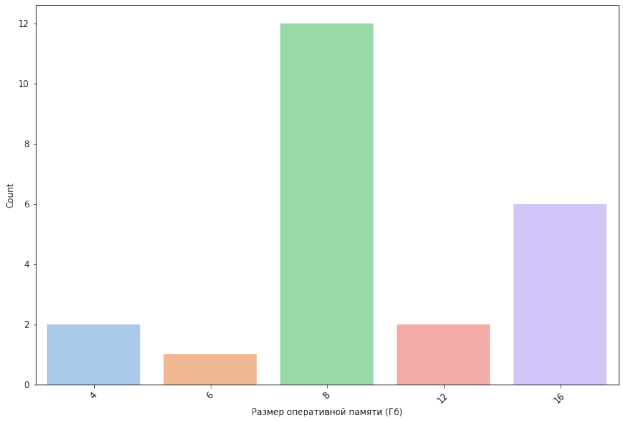

Эталон
1500
2000
2500
3000
3500
4000
4500
5000
Эталон

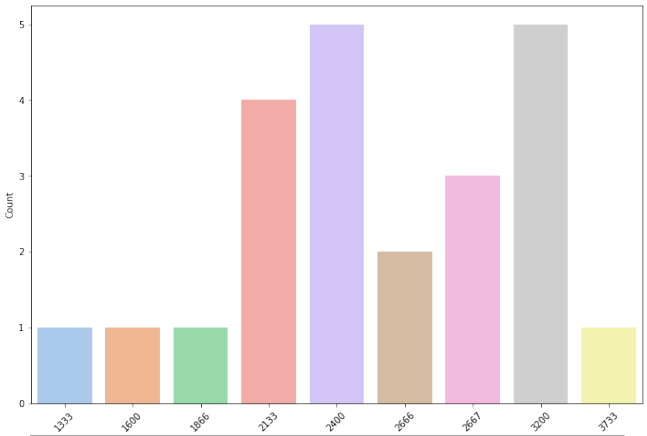
Подведём итог о диаграммах рассеивания: видно, что выбросов нет, с данными можно работать без дополнительной обработки выбросов.

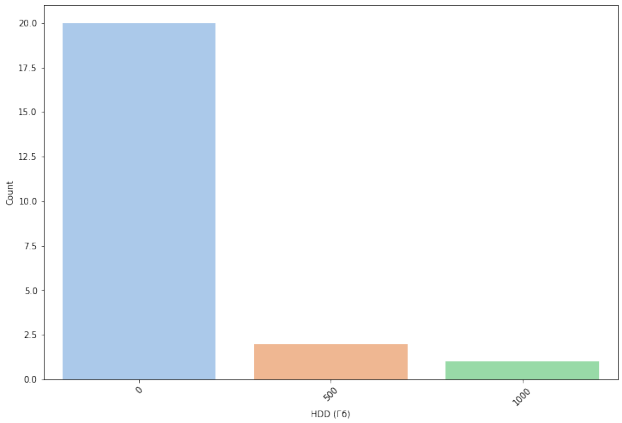
ГИСТОГРАММА НОМИНАЛЬНЫХ ЗНАЧЕНИЙ

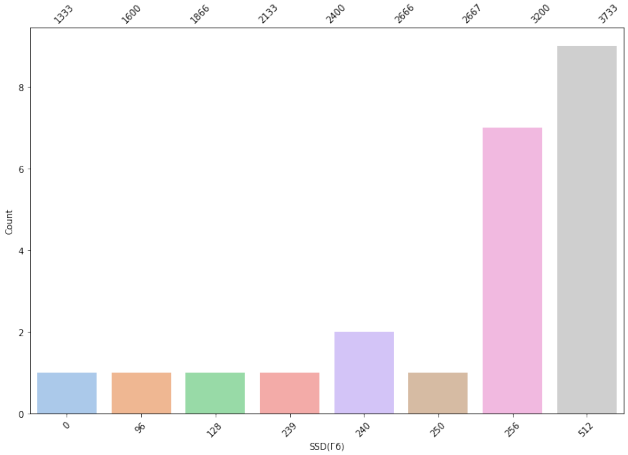
Посмотрим, как распределены наши данные (каждый столбец) (рис. 21 - 31):




Count
Размер оперативной памяти (Гб)
4
6
8
12
16


Count
1333
1600
1866
2133
2400
2666
2667
3200
3733


Count
HDD (Гб)
0
500
1000


Count
SSD(Гб)
0
96
128
239
240
250
256
512

ИТОГОВАЯ ПРОВЕРКА НАБОРА

Проверим обработанный набор данных на наличие NaN значений:

```
In [37]: # --- Итоговая проверка на наличие NaN значений ---
         df.isnull().sum()
```

```
Out[37]: Количество ядер                                  0
         Количество логических процессоров (потоков)     0
         Тактовая частота процессора (ГГц)                0
         Максимальная тактовая частота (ГГц)              0
         Объем кэша L2 процессора (Кб)                    0
         Объем кэша L3 процессора (Кб)                    0
         Частота оперативной памяти (МГц)                 0
         HDD (Гб)                                         0
         SSD(Гб)                                          0
         Объем видеопамяти (Гб) [дискретной]              0
         Объем видеопамяти (Гб) [встроенной]              0
         Учебный                                          0
         Развлекательный                                  0
         Эталон                                           0
         Наличие графического ускорителя: дискретный      0
         Тип видеопамяти: DDR4                            0
         Тип видеопамяти: GDDR5                           0
         Тип видеопамяти: GDDR6                           0
         Тип видеопамяти: LPDDR4X                         0
         Тип видеопамяти: SMA                             0
         Оперативная память > 8                           0
         benchmark                                        0
         dtype: int64
```

Рис. 32

NaN значения отсутствуют, следовательно, набор данных обработан верно и готов к построению модели.

Типы данных:

```
In [39]: # --- Типы данных ---
         df.dtypes

Out[39]: Количество ядер                                    int64
         Количество логических процессоров (потоков)        int64
         Тактовая частота процессора (ГГц)                float64
         Максимальная тактовая частота (ГГц)              float64
         Объем кэша L2 процессора (Кб)                      int64
         Объем кэша L3 процессора (Кб)                      int64
         Частота оперативной памяти (МГц)                   int64
         HDD (Гб)                                           int64
         SSD(Гб)                                            int64
         Объем видеопамяти (Гб) [дискретной]              float64
         Объем видеопамяти (Гб) [встроенной]              float64
         Учебный                                            int64
         Развлекательный                                    int64
         Эталон                                             int64
         Наличие графического ускорителя: дискретный      float64
         Тип видеопамяти: DDR4                              uint8
         Тип видеопамяти: GDDR5                             uint8
         Тип видеопамяти: GDDR6                             uint8
         Тип видеопамяти: LPDDR4X                           uint8
         Тип видеопамяти: SMA                               uint8
         Оперативная память > 8                             int64
         benchmark                                          int64
         dtype: object
```

Рис. 33

Типы данных соблюдены, следовательно, набор данных обработан верно и готов к построению модели.

ЗАКЛЮЧЕНИЕ

Итоговая программа верно предобрабатывает набор данных. Сценарий предобработки полностью повторяет сценарий предобработки ВКР Тумановой Веры Дмитриевны.
В ходе предобработки были закодированы номинальные и категориальные значения, удалены пропуски и значения NaN. Выборка проверена на выбросы и произведён корреляционный анализ.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

- *Введение в машинное обучение с помощью Python. Руководство для специалистов по работе с данными. Андреас Мюллер, Сара Гвидо*

- *Python для сложных задач: наука о данных и машинное обучение | Вандер Плас Дж.*

ЛИСТИНГ

```
from math import sqrt
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings

warnings.filterwarnings('ignore')

from scipy import stats

import statsmodels.api as sm
import statsmodels.stats.api as sms
from statsmodels.stats.outliers_influence import OLSInfluence

from statsmodels.stats.outliers_influence import variance_inflation_factor
from scipy.stats import shapiro


# Предобработка (часть 1)

def rename_cols(old_name, one_hot):
    new_cols = {}
    for column in one_hot:
        new_cols[column] = old_name + ": " + column
    return one_hot.rename(columns=new_cols)

def add_cols(df, add_df):
    for column in add_df.columns:
        df[column] = add_df[column]


def preprocessing(path):
    df = pd.read_csv(path)

    print(df.isnull().sum())
    # --- Ненужные для обработки столбцы ---
    useles_columns = ['№', 'Модель (полное название)', 'Модель процессора',
'Год выпуска процессора',
                      'Модель встроенной видеокарты', 'Тип (марка) куллера
материнской платы', 'Браузер для тестов',
                      'Исполнитель', 'Модель дискретной видеокарты']
    df = df.drop(labels=useles_columns, axis=1)

    new_names = {'Объем видеопамяти (Гб)': 'Объем видеопамяти (Гб)
[дискретной]',
                 'Объем видеопамяти (Гб).1': 'Объем видеопамяти (Гб)
[встроенной]'}

    # --- Удаление NaN значений ---
    df = df.rename(columns=new_names)
    df.dropna(subset=['Количество ядер',
                      'Количество логических процессоров (потоков)',
                      'Тактовая частота процессора (ГГц)',
                      'Максимальная тактовая частота (ГГц)', 'Объем кэша L2
процессора (Кб)',
```

```
                      'Объем кэша L3 процессора (Кб)', 'Размер оперативной
памяти (Гб)',
                      'Частота оперативной памяти (МГц)',
                      'Учебный',
                      'Развлекательный',
                      'Эталон'],
              inplace=True)

    num_features = ['Количество ядер', 'Количество логических процессоров
(потоков)',
                    'Тактовая частота процессора (ГГц)', 'HDD (Гб)',
'SSD(Гб)', 'Объем видеопамяти (Гб) [встроенной]',
                    'Объем видеопамяти (Гб) [дискретной]']
    cat_features = ['Вид графического ускорителя (Дискретный / Встроенный)',
'Тип видеопамяти',
                    'Размер оперативной памяти (Гб)']

    # --- Обработка числовых значений ---
    obj_df = df[num_features].copy()
    obj_df['Объем видеопамяти (Гб) [дискретной]'] =
pd.to_numeric(obj_df['Объем видеопамяти (Гб) [дискретной]'],

errors='coerce')
    obj_df['Объем видеопамяти (Гб) [встроенной]'] =
pd.to_numeric(obj_df['Объем видеопамяти (Гб) [встроенной]'],

errors='coerce')
    obj_df = obj_df.fillna({"Объем видеопамяти (Гб) [встроенной]": 0,
                            "Объем видеопамяти (Гб) [дискретной]": 0})
    df[num_features] = obj_df

    # --- Кодирование категориальных значений ---

    # --- Вид графического ускорителя ---
    graph_acc = pd.get_dummies(df['Вид графического ускорителя (Дискретный /
Встроенный)'])
    graph_acc.head()
    graph_acc.columns = graph_acc.columns.str.strip()
    graph_acc.loc[graph_acc['дискретный и встроенный'] == 1, ['встроенный',
'дискретный']] = 1
    graph_acc = graph_acc.drop(['дискретный и встроенный', 'встроенный'],
axis=1)
    graph_acc = rename_cols('Наличие графического ускорителя', graph_acc)
    graph_acc = graph_acc.fillna(0)

    # --- Тип видеопамяти ---
    video_mem = pd.get_dummies(df['Тип видеопамяти'])
    video_mem = rename_cols('Тип видеопамяти', video_mem)

    # --- Размер оперативной памяти ---
    RAM_size = (df['Размер оперативной памяти (Гб)'] // 8 > 0).astype(int)

    df = df.drop(cat_features, axis=1)
    add_cols(df, graph_acc)
    add_cols(df, video_mem)
    df['Оперативная память > 8'] = RAM_size

    # --- Сумма значений Бенчмарк ---
```

```
    benchmark_columns = ['Учебный', 'Развлекательный', 'Эталон']
    benchmark_df = df[benchmark_columns].dropna()
    benchmark_df['sum'] = benchmark_df[benchmark_columns].sum(axis=1)
    df['benchmark'] = benchmark_df['sum']
    return df

df = preprocessing(r"/Users/aleksandr/laptop-performance/data/
laptop_info.csv")

df.head()

plt.figure(figsize=(20, 20))
p = sns.heatmap(df.corr(), annot=True, cmap=plt.cm.PuBu)

numeric_columns = df._get_numeric_data().columns

for col in numeric_columns:
    plt.title(col)
    sns.boxplot(df[col])
    plt.show()

# --- Итоговая проверка на наличие NaN значений ---
df.isnull().sum()

# --- Типы данных ---
df.dtypes
```