Федеральное государственное автономное образовательное учреждение высшего образования «Национальный Исследовательский Университет ИТМО»

ЛАБОРАТОРНАЯ РАБОТА №1
ПРЕДМЕТ «МАТЕМАТИЧЕСКАЯ СТАТИСТИКА»

Вариант 4, 2

Преподаватель: Лимар И. А.
Студент: Румянцев А. А.
Поток: Мат Стат 31.2

Факультет: СУиР
Группа: R3341

Санкт-Петербург
2024

Содержание

1 Задание 1

1.1 Условие

В файле mobile_phones.csv приведены данные о мобильных телефонах. В сколько моделей можно вставить 2 сим-карты, сколько поддерживают 3-G, каково наибольшее число ядер у процессора? Рассчитайте выборочное среднее, выборочную дисперсию, выборочную медиану и выборочную квантиль порядка 2/5, построить график эмпирической функции распределения, гистограмму и box-plot для емкости аккумулятора для всей совокупности и в отдельности для поддерживающих/не поддерживающих Wi-Fi

1.2 Выполнение

Для начала импортируем необходимые библиотеки

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

Листинг 1: Импортирование библиотек

Теперь считаем таблицу по ссылке на представленный гугл-диск в переменную df

```
url='https://drive.google.com/file/d/104rFr9xg9aFmkjx4-hl_X0c5O9q65_EW\
    /view?usp=sharing'
url='https://drive.google.com/uc?id=' + url.split('/')[-2]
df = pd.read_csv(url)
```

Листинг 2: Считывание таблицы

В колонках «dual_sim» и «three_g» наличие или отсутствие параметра определяется единицей или нулем соответственно, следовательно, просуммировав значения в этих столбцах, получим количество моделей с наличием данных параметров. Для ядер просто выведем максимум из столбца, используя методы библиотеки `pandas`

```
# how many models can you insert 2 SIM cards into?
dual_sim_count = df['dual_sim'].sum()
print(f'dual_sim_count={dual_sim_count}')

# how many models support 3-G?
three_g_count = df['three_g'].sum()
print(f'three_g_count={three_g_count}')

# what is the highest number of cores a processor has?
max_cores = df['n_cores'].max()
print(f'max_cores={max_cores}')
```

Листинг 3: Код на ответы на первые три вопроса

Получим следующий вывод в консоль

```
dual_sim_count=1019
three_g_count=1523
max_cores=8
```

Листинг 4: Вывод в консоль: ответы на первые три вопроса

Для расчета необходимых характеристик я написал отдельный метод, куда достаточно передать выборку и ее именование для удобного вывода в консоль. Используем методы библиотеки `pandas` – `mean` посчитает выборочное среднее, `var` выборочную дисперсию, `median` выборочную медиану и `quantile` с параметром `q=2/5` квантиль порядка 2/5

```
# calculating the main values
def calculate_print_values(df: pd.Series, name: str):
    mean = df.mean()
    print(f'mean_{name}={mean}')

    var = df.var()
    print(f'var_{name}={var}')

    median = df.median()
    print(f'median_{name}={median}')

    quantile_2d5 = df.quantile(q=2/5)
    print(f'quantile_2/5_{name}={quantile_2d5}')
```

Листинг 5: Код для подсчета основных характеристик

Зададим сразу все три выборки – всю, только модели с наличием Wi-Fi и только с отсутствием. Библиотека `pandas` позволяет удобно отбирать нужные значения – для выборок с Wi-Fi оставляем все строки, для которых выполняется равенство единице или нулю. Добавим для удобства метод разделитель, который будем вызывать между выборками. Вызовем подсчитывающий метод три раза для трех выборок

```
# common separator between unrelated outputs
def print_separate():
    print('----------------')

print_separate()

# the entire sample
all_battery = df['battery_power']
calculate_print_values(all_battery, name='all_battery')

print_separate()

# selection with the condition of wifi availability
wifi_table = df[df['wifi']==1]
wifi_battery = wifi_table['battery_power']
calculate_print_values(wifi_battery, name='wifi_battery')

print_separate()

# selection with the condition of wifi unavailability
nowifi_table = df[df['wifi']==0]
no_wifi_battery = nowifi_table['battery_power']
calculate_print_values(no_wifi_battery, name='no_wifi_battery')
```

Листинг 6: Подготовка для удобного и быстрого получения результатов

С заданными ранее параметрами получаем следующий вывод в консоль

```
----------------
mean_all_battery=1238.5185
var_all_battery=193088.35983766883
median_all_battery=1226.0
quantile_2/5_all_battery=1076.0
----------------
```

```
mean_wifi_battery=1234.9043392504932
var_wifi_battery=190296.40051422242
median_wifi_battery=1233.0
quantile_2/5_wifi_battery=1077.8000000000002
----------------
mean_no_wifi_battery=1242.235294117647
var_no_wifi_battery=196128.43798148702
median_no_wifi_battery=1222.0
quantile_2/5_no_wifi_battery=1076.0
----------------
```

Листинг 7: Вывод в консоль: посчитанные основные характеристики

Результаты получились хорошие. Теперь построим графики в соответствии с заданием. Напишем метод, который принимает выборку и название графика – таким образом, достаточно будет вызвать метод для каждой выборки и получить все графики. Используем библиотеку `matplotlib` для отрисовки. Для подсчета необходимых данных все также нужна библиотека `pandas`. Для построения графика эмпирической функции распределения находим по сортированным данным без сохранения индексов ключ-значение, где ключ – `battery_power`, значение – вероятность встретить именно такую `battery_power`. Однако для графика нужны не сами значения, а кумулятивные суммы. Для гистограммы определяем количество интервалов правилом Стёрджеса: $n = 1 + \log_2 N$ и округляем вниз

```python
# plotting basic graphs
def show_graphs(df: pd.Series, name='sample'):
    # the resulting axis will be labeled 0, 1, ..., n - 1
    # (not saving original indexes)
    sorted_ = df.sort_values(ignore_index=True)

    # normalize for proportions (probabilities) instead of freqs
    # sorting by DataFrame column values (not by freqs)
    idx_prob = sorted_.value_counts(normalize=True, sort=False)

    # parsing x & y then cumsum for distribution function
    plt.plot(idx_prob.index, idx_prob.values.cumsum())
    plt.title(f'Empirical distribution function of {name}')
    plt.xlabel('battery_power')
    plt.ylabel('probability')
    plt.grid()
    plt.gcf().set_size_inches(10, 5)
    plt.show()

    # Sturges' rule
    n = np.int64(np.floor(1+3.322*np.log10(df.shape[0])))
    plt.hist(df, bins=n)
    plt.title(f'Histogram of {name}')
    plt.xlabel('battery_power')
    plt.ylabel('count')
    plt.grid()
    plt.gcf().set_size_inches(10, 5)
    plt.show()

    plt.boxplot(df)
    plt.title(f'Boxplot of {name}')
    plt.ylabel('battery_power')
    plt.grid()
    plt.gcf().set_size_inches(10, 5)
    plt.show()
```

Листинг 8: Код для построения необходимых графиков

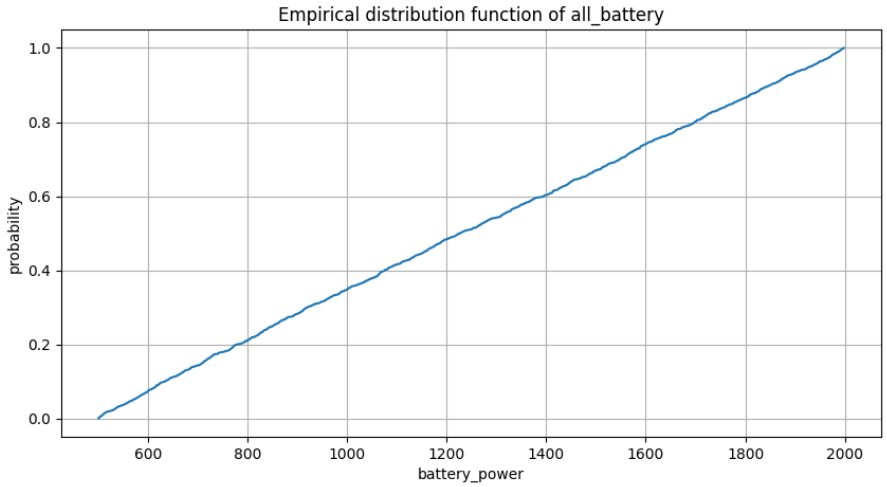

Рис. 1: График эмпирической функции распределения для всей выборки

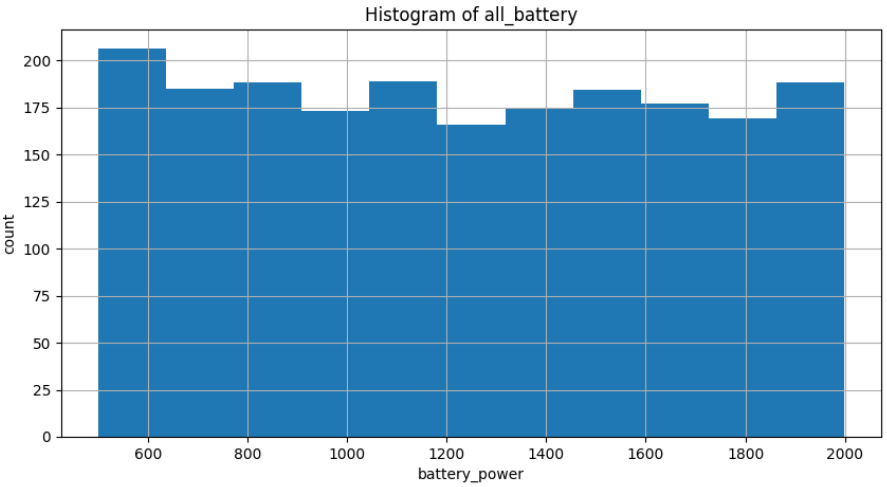

Рис. 2: Гистограмма для всей выборки

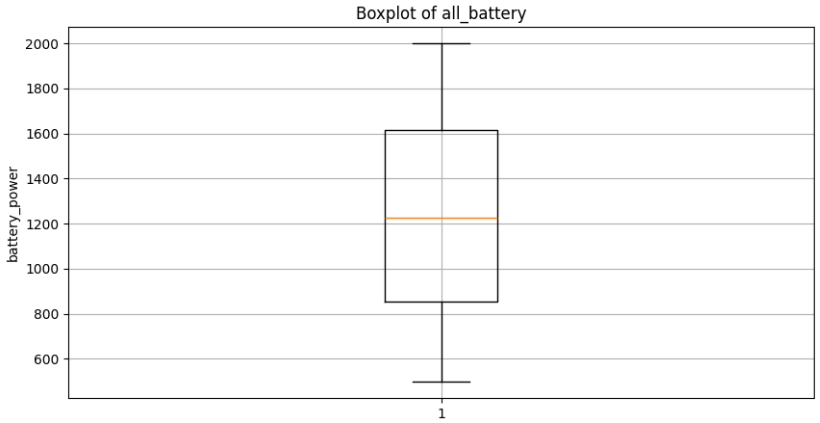

Рис. 3: Ящик с усами для всей выборки

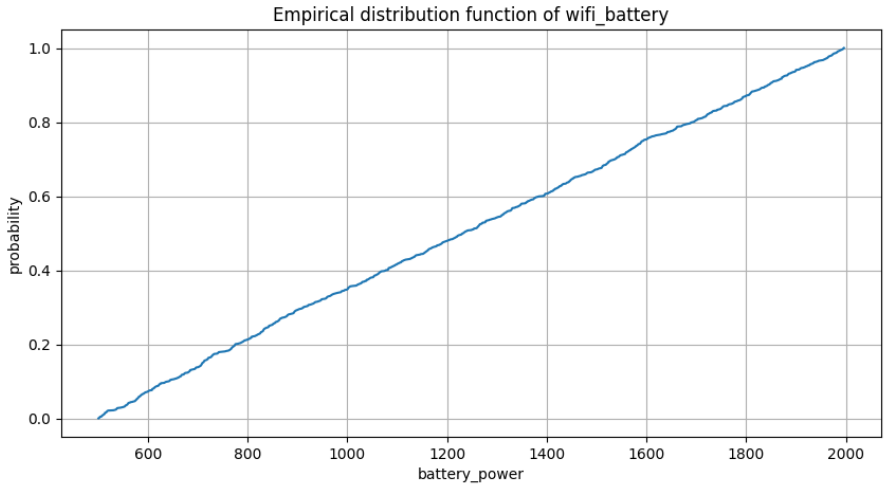

Рис. 4: График эмпирической функции распределения для выборки моделей с Wi-Fi

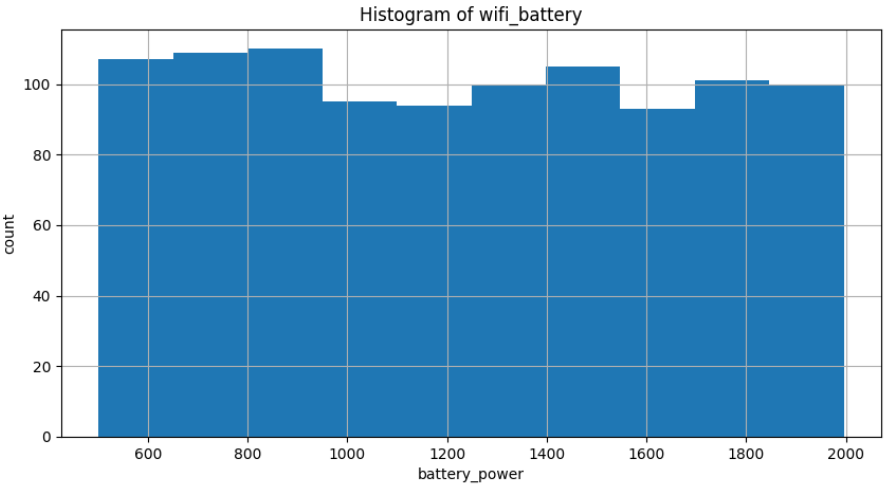

Рис. 5: Гистограмма для выборки моделей с Wi-Fi

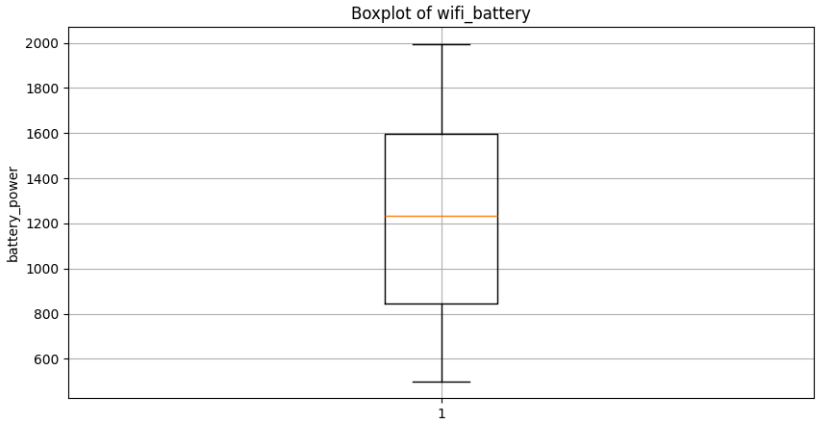

Рис. 6: Ящик с усами для выборки моделей с Wi-Fi

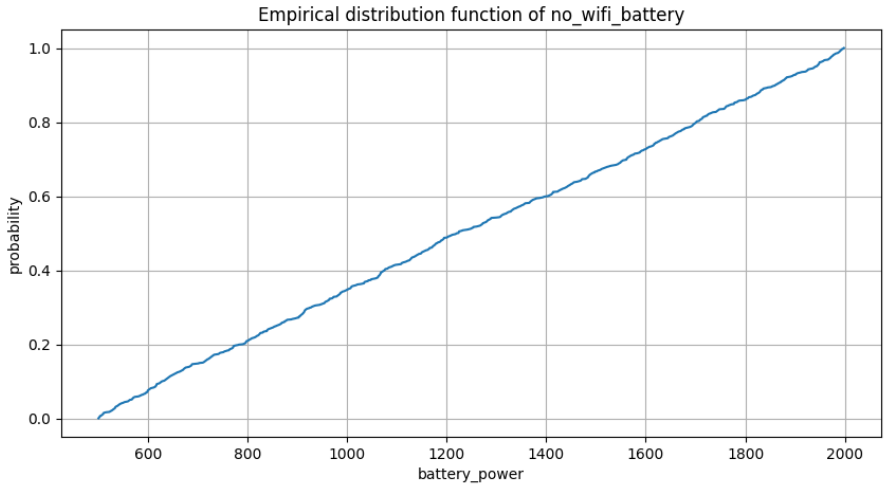

Рис. 7: График эмпирической функции распределения для выборки моделей без Wi-Fi

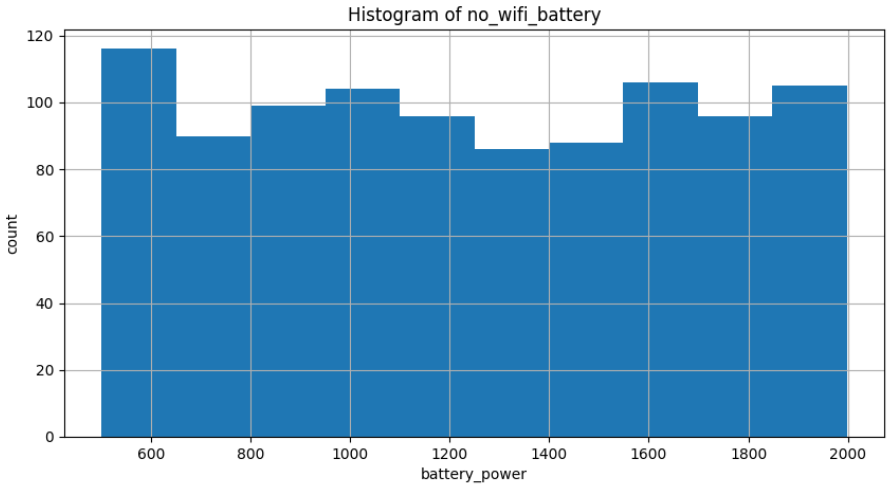

Рис. 8: Гистограмма для выборки моделей без Wi-Fi

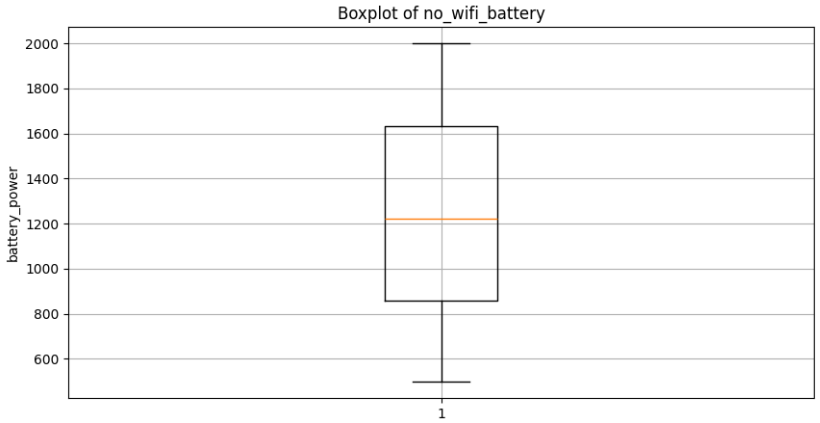

Рис. 9: Ящик с усами для выборки моделей без Wi-Fi