|  |  |  |
| --- | --- | --- |
|  | |  |
| МИНОБРНАУКИ РОССИИ | |  |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **"МИРЭА - Российский технологический университет"**  **РТУ МИРЭА** | |  |
|  | Институт информационных технологий (ИТ) | |
|  | Кафедра прикладной математики | |

|  |  |
| --- | --- |
| **Отчет по практической работе №3.** | |
| **по дисциплине** | |
| **«Технологии и инструментарий анализа больших данных»** | |
|  | |
| Выполнил студент группы ИКБО- 22-20 | Гологузов К.А. |
| Принял | Парамонов А.А. |

Москва 2023

Решение

1. Загрузить данные из файла “insurance.csv”.

2. С помощью метода describe() посмотреть статистику по данным. Сделать выводы.

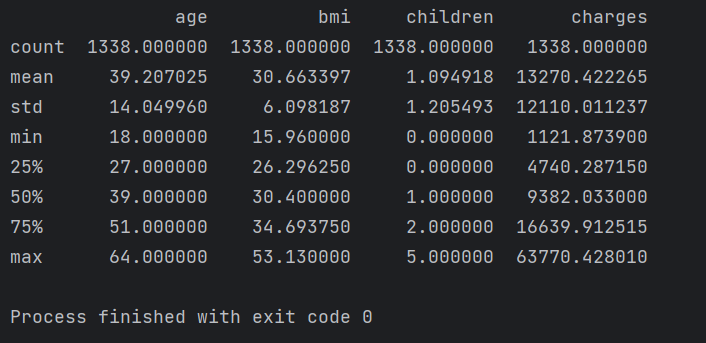
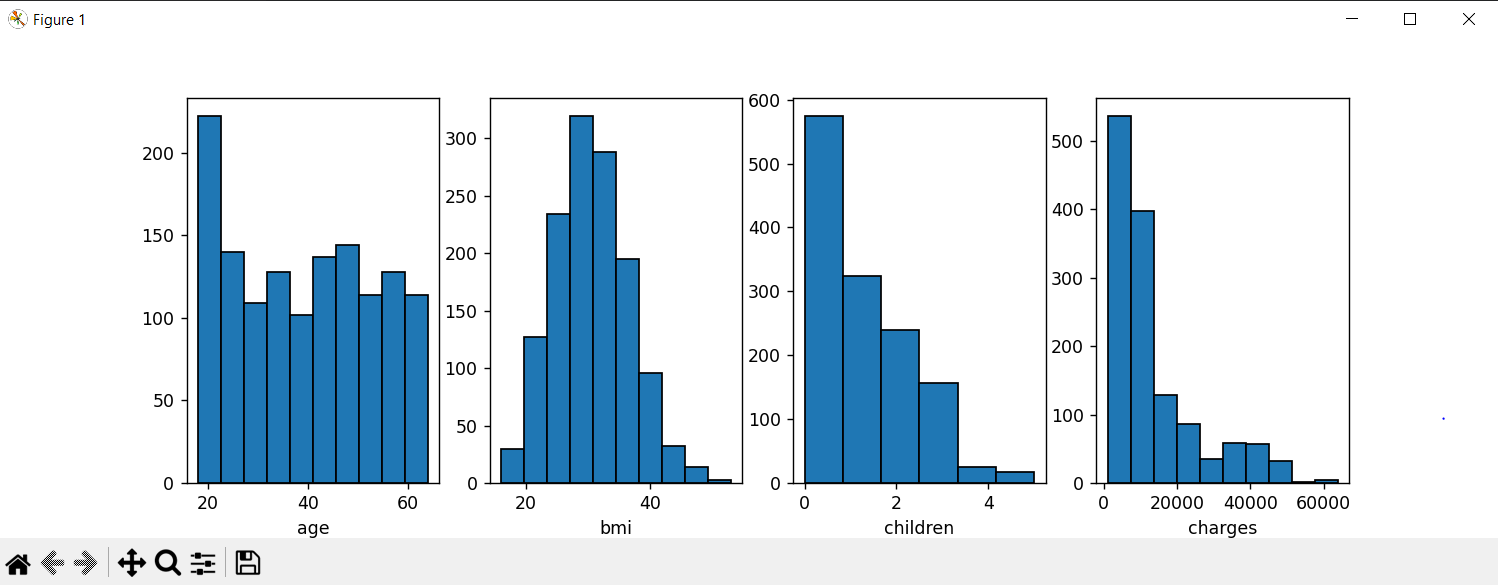


Рис. 1 – Результат describe()

Метод describe() показывает описательную статистику числовых данных файла. Count – количество строк, mean – среднее значение, std – среднеквадратическое отклонение, min – минимальное значение, 25% - четверть значений данных лежит в диапазоне от min do 25%, аналогично 50% и 75%, max – максимальное значение.

3. Построить гистограммы для числовых показателей. Сделать выводы.

Рис. 2 – Гистограммы значений

4. Найти меры центральной тенденции и меры разброса для индекса массы тела (bmi) и расходов (charges). Отобразить результаты в виде текста и на гистограммах (3 вертикальные линии). Добавить легенду на графики. Сделать выводы.

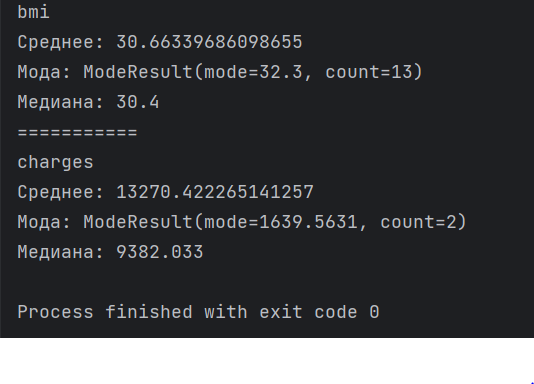


Рис. 3 – Результаты в виде текста

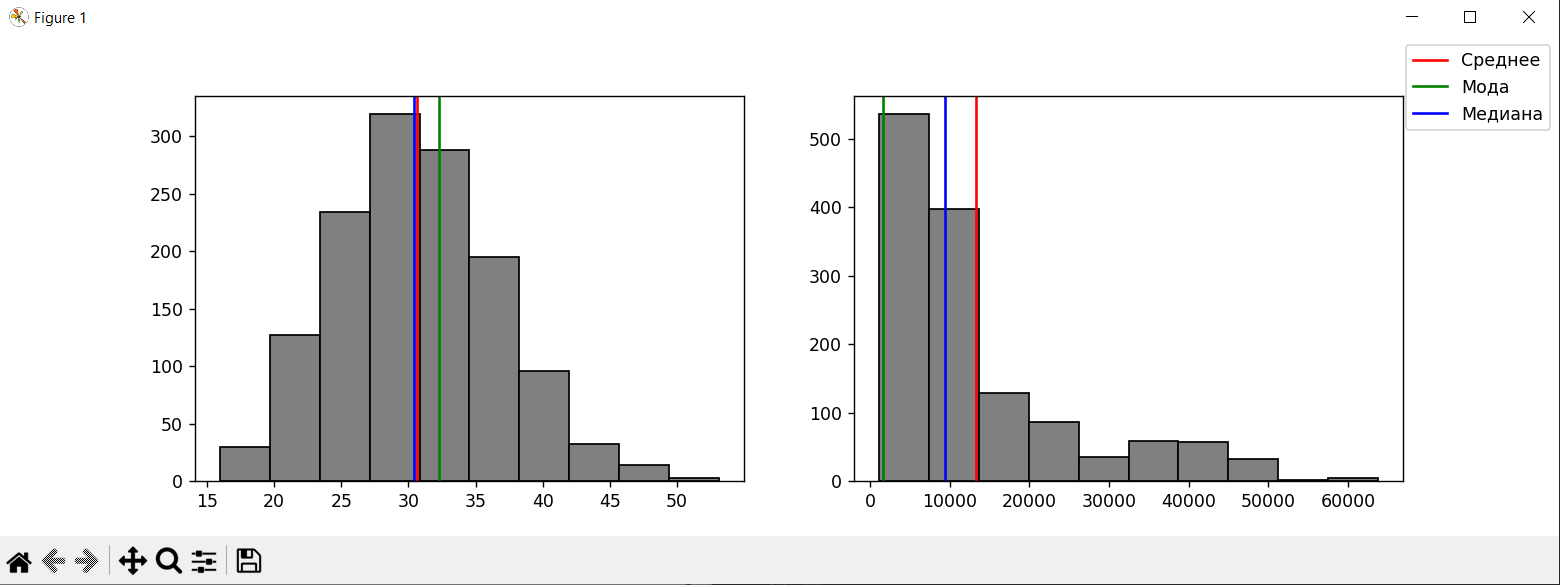


Рис. 4 – Гистограммы с легендой

5. Построить box-plot для числовых показателей. Названия графиков должны соответствовать названиям признаков. Сделать выводы.

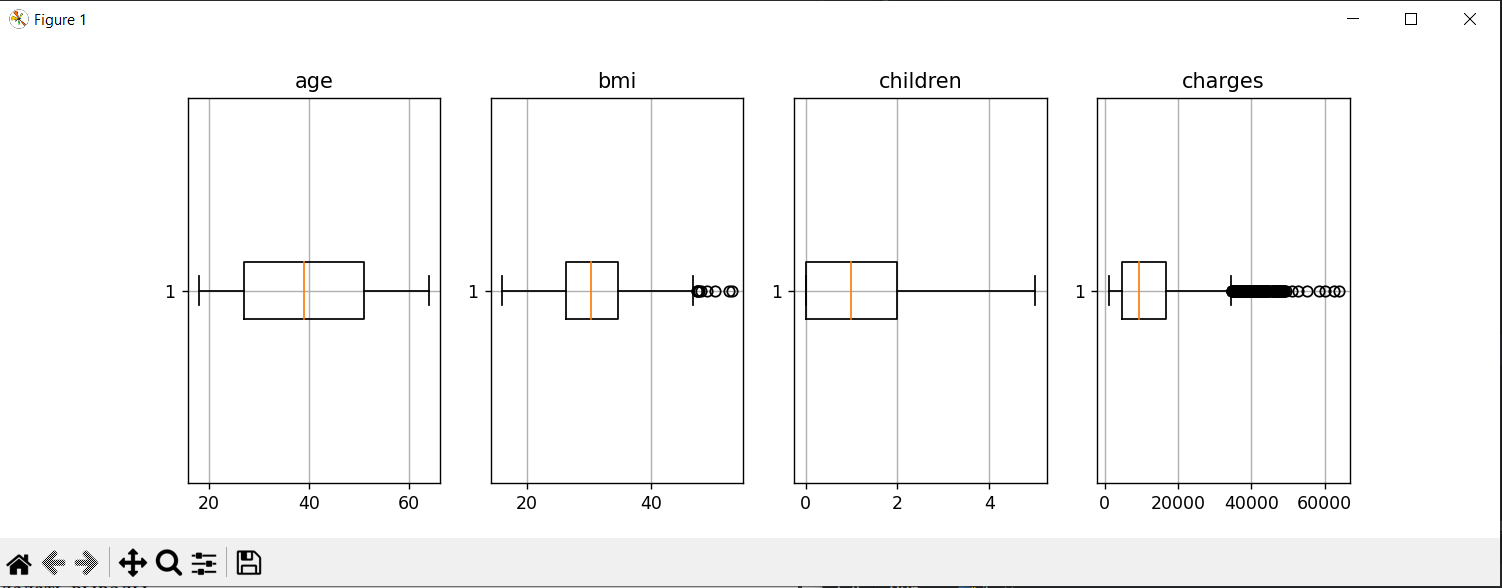


Рис. 5 – box-plot диаграммы

Оранжевая линия – медиана или Q2. Границы коробки – это квартили Q1 и Q3. Точки за пределами “усов” – выбросы.

6. Используя признак charges или imb, проверить, выполняется ли центральная предельная теорема. Использовать различные длины выборок n. Количество выборок = 300. Вывести результат в виде гистограмм. Найти стандартное отклонение и среднее для полученных распределений. Сделать выводы.

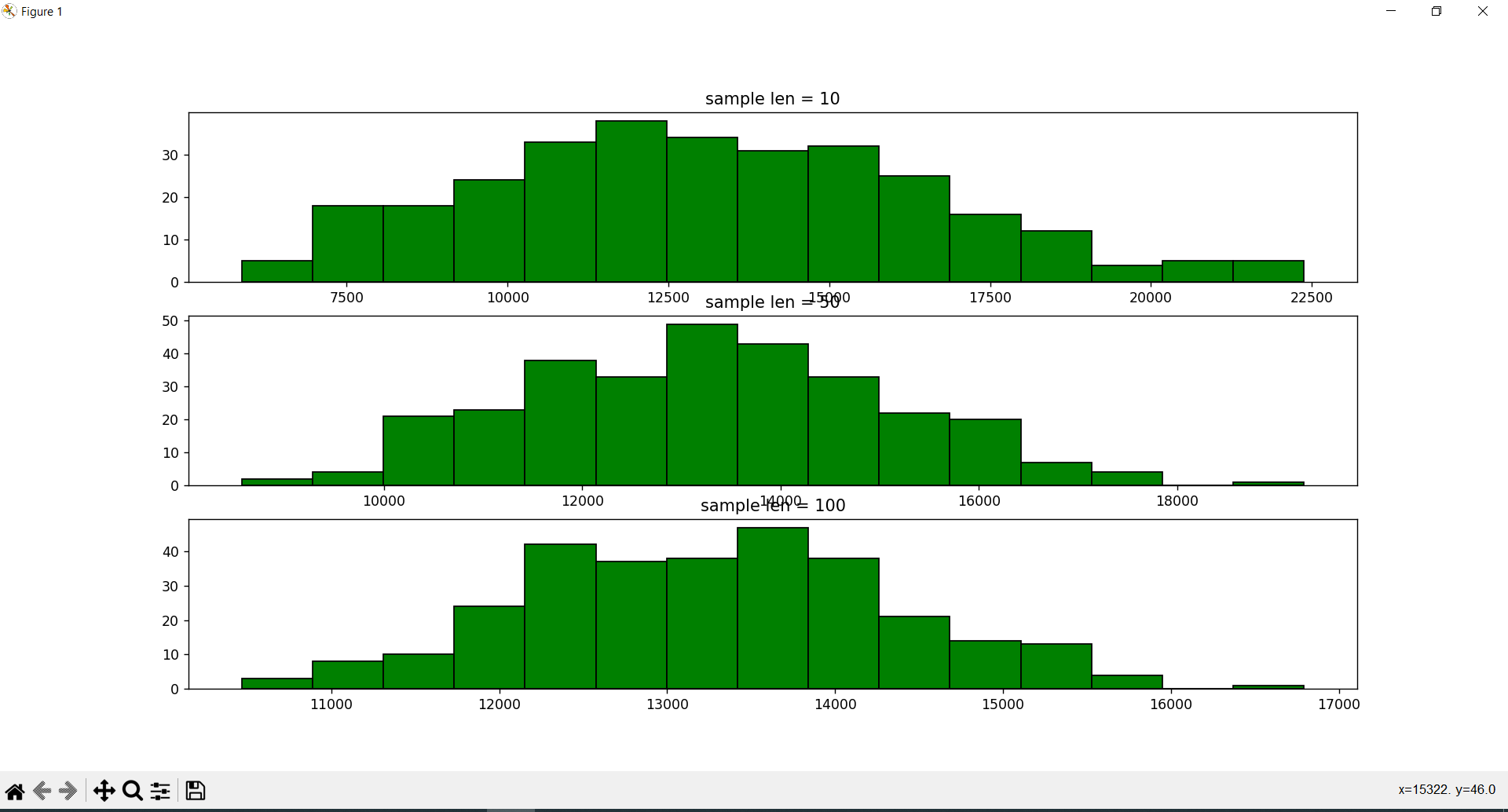


Рис. 6 – Центральная предельная теорема

7. Построить 95% и 99% доверительный интервал для среднего значения расходов и среднего значения индекса массы тела

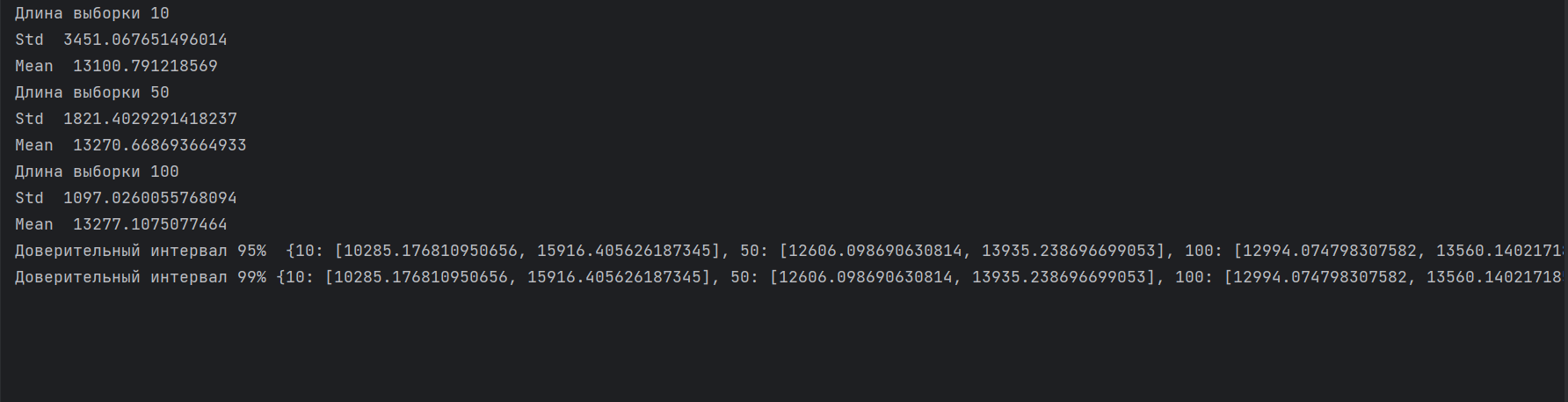


Рис. 7 – Доверительный интервал

8. Проверить распределения следующих признаков на нормальность: индекс массы тела, расходы. Сформулировать нулевую и альтернативную гипотезы. Для каждого признака использовать KS-тест и q-q plot. Сделать выводы на основе полученных p-значений.

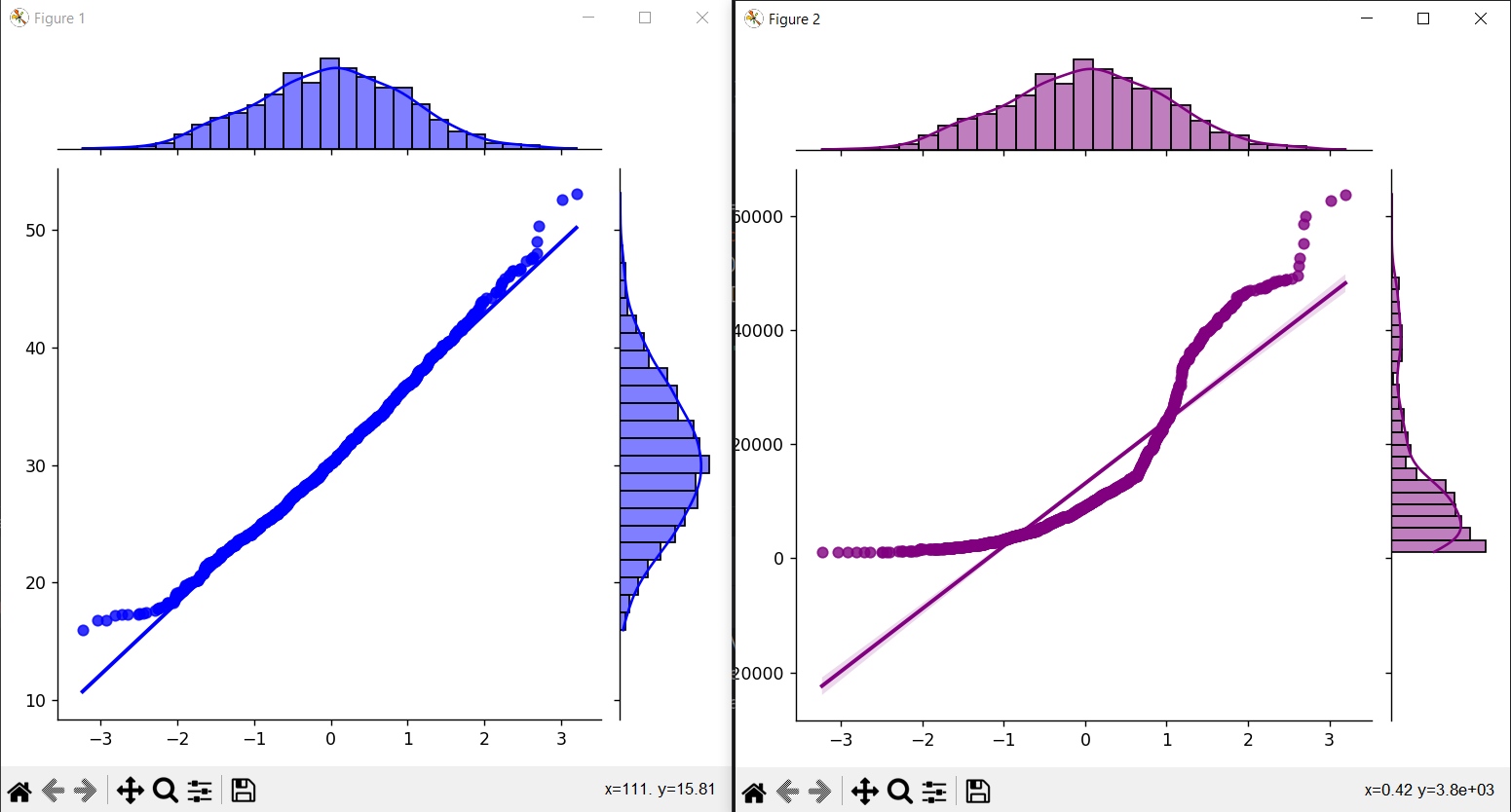


Рис. 8 – график Q-Q plot

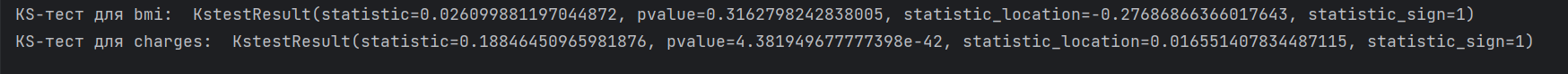


Рис. 9 – Результаты тестов

Судя по тестам Колмогорова, p-значение для первой выборки больше 0.05, в то время как p-значение для второй выборки намного меньше 0.05, следовательно нулевая гипотеза отвергается и вторая выборка не имеет нормального распределения.

9. Загрузить данные из файла “ECDCCases.csv”.

10. Проверить в данных наличие пропущенных значений. Вывести количество пропущенных значений в процентах. Удалить два признака, в которых больше всех пропущенных значений. Для оставшихся признаков обработать пропуски: для категориального признака использовать заполнение значением по умолчанию (например, «other»), для числового признака использовать заполнение медианным значением. Показать, что пропусков больше в данных нет. 11. Посмотреть статистику по данным

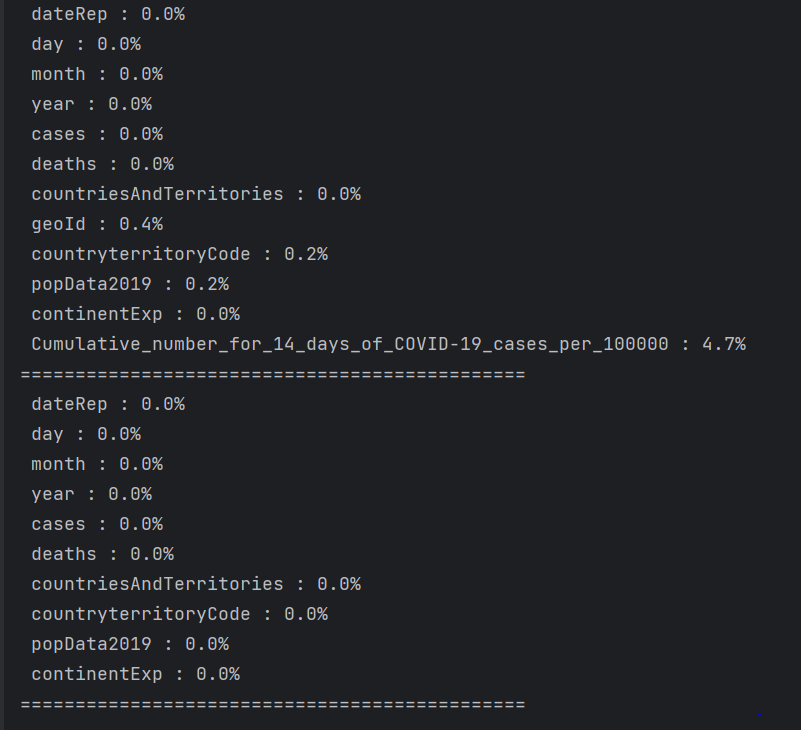


Рис. 10 – Обработка пустых значений

11. Посмотреть статистику по данным, используя describe(). Сделать выводы о том, какие признаки содержат выбросы. Посмотреть, для каких стран количество смертей в день превысило 3000 и сколько таких дней было.

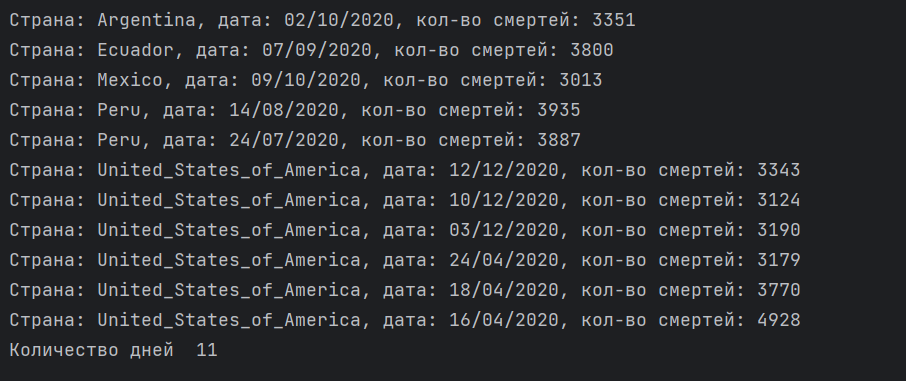


Рис. 11 – Количество дней

12. Найти дублирование данных. Удалить дубликаты.



Рис. 12 – Дубликаты

13. Загрузить данные из файла “bmi.csv”. Взять оттуда две выборки. Одна выборка – это индекс массы тела людей c региона northwest, вторая выборка – это индекс массы тела людей с региона southwest. Сравнить средние значения этих выборок, используя t-критерий Стьюдента. Предварительно проверить выборки на нормальность (критерий ШопироУилка) и на гомогенность дисперсии (критерий Бартлетта).

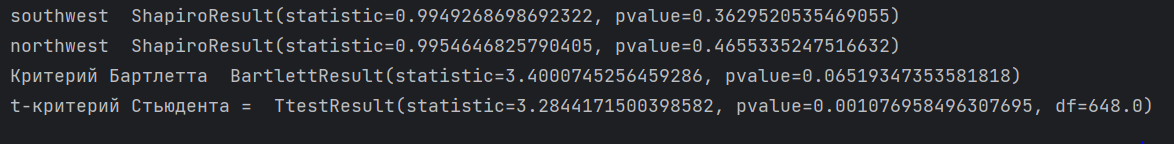


Рис. 13 – Проверки выборок

14. Кубик бросили 600 раз, получили следующие результаты:

|  |  |
| --- | --- |
| **N** | **Кол-во выпадений** |
| 1 | 97 |
| 2 | 98 |
| 3 | 109 |
| 4 | 95 |
| 5 | 97 |
| 6 | 104 |

С помощью критерия Хи-квадрат проверить, является ли полученное распределение равномерным. Использовать функцию scipy.stats.chisquare().



Рис. 14 – Проверка распределения бросков кубика

15. С помощью критерия Хи-квадрат проверить, являются ли переменные зависимыми.

Использовать функцию scipy.stats.chi2\_contingency(). Влияет ли семейное положение на занятость?

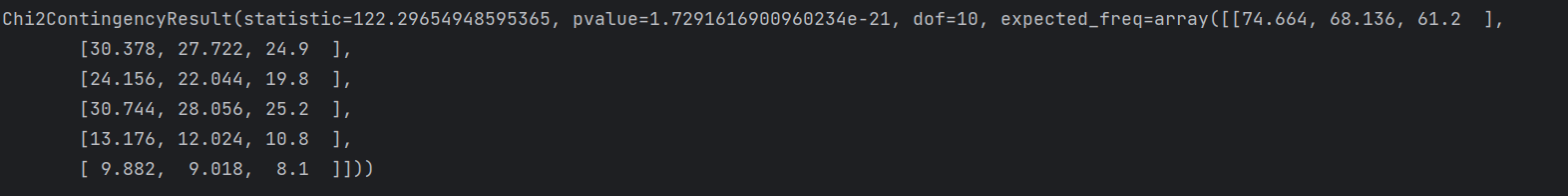


Рис. 15 – Проверка переменных

ПРИЛОЖЕНИЕ

Приложение 1 – 1.py

import matplotlib.pyplot as plt  
import numpy as np  
import pandas as pd  
import seaborn as sns  
import scipy.stats as sts  
import plotly.graph\_objs as go  
import random  
  
  
df = pd.read\_csv("./insurance.csv")  
df = df.drop("sex", axis=1)  
df = df.drop("smoker", axis=1)  
df = df.drop("region", axis=1)  
columns = df.columns.values.tolist()  
def bars():  
 fig, ax = plt.subplots(1, len(columns), figsize=(12,4))  
 for column in columns:  
 ind = columns.index(column)  
 if ind == 2:  
 ax[ind].hist(df[column].values, edgecolor='black', bins=6)  
 ax[ind].set\_xlabel(column)  
 continue  
 ax[ind].hist(df[column], edgecolor='black')  
 ax[ind].set\_xlabel(column)  
 print(column, ind)  
 plt.show()  
 return  
  
  
def box():  
 fig, ax = plt.subplots(1, len(columns), figsize=(12, 4))  
 for col in columns:  
 ind =columns.index(col)  
 print(ind)  
 ax[ind].boxplot(df[col], vert=False)  
 ax[ind].set\_title(col)  
 ax[ind].grid()  
 plt.show()  
  
  
def theorem():  
 n = 300  
 len\_samples = [10, 50, 100]  
 fig, ax = plt.subplots(len(len\_samples), figsize=(30, 8))  
 means = []  
 dist95 = {}  
 dist99 = {}  
 for i in range(len(len\_samples)):  
 sample\_means = []  
 for k in range(n):  
 sample = np.random.choice(df["charges"], size=len\_samples[i], replace=True)  
 sample\_means.append(np.mean(sample))  
 means.append(sample\_means)  
 ax[i].hist(sample\_means, bins=15, edgecolor="black", color= "green")  
 ax[i].set\_title("sample len = " + str(len\_samples[i]))  
 sample\_means = np.array(sample\_means)  
 print("Длина выборки", len\_samples[i])  
 print("Std ", sample\_means.std())  
 print("Mean ", np.mean(sample\_means))

Продолжение – 1.py

dist95[len\_samples[i]] = [  
 np.mean(sample\_means) - 1.96 \* sample\_means.std() / np.sqrt(len\_samples[i]),  
 np.mean(sample\_means) + 1.96 \* sample\_means.std() / np.sqrt(len\_samples[i])  
 ]  
 dist99[len\_samples[i]] = [  
 np.mean(sample\_means) - 2.58 \* sample\_means.std() / np.sqrt(len\_samples[i]),  
 np.mean(sample\_means) + 2.58 \* sample\_means.std() / np.sqrt(len\_samples[i])  
 ]  
 print("Доверительный интервал 95% ", dist99)  
 print("Доверительный интервал 99%", dist99)  
 plt.show()  
  
def normal\_distribution():  
 normal\_quantiles = np.random.normal(0, 1, 1338)  
 x = np.sort(normal\_quantiles)  
 y\_bmi = np.sort(df['bmi'])  
 y\_charges = np.sort(df['charges'])  
 sns.jointplot(x=x, y=y\_bmi, kind="reg", truncate=True, color="blue")  
 sns.jointplot(x=x, y=y\_charges, kind="reg", truncate=True, color="purple")  
 test\_bmi = (df['bmi'] - np.mean(df['bmi'])) / df['bmi'].std()  
 test\_charges = (df['charges'] - np.mean(df['charges'])) / df['charges'].std()  
 print("KS-тест для bmi: ", sts.kstest(test\_bmi, 'norm'))  
 print("KS-тест для charges: ", sts.kstest(test\_charges, 'norm'))  
 plt.show()  
  
  
normal\_distribution()

Приложение 2 – 4.py

import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import scipy.stats as sts  
  
  
df = pd.read\_csv("insurance.csv")  
bmi = df["bmi"].values  
charges = df["charges"].values  
bmi\_mean = np.mean(bmi)  
charges\_mean = np.mean(charges)  
bmi\_moda = sts.mode(bmi)  
charges\_moda = sts.mode(charges)  
bmi\_med = np.median(bmi)  
charges\_med = np.median(charges)  
  
print("bmi")  
print("Среднее:", bmi\_mean)  
print("Мода:", bmi\_moda)  
print("Медиана:", bmi\_med)  
print("===========")  
print("charges")  
print("Среднее:", charges\_mean)  
print("Мода:", charges\_moda)  
print("Медиана:", charges\_med)  
  
def histogramms():  
 fig, ax = plt.subplots(1, 2, figsize=(12,4))  
 ax[0].hist(bmi, edgecolor='black', color="gray")  
 ax[0].axvline(x=bmi\_mean, color="red")  
 ax[0].axvline(x=bmi\_moda[0], color="green")  
 ax[0].axvline(x=bmi\_med, color="blue")  
 ax[1].hist(charges, edgecolor="black", color="grey")  
 ax[1].axvline(x=charges\_mean, color="red")  
 ax[1].axvline(x=charges\_moda[0], color="green")  
 ax[1].axvline(x=charges\_med, color="blue")  
 fig.legend(["Среднее", "Мода", "Медиана"])  
 plt.show()  
  
  
def box\_plots():  
 pass

Приложение 3 – part 2\main.py

import pandas as pd  
import numpy as np  
import scipy.stats as sts  
  
  
def missing\_val(df):  
 for column in df.columns:  
 missing = np.mean(df[column].isna() \* 100)  
 print(f" {column} : {round(missing, 1)}%")  
 print("==============================================")  
 return  
  
  
def find\_duplicates(df):  
 a = sum(df.duplicated())  
 print("Дубликатов: ", a)  
 df.drop\_duplicates()  
 return  
  
  
def deaths(df):  
 a = 0  
 for i in df.values:  
 if i[5] > 3000:  
 a += 1  
 print(f"Страна: {i[6]}, дата: {i[0]}, кол-во смертей: {i[5]}")  
 print("Количество дней ", a)  
 return  
  
  
def main():  
 df = pd.read\_csv("ECDCCases.csv")  
 # missing\_val(df)  
 df.drop(columns=["geoId", 'Cumulative\_number\_for\_14\_days\_of\_COVID-19\_cases\_per\_100000'], inplace=True)  
 median = np.mean(df['popData2019'])  
 df['popData2019'].fillna(median, inplace=True)  
 df['countryterritoryCode'].fillna("other", inplace=True)  
 # missing\_val(df)  
 # print(df.info())  
 # deaths(df)  
 find\_duplicates(df)  
 return  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Приложение 4 – part 3\main.py

import pandas as pd  
import numpy as np  
import scipy.stats as sts  
  
  
def sample(df):  
 south = []  
 north = []  
 for i in df.values:  
 if i[1] == 'northwest':  
 north.append(i[0])  
 else:  
 south.append(i[0])  
 south = np.array(south)  
 north = np.array(north)  
 print("southwest ", sts.shapiro(south))  
 print("northwest ", sts.shapiro(north))  
 print("Критерий Бартлетта ", sts.bartlett(south, north))  
 print("t-критерий Стьюдента = ", sts.ttest\_ind(south, north))  
 # print("t-критерий Стьюдента = ", t\_criter\_south)  
 return  
  
  
def cube():  
 kybik = [97, 98, 109, 95, 97, 104]  
 print(sts.chisquare(kybik))  
 print("===============================")  
 data = pd.DataFrame({'Женат': [89, 17, 11, 43, 22, 1],  
 'Гражданский брак': [80, 22, 20, 35, 6, 4],  
 'Не состоит в отношениях': [35, 44, 35, 6, 8, 22]})  
 data.index = ['Полный рабочий день', 'Частичная занятость', 'Временно не работает',  
 'На домохозяйстве', 'На пенсии', 'Учёба']  
 print(sts.chi2\_contingency(data))  
 return  
  
  
def main():  
 df = pd.read\_csv("bmi.csv")  
 # print(df.head())  
 # sample(df)  
 cube()  
 return  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()