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

|  |  |
| --- | --- |
| **Отчет по третьей практической работе.** | |
| **по дисциплине** | |
| **«Технологии и инструментарий анализа больших данных»** | |
|  | |
| Выполнил студент группы ИКБО- 22-20 | Никулин К.В. |
| Принял | Парамонов А.А. |

Москва 2023 г.

Оглавление

[1 ИНФОРМАЦИЯ О ДАТАСЕТЕ YOUTUBE STATISTIC 6](#_Toc146809030)

[2 СТОЛБЧАТАЯ ДИАГРАММА 8](#_Toc146809031)

[3 КРУГОВАЯ ДИАГРАММА 9](#_Toc146809032)

[4 ЛИНЕЙНЫЕ ГРАФИКИ 10](#_Toc146809033)

[5 ВИЗУАЛИЗАЦИЯ МНОГОМЕРНЫХ ДАННЫХ T-SNE 11](#_Toc146809034)

[6 ВИЗУАЛИЗАЦИЯ МНОГОМЕРНЫХ ДАННЫХ UMAP 13](#_Toc146809035)

[7 ПРИЛОЖЕНИЕ 16](#_Toc146809036)

# ИНФОРМАЦИЯ О ДАТАСЕТЕ INSURANCE

Датасет insurance содержит данные о, в нем содержится 7 параметров, по 1338 записей.

Описание используемых параметров:

* age – информация о количестве подписчиков;
* sex – информация о том, на каком месте в топе находится канал ютубера;
* bmi – информация о общем количестве просмотров канала;
* children – наибольшая сумма заработка в год;
* smoker – ;
* region – ;
* charges – .

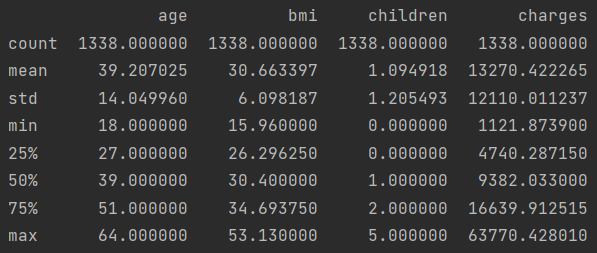


Рисунок 1 – Информация о датасете

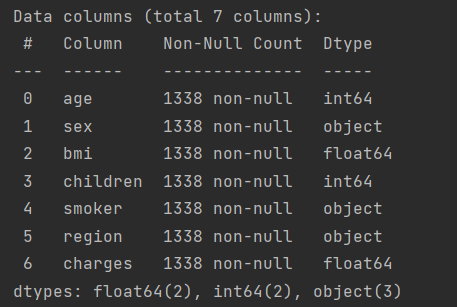


Рисунок 2 – Параметры датасета

Исследуем форму распределения выборки, с помощью гистограммы частот. По оси абсцисс откладывается значение переменной, а по оси ординат указывается как часто значение этой переменной встречается на определенном интервале.

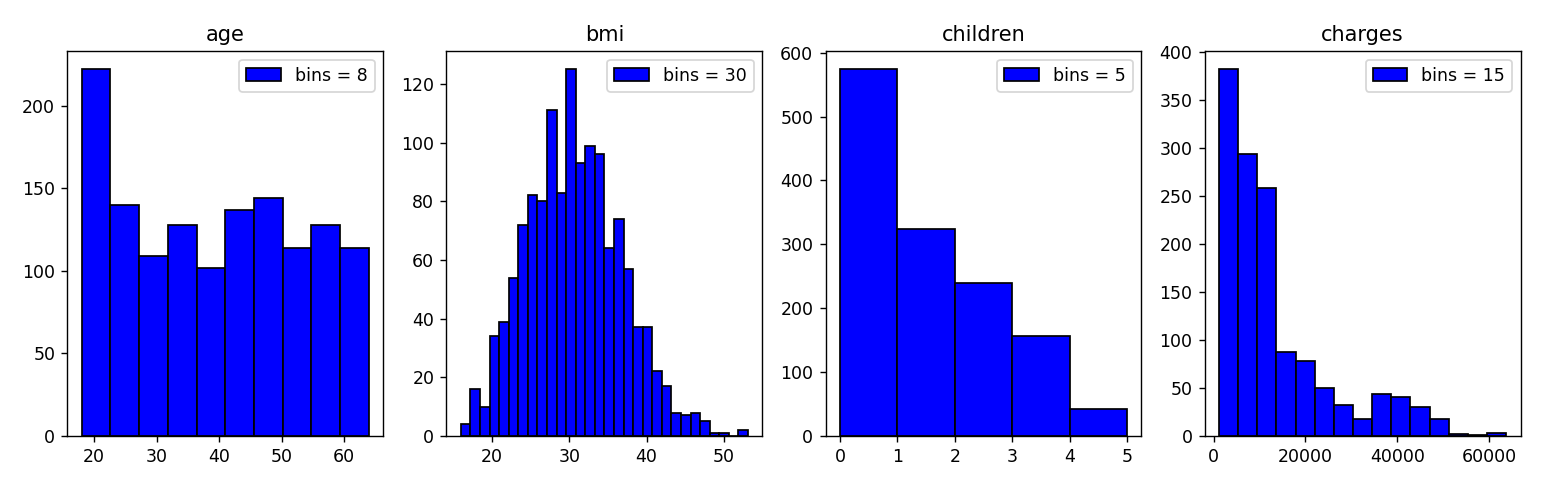


Рисунок 3 – Гистограммы частот

− Меры центральной тенденции:

* **Мода**. Это значение, которое наиболее часто встречается в выборке.
* **Медиана**. Для нечетного количества элементов медиана равна центральному элементу в отсортированном массиве.

Для четного количества элементов медиана равна среднему двух центральных элементов в отсортированном массиве.

* **Среднее**. Сумма значений всех элементов выборки, деленное на их количество.

− Меры изменчивости:

• **Размах**. Разница между максимальным и минимальным значением выборки. Очень простая мера, но она использует только два значения из всей выборки. Правильнее использовать каждое значение из выборки для расчета изменчивости данных.

• **Стандартное** **отклонение**. Это оценка для выборки. Считается, что стандартное отклонение выборки немного недооценивается, поэтому ее чуть-чуть увеличивают, делив на n-1, а не на n, как для генеральной совокупности. Для генеральной совокупности такой показатель называется среднеквадратическим отклонением. Этот показатель позволяет оценить, как сильно меняются данные относительно их среднего. Стандартное отклонение не устойчиво к выбросам.

• **Межквартильный** **размах** **(IQR) .** Для всех выборок существуют такие отсечки, которые называются «квартили», их всего три: Q1, Q2 и Q3. Межквартильный размах – разность между Q3 (75%) и Q1 (25%), это ширина интервала, который содержит 50% данных. Это метрика полезна для описания данных, она устойчива к выбросам.

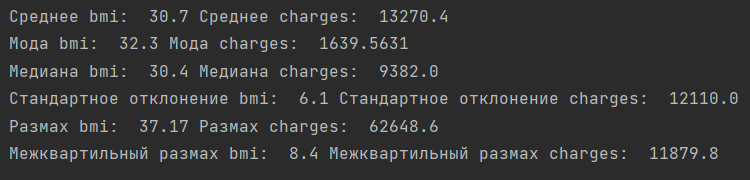


Рисунок 4 – Меры центральной тенденции и меры разброса для индекса массы тела (bmi) и расходов (charges)

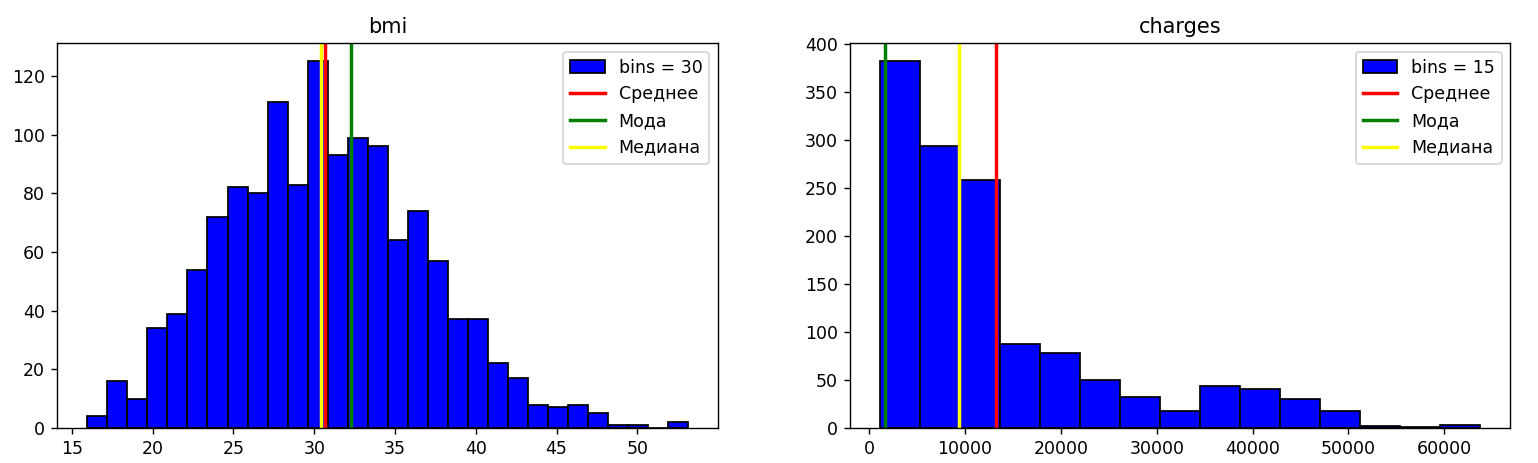


Рисунок 5 – Меры центральной тенденции и меры разброса для индекса массы тела (bmi) и расходов (charges)

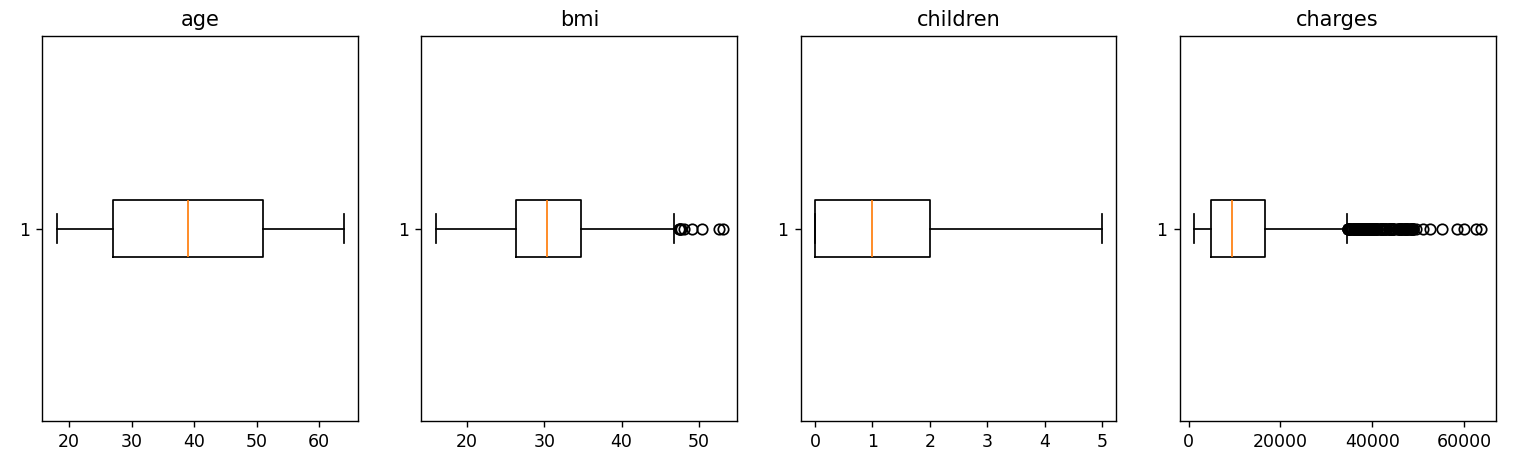


Рисунок 6 – box-plot для числовых показателей

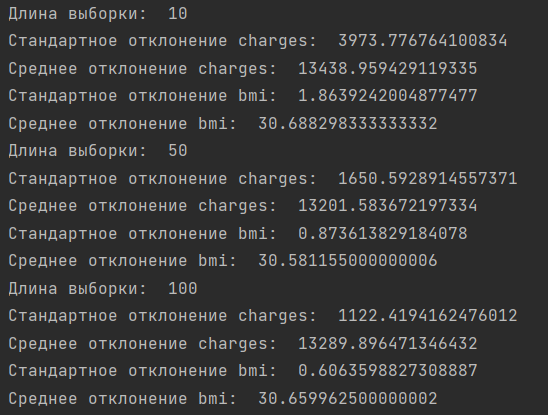


Рисунок 7 – Стандартное и среднее отклонение для признаков (charges) и (bmi)

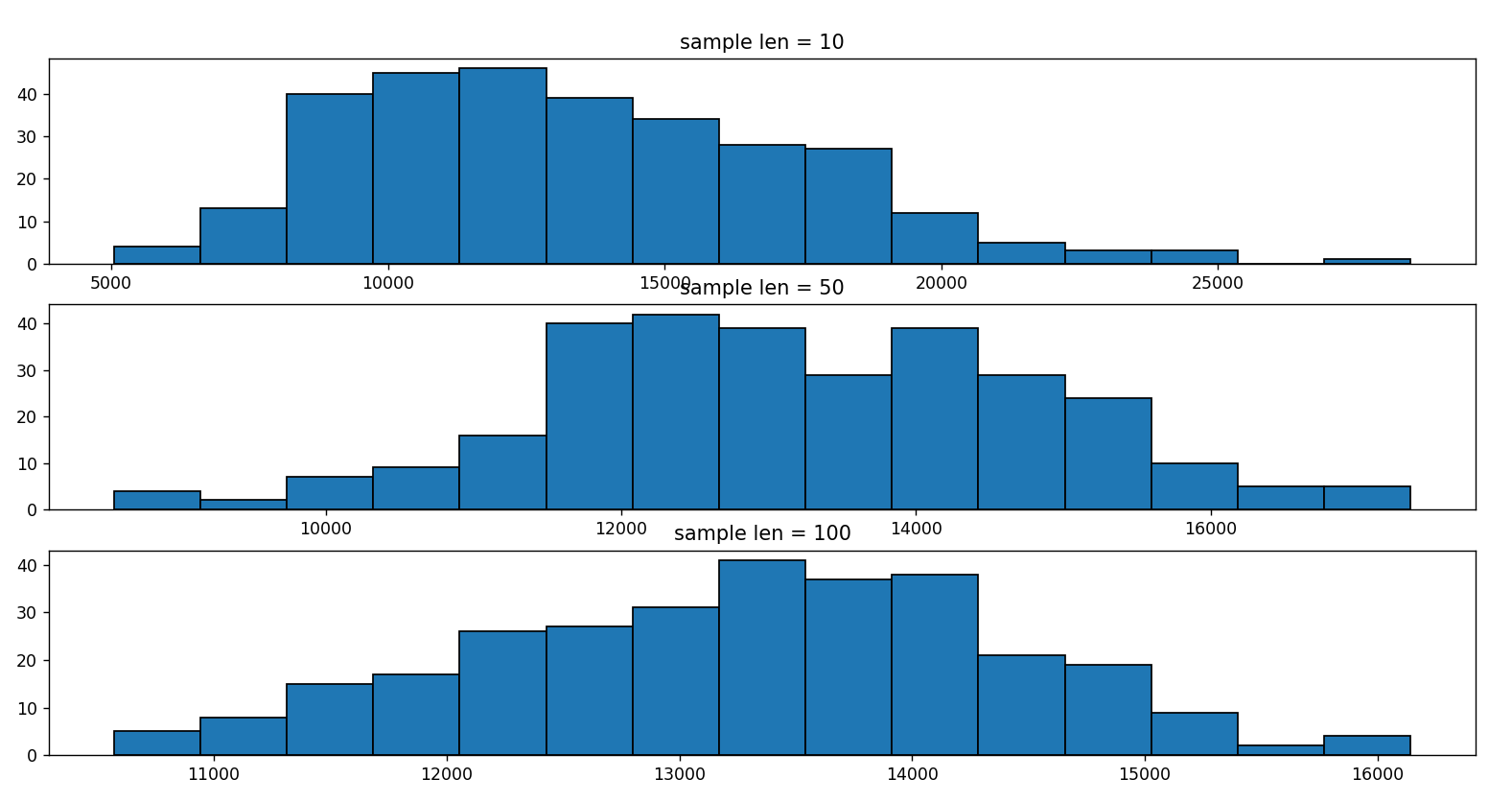


Рисунок 8 – Результат выборок в виде гистограмм

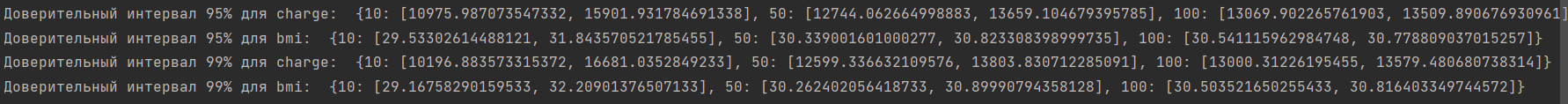


Рисунок 9 – Доверительные интервалы 95% и 99% для выборок (charges) и (bmi)

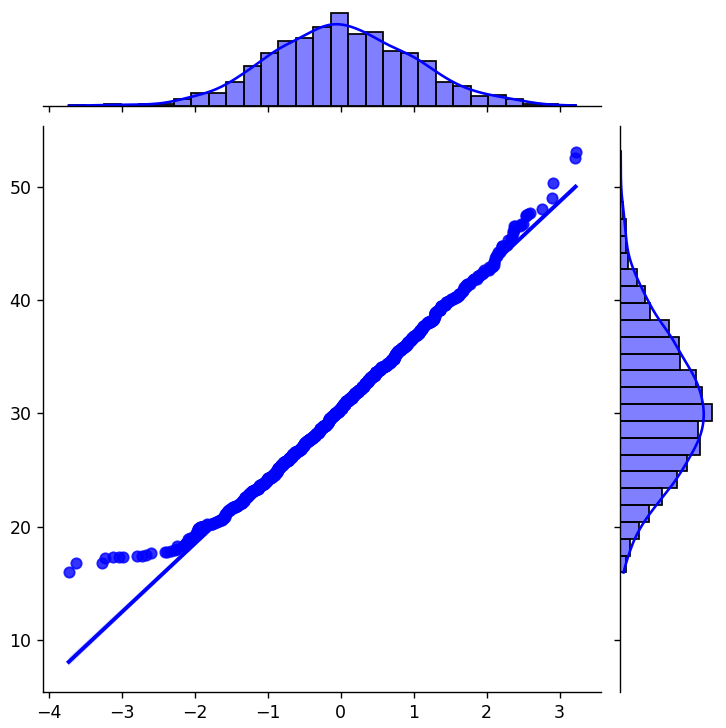


Рисунок 10 – q-q plot для параметра (bmi)

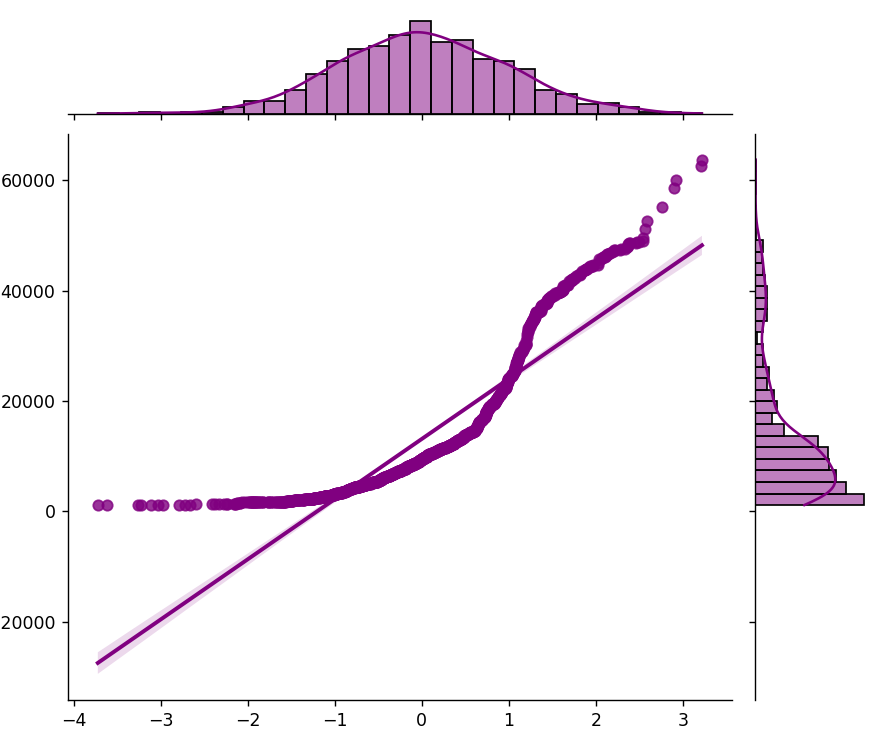


Рисунок 11 – q-q plot для параметра (charges)

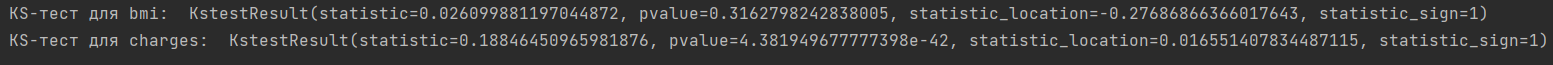


Рисунок 12 – KS-тест для двух параметров

# ECDCCases

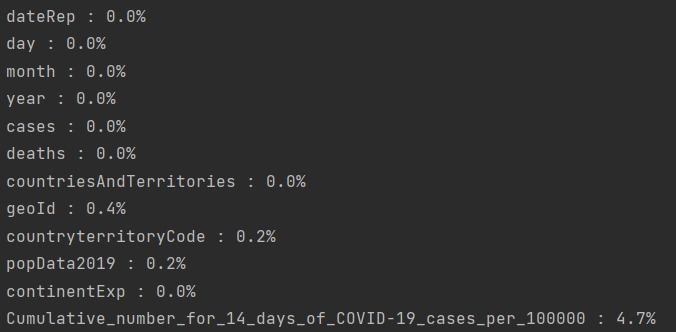


Рисунок 13 – Изначальное количество пропущенных значений

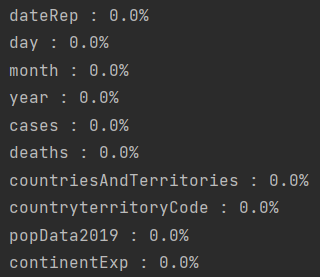


Рисунок 13 – Количество пропущенных значений после редактирования

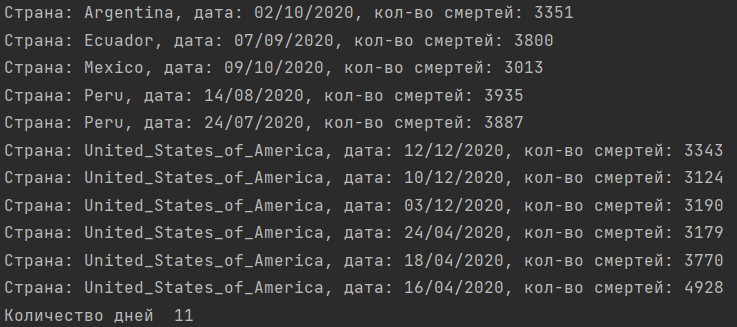


Рисунок 13 – Информация для каких стран количество смертей в день превысило 3000 и сколько таких дней было



Рисунок 14 – Количество дупликатов

# BMI

Построим линейные графики для сравнения параметров, будем смотреть зависимость параметра subscribers, highest yearly earnings и video views от параметра rank.

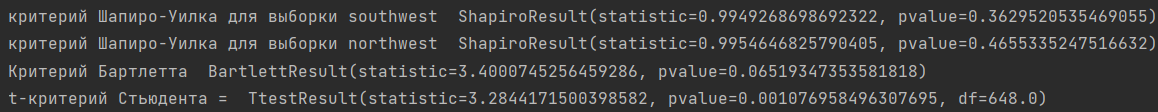


Рисунок 15 – Линейный графики для 3-ех параметров от параметра rank



Рисунок 16 – Критерий Хи-квадрат для кубика

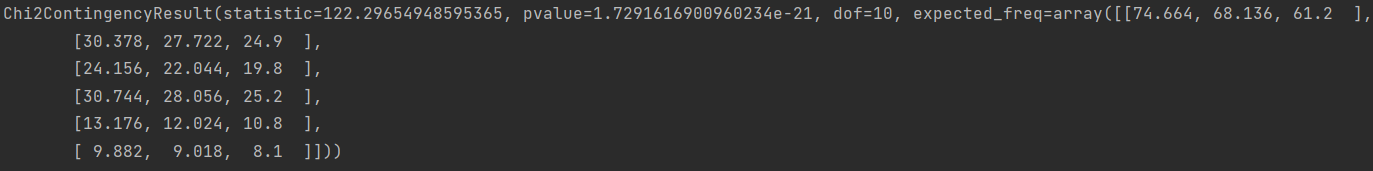


Рисунок 17 – Критерий Хи-квадрат

# ПРИЛОЖЕНИЕ

Приложение 1 – Код программы для зачад 1-8

import pandas as pd  
import matplotlib.pyplot as plt  
import numpy as np  
import scipy.stats as sts  
import seaborn as sns  
  
  
def bar\_grapf(df):  
 fig, ax = plt.subplots(1, 4, figsize=(15, 4))  
 ax[0].hist(df['age'], bins=10, edgecolor='black', color='blue', label='bins = 8')  
 ax[0].set\_title("age")  
 ax[0].legend()  
 ax[1].hist(df['bmi'], bins=30, edgecolor='black', color='blue', label='bins = 30')  
 ax[1].set\_title("bmi")  
 ax[1].legend()  
 ax[2].hist(df['children'], bins=5, edgecolor='black', color='blue', label='bins = 5')  
 ax[2].set\_title("children")  
 ax[2].legend()  
 ax[3].hist(df['charges'], bins=15, edgecolor='black', color='blue', label='bins = 15')  
 ax[3].set\_title("charges")  
 ax[3].legend()  
 mean1, mean2 = np.mean(df['bmi']), np.mean(df['charges'])  
 moda1, moda2 = sts.mode(df['bmi']), sts.mode(df['charges'])  
 med1, med2 = np.median(df['bmi']), np.median(df['charges'])  
 std1, std2 = df['bmi'].std(), df['charges'].std()  
 raz1, raz2 = df['bmi'].max() - df['bmi'].min(), df['charges'].max() - df['charges'].min()  
 iqr1, iqr2 = sts.iqr(df['bmi'], interpolation='midpoint'), sts.iqr(df['charges'], interpolation='midpoint')  
 print('Среднее bmi: ', round(mean1, 1), 'Среднее charges: ', round(mean2, 1))  
 print('Мода bmi: ', moda1[0], 'Мода charges: ', moda2[0])  
 print('Медиана bmi: ', med1, 'Медиана charges: ', round(med2, 1))  
 print('Стандартное отклонение bmi: ', round(std1, 1), 'Стандартное отклонение charges: ', round(std2, 1))  
 print('Размах bmi: ', raz1, 'Размах charges: ', round(raz2, 1))  
 print('Межквартильный размах bmi: ', round(iqr1, 1), 'Межквартильный размах charges: ', round(iqr2, 1))  
 fig, bx = plt.subplots(1, 2, figsize=(15, 4))  
 bx[0].hist(df['bmi'], bins=30, edgecolor='black', color='blue', label='bins = 30')  
 bx[0].axvline(mean1, color='red', linewidth=2, label='Среднее')  
 bx[0].axvline(moda1[0], color='green', linewidth=2, label='Мода')  
 bx[0].axvline(med1, color='yellow', linewidth=2, label='Медиана')  
 bx[0].set\_title('bmi')  
 bx[0].legend()  
 bx[1].hist(df['charges'], bins=15, edgecolor='black', color='blue', label='bins = 15')  
 bx[1].axvline(mean2, color='red', linewidth=2, label='Среднее')  
 bx[1].axvline(moda2[0], color='green', linewidth=2, label='Мода')  
 bx[1].axvline(med2, color='yellow', linewidth=2, label='Медиана')  
 bx[1].set\_title('charges')  
 bx[1].legend()  
 plt.show()  
 return  
  
  
def box\_plot(df):  
 fig, ax = plt.subplots(1, 4, figsize=(15, 4))  
 ax[0].boxplot(df['age'], vert=False)  
 ax[0].set\_title('age')  
 ax[1].boxplot(df['bmi'], vert=False)  
 ax[1].set\_title('bmi')  
 ax[2].boxplot(df['children'], vert=False)  
 ax[2].set\_title('children')  
 ax[3].boxplot(df['charges'], vert=False)  
 ax[3].set\_title('charges')  
 plt.show()  
 return  
  
  
def theorem(df):  
 num\_samples = 300  
 len\_samples = [10, 50, 100]  
 fig, ax = plt.subplots(len(len\_samples), figsize=(30, 8))  
 means\_charges = []  
 means\_bmi = []  
 distance95\_charge = {}  
 distance95\_bmi = {}  
 distance99\_charge = {}  
 distance99\_bmi = {}  
 for i in range(len(len\_samples)):  
 sample\_means\_charges = []  
 sample\_means\_bmi = []  
 for k in range(num\_samples):  
 sample\_charges = np.random.choice(df['charges'], size=len\_samples[i], replace=True)  
 sample\_mean\_charges = np.mean(sample\_charges)  
 sample\_means\_charges.append(sample\_mean\_charges)  
 sample\_bmi = np.random.choice(df['bmi'], size=len\_samples[i], replace=True)  
 sample\_mean\_bmi = np.mean(sample\_bmi)  
 sample\_means\_bmi.append(sample\_mean\_bmi)  
 means\_charges.append(sample\_means\_charges)  
 means\_bmi.append(sample\_means\_bmi)  
 ax[i].hist(sample\_means\_charges, bins=15, edgecolor='black')  
 ax[i].set\_title('sample len = ' + str(len\_samples[i]))  
 sample\_means\_charges = np.array(sample\_means\_charges)  
 sample\_means\_bmi = np.array(sample\_means\_bmi)  
 print("Длина выборки: ", len\_samples[i])  
 print("Стандартное отклонение charges: ", sample\_means\_charges.std())  
 print("Среднее отклонение charges: ", np.mean(sample\_means\_charges))  
 print("Стандартное отклонение bmi: ", sample\_means\_bmi.std())  
 print("Среднее отклонение bmi: ", np.mean(sample\_means\_bmi))  
 distance95\_charge[len\_samples[i]] = [  
 np.mean(sample\_means\_charges) - 1.96 \* sample\_means\_charges.std() / np.sqrt(len\_samples[i]),  
 np.mean(sample\_means\_charges) + 1.96 \* sample\_means\_charges.std() / np.sqrt(len\_samples[i])]  
 distance95\_bmi[len\_samples[i]] = [  
 np.mean(sample\_means\_bmi) - 1.96 \* sample\_means\_bmi.std() / np.sqrt(len\_samples[i]),  
 np.mean(sample\_means\_bmi) + 1.96 \* sample\_means\_bmi.std() / np.sqrt(len\_samples[i])]  
 distance99\_charge[len\_samples[i]] = [  
 np.mean(sample\_means\_charges) - 2.58 \* sample\_means\_charges.std() / np.sqrt(len\_samples[i]),  
 np.mean(sample\_means\_charges) + 2.58 \* sample\_means\_charges.std() / np.sqrt(len\_samples[i])]  
 distance99\_bmi[len\_samples[i]] = [  
 np.mean(sample\_means\_bmi) - 2.58 \* sample\_means\_bmi.std() / np.sqrt(len\_samples[i]),  
 np.mean(sample\_means\_bmi) + 2.58 \* sample\_means\_bmi.std() / np.sqrt(len\_samples[i])]  
 print("Доверительный интервал 95% для charge: ", distance95\_charge)  
 print("Доверительный интервал 95% для bmi: ", distance95\_bmi)  
 print("Доверительный интервал 99% для charge: ", distance99\_charge)  
 print("Доверительный интервал 99% для bmi: ", distance99\_bmi)  
 plt.show()  
 return  
  
  
def normal\_distribution(df):  
 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()  
 return  
  
  
def main():  
 df = pd.read\_csv("insurance.csv")  
 print(df.describe())  
 print(df.info())  
 # bar\_grapf(df)  
 # box\_plot(df)  
 # theorem(df)  
 normal\_distribution(df)  
 return  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Приложение 2 – Код программы для зачад 9-12

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 = 0  
 for item in df.duplicated():  
 if item:  
 a += 1  
 print("Дупликатов: ", 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 3 – Код программы для зачад 13-15

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)  
 # t\_criter\_south = (np.mean(south) - #np.mean(north))/np.sqrt((south.std()\*\*2/len(south)) + #(north.std()\*\*2/len(north)))  
 print("критерий Шапиро-Уилка для выборки southwest ", sts.shapiro(south))  
 print("критерий Шапиро-Уилка для выборки 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))  
 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()