МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИТ)

Кафедра прикладной математики

ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 3

по дисциплине «Технологии и инструментарий анализа больших данных»

Выполнил студент группы ИКБО-20-21 Сидоров С.Д.

Проверил ассистент кафедры ПМ ИИТ Тетерин Н.Н.

Москва 2024

Практическая работа

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

Листинг 1:

```
insurance_data = pd.read_csv('insurance.csv')
```

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

Листинг 2:

```
print(insurance_data.describe())
```

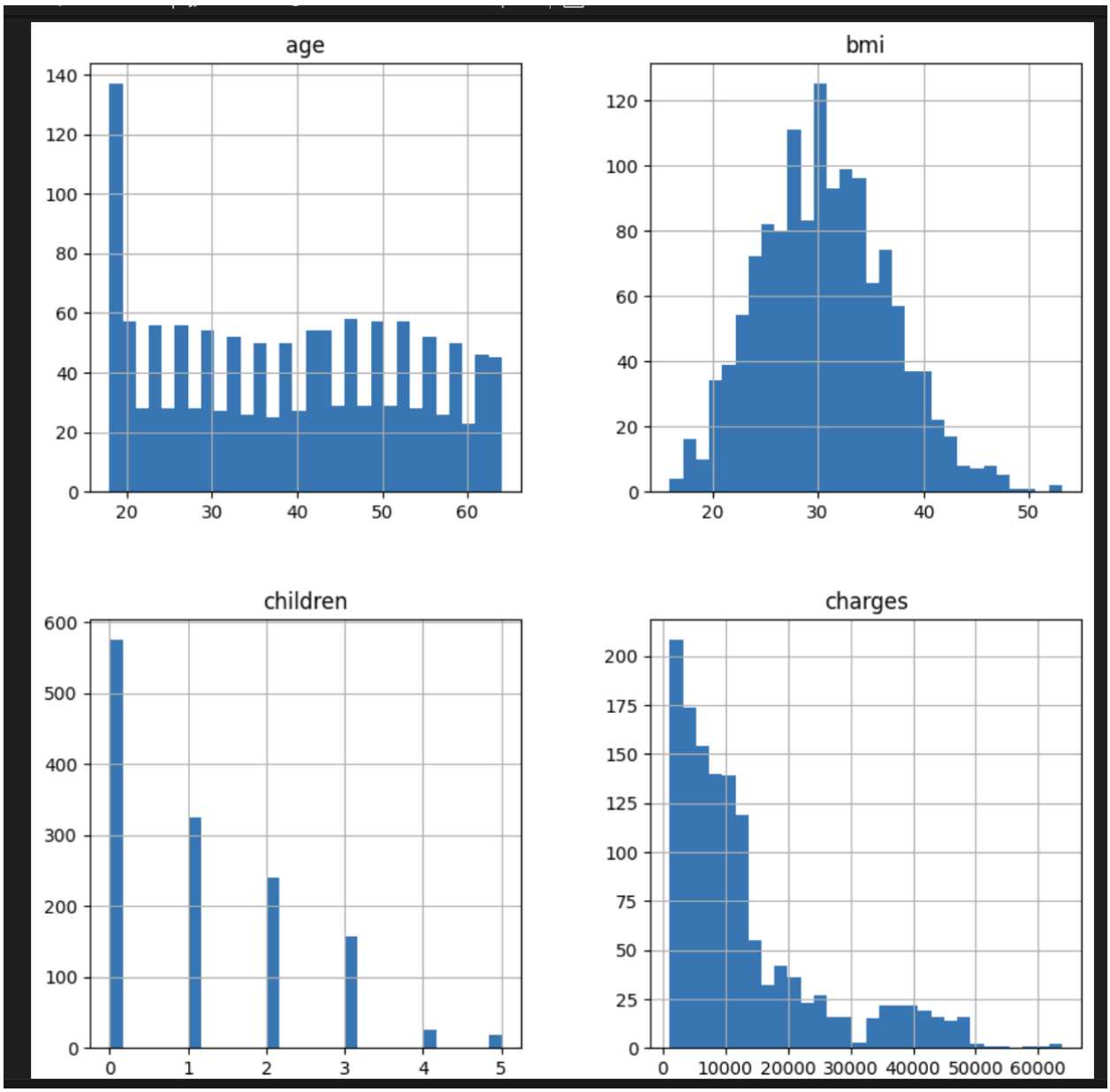
```
=== Задание 2: Статистика по данным ===
                age          bmi     children       charges
count  1338.000000  1338.000000  1338.000000   1338.000000
mean     39.207025    30.663397     1.094918  13270.422265
std      14.049960     6.098187     1.205493  12110.011237
min      18.000000    15.960000     0.000000   1121.873900
25%      27.000000    26.296250     0.000000   4740.287150
50%      39.000000    30.400000     1.000000   9382.033000
75%      51.000000    34.693750     2.000000  16639.912515
max      64.000000    53.130000     5.000000  63770.428010
```

Выводы: датасет содержит данные о страховании, включающие возраст, индекс массы тела, количество детей, расходы

3) Построить гистограммы для числовых показателей. Сделать выводы

Листинг 3:

```
import matplotlib.pyplot as plt
insurance_data.hist(bins=30, figsize=(10,10))
plt.show()
#гистограммы частот
```



Вывод:

Средний возраст участников составляет примерно `39` лет

Средний BMI равен `30.66`, что соответствует категории избыточного веса или ожирения.

В среднем у респондентов `1` ребенок. Большинство - либо `бездетные`, либо имеют `одного` или `двух` детей (максимум - `5`)

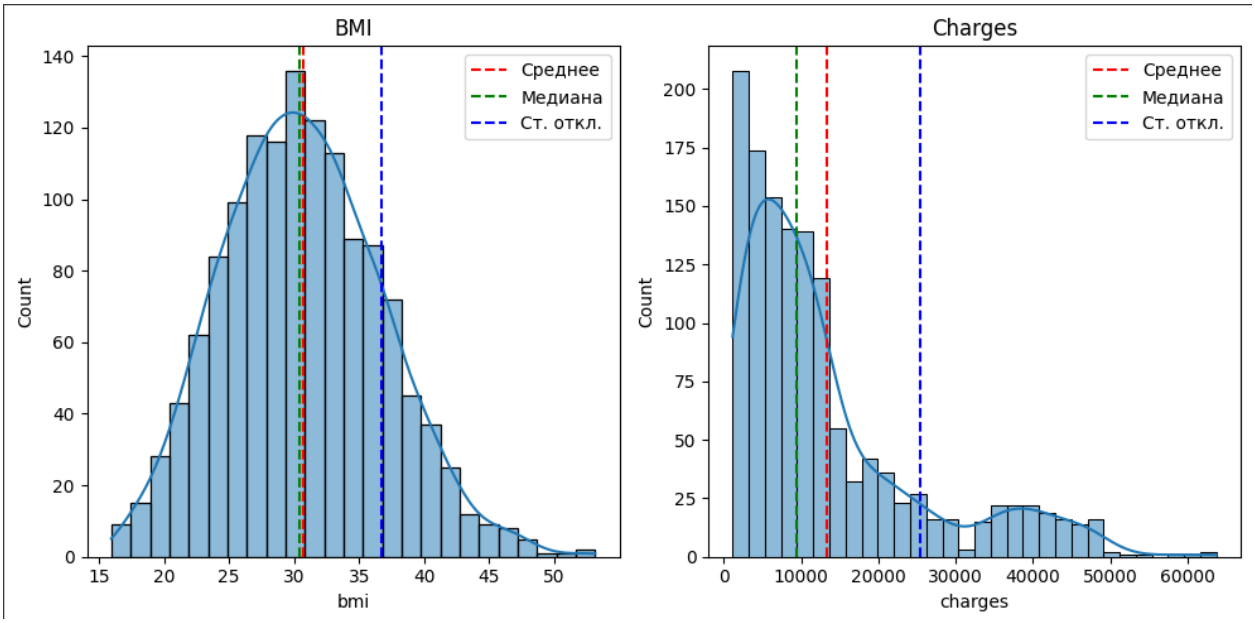
Средний уровень страховых взносов составляет около `13270` долларов с большим разбросом, что указывает на значительное разнообразие в стоимости страхования.

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.

Листинг 4:

```
print("\n=== Задание 4: Меры центральной тенденции и разброса для bmi и
charges ===")
bmi = insurance_data['bmi']
charges = insurance_data['charges']

# Расчет средней, медианы и стандартного отклонения
mean_bmi = bmi.mean()
median_bmi = bmi.median()
std_bmi = bmi.std()

mean_charges = charges.mean()
median_charges = charges.median()
std_charges = charges.std()

print(f"Среднее BMI: {mean_bmi}, Медиана BMI: {median_bmi}, Стандартное
отклонение BMI: {std_bmi}")
print(f"Среднее Charges: {mean_charges}, Медиана Charges:
{median_charges}, Стандартное отклонение Charges: {std_charges}")

# Построение гистограмм с добавлением линий
plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)
sns.histplot(bmi, kde=True)
plt.axvline(mean_bmi, color='r', linestyle='--', label='Среднее')
plt.axvline(median_bmi, color='g', linestyle='--', label='Медиана')
plt.axvline(mean_bmi + std_bmi, color='b', linestyle='--', label='Ст.
откл.')
plt.legend()
plt.title('BMI')

plt.subplot(1, 2, 2)
sns.histplot(charges, kde=True)
plt.axvline(mean_charges, color='r', linestyle='--', label='Среднее')
plt.axvline(median_charges, color='g', linestyle='--', label='Медиана')
plt.axvline(mean_charges + std_charges, color='b', linestyle='--',
label='Ст. откл.')
plt.legend()
plt.title('Charges')

plt.tight_layout()
plt.show()
```

```
=== Задание 4: Меры центральной тенденции и разброса для bmi и charges ===
Среднее BMI: 30.66339686098655, Медиана BMI: 30.4, Стандартное отклонение BMI: 6.098186911679017
Среднее Charges: 13270.422265141257, Медиана Charges: 9382.033, Стандартное отклонение Charges: 12110.011236693994
```



Вывод:

Индекс массы тела (BMI)

Средний BMI составляет `30.66`, медиана - `30.40`.

Наиболее часто встречающееся значение (мода) `BMI` равно `32.30`, большинство участников имеют избыточный вес.

Стандартное отклонение `6.10` и дисперсия `37.19` указывают на умеренное разнообразие значений.

Страховые взносы

Средний размер страховых взносов составляет `13270.42` долларов, в то время как медиана - `9382.03` долларов. Это говорит о том, что среднее значение значительно выше медианы, что может свидетельствовать о наличии некоторых участников с высокими страховками.

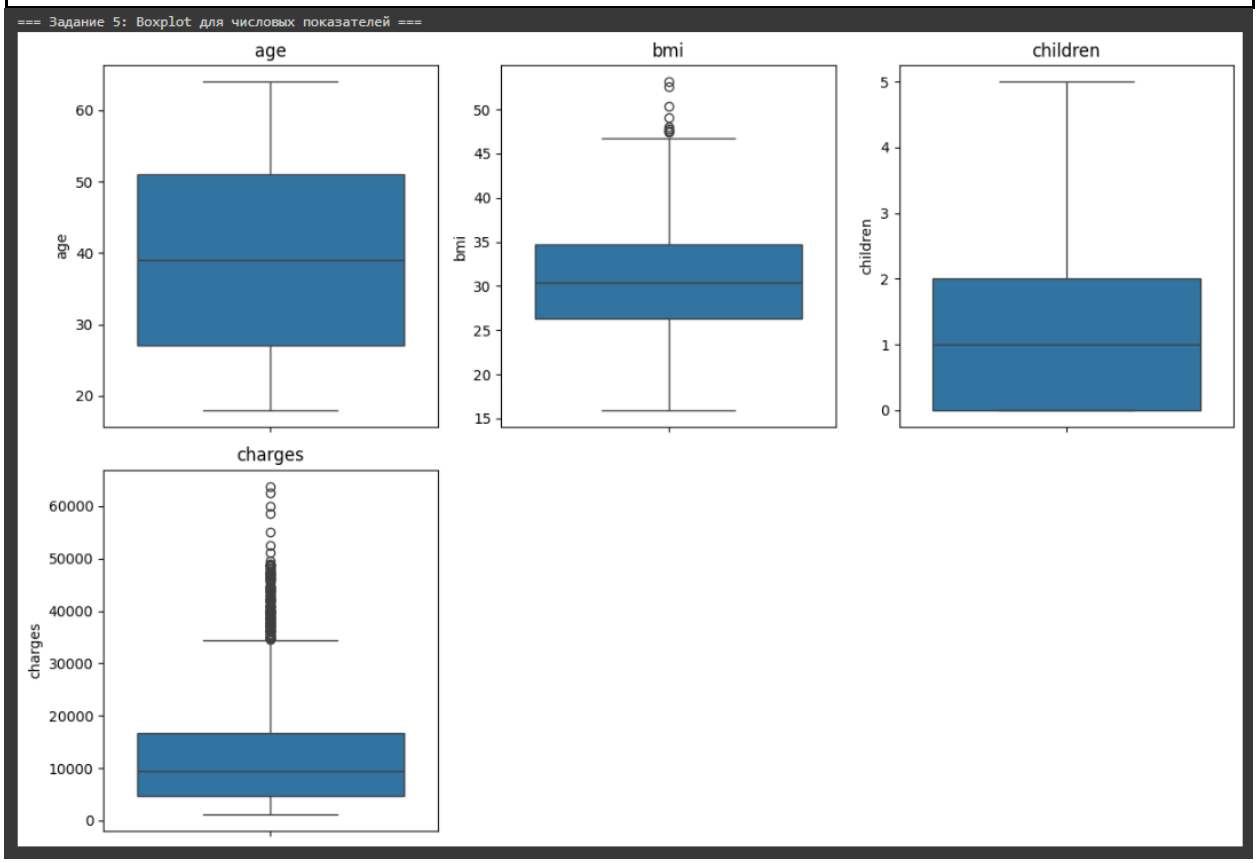
мода - `1639.56` долларов, что сильно ниже среднего значения. Это говорит о присутствие группы людей с низкими расходами на страхование.

Стандартное отклонение `12110.01` указывает на широкий разброс данных.

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

Листинг 5:

```
print("\n=== Задание 5: Boxplot для числовых показателей ===")
plt.figure(figsize=(12, 8))

for i, column in enumerate(insurance_data.select_dtypes(include=[np.number]).columns, 1):
    plt.subplot(2, 3, i)
    sns.boxplot(y=insurance_data[column])
    plt.title(column)

plt.tight_layout()
plt.show()
```

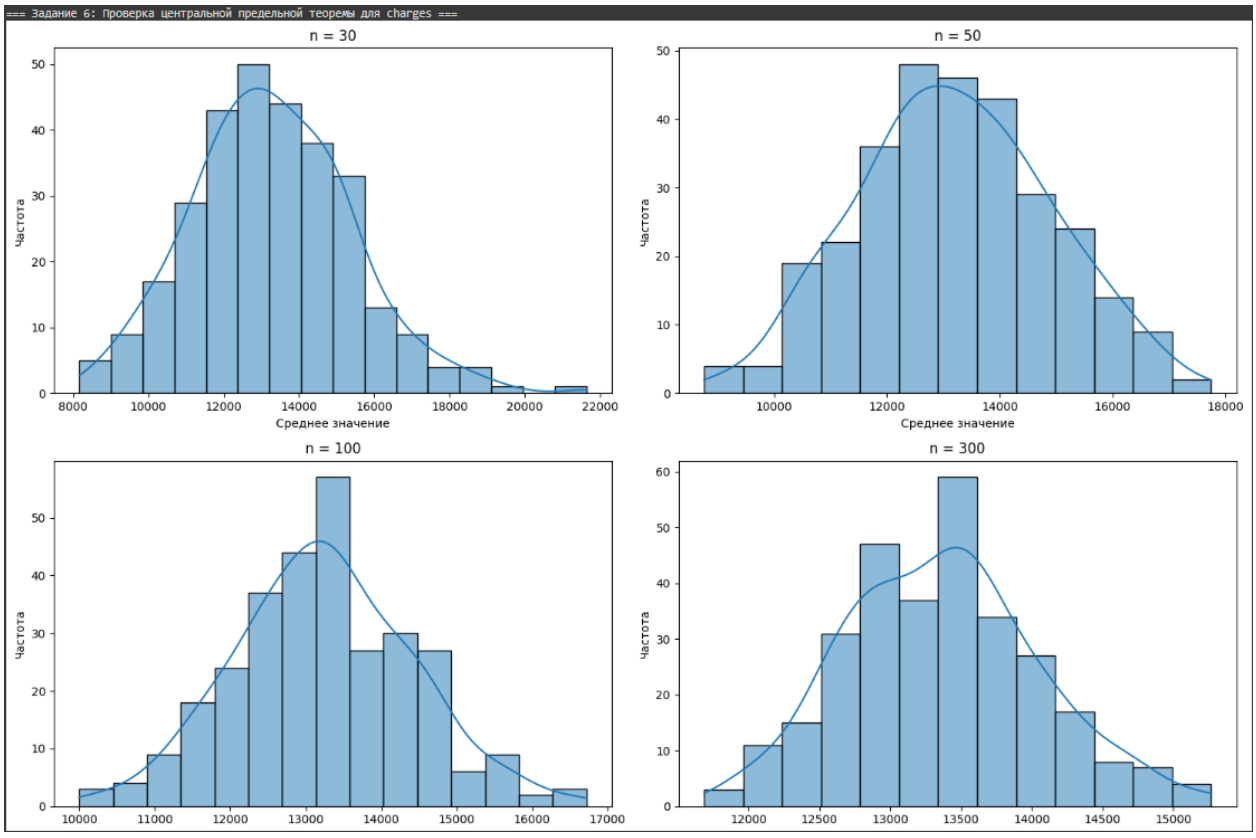


6) Используя признак charges или imb, проверить, выполняется ли центральная предельная теорема. Использовать различные длины выборок n. Количество выборок = 300. Вывести результат в виде гистограмм. Найти стандартное отклонение и среднее для полученных распределений. Сделать выводы

Листинг 6:

```
print("\n=== Задание 6: Проверка центральной предельной теоремы для
charges ===")
sample_means = []
n_values = [30, 50, 100, 300]
```

```
for size in n_values:
    means = [np.mean(np.random.choice(charges, size)) for _ in
range(300)]
    sample_means.append(means)

# Построение гистограмм для каждой длины выборки
plt.figure(figsize=(15, 10))
for i, size in enumerate(n):
    plt.subplot(2, 2, i+1)
    sns.histplot(sample_means[i], kde=True)
    plt.title(f"n = {size}")
    plt.xlabel('Среднее значение')
    plt.ylabel('Частота')

plt.tight_layout()
plt.show()

for i, n in enumerate(n_values):
    print(f'Длина выборки = {n}: Среднее = {np.mean(sample_means[i])},
Станд.отклонение = {np.std(sample_means[i])}')
```




```
Длина выборки = 30: Среднее = 13276.585324722777, Станд.отклонение = 2113.0356296196114
Длина выборки = 50: Среднее = 13200.515166053468, Станд.отклонение = 1724.6835548064091
Длина выборки = 100: Среднее = 13242.921941256034, Станд.отклонение = 1186.235297785789
Длина выборки = 300: Среднее = 13365.154524655534, Станд.отклонение = 683.155818636932
```

Выводы: ЦПТ подтверждается

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

Листинг 7:

```
print("\n=== Задание 7: Доверительные интервалы для BMI и Charges ===")
n = len(bmi)
mean = bmi.mean()
stderr = bmi.std() / np.sqrt(n)
margin = stderr * stats.t.ppf((1 + 0.95) / 2.0, n - 1)
print("BMI 95:", mean - margin, mean + margin)

n = len(charges)
mean = charges.mean()
stderr = charges.std() / np.sqrt(n)
margin = stderr * stats.t.ppf((1 + 0.95) / 2.0, n - 1)
print("Charges 95:", mean - margin, mean + margin)

n = len(bmi)
mean = bmi.mean()
stderr = bmi.std() / np.sqrt(n)
margin = stderr * stats.t.ppf((1 + 0.99) / 2.0, n - 1)
print("BMI 99:", mean - margin, mean + margin)

n = len(charges)
mean = charges.mean()
stderr = charges.std() / np.sqrt(n)
margin = stderr * stats.t.ppf((1 + 0.99) / 2.0, n - 1)
print("Charges 99:", mean - margin, mean + margin)
```

```
=== Задание 7: Доверительные интервалы для BMI и Charges ===
BMI 95: 30.336346903054107 30.99044681891899
Charges 95: 12620.954034192644 13919.890496089869
BMI 99: 30.233355575431624 31.093438146541473
Charges 99: 12416.429943203952 14124.414587078561
```

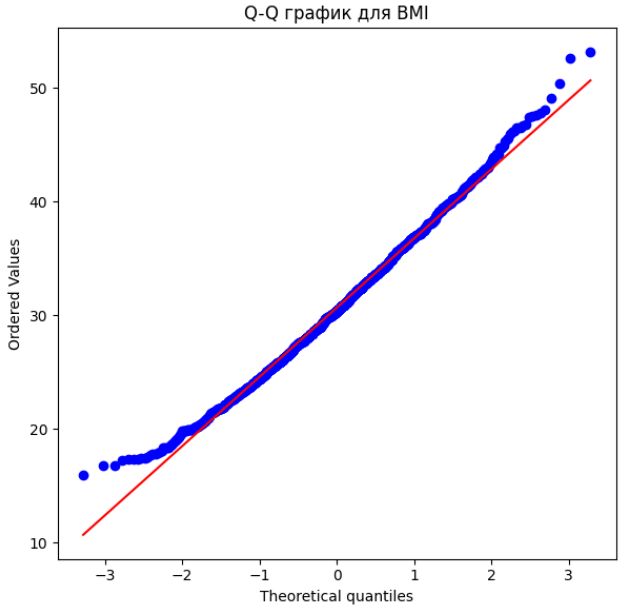
8) Проверить распределения следующих признаков на нормальность: индекс массы тела, расходы. Сформулировать нулевую и альтернативную гипотезы. Для каждого признака использовать KS-тест и q-q plot. Сделать выводы на основе полученных p-значений.

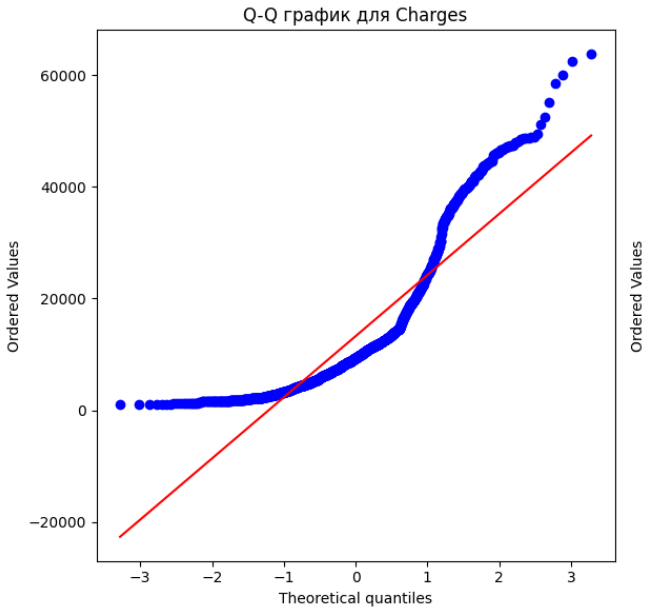
Листинг 8:

```
print("\n=== Задание 8: Проверка нормальности для BMI и Charges ===")
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
stats.probplot(bmi, dist="norm", plot=plt)
plt.title('Q-Q график для BMI')

plt.tight_layout()
plt.show()

plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
stats.probplot(charges, dist="norm", plot=plt)
plt.title('Q-Q график для Charges')

plt.tight_layout()
```

```
plt.show()

# KS-тест
for feature in ['bmi', 'charges']:
    stat = stats.kstest(insurance_data[feature], 'norm', args=(insurance_data[feature].mean(),
insurance_data[feature].std()))
    print(stat)
#нулевая - распределение нормально
#альтернативная - распределение не нормально
```

```
KstestResult(statistic=0.026099881197044872, pvalue=0.3162798242838005, statistic_location=28.975, statistic_sign=1)
KstestResult(statistic=0.18846450965981876, pvalue=4.381949677777398e-42, statistic_location=13470.86, statistic_sign=1)
```





Вывод:

Основная гипотеза (H0)

Нет значительных различий/влияний в зависимости от рассматриваемых факторов на изучаемую величину (или переменную)

Альтернативная гипотеза (H1)

Существуют значительные различия/влияния в зависимости от рассматриваемых факторов на изучаемую величину (или переменную)

Индекс массы тела

Поскольку p-value (`0.316`) больше уровня значимости (`0.05`), мы не можем отвергнуть нулевую гипотезу. Распределение индекса массы тела можно считать нормальным

Расходы

Поскольку p-value (`0.000`) меньше уровня значимости (`0.05`), мы отвергаем нулевую гипотезу. Распределение расходов не является нормальным

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

Листинг 10:

```
ecdccases_data = pd.read_csv('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.

Листинг 11:

```
print("\n=== Задание 10: Пропущенные значения в данных ===")
missing_values = ecdccases_data.isnull().sum()
percentage_missing = (missing_values / len(ecdccases_data)) * 100
print("Пропущенные значения: " + str(missing_values))
print("Процент пропущенных значений: " + str(percentage_missing))
```

```
most_missing_features = missing_values.nlargest(2).index
ecdccases_data.drop(columns=most_missing_features)

ecdccases_data['countryterritoryCode'] = ecdccases_data['countryterritoryCode'].fillna('other')
ecdccases_data['popData2019'] =
ecdccases_data['popData2019'].fillna(ecdccases_data['popData2019'].median())

print("Пустых значений нет:", ecdccases_data.isnull().sum().sum() == 0)
```

```
=== Задание 10: Пропущенные значения в данных ===
Пропущенные значения: dateRep                                    0
day                                                              0
month                                                            0
year                                                             0
cases                                                            0
deaths                                                           0
countriesAndTerritories                                          0
geoId                                                          275
countryterritoryCode                                           123
popData2019                                                    123
continentExp                                                     0
Cumulative_number_for_14_days_of_COVID-19_cases_per_100000    2879
dtype: int64
Процент пропущенных значений: dateRep                                 0.000000
day                                                           0.000000
month                                                         0.000000
year                                                          0.000000
cases                                                         0.000000
deaths                                                        0.000000
countriesAndTerritories                                       0.000000
geoId                                                         0.444236
countryterritoryCode                                          0.198695
popData2019                                                   0.198695
continentExp                                                  0.000000
Cumulative_number_for_14_days_of_COVID-19_cases_per_100000    4.650750
dtype: float64
```

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.

Листинг 12:

```
print("\n=== Задание 11: Описание данных и отбор записей ===")
print(ecdccases_data.describe())
over3000 = ecdccases_data.loc[ecdccases_data['deaths'] > 3000]
print(over3000['countriesAndTerritories'].value_counts())
```

```
                day         month          year          cases        deaths  \
count  61904.000000  61904.000000  61904.000000   61904.000000  61904.000000
mean      15.629232      7.067104   2019.998918    1155.079026     26.053987
std        8.841624      2.954816      0.032881    6779.010824    131.222948
min        1.000000      1.000000   2019.000000   -8261.000000  -1918.000000
25%        8.000000      5.000000   2020.000000       0.000000      0.000000
50%       15.000000      7.000000   2020.000000      15.000000      0.000000
75%       23.000000     10.000000   2020.000000     273.000000      4.000000
max       31.000000     12.000000   2020.000000  234633.000000   4928.000000

        popData2019  \
count  6.190400e+04
mean   4.091909e+07
std    1.529798e+08
min    8.150000e+02
25%    1.324820e+06
50%    7.169456e+06
75%    2.851583e+07
max    1.433784e+09

       Cumulative_number_for_14_days_of_COVID-19_cases_per_100000
count                                        59025.000000
mean                                            66.316369
std                                            162.324550
min                                           -147.419587
25%                                              0.757526
50%                                              6.724045
75%                                             52.561206
max                                           1900.836210
countriesAndTerritories
United_States_of_America    6
Peru                        2
Argentina                   1
Ecuador                     1
Mexico                      1
Name: count, dtype: int64
```

12) Найти дублирование данных. Удалить дубликаты.

Листинг 13:

```
print("\n=== Задание 12: Поиск и удаление дубликатов ===")
dupl = ecdccases_data.duplicated()
print(f'Количество одинаковых строк: { dupl.sum() }')
ecdccases_data = ecdccases_data.drop_duplicates()
print(ecdccases_data)
```

13) Загрузить данные из файла “bmi.csv”. Взять оттуда две выборки. Одна выборка – это индекс массы тела людей с 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).

Листинг 14:

```
print("\n=== Задание 13: Сравнение выборок по BMI из разных регионов ===")
bmi_data = pd.read_csv('bmi.csv')

northwest_bmi = bmi_data[bmi_data['region'] == 'northwest']['bmi']
southwest_bmi = bmi_data[bmi_data['region'] == 'southwest']['bmi']

_, p_value_northwest = stats.shapiro(northwest_bmi)
_, p_value_southwest = stats.shapiro(southwest_bmi)

_, p_value_bartlett = stats.bartlett(northwest_bmi, southwest_bmi)

print(f"p-value (Шапиро-Уилка) для выборки из northwest: {p_value_northwest:.4f}")
print(f"p-value (Шапиро-Уилка) для выборки из southwest: {p_value_southwest:.4f}")
print(f"p-value (Бартлетт) для проверки гомогенности дисперсии: {p_value_bartlett:.4f}")

if p_value_northwest > 0.05 and p_value_southwest > 0.05 and p_value_bartlett > 0.05:
    t_statistic, p_value_ttest = stats.ttest_ind(northwest_bmi, southwest_bmi)
    print(f"t-статистика: {t_statistic:.4f}")
    print(f"p-value (t-критерии Стьюдента): {p_value_ttest:.4f}")

    if p_value_ttest < 0.05:
        print("Различия в средних значениях выборок статистически значимы.")
    else:
        print("Нет статистически значимых различий в средних значениях выборок.")

else:
    print("Условия для использования t-критерия Стьюдента не выполняются.")
```

```
=== Задание 13: Сравнение выборок по BMI из разных регионов ===
p-value (Шапиро-Уилка) для выборки из northwest: 0.4656
p-value (Шапиро-Уилка) для выборки из southwest: 0.3630
p-value (Бартлетт) для проверки гомогенности дисперсии: 0.0652
t-статистика: -3.2844
p-value (t-критерии Стьюдента): 0.0011
Различия в средних значениях выборок статистически значимы.
```

14) Кубик бросили 600 раз, получили следующие результаты: N Количество 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().

Листинг 15:

```
print("\n=== Задание 14: Проверка равномерности распределения кубика ===")
observed_frequencies = np.array([97, 98, 109, 95, 97, 104])
expected_frequencies = np.array([100] * 6)
```

```
chi2_statistic, p_value = stats.chisquare(observed_frequencies, expected_frequencies)

print(f"Значение критерия Хи-квадрат: {chi2_statistic:.2f}")
print(f"p-значение: {p_value:.4f}")

alpha = 0.05
if p_value < alpha:
    print("Отвергаем нулевую гипотезу: распределение не является равномерным.")
else:
    print("Не отвергаем нулевую гипотезу: распределение равномерное.")
```

```
=== Задание 14: Проверка равномерности распределения кубика ===
Значение критерия Хи-квадрат: 1.44
p-значение: 0.9199
Не отвергаем нулевую гипотезу: распределение равномерное.
```

15) С помощью критерия Хи-квадрат проверить, являются ли переменные зависимыми. Создать датафрейм, используя следующий код:

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']

Использовать функцию scipy.stats.chi2_contingency(). Влияет ли семейное положение на занятость?

Листинг 16:

```
print("\n=== Задание 15: Проверка зависимости семейного положения и занятости ===")
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']

chi2_statistic, p_value, dof, expected = stats.chi2_contingency(data)

print(f"Значение критерия Хи-квадрат: {chi2_statistic:.2f}")
print(f"p-значение: {p_value:.4f}")
```

```
=== Задание 15: Проверка зависимости семейного положения и занятости ===
Значение критерия Хи-квадрат: 122.30
p-значение: 0.0000
Отвергаем нулевую гипотезу: семейное положение влияет на занятость.
```