|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра прикладной математики (ПМ)**

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ**

по дисциплине «Технологии и инструментарий анализа больших данных»

**Практическое занятие № 3**

|  |  |  |  |
| --- | --- | --- | --- |
| Студент группы ИНБО-01-17 | *ИМБО-02-22, Ким Кирилл Сергеевич* | (подпись) | |
| Преподаватель | *Тетерин Николай Николаевич, преподаватель* | (подпись) | |
| Отчет представлен | «\_\_\_»\_\_\_\_\_\_\_\_202\_\_г. | |  | |

Москва 2025 г.

СОДЕРЖАНИЕ

[ПРАКТИЧЕСКАЯ РАБОТА 3 3](#_Toc210826472)

[Введение 3](#_Toc210826473)

[Шаги выполнения 3](#_Toc210826474)

[Результат работы: 20](#_Toc210826475)

[Вывод: 20](#_Toc210826476)

[Список использованных источников и литературы: 21](#_Toc210826477)

ПРАКТИЧЕСКАЯ РАБОТА 3

Введение

**Цель**

Вспомнить такие базовые понятия, как выборка и генеральная совокупность.

Шаги выполнения

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

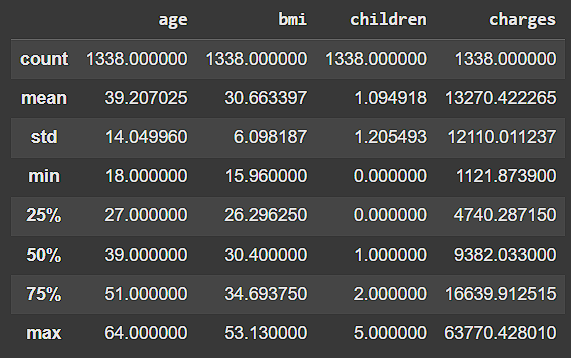
Листинг 1 – Загрузка данных

|  |
| --- |
| import pandas as pd  import numpy as np  import matplotlib.pyplot as plt  import seaborn as sns  import scipy.stats as sts  from scipy.stats import norm, shapiro, bartlett, ttest\_ind, kstest, chi2\_contingency  data = pd.read\_csv("insurance.csv")  data.isna().sum()  data |

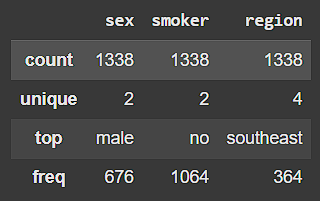
1. С помощью метода describe() посмотреть статистику по данным. Сделать выводы.

Листинг 2 – Статистика данных

|  |
| --- |
| data.describe()  data.describe(include=['object']) |



**Рисунок 1 – Статистика по числовым признакам**



**Рисунок 2 – Статистика по категориальным признакам**

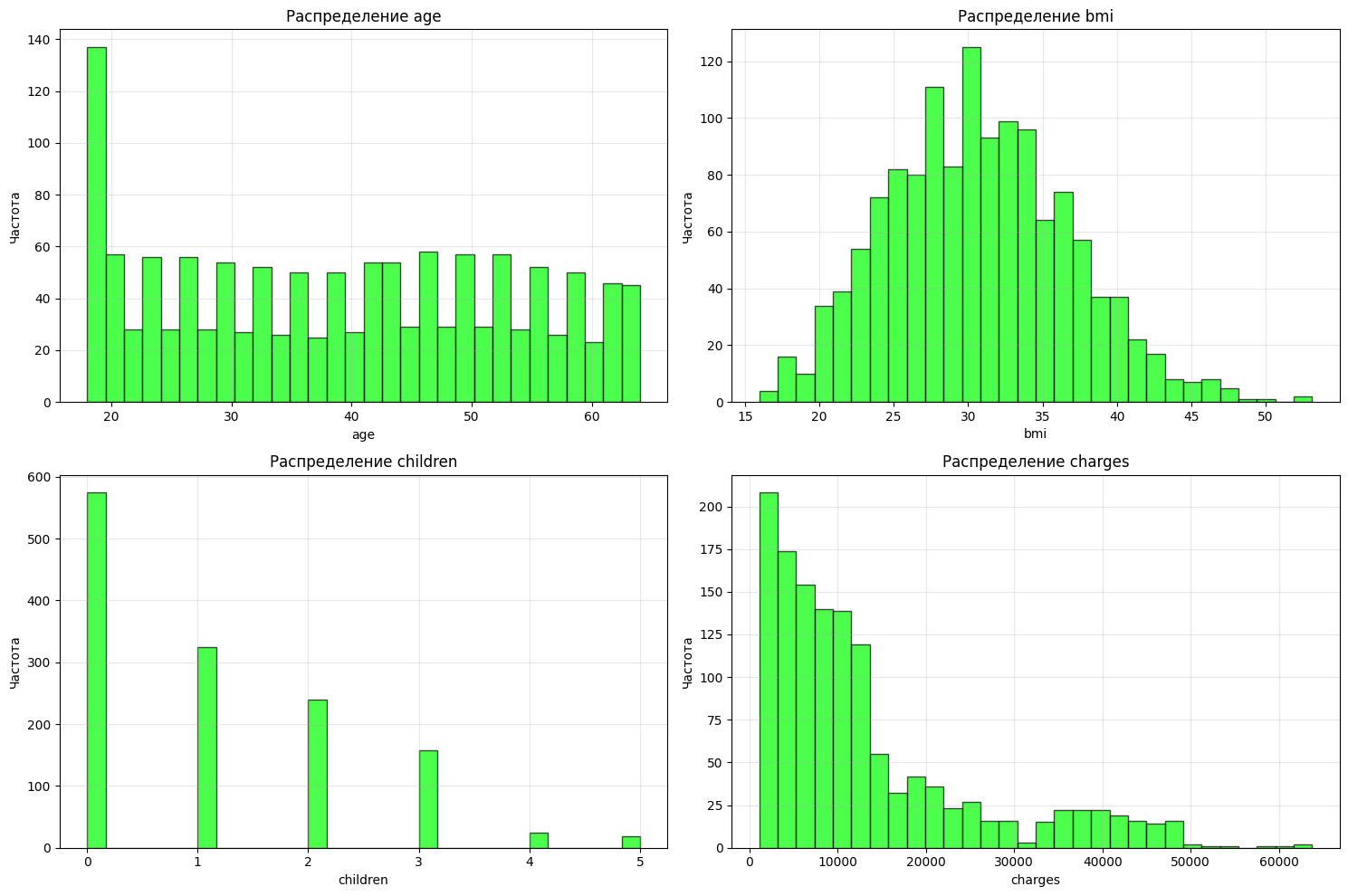
Выводы:

* В данных 1338 наблюдений
* Средний возраст клиентов +- 39 лет
* Средний BMI +- 30.66 (что указывает на избыточный вес)
* Средние страховые расходы +- 13270 US
* Большинство клиентов не имеют детей (75% имеют 0-1 ребенка)
* Большинство клиентов не курят

1. Построить гистограммы для числовых показателей. Сделать выводы.

Листинг 3 – Гистограмма

|  |
| --- |
| # Гистограммы для всех числовых признаков  numeric\_cols = ['age', 'bmi', 'children', 'charges']  fig, axes = plt.subplots(2, 2, figsize=(15, 10))  axes = axes.ravel()  for i, col in enumerate(numeric\_cols):      axes[i].hist(data[col], bins=30, color='lime', edgecolor='black', alpha=0.7)      axes[i].set\_title(f'Распределение {col}')      axes[i].set\_xlabel(col)      axes[i].set\_ylabel('Частота')      axes[i].grid(True, alpha=0.3)  plt.tight\_layout()  plt.show() |



**Рисунок 3 – Статистика по категориальным признакам**

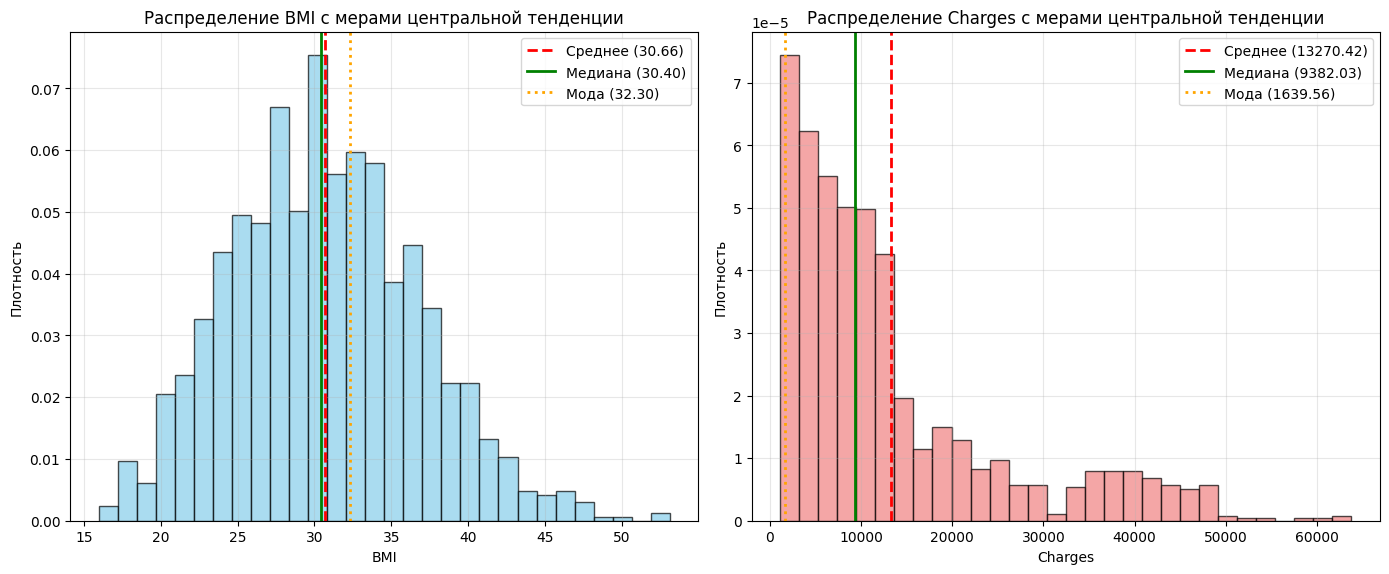
Выводы:

* Age: Распределение относительно равномерное с пиками на круглых возрастах
* BMI: Распределение близко к нормальному с центром около 30
* Children: Дискретное распределение, большинство клиентов без детей или с 1 ребенком
* Charges: Сильно правоскошенное распределение с длинным хвостом больших значений

1. 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 – Статистика BMI и Charges

|  |
| --- |
| def calculate\_statistics(series, name):      stats = {}      stats['mean'] = np.mean(series)      stats['median'] = np.median(series)      stats['mode'] = sts.mode(series, keepdims=True).mode[0]      stats['std'] = np.std(series, ddof=1)      stats['var'] = np.var(series, ddof=1)      stats['range'] = series.max() - series.min()      stats['q1'] = np.percentile(series, 25)      stats['q3'] = np.percentile(series, 75)      stats['iqr'] = stats['q3'] - stats['q1']        print(f"Статистика для {name}")      for key, value in stats.items():          print(f"{key}: {value:.2f}")        return stats  # Статистика для BMI  stats\_bmi = calculate\_statistics(data['bmi'], 'BMI')  print()  # Статистика для Charges  stats\_charges = calculate\_statistics(data['charges'], 'Charges')  # Построение гистограмм с мерами центральной тенденции  fig, ax = plt.subplots(1, 2, figsize=(14, 6))  # Гистограмма для BMI  ax[0].hist(data['bmi'], bins=30, color='skyblue', edgecolor='black', alpha=0.7, density=True)  ax[0].axvline(stats\_bmi['mean'], color='red', linestyle='--', linewidth=2, label=f'Среднее ({stats\_bmi["mean"]:.2f})')  ax[0].axvline(stats\_bmi['median'], color='green', linestyle='-', linewidth=2, label=f'Медиана ({stats\_bmi["median"]:.2f})')  ax[0].axvline(stats\_bmi['mode'], color='orange', linestyle=':', linewidth=2, label=f'Мода ({stats\_bmi["mode"]:.2f})')  ax[0].set\_title('Распределение BMI с мерами центральной тенденции')  ax[0].set\_xlabel('BMI')  ax[0].set\_ylabel('Плотность')  ax[0].legend()  ax[0].grid(True, alpha=0.3)  # Гистограмма для Charges  ax[1].hist(data['charges'], bins=30, color='lightcoral', edgecolor='black', alpha=0.7, density=True)  ax[1].axvline(stats\_charges['mean'], color='red', linestyle='--', linewidth=2, label=f'Среднее ({stats\_charges["mean"]:.2f})')  ax[1].axvline(stats\_charges['median'], color='green', linestyle='-', linewidth=2, label=f'Медиана ({stats\_charges["median"]:.2f})')  ax[1].axvline(stats\_charges['mode'], color='orange', linestyle=':', linewidth=2, label=f'Мода ({stats\_charges["mode"]:.2f})')  ax[1].set\_title('Распределение Charges с мерами центральной тенденции')  ax[1].set\_xlabel('Charges')  ax[1].set\_ylabel('Плотность')  ax[1].legend()  ax[1].grid(True, alpha=0.3)  plt.tight\_layout()  plt.show() |



**Рисунок 4 – Статистика BMI и Charges**

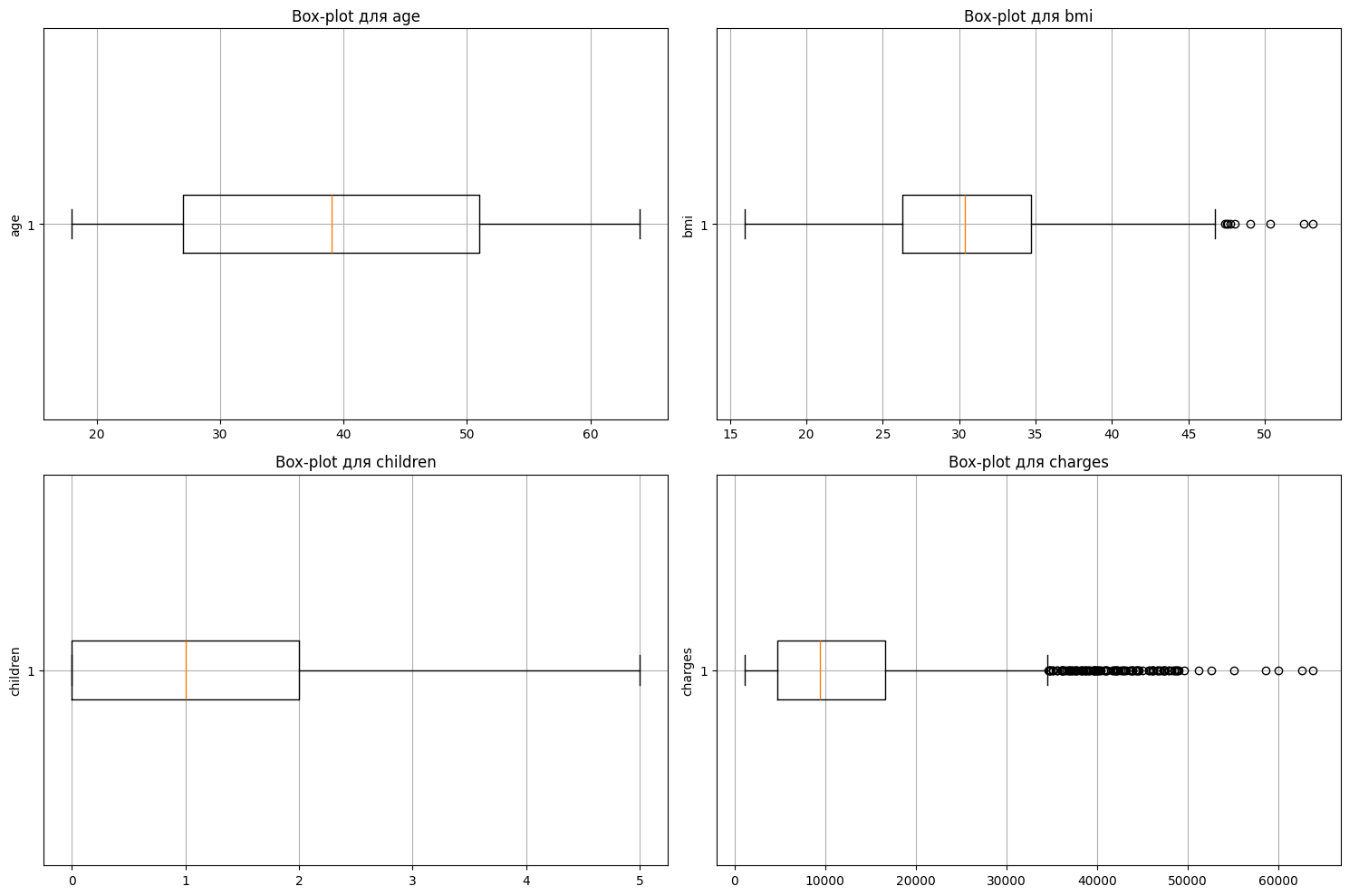
Выводы:

* BMI: Распределение близко к нормальному, все три меры центральной тенденции близки друг к другу
* Charges: Сильное правостороннее смещение - медиана значительно меньше среднего, что указывает на наличие выбросов с высокими значениями

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

Листинг 5 – Box-plot

|  |
| --- |
| # Box-plot для всех числовых признаков  fig, axes = plt.subplots(2, 2, figsize=(15, 10))  axes = axes.ravel()  for i, col in enumerate(numeric\_cols):      axes[i].boxplot(data[col], vert=False)      axes[i].set\_title(f'Box-plot для {col}')      axes[i].set\_ylabel(col)      axes[i].grid()  plt.tight\_layout()  plt.show()  # Детальный анализ выбросов для charges  Q1\_charges = data['charges'].quantile(0.25)  Q3\_charges = data['charges'].quantile(0.75)  IQR\_charges = Q3\_charges - Q1\_charges  lower\_bound = Q1\_charges - 1.5 \* IQR\_charges  upper\_bound = Q3\_charges + 1.5 \* IQR\_charges  outliers\_charges = data[(data['charges'] < lower\_bound) | (data['charges'] > upper\_bound)]  print("Количество выбросов в charges: ", len(outliers\_charges)) |



**Рисунок 5 – Box-plot для числовых показателей**

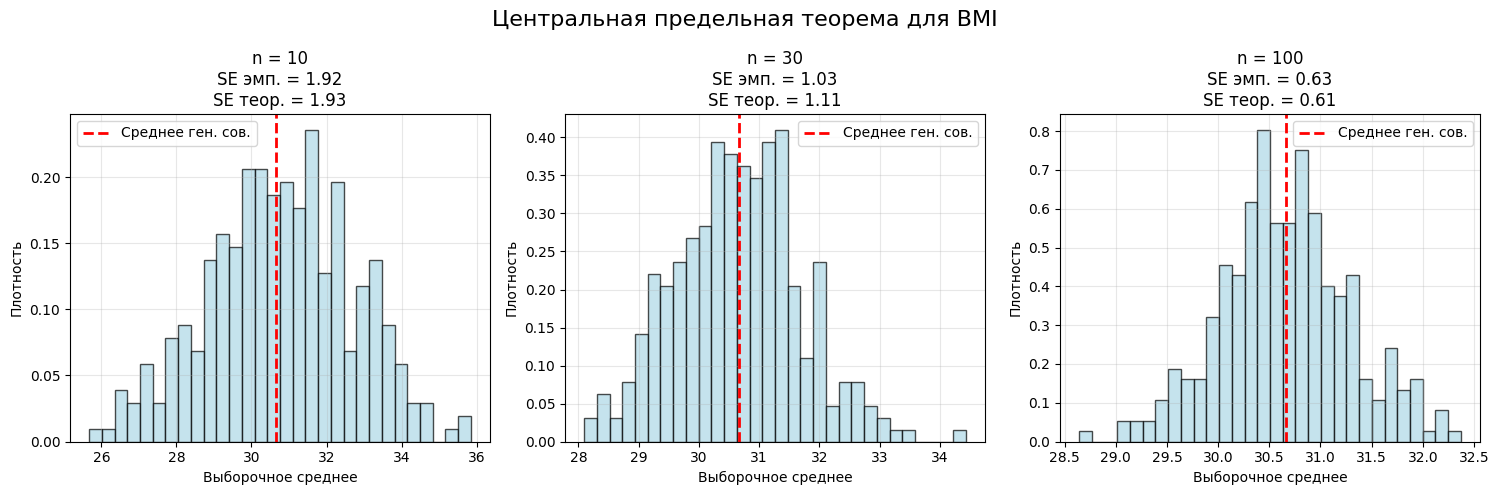
Выводы:

* Age: Нет выбросов, равномерное распределение
* BMI: Несколько выбросов в верхней части (высокий BMI)
* Children: Нет выбросов, дискретное распределение
* Charges: Много выбросов в верхней части - высокие страховые выплаты

1. 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 –

|  |
| --- |
| def check\_clt(series, name, sample\_sizes=[10, 30, 100], n\_samples=300):      fig, axes = plt.subplots(1, len(sample\_sizes), figsize=(15, 5))      population\_mean = series.mean()      population\_std = series.std(ddof=1)      print(f"Проверка ЦПТ для {name}")      print(f"Среднее генеральной совокупности: {population\_mean:.2f}")      print(f"Стандартное отклонение генеральной совокупности: {population\_std:.2f}")      for i, n in enumerate(sample\_sizes):          sample\_means = []          for \_ in range(n\_samples):              sample = np.random.choice(series, size=n, replace=True)              sample\_means.append(sample.mean())          sample\_means = np.array(sample\_means)          se\_empirical = sample\_means.std(ddof=1)          se\_theoretical = population\_std / np.sqrt(n)          axes[i].hist(sample\_means, bins=30, color='lightblue', edgecolor='black', alpha=0.7, density=True)          axes[i].axvline(population\_mean, color='red', linestyle='--', linewidth=2, label='Среднее ген. сов.')          axes[i].set\_title(f'n = {n}\nSE эмп. = {se\_empirical:.2f}\nSE теор. = {se\_theoretical:.2f}')          axes[i].set\_xlabel('Выборочное среднее')          axes[i].set\_ylabel('Плотность')          axes[i].legend()          axes[i].grid(True, alpha=0.3)          print(f"n = {n}: SE эмпирическое = {se\_empirical:.2f}, SE теоретическое = {se\_theoretical:.2f}")      plt.suptitle(f'Центральная предельная теорема для {name}', fontsize=16)      plt.tight\_layout()      plt.show()  # Проверка ЦПТ для bmi  check\_clt(data['bmi'], 'BMI') |



**Рисунок 5 – Box-plot для числовых показателей**

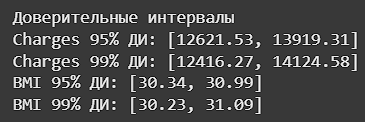
Выводы: Центральная предельная теорема подтверждается - с увеличением размера выборки:

* Распределение выборочных средних приближается к нормальному
* Стандартная ошибка уменьшается
* Эмпирическая и теоретическая стандартные ошибки близки друг к другу

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

Листинг 7 – Доверительный интервал

|  |
| --- |
| def confidence\_interval(series, confidence=0.95):      n = len(series)      mean = series.mean()      std = series.std(ddof=1)      se = std / np.sqrt(n)      if confidence == 0.95:          z = 1.96      elif confidence == 0.99:          z = 2.58      else:          z = norm.ppf(1 - (1 - confidence) / 2)      lower = mean - z \* se      upper = mean + z \* se      return lower, upper, mean, se  # Доверительные интервалы для charges  ci\_95\_charges = confidence\_interval(data['charges'], 0.95)  ci\_99\_charges = confidence\_interval(data['charges'], 0.99)  # Доверительные интервалы для bmi  ci\_95\_bmi = confidence\_interval(data['bmi'], 0.95)  ci\_99\_bmi = confidence\_interval(data['bmi'], 0.99)  print("Доверительные интервалы")  print(f"Charges 95% ДИ: [{ci\_95\_charges[0]:.2f}, {ci\_95\_charges[1]:.2f}]")  print(f"Charges 99% ДИ: [{ci\_99\_charges[0]:.2f}, {ci\_99\_charges[1]:.2f}]")  print(f"BMI 95% ДИ: [{ci\_95\_bmi[0]:.2f}, {ci\_95\_bmi[1]:.2f}]")  print(f"BMI 99% ДИ: [{ci\_99\_bmi[0]:.2f}, {ci\_99\_bmi[1]:.2f}]") |

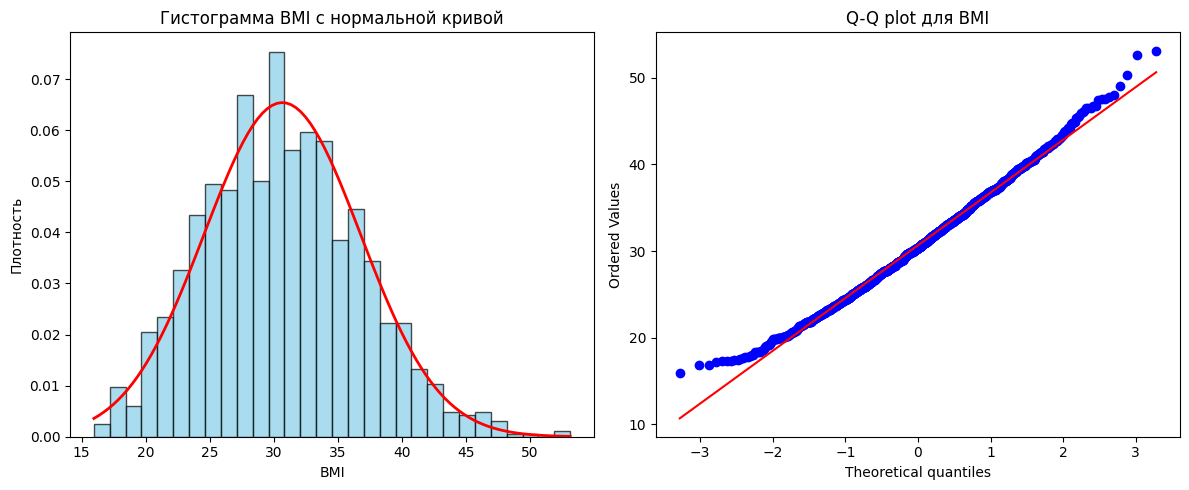


**Рисунок 6 – Интервал**

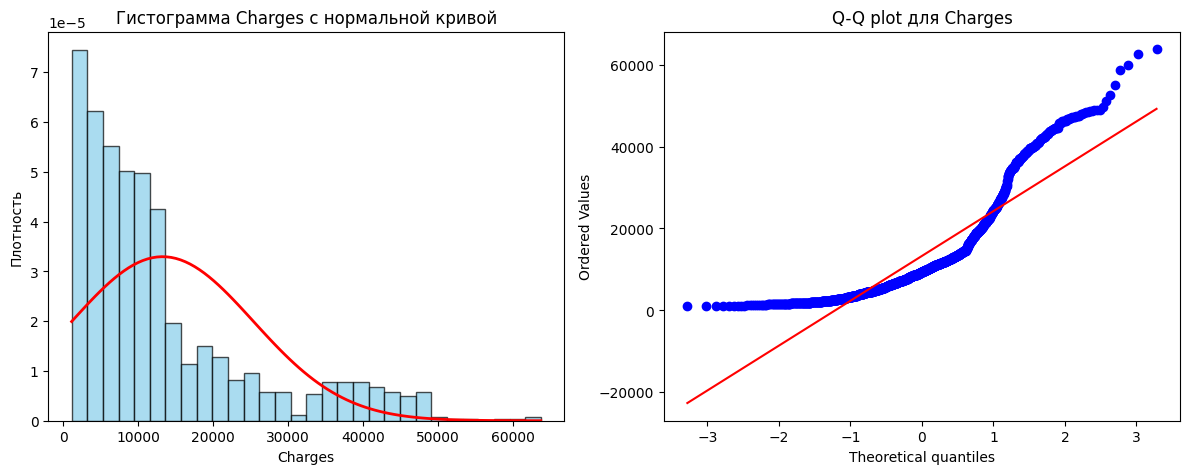
1. Проверить распределения следующих признаков на нормальность: индекс массы тела, расходы. Сформулировать нулевую и альтернативную гипотезы. Для каждого признака использовать KS-тест и q-q plot. Сделать выводы на основе полученных p-значений.

Листинг 8 – Распределение

|  |
| --- |
| def normality\_test(series, name):      # Тест Колмогорова-Смирнова      ks\_stat, ks\_p = kstest((series - series.mean()) / series.std(ddof=1), 'norm')      # Тест Шапиро-Уилка (для выборок < 5000)      if len(series) <= 5000:          shapiro\_stat, shapiro\_p = shapiro(series)      else:          shapiro\_stat, shapiro\_p = np.nan, np.nan      # Q-Q plot      fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 5))      # Гистограмма с нормальной кривой      ax1.hist(series, bins=30, density=True, alpha=0.7, color='skyblue', edgecolor='black')      x = np.linspace(series.min(), series.max(), 100)      ax1.plot(x, norm.pdf(x, series.mean(), series.std(ddof=1)), 'r-', linewidth=2)      ax1.set\_title(f'Гистограмма {name} с нормальной кривой')      ax1.set\_xlabel(name)      ax1.set\_ylabel('Плотность')      # Q-Q plot      sts.probplot(series, dist="norm", plot=ax2)      ax2.set\_title(f'Q-Q plot для {name}')      plt.tight\_layout()      plt.show()      print(f"Проверка нормальности для {name}")      print(f"Тест Колмогорова-Смирнова: статистика = {ks\_stat:.4f}, p-value = {ks\_p:.4f}")      if not np.isnan(shapiro\_p):          print(f"Тест Шапиро-Уилка: статистика = {shapiro\_stat:.4f}, p-value = {shapiro\_p:.4f}")      # Формулировка гипотез      print()      print("Гипотезы:")      print("H0: Распределение соответствует нормальному")      print("H1: Распределение не соответствует нормальному")      alpha = 0.05      if ks\_p > alpha:          print(f"p-value ({ks\_p:.4f}) > {alpha} - нет оснований отвергать H0")      else:          print(f"p-value ({ks\_p:.4f}) ≤ {alpha} - отвергаем H0 в пользу H1")  # Проверка нормальности для BMI  normality\_test(data['bmi'], 'BMI')  # Проверка нормальности для Charges  normality\_test(data['charges'], 'Charges') |



**Рисунок 7 – Гистограмма и q-q plot для BMI**

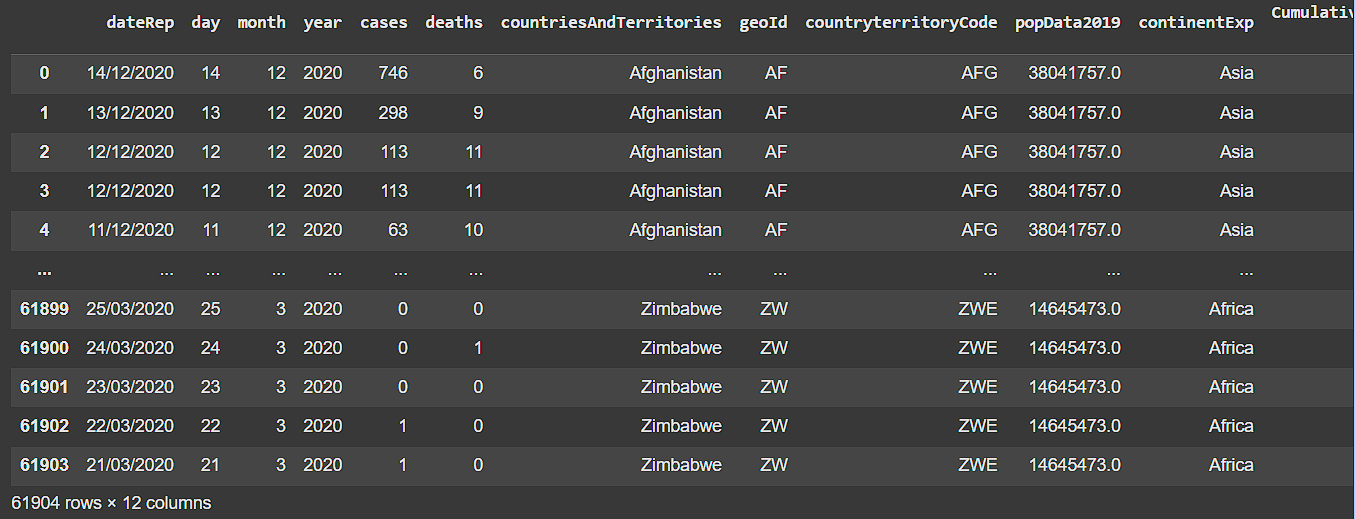


**Рисунок 8 – Гистограмма и q-q plot для Charges**

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

Листинг 9 – Загрузка файла

|  |
| --- |
| ecdc\_data = pd.read\_csv("ECDCCases.csv")  ecdc\_data |



**Рисунок 9 – ECDCC.csv**

1. 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.

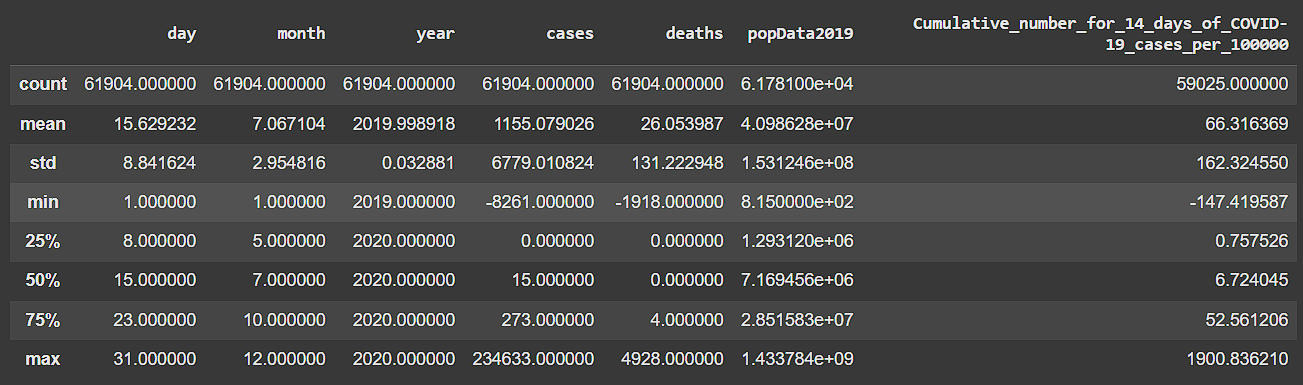
Листинг 10 – Проверка на наличие пропуска

|  |
| --- |
| ecdc\_data.isna().sum()  missing\_percent = (ecdc\_data.isna().sum() / len(ecdc\_data)) \* 100  print("Процент пропусков по признакам:")  print(missing\_percent.sort\_values(ascending=False))  # Удаление двух признаков с наибольшим количеством пропусков  columns\_to\_drop = missing\_percent.nlargest(2).index  ecdc\_cleaned = ecdc\_data.drop(columns=columns\_to\_drop)  print(f"Удалены признаки: {list(columns\_to\_drop)}")  # Обработка оставшихся пропусков  for col in ecdc\_cleaned.columns:      if ecdc\_cleaned[col].isna().sum() > 0:          if ecdc\_cleaned[col].dtype == 'object':              ecdc\_cleaned[col].fillna('other', inplace=True)          else:              ecdc\_cleaned[col].fillna(ecdc\_cleaned[col].median(), inplace=True)  print("Пропуски обработаны")  print(f"Осталось пропусков: {ecdc\_cleaned.isna().sum().sum()}") |

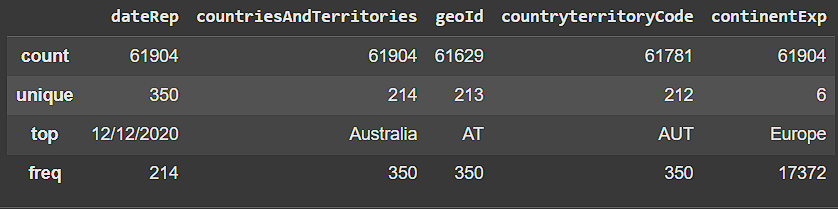
1. Посмотреть статистику по данным, используя describe(). Сделать выводы о том, какие признаки содержат выбросы. Посмотреть, для каких стран смертей в день превысило 3000 и сколько таких дней было.

Листинг 11 – Статистика данных

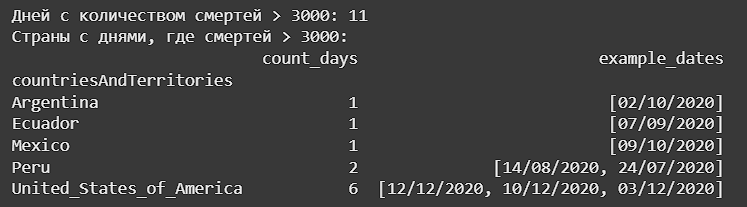
|  |
| --- |
| ecdc\_data.describe()  ecdc\_data.describe(include=['object'])  high\_death\_days = ecdc\_data[ecdc\_data['deaths'] > 3000]  print("Дней с количеством смертей > 3000:", len(high\_death\_days))  if len(high\_death\_days) > 0:      print("Страны с днями, где смертей > 3000:")      high\_death\_countries = high\_death\_days.groupby('countriesAndTerritories').agg({          'deaths': 'count',          'dateRep': lambda x: list(x)[:3]  # первые 3 даты      }).rename(columns={'deaths': 'count\_days', 'dateRep': 'example\_dates'})      print(high\_death\_countries) |



**Рисунок 10 – Статистика по числовым признакам ECDCC.csv**



**Рисунок 11 – Статистика по категориальным признакам ECDCC.csv**



**Рисунок 12 – ECDCC.csv смертей >3000**

1. Найти дублирование данных. Удалить дубликаты.

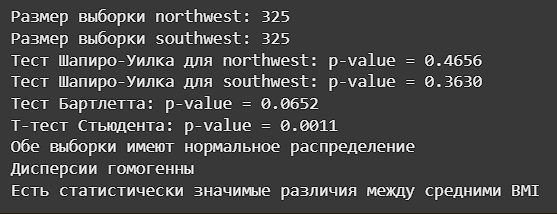
Листинг 12 – Удаление дубликатов

|  |
| --- |
| ecdc\_data.shape  # Поиск дубликатов  duplicates = ecdc\_data.duplicated().sum()  print("Найдено полных дубликатов:", duplicates)  # Удаление дубликатов  ecdc\_no\_duplicates = ecdc\_data.drop\_duplicates()  print("Размер данных после удаления дубликатов:", ecdc\_no\_duplicates.shape) |

1. 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 Шопиро Уилка) и на гомогенность дисперсии (критерий Бартлетта).

Листинг 13 – Загрузка данных и работа с тестами

|  |
| --- |
| bmi\_data = pd.read\_csv("bmi.csv")  bmi\_data  # Создание выборок для northwest и southwest  bmi\_nw = bmi\_data[bmi\_data['region'] == 'northwest']['bmi']  bmi\_sw = bmi\_data[bmi\_data['region'] == 'southwest']['bmi']  print(f"Размер выборки northwest: {len(bmi\_nw)}")  print(f"Размер выборки southwest: {len(bmi\_sw)}")  # Проверка нормальности  shapiro\_nw = shapiro(bmi\_nw)  shapiro\_sw = shapiro(bmi\_sw)  print(f"Тест Шапиро-Уилка для northwest: p-value = {shapiro\_nw[1]:.4f}")  print(f"Тест Шапиро-Уилка для southwest: p-value = {shapiro\_sw[1]:.4f}")  # Проверка гомогенности дисперсий  bartlett\_test = bartlett(bmi\_nw, bmi\_sw)  print(f"Тест Бартлетта: p-value = {bartlett\_test[1]:.4f}")  # T-тест  t\_test = ttest\_ind(bmi\_nw, bmi\_sw, equal\_var=True)  print(f"T-тест Стьюдента: p-value = {t\_test[1]:.4f}")  # Интерпретация  alpha = 0.05  if shapiro\_nw[1] > alpha and shapiro\_sw[1] > alpha:      print("Обе выборки имеют нормальное распределение")  else:      print("Нарушено предположение о нормальности")  if bartlett\_test[1] > alpha:      print("Дисперсии гомогенны")  else:      print("Дисперсии негомогенны")  if t\_test[1] > alpha:      print("Нет статистически значимых различий между средними BMI")  else:      print("Есть статистически значимые различия между средними BMI") |

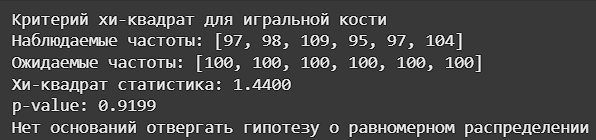


**Рисунок 13 – Вывод результатов**

1. Кубик бросили 600 раз, получили следующие результаты: С помощью критерия Хи-квадрат проверить, является ли полученное распределение равномерным. scipy.stats.chisquare().

Листинг 14 – Критерий Хи-квадрат

|  |
| --- |
| # Данные о бросках кубика  observed = [97, 98, 109, 95, 97, 104]  expected = [100] \* 6  # Критерий хи-квадрат  chi2\_stat, p\_value = sts.chisquare(observed, expected)  print("Критерий хи-квадрат для игральной кости")  print(f"Наблюдаемые частоты: {observed}")  print(f"Ожидаемые частоты: {expected}")  print(f"Хи-квадрат статистика: {chi2\_stat:.4f}")  print(f"p-value: {p\_value:.4f}")  alpha = 0.05  if p\_value > alpha:  print("Нет оснований отвергать гипотезу о равномерном распределении")  else:  print("Распределение не является равномерным") |

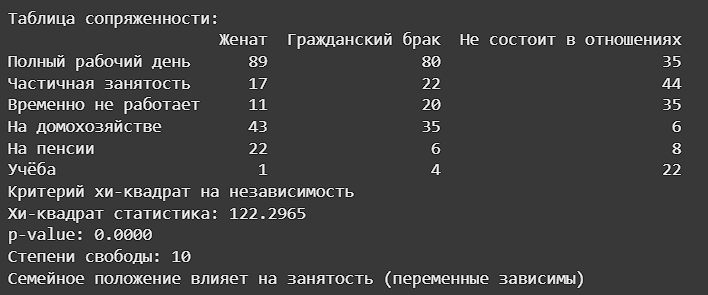


**Рисунок 14 – Результат Хи-квадрат**

1. С помощью критерия Хи-квадрат проверить, являются ли переменные зависимыми. Создать датафрейм, используя следующий код: Использовать функцию scipy.stats.chi2\_contingency(). Влияет ли семейное положение на занятость?

Листинг 15 – Создание датафрейма

|  |
| --- |
| # Создание датафрейма  data\_employment = 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\_employment.index = [      'Полный рабочий день', 'Частичная занятость', 'Временно не работает',      'На домохозяйстве', 'На пенсии', 'Учёба'  ]  print("Таблица сопряженности:")  print(data\_employment)  # Критерий хи-квадрат на независимость  chi2\_stat, p\_value, dof, expected = chi2\_contingency(data\_employment)  print(f"Критерий хи-квадрат на независимость")  print(f"Хи-квадрат статистика: {chi2\_stat:.4f}")  print(f"p-value: {p\_value:.4f}")  print(f"Степени свободы: {dof}")  alpha = 0.05  if p\_value > alpha:      print("Семейное положение НЕ влияет на занятость (переменные независимы)")  else:      print("Семейное положение влияет на занятость (переменные зависимы)") |



**Рисунок 15 – Вывод таблицы**

Результат работы:

Данную работу можете увидеть в блокноте Jupyter Notebook.

<https://drive.google.com/file/d/18bXovOCyW-UwiRWj0hgBeRW4eRb_tIwU/view?usp=sharing>

Вывод:

Были изучены и применены ключевые статистические методы, включая проверку гипотез, анализ распределений, построение гистограмм, доверительных интервалов и проверку на нормальность.

Список использованных источников и литературы:

1. Ростовцев В.С. Искусственные нейронные сети,   
   Издательство "Лань", 2019. — 216 с. — URL: https://e.lanbook.com/book/122180
2. Араки М. Манга: Машинное обучение,   
   Издательство "ДМК Пресс", 2020. — 214 с. — URL: <https://e.lanbook.com/book/179473>
3. Платонов, А. В. Машинное обучение : учебное пособие для вузов / А. В. Платонов. — Москва : Издательство Юрайт, 2022. — 85 с. — (Высшее образование). — ISBN 978-5-534-15561-7. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/508804