Лабораторная работа №2

Выполнил: Величко Максим Иванович, М32061

Варианты для задания: 1,3.

Весь код для удобства выложен на мой GitHub(https://github.com/maksve11/ITMO\_MatStat\_sem4/tree/main/lab2)

№1

Методом моментов найти оценку параметра 𝜃 равномерного распределения на [−𝜃, 𝜃]. Найти смещение оценки, дисперсию, среднеквадратическую ошибку. Эксперимент для 𝜃 = 10.

Для начала определимся с методом моментов для равномерного распределения. Пусть 𝑋1, ..., 𝑋𝑛 — выборка из равномерного распределения на [−𝜃, 𝜃]. Тогда оценка параметра 𝜃 методом моментов будет равна:

𝜃̂ = √(3(𝑋̅²)),

где 𝑋̅ — выборочное среднее.

Смещение оценки:

Смещение оценки равно разности между ожидаемым значением оценки и истинным значением параметра:

Bias(𝜃̂) = E(𝜃̂) - 𝜃.

Найдем ожидаемое значение оценки:

E(𝜃̂) = E[√(3(𝑋̅²))] = √(3)E[𝑋̅²]

Используя то, что E(𝑋̅²) = Var(𝑋̅) + (E(𝑋̅))^2, где Var(𝑋̅) = Var(𝑋) / n, где 𝑋 имеет равномерное распределение на [−𝜃, 𝜃], получим:

E(𝜃̂) = √(3) \* (Var(𝑋) / n + (E(𝑋))^2) = √(3) \* (𝜃² / (3n) + 0) = 𝜃 / √(n)

Тогда смещение оценки равно:

Bias(𝜃̂) = 𝜃 / √(n) - 𝜃 = -𝜃 / √(n)

Дисперсия оценки:

Дисперсия оценки равна Var(𝜃̂) = E(𝜃̂²) - (E(𝜃̂))^2.

Найдем E(𝜃̂²):

E(𝜃̂²) = E[3(𝑋̅²)] = 3E[𝑋̅²] = 3(Var(𝑋) / n + (E(𝑋))^2) = 3(𝜃² / (3n) + 0) = 𝜃² / n

Тогда дисперсия оценки равна:

Var(𝜃̂) = 𝜃² / n - (𝜃 / √(n))^2 = (2/𝑛)𝜃²

Среднеквадратическая ошибка:

Среднеквадратическая ошибка равна MSE(𝜃̂) = E[(𝜃̂ - 𝜃)^2].

MSE(𝜃̂) = Var(𝜃̂) + Bias(𝜃̂)² = (2/𝑛)𝜃² + (𝜃 / √(n))^2 = (2/𝑛)𝜃² + (𝜃² / n) = (3/𝑛)𝜃².

Теперь, решим задачу для конкретного значения параметра 𝜃 = 10:

import numpy as np  
import matplotlib.pyplot as plt  
  
np.random.seed(42)  
n = 1000  
theta = 10  
X = np.random.uniform(low=-theta, high=theta, size=n)  
theta\_hat = np.sqrt(3 \* np.mean(X\*\*2))  
bias = -theta / np.sqrt(n)  
variance = (2 / n) \* theta\*\*2  
mse = (3 / n) \* theta\*\*2  
  
print("Оценка методом моментов:", theta\_hat)  
print("Смещение оценки:", bias)  
print("Дисперсия оценки:", variance)  
print("Среднеквадратическая ошибка:", mse)

Результат:

Изображение выглядит как текст

Автоматически созданное описание

Сгенерируйте 500 выборок объема 50 с указанным значением параметра 𝜃. Сколько раз оценка отклонится от истинного значения параметра более чем на 0.01? То же самое сде-лать для объемов выборки 100, 500, 1000, 2500. Визуализируйте результат. Как объяснить полученный результат?

Для решения этой задачи мы можем написать функцию, которая будет генерировать 500 выборок заданного размера и оценивать параметр методом моментов. Затем, мы можем посчитать количество раз, когда оценка отклонилась от истинного значения более чем на 0.01.

def generate\_samples(n, theta):  
 np.random.seed(42)  
 samples = np.random.uniform(low=-theta, high=theta, size=(500, n))  
 estimates = np.sqrt(3 \* np.mean(samples\*\*2, axis=1))  
 deviations = np.abs(estimates - theta)  
 return np.sum(deviations > 0.01)  
  
  
theta = 10  
sample\_sizes = [50, 100, 500, 1000, 2500]  
  
deviations = []  
for n in sample\_sizes:  
 deviation = generate\_samples(n, theta)  
 deviations.append(deviation)  
 print(f"Объем выборки: {n}, Смещение: {deviation}")  
  
plt.plot(sample\_sizes, deviations, 'bo-')  
plt.xlabel('Объем выборки')  
plt.ylabel('номер смещения > 0.01')  
plt.show()

Результаты:

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как диаграмма

Автоматически созданное описание

№2

Методом максимального правдоподобия найти оценку параметра 𝜃 биномиального распределения Bin(𝑛, 𝜃), считая 𝑛 известным. Найти смещение оценки, дисперсию, среднеквадратическую ошибку. Является ли найденная оценка эффективной? Эксперимент при 𝑛 = 4, 𝜃 = 1/5.

Для нахождения оценки параметра методом максимального правдоподобия мы должны записать функцию правдоподобия и найти ее максимум. Для биномиального распределения Bin(𝑛, 𝜃), функция правдоподобия имеет вид:

L(𝜃|x1, x2, ..., xn) = f(x1; 𝜃) \* f(x2; 𝜃) \* ... \* f(xn; 𝜃) = 𝑛!/(𝑥1! \* 𝑥2! \* ... \* 𝑥𝑛!) \* 𝜃^(𝑥1 + 𝑥2 + ... + 𝑥𝑛) \* (1-𝜃)^(𝑛-𝑥1 + 𝑛-𝑥2 + ... + 𝑛-𝑥𝑛)

где xi - результат i-го испытания, 𝑥1 + 𝑥2 + ... + 𝑥𝑛 - общее число успехов, 𝑛 - общее число испытаний.

Прологарифмируем функцию правдоподобия и найдем ее максимум, находим производную логарифма функции правдоподобия по 𝜃 и приравниваем ее к нулю:

ln L(𝜃|x1, x2, ..., xn) = ln(𝑛!/(𝑥1! \* 𝑥2! \* ... \* 𝑥𝑛!)) + (𝑥1 + 𝑥2 + ... + 𝑥𝑛) \* ln 𝜃 + (𝑛-𝑥1 + 𝑛-𝑥2 + ... + 𝑛-𝑥𝑛) \* ln (1-𝜃)

d/d𝜃 ln L(𝜃|x1, x2, ..., xn) = (𝑥1 + 𝑥2 + ... + 𝑥𝑛)/𝜃 - (𝑛-𝑥1 + 𝑛-𝑥2 + ... + 𝑛-𝑥𝑛)/(1-𝜃) = 0

Отсюда получаем, что оценка методом максимального правдоподобия равна:

𝜃\_hat = (𝑥1 + 𝑥2 + ... + 𝑥𝑛)/𝑛

Найдем смещение оценки, дисперсию и среднеквадратическую ошибку:

Смещение:

E(𝜃\_hat) - 𝜃 = E((𝑥1 + 𝑥2 + ... + 𝑥𝑛)/𝑛) - 𝜃 = 𝜃

Таким образом, оценка методом максимального правдоподобия несмещенная.

Дисперсия:

Var(𝜃\_hat) = Var((𝑥1 + 𝑥2 + ... + 𝑥𝑛)/𝑛) = (1/𝑛^2) \* Var(𝑥1 + 𝑥2 + ... + 𝑥𝑛) = (1/𝑛^2) \* (𝑛 \* 𝜃 \* (1-𝜃))

Таким образом, дисперсия оценки равна:

Var(𝜃\_hat) = (𝜃 \* (1-𝜃))/𝑛

Среднеквадратическая ошибка:

MSE(𝜃\_hat) = E((𝜃\_hat - 𝜃)^2) = Var(𝜃\_hat) + Bias^2(𝜃\_hat, 𝜃) = (𝜃 \* (1-𝜃))/𝑛

Оценка методом максимального правдоподобия является эффективной, так как она достигает нижней границы для дисперсии любой другой несмещенной оценки.

Теперь проведем эксперимент в Python для 𝑛 = 4, 𝜃 = 1/5:

import numpy as np  
  
# Задаем параметры распределения  
n = 4  
theta = 1/5  
  
# Генерируем выборку размера 50  
sample = np.random.binomial(n, theta, size=50)  
  
# Находим оценку параметра методом максимального правдоподобия  
theta\_hat = np.mean(sample)/n  
  
# Находим смещение, дисперсию и среднеквадратическую ошибку  
bias = theta\_hat - theta  
variance = (theta \* (1-theta))/n  
mse = variance + bias\*\*2  
  
print("Оценка методом максимального правдоподобия:", theta\_hat)  
print("Смещение:", bias)  
print("Дисперсия:", variance)  
print("Среднеквадратическая ошибка:", mse)

Результаты:

Изображение выглядит как текст

Автоматически созданное описание

Сгенерируйте 500 выборок объема 50 с указанным значением параметра 𝜃. Сколько раз оценка отклонится от истинного значения параметра более чем на 0.01? То же самое сде-лать для объемов выборки 100, 500, 1000, 2500. Визуализируйте результат. Как объяснить полученный результат?

Для проведения эксперимента посчитаем, сколько раз оценка методом максимального правдоподобия отклонится от истинного значения параметра более чем на 0.01 для каждого из объемов выборки. Для этого сгенерируем 500 выборок и для каждой выборки найдем оценку параметра методом максимального правдоподобия и посчитаем ее отклонение от истинного значения параметра. Затем посчитаем количество таких выборок и выразим это количество в процентах от общего числа выборок:

delta = 0.01  
sample\_sizes = [50, 100, 500, 1000, 2500]  
  
# Определяем число экспериментов  
n\_experiments = 500  
  
# Создаем массивы для хранения результатов  
proportions = np.zeros(len(sample\_sizes))  
  
for i, sample\_size in enumerate(sample\_sizes):  
 # Создаем массив для хранения отклонений  
 deviations = np.zeros(n\_experiments)  
  
 for j in range(n\_experiments):  
 # Генерируем выборку  
 sample = np.random.binomial(n, theta, size=sample\_size)  
  
 # Находим оценку параметра методом максимального правдоподобия  
 theta\_hat = np.mean(sample) / n  
  
 # Считаем отклонение от истинного значения параметра  
 deviation = abs(theta\_hat - theta)  
  
 # Добавляем отклонение в массив  
 deviations[j] = deviation  
  
 # Считаем количество выборок с отклонением более, чем на delta  
 proportion = (deviations > delta).sum() / n\_experiments  
  
 # Записываем результат  
 proportions[i] = proportion  
  
# Выводим результаты  
for i, sample\_size in enumerate(sample\_sizes):  
 print(f"Доля выборок объема {sample\_size}, отклонившихся более чем на {delta}: {proportions[i] \* 100:.2f}%")

Результаты:

Изображение выглядит как текст

Автоматически созданное описание

Вывод: из результатов видно, что при малых размерах выборки оценка методом моментов достаточно неточна и существенно отклоняется от истинного значения параметра. С увеличением размера выборки точность оценки улучшается и число ошибок существенно уменьшается. При размере выборки 2500 ошибка практически отсутствует.