Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет программной инженерии и вычислительной техники

Отчет

По лабораторной работе № 4

по дисциплине «Математическая статистика»

Вариант 1, 4

Выполнила: Старостина Е. П., группа P3219

Преподаватель: Лимар И. А.

г. Санкт-Петербург
2025

Цель работы:

Попрактиковаться в построении линейных моделей, вычислении оценок коэффициентов модели и остаточной дисперсии, построении доверительных интервалов.

Задание 1:

Код:

```
import numpy as np
import scipy.stats as stats

variables = []
prices = []

with open('cars93.csv') as f:
    lines = f.readlines()
    for i in lines[1:]:
        line = i.strip().split(',')
        variables.append([float(line[6]), float(line[7]), float(line[12])])
        prices.append(float(line[4]))

X = np.array(variables)
y = np.array(prices)

X = np.hstack([np.ones((X.shape[0], 1)), X])

XtX = X.T @ X
Xty = X.T @ y
beta = np.linalg.solve(XtX, Xty)

y_pred = X @ beta
residuals = y - y_pred
sigma_squared = (residuals.T @ residuals) / (len(y) - X.shape[1])

total_variance = ((y - y.mean()).T @ (y - y.mean()))
r_squared = 1 - (residuals.T @ residuals) / total_variance

XtX_inv = np.linalg.inv(XtX)
```

```
se_beta = np.sqrt(np.diag(sigma_squared * XtX_inv))


n, p = X.shape
t_value = stats.t.ppf(1 - 0.05 / 2, df=n - p)


ci_beta = [(beta[i] - t_value * se_beta[i], beta[i] + t_value * se_beta[i])
for i in range(len(beta))]


chi2_lower = stats.chi2.ppf(0.05 / 2, df=n - p)
chi2_upper = stats.chi2.ppf(1 - 0.05 / 2, df=n - p)
ci_sigma_squared = ((n - p) * sigma_squared / chi2_upper, (n - p) *
sigma_squared / chi2_lower)


print("Коэффициенты модели:", beta)
print("Доверительные интервалы для коэффициентов:")
for i, ci in enumerate(ci_beta):
    print(f"b{i}: {ci}")
print("Остаточная дисперсия:", sigma_squared)
print("Доверительный интервал для остаточной дисперсии:", ci_sigma_squared)
print("Коэффициент детерминации (R^2):", r_squared)



# второе задание
t_stat = beta[1] / se_beta[1]
p_value = 2 * (1 - stats.t.cdf(abs(t_stat), df=len(y) - X.shape[1]))
print(f"t-статистика расхода топлива в городе: {t_stat}")
print(f"p-value расхода топлива в городе: {p_value}")


if p_value < 0.05:
    print("Коэффициент расхода в городе значим")
else:
    print("Коэффициент расхода в городе не значим")



# третье задание
t_stat = beta[2] / se_beta[2]
p_value = 2 * (1 - stats.t.cdf(abs(t_stat), df=len(y) - X.shape[1]))
```

```
print(f"t-статистика расхода топлива на шоссе: {t_stat}")
print(f"p-value расхода топлива на шоссе: {p_value}")


if p_value < 0.05:
    print("Коэффициент расхода на шоссе значим")
else:
    print("Коэффициент расхода на шоссе не значим")
```

Результат работы программы:

```
Коэффициенты модели: [ 6.68867892 -0.03860018 -0.17885863  0.13131383]
Доверительные интервалы для коэффициентов:
b0: (-5.210205050961368, 18.587562894626235)
b1: (-0.7481224455443461, 0.6709220880283578)
b2: (-0.883501689262523, 0.5257844353877957)
b3: (0.099311208691445, 0.16331645547598622)
Остаточная дисперсия: 35.69468573318614
Доверительный интервал для остаточной дисперсии: (27.154902343428706,
49.03013144303709)
Коэффициент детерминации (R^2): 0.6299138920082776
t-статистика расхода топлива в городе: -0.10809771106501996
p-value расхода топлива в городе: 0.9141615110842902
Коэффициент расхода в городе не значим
t-статистика расхода топлива на шоссе: -0.504352204610917
p-value расхода топлива на шоссе: 0.6152602731348269
Коэффициент расхода на шоссе не значим
```

(В коэффициентах первое число – свободная переменная, далее расход в городе, расход на шоссе, мощность)

Проверка подозрений:

1. Чем больше мощность, тем больше цена – верно, коэффициент при мощности (3-ая переменная) положителен
2. Цена изменяется в зависимости от расхода в городе: коэффициент при расходе в городе (1-ая переменная) не равен нулю, но близок к нему. Проверяя гипотезу о том, что коэффициент расхода в городе значим против альтернативы (не значим), получаем p_value 0.91, отвергаем гипотезу о значимости расхода в городе.
3. H_0: коэффициенты при расходе в городе и расходе на шоссе одновременно равны нулю. Из предыдущего задания: коэффициент расхода топлива в городе не значим. Аналогичным образом устанавливаем, что коэффициент расхода топлива на шоссе также не значим. Принимаем H_0.

Задание 2:

Код:

```
import numpy as np
from scipy.stats import f

price_range = []
battery = []

with open('mobile_phones.csv') as file:
    lines = file.readlines()
    for i in lines[1:]:
        line = i.strip().split(',')
        price_range.append(int(line[-1]))
        battery.append(int(line[0]))

price_range = np.array(price_range)
battery = np.array(battery)

groups = [battery[price_range == level] for level in np.unique(price_range)]

overall_mean = np.mean(battery)

ssb = sum(len(group) * (np.mean(group) - overall_mean) ** 2 for group in
groups)
ssw = sum(np.sum((group - np.mean(group)) ** 2) for group in groups)

df_between = len(groups) - 1
df_within = len(battery) - len(groups)

msb = ssb / df_between
msw = ssw / df_within

f_stat = msb / msw

p_value = f.sf(f_stat, df_between, df_within)
```

```
print("f-статистика:", f_stat)
print("p-value:", p_value)

if p_value < 0.05:
    print("Отвергнуть H0")
else:
    print("Принять H0")
```

Результат работы программы:

```
f-статистика: 31.59815753989732
p-value: 5.948688277085545e-20
Отвергнуть H0
```

Выводы по работе:

В ходе лабораторной работы я попрактиковалась в построении линейных моделей, вычислении оценок коэффициентов модели и остаточной дисперсии, построении доверительных интервалов.

Приложение:

Ссылка на код:

https://github.com/Starostina-elena/math_stat_lab4