МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Теория принятия решений»

Тема: Использование модели упреждающей связи при принятии решений

Студентка гр. 1384	____________________	Усачева Д.В.
Преподаватель	____________________	Попова Е.В.

Санкт-Петербург

2025

Цель работы.

Смоделировать упреждающую связь и выявить закономерности изменения данных при помощи корректировки вероятности гипотез с помощью теоремы Байеса.

1. Изменить условие задачи, считая полную вероятность антиципацитным значением. Сосчитать не 2 итерации, а 20.
2. Подсчитать первые 3 итерации вручную и подробно их отобразить в работе с помощью вставки/уравнения или фотографии аккуратно написанного текста.
3. Сосчитать последующие итерации с помощью языков программирования. Предоставить код в Приложении. (пояснения в работе)
4. Прибегнуть к аппроксимации с помощью инструментальных средств. (пояснения в работе)
5. Сделать вывод на каком шаге с погрешностью $\varepsilon=0.01$ аппроксимационная кривая отличается от порогового значения.

Задание.

Вариант 17.

Однотипные приборы выпускаются 3 заводами в отношении 2:4:5, причём вероятность брака для этих заводов соответственно равны 0,06; 0,04; 0,07. Приобретённый прибор оказался не бракованным.

1) Какова вероятность того, что он изготовлен заводом, содержащим меньший процент брака.

2) Найти ту же вероятность, если взятый прибор возвращен на завод, после чего из этого завода взяли прибор, который оказался не бракованным.

Выполнение работы.

Формула Байеса

$$P(A \mid B) = \frac{P(B \mid \text{A})P(\text{A})}{P(B)}$$

1) Найдем вероятность того, что не бракованный прибор изготовлен заводом, содержащим меньший процент брака.

$P(B) = P(B \mid A_1)P(A_1) + P(B \mid A_2)P(A_2) + P(B \mid A_3)P(A_3) = \frac{2}{11} \cdot 0.94 + \frac{4}{11} \cdot 0.96 + \frac{5}{11} \cdot 0.93 = \frac{10.37}{11} \approx 0.9427$ - вероятность того, что приобретённый прибор оказался не бракованным.

$P(A_1) = 2/11$ - вероятность того, что прибор изготовлен 1 заводом.

$P(A_2) = 4/11$ - вероятность того, что прибор изготовлен 2 заводом.

$P(A_3) = 5/11$ - вероятность того, что прибор изготовлен 3 заводом.

Соответственно, вероятность:

$$P(A_2|B) = \frac{\frac{4}{11} \cdot 0.96}{\frac{10.37}{11}} = \frac{384}{1037} \approx 0.3703$$

Найдем остальные исходы:

$$P(A_1|B) = \frac{\frac{2}{11} \cdot 0.94}{\frac{10.37}{11}} = \frac{188}{1037} \approx 0.1813$$

$$P(A_3|B) = \frac{\frac{5}{11} \cdot 0.93}{\frac{10.37}{11}} = \frac{465}{1037} \approx 0.4484$$

2) Найдем ту же вероятность, если взятый прибор возвращен на завод, после чего из этого завода взяли прибор, который оказался не бракованным.

Заметим, что $P\big(B \mid (A_i \mid B)\big) = P(B \mid A_i)$. Обозначим $P(A_i \mid B)$ как $P(\text{C}_i)$.

Пересчитаем $P(B)$:

$$P(B) = \frac{188}{1037} \cdot 0.94 + \frac{384}{1037} \cdot 0.96 + \frac{465}{1037} \cdot 0.93 = \frac{977.81}{1037} \approx 0.9429$$

$$P(C_2 \mid B) = \frac{\frac{384}{1037} \cdot 0.96}{\frac{977.81}{1037}} = \frac{36864}{97781} \approx 0.3770$$

Вероятность события увеличилась.

Найдем остальные исходы:

$$P(C_1|B) = \frac{\frac{188}{1037} \cdot 0.94}{\frac{977.81}{1037}} = \frac{17672}{97781} \approx 0.1807$$

$$P(C_3|B) = \frac{\frac{465}{1037} \cdot 0.93}{\frac{977.81}{1037}} = \frac{43245}{97781} \approx 0.4422$$

3) Проведем 3 итерацию

Обозначим $P(C_i \mid B)$ как $P(D_i)$.

Пересчитаем $P(B)$:

$$P(B) = \frac{17672}{97781} \cdot 0.94 + \frac{36864}{97781} \cdot 0.96 + \frac{43245}{97781} \cdot 0.93 = \frac{92218.97}{97781} \approx 0.9431$$

$$P(D_2 \mid B) = \frac{\frac{36864}{97781} \cdot 0.96}{\frac{92218.97}{97781}} = \frac{3538944}{9221897} \approx 0.3837$$

Вероятность события увеличилась.

Сосчитаем последующие итерации с помощью программы на Python (Приложение А).

Листинг 1 – Результаты итераций

```
Шаг 1:
Вероятность: 0.3703
Шаг 2:
Вероятность: 0.3770
Шаг 3:
Вероятность: 0.3838
Шаг 4:
Вероятность: 0.3905
Шаг 5:
Вероятность: 0.3974
```

```
Шаг 6:
Вероятность: 0.4042
Шаг 7:
Вероятность: 0.4111
Шаг 8:
Вероятность: 0.4180
Шаг 9:
Вероятность: 0.4250
Шаг 10:
Вероятность: 0.4320
Шаг 11:
Вероятность: 0.4389
Шаг 12:
Вероятность: 0.4460
Шаг 13:
Вероятность: 0.4530
Шаг 14:
Вероятность: 0.4600
Шаг 15:
Вероятность: 0.4671
Шаг 16:
Вероятность: 0.4741
Шаг 17:
Вероятность: 0.4812
Шаг 18:
Вероятность: 0.4882
Шаг 19:
Вероятность: 0.4953
Шаг 20:
Вероятность: 0.5024
```

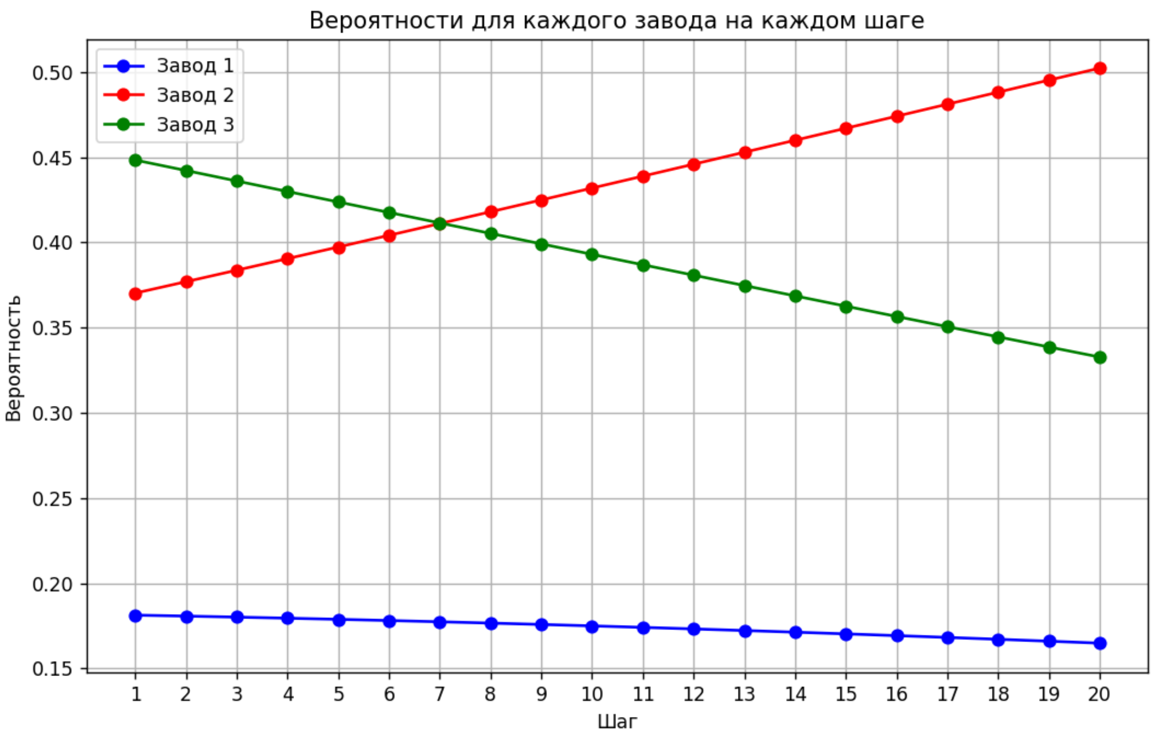

Рисунок 1 – Результаты итераций

Аппроксимация выполняется с помощью полиномиальной регрессии. Мы используем функцию np.polyfit из библиотеки NumPy для нахождения коэффициентов полинома заданной степени, который наилучшим образом описывает данные (вероятности 20 итераций). Затем с помощью np.poly1d строим полиномиальную функцию и вычисляем её значения для построения аппроксимирующей кривой. Для визуализации используется библиотека Matplotlib.

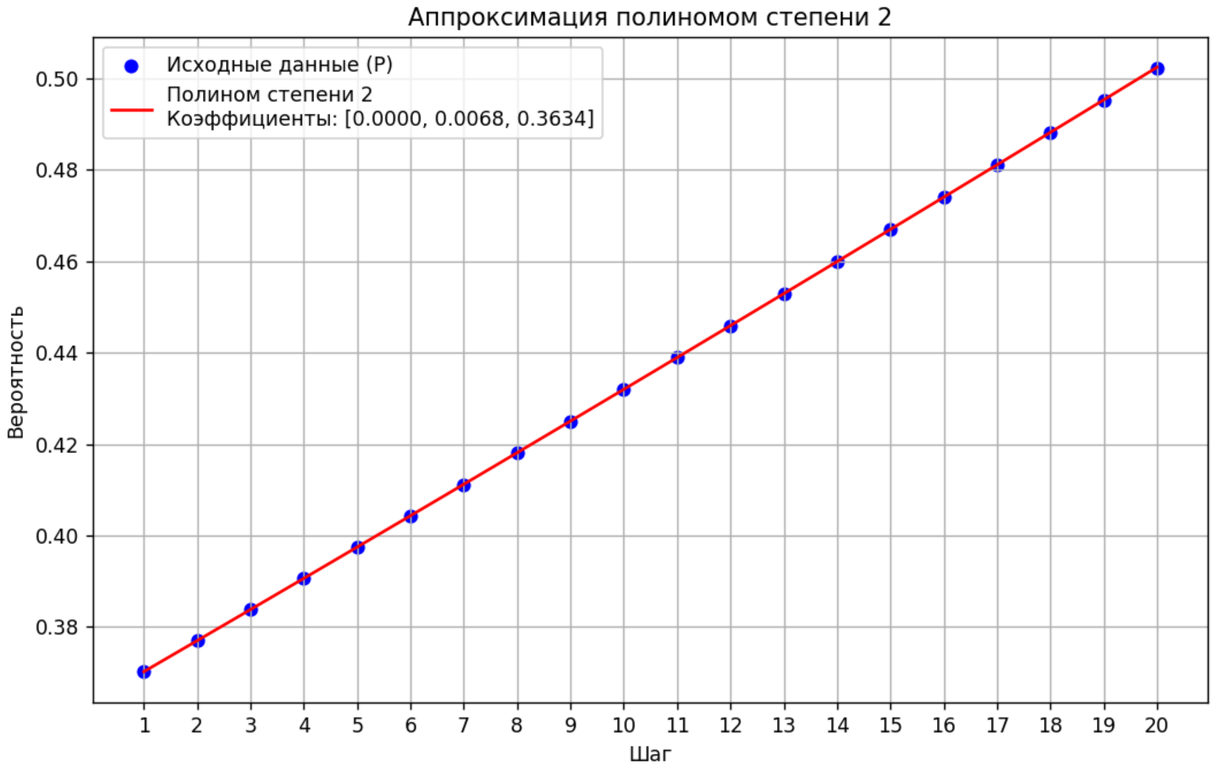

Рисунок 2 – Аппроксимация полиномом 2 степени

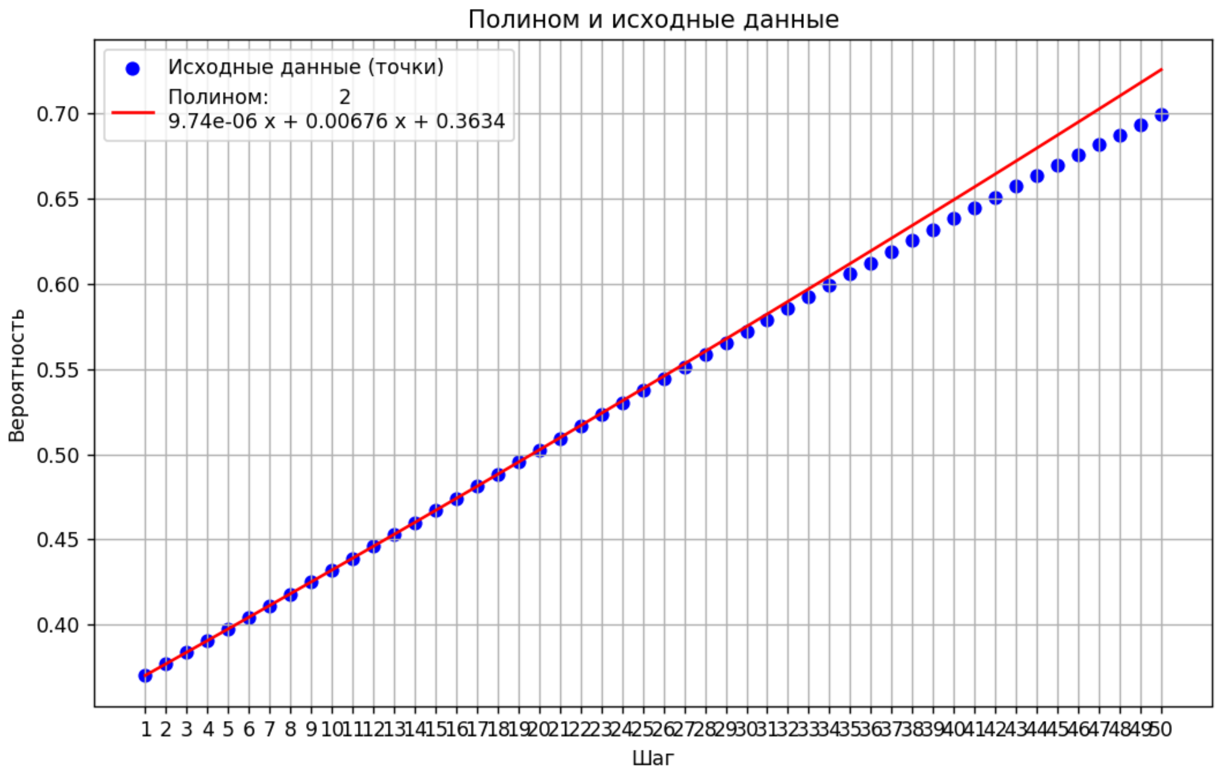

Рисунок 3 – Расхождение полинома 2 степени и результатов итераций

Рисунок 4 – Аппроксимация полиномом 3 степени

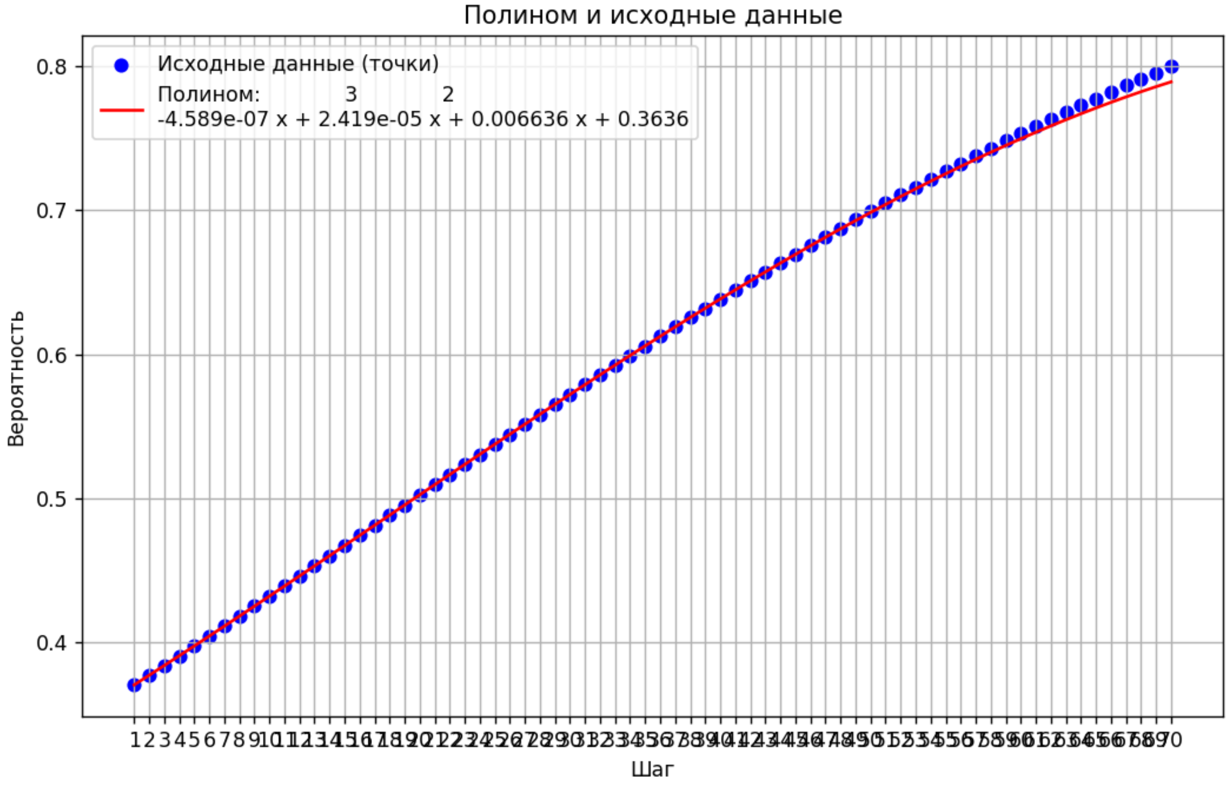

Рисунок 5 – Расхождение полинома 3 степени и результатов итераций

Полином 2-й степени отклоняется от исходных данных на 39 шаге, в то время как полином 3-й степени сохраняет точность до 69 шага.

Вывод.

В ходе работы была успешно смоделирована упреждающая связь с использованием теоремы Байеса для корректировки вероятностей гипотез. Первые три итерации были рассчитаны вручную, что позволило наглядно продемонстрировать процесс обновления вероятностей и убедиться в правильности расчетов. Для последующих итераций был разработан рекурсивный алгоритм на Python, который автоматизировал процесс и позволил провести 20 шагов обновления. С помощью полиномиальной регрессии была построена аппроксимационная кривая.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

```
import numpy as np
import matplotlib.pyplot as plt

def calculate_probabilities(P, Q, steps, current_step=1):
    P_B = sum(P[i] * Q[i] for i in range(len(P)))
    P_new = [P[i] * Q[i] / P_B for i in range(len(P))]
    for i, prob in enumerate(P_new):
        if i == 0:
            P_A1.append(prob)
        elif i == 1:
            P_A2.append(prob)
        elif i == 2:
            P_A3.append(prob)
    if current_step == steps:
        return P_new
    return calculate_probabilities(P_new, Q, steps, current_step + 1)

def plot_probabilities(x, P_A1, P_A2, P_A3):
    plt.figure(figsize=(10, 6))
    plt.plot(x, P_A1, marker='o', label='Завод 1', color='blue')
    plt.plot(x, P_A2, marker='o', label='Завод 2', color='red')
    plt.plot(x, P_A3, marker='o', label='Завод 3', color='green')
    plt.xticks(x)
    plt.xlabel('Шаг')
    plt.ylabel('Вероятность')
    plt.title('Вероятности для каждого завода на каждом шаге')
    plt.legend()
    plt.grid(True)
    plt.show()

def approximate_with_polynomial(x, y, degree):
    coefficients = np.polyfit(x, y, degree)
    polynomial = np.poly1d(coefficients)
    coeff_str = ", ".join([f"{coeff:.4f}" for coeff in coefficients])
    legend_label = f"Полином степени {degree}\nКоэффициенты: [{coeff_str}]"
    y_approx = polynomial(x)
    plt.figure(figsize=(10, 6))
    plt.scatter(x, y, color='blue', label='Исходные данные (P)')
    plt.plot(x, y_approx, color='red', label=legend_label)
    plt.xticks(x)
    plt.xlabel('Шаг')
    plt.ylabel('Вероятность')
    plt.title(f'Аппроксимация полиномом степени {degree}')
    plt.legend()
    plt.grid(True)
    plt.show()
    return coefficients

def plot_polynomial_with_points(x, y, coefficients):
    polynomial = np.poly1d(coefficients)
    y_approx = polynomial(x)
    for i in range(len(x)):
```

```
        if abs(y[i] - y_approx[i]) > 0.01:
            print(f"Отклонение > 0.01 на шаге {i + 1}")
            break
    plt.figure(figsize=(10, 6))
    plt.scatter(x, y, color='blue', label='Исходные данные (точки)')
    plt.plot(x, y_approx, color='red', label=f'Полином: {polynomial}')
    plt.xticks(x)
    plt.xlabel('Шаг')
    plt.ylabel('Вероятность')
    plt.title('Полином и исходные данные')
    plt.legend()
    plt.grid(True)
    plt.show()

P = [2 / 11, 4 / 11, 5 / 11]
Q = [0.94, 0.96, 0.93]
steps = 20
P_A1 = []
P_A2 = []
P_A3 = []

result = calculate_probabilities(P, Q, steps)

x = np.arange(1, steps + 1)
y = np.array(P_A2)
degree = 2

coefficients = approximate_with_polynomial(x, y, degree)
print(f"Коэффициенты полинома степени {degree}: {coefficients}")

P_A2 = []
steps = 70
result = calculate_probabilities(P, Q, steps)
x = np.arange(1, steps + 1)
y = np.array(P_A2)

plot_polynomial_with_points(x, y, coefficients)
```