НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Системи штучного інтелекту

Логістична регресія

Практична робота #3

Виконав:
Ілля Стародубець
Група:
ІМ-32мп
Курс:
1

3 листопада 2023 р.

1 Логістична регресія

1.1 Реалізація логістичної регресії

1.1.1 Ініціалізувати ваги та зсув

```python
def parameters_inititalization(m):
  # BEGIN_YOUR_CODE
  W = np.random.randn(1, m)
  b = 0.0

  return W, b
  # END_YOUR_CODE
```

1.1.2 Обчислити лінійну комбінацію вхідних ознак та ваг, включачи зсувю Застосувати нелінійну функцію активації (сигмоїду) до отриманого значення з крок 1

```python
# TODO
def forwardPropagate(X, W, b):
  # BEGIN_YOUR_CODE
  z = np.dot(W, X.T) + b
  y_hat = sigmoid(z)

  return z, y_hat
  # END_YOUR_CODE
```

1.1.3 Обчислити усереднену втрату на всьому навчальному наборі даних

```python
# TODO
def cost(n, y_hat, y_true):
  ep = 10E-10
  # BEGIN_YOUR_CODE
  J = -(1 / n) * np.sum(y_true * np.log(y_hat + ep) + (1 - y_true) * np.log(1 -   y_hat + ep
                                                                              ))

  return J
  # END_YOUR_CODE
```

1.1.4 Розрахувати градієнти цільвої функції відносно ваг та зсуву

```python
# TODO
def backwardPropagate(n, X, y_hat, y_true):
  # BEGIN_YOUR_CODE
  dW = (1 / n) * np.dot(y_hat - y_true, X)
  db = (1 / n) * np.sum(y_hat - y_true)

  return dW, db
  # END_YOUR_CODE
```

1.1.5 Оновити ваги та зсув

```python
# TODO
def update(lr, dW, db, W, b):
  # BEGIN_YOUR_CODE
  W -= lr * dW
  b -= lr * db

  return W, b
  # END_YOUR_CODE
```

1.2 Результати експериментів

При ручній маніпуляції швидкостями навчання та кількості ітерацій вийшли наступні результати

1.2.1 lr = 0.001

Спочатку протестуємо з данним з завданням lr.

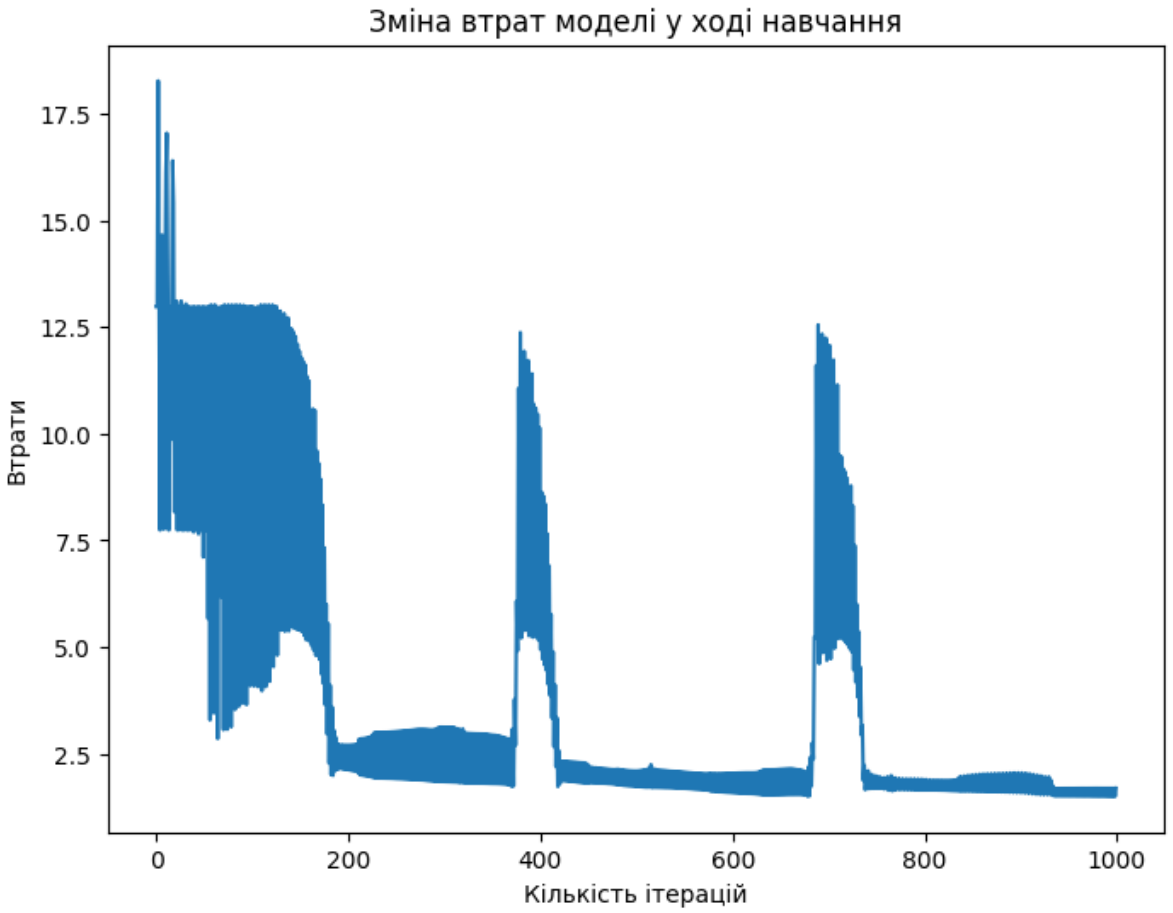

Рис. 1: Усереднена втрата моделі = 1.4424301845155454.

Оскільки рисунок 1 показує перший варінт графіку, то відповідно немає з чим порівнювати. Для початку збільшимо кількість ітерацій в 10 разів щоб побачити певну закономірність

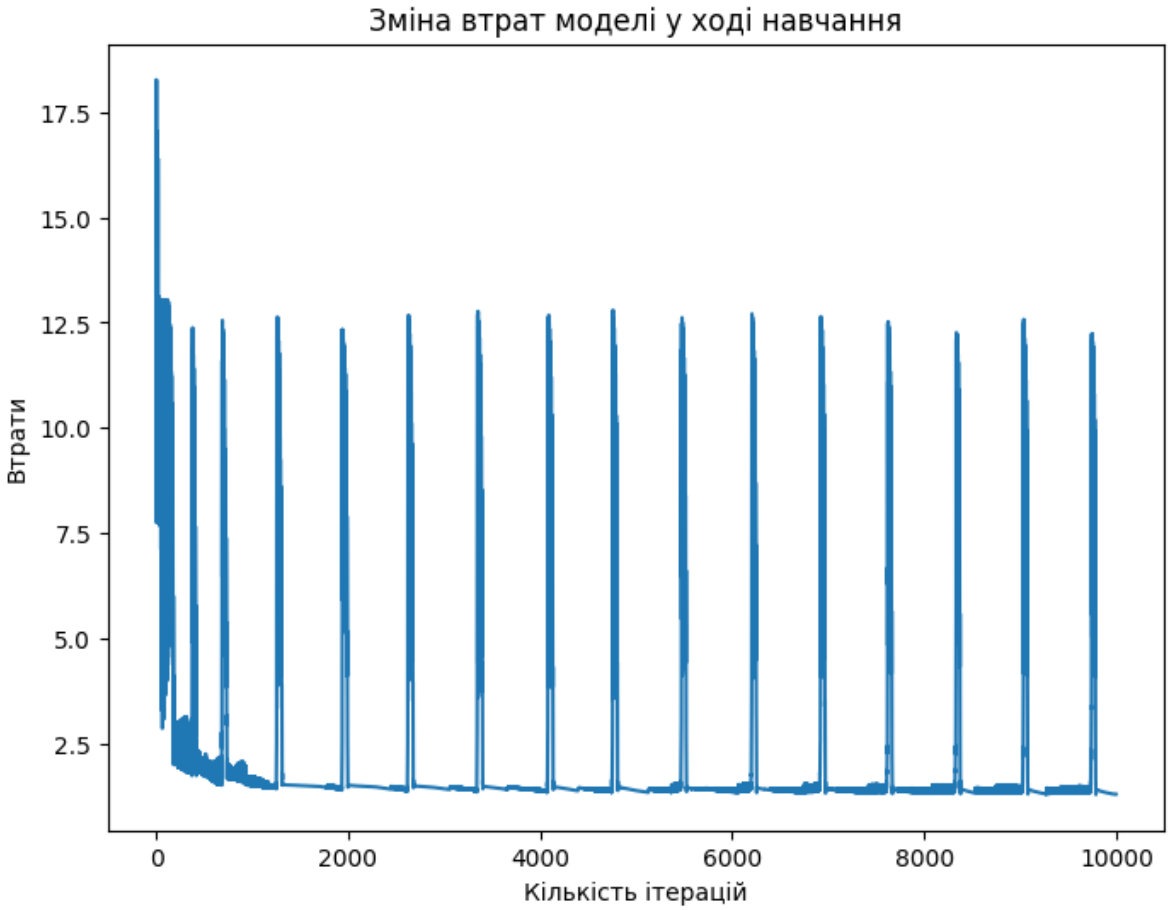

Рис. 2: Усереднена втрата моделі = 1.0749513733008609.

Як ми бачимо на рисунку 2 після збільшення ітерацій - збільшилось і кількість стрибків втрат. Спробуємо зменшити кількість ітерацій

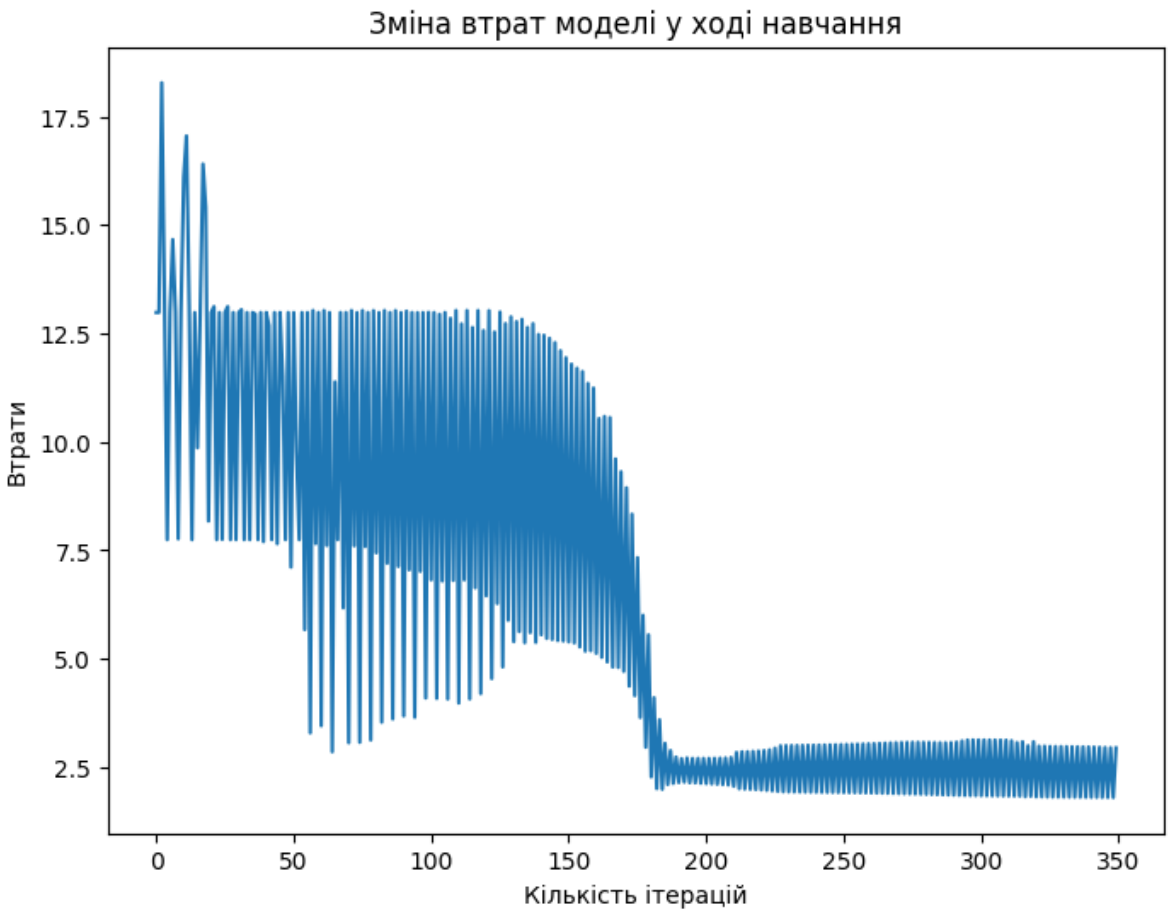

Рис. 3: Усереднена втрата моделі = 1.9528048260687907.

Як ми бачимо на рисунку 3 графік стрибає, проте поступово підходить до мінімального значення втрат. Але все ж таки присутня ось цей стрибок у втратах та здається маються ознаки перетренування

1.2.2 lr = 0.0001

Зараз спробуємо подивитися на поведінку графіка при зменшенні швидкості навчання

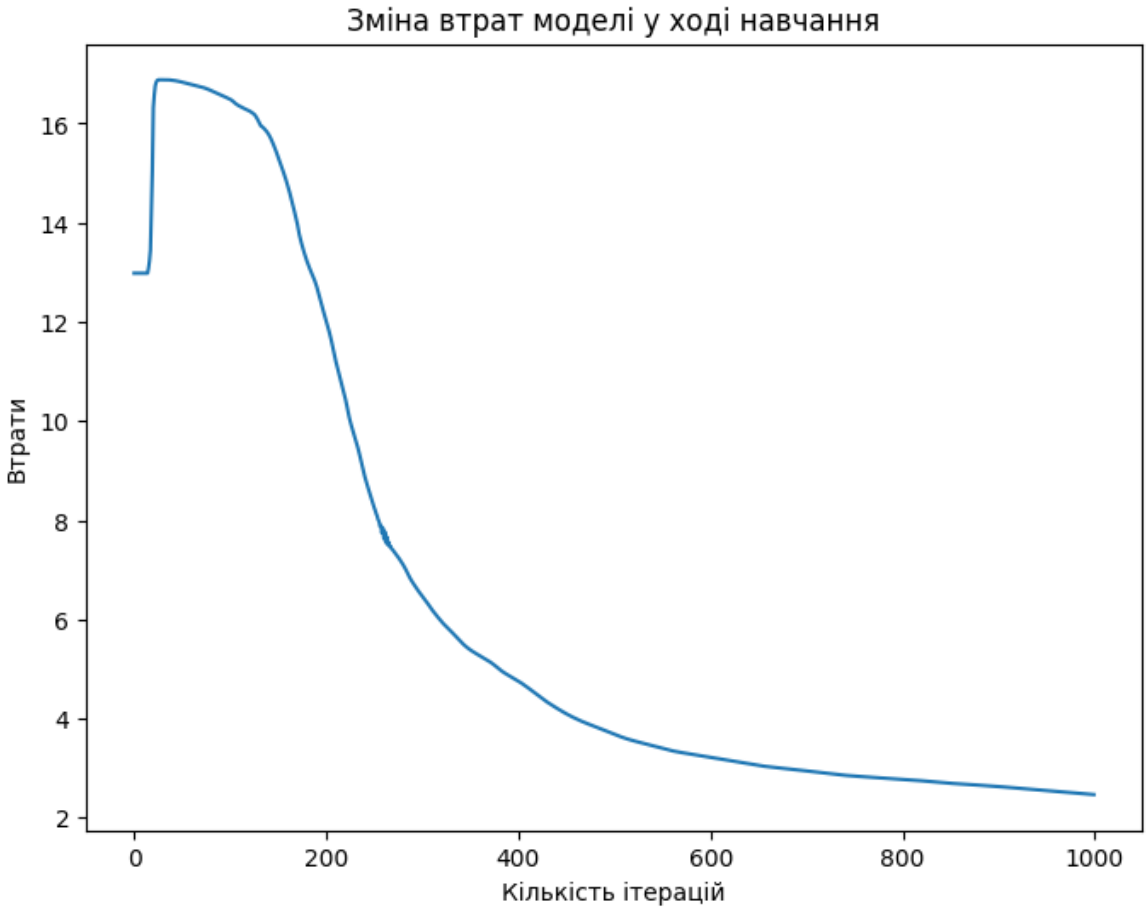

Рис. 4: Усереднена втрата моделі = 2.5209633875774364.

Судячи з рисунку 4 бачимо що втрата хоча і більша, проте графік втрат стрибає менше. Спробуємо збільшити кількість ітерацій.

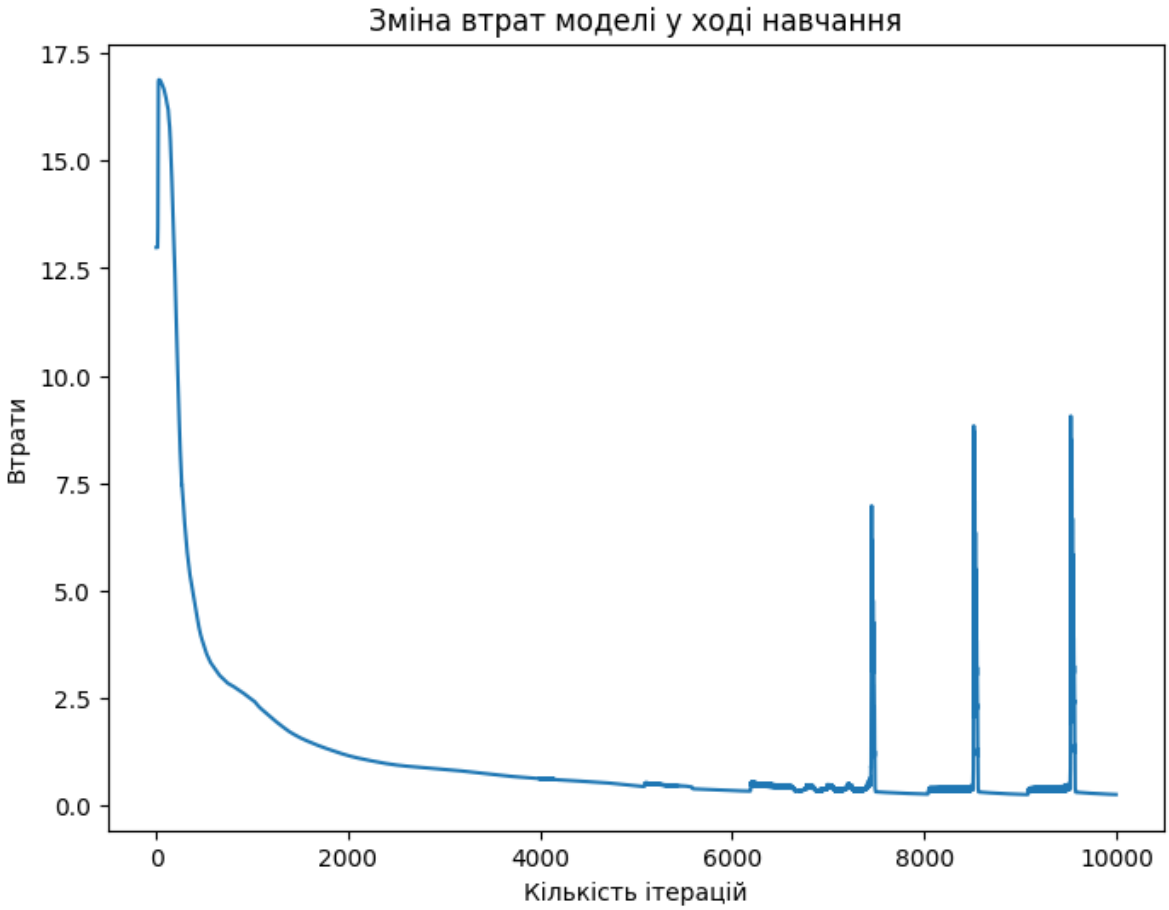

Рис. 5: Усереднена втрата моделі = 0.29179701311397743.

Як бачимо на рисунку 5 ознаки перетренування з'являються тільки після 6000 ітерацій. Та втрата моделі мінімальна за всі спроби тестування. Спробуємо зменшити кількість ітерацій.

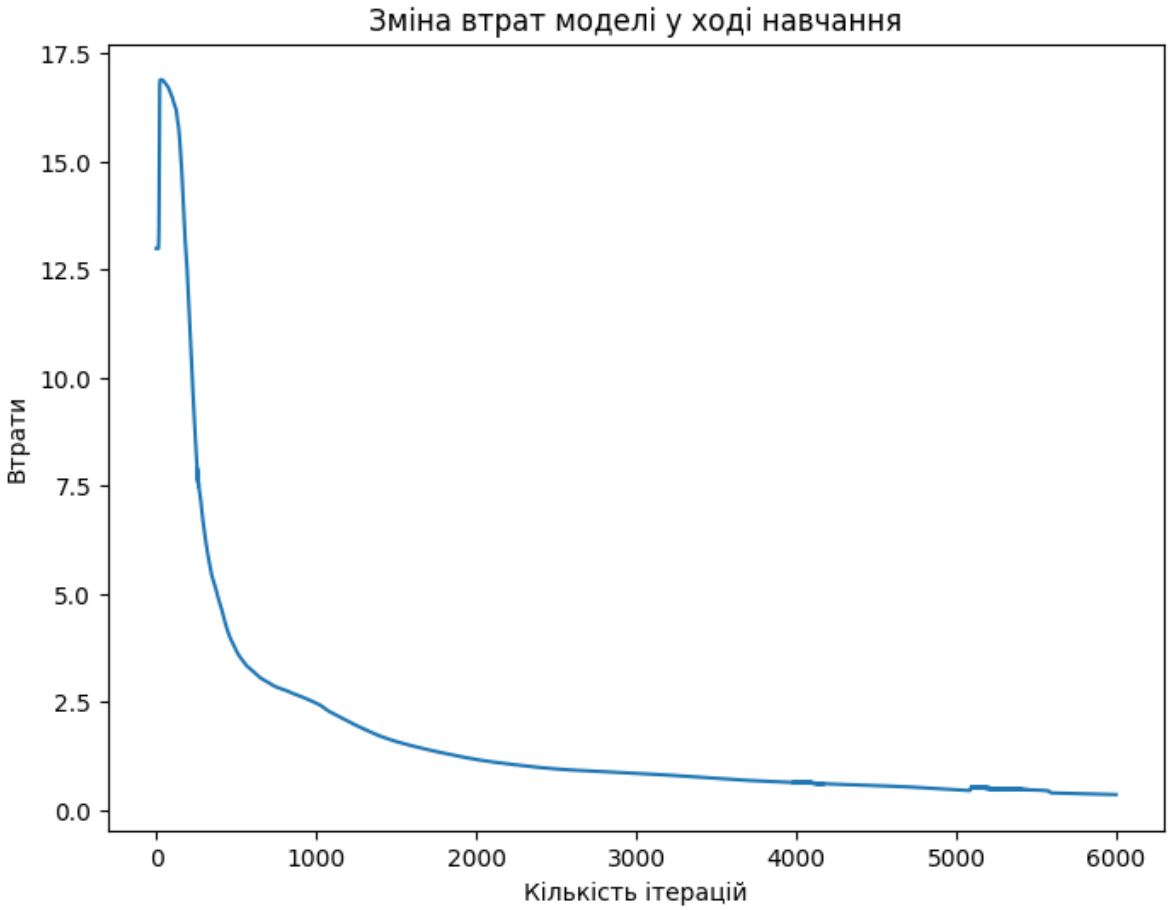

Рис. 6: Усереднена втрата моделі = 0.5231105268884676.

Як ми бачимо на рисунку 6 графік виглядає достатньо гладким та відсутні ознаки перетренування

1.2.3 lr = 0.01

Зараз спробуємо збільшити швідкість навчання порівнянно зі стартовим значенням та проведемо відповідні спостереження.

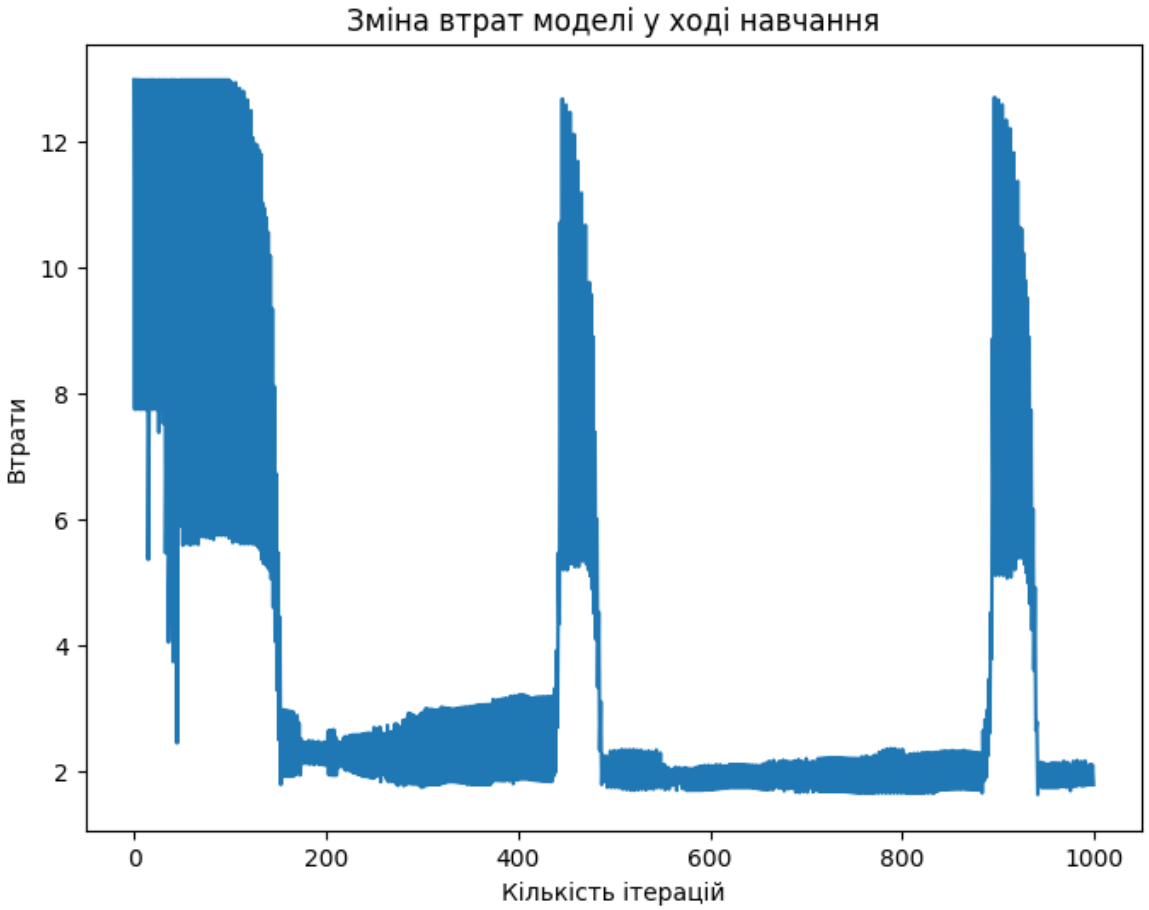

Рис. 7: Усереднена втрата моделі = 2.389146863129543.

Як ми бачимо на рисунку 7 через збільшену швидкість навчання графік втрат також сильно стрибає, про що і кажуть жирні лінії.

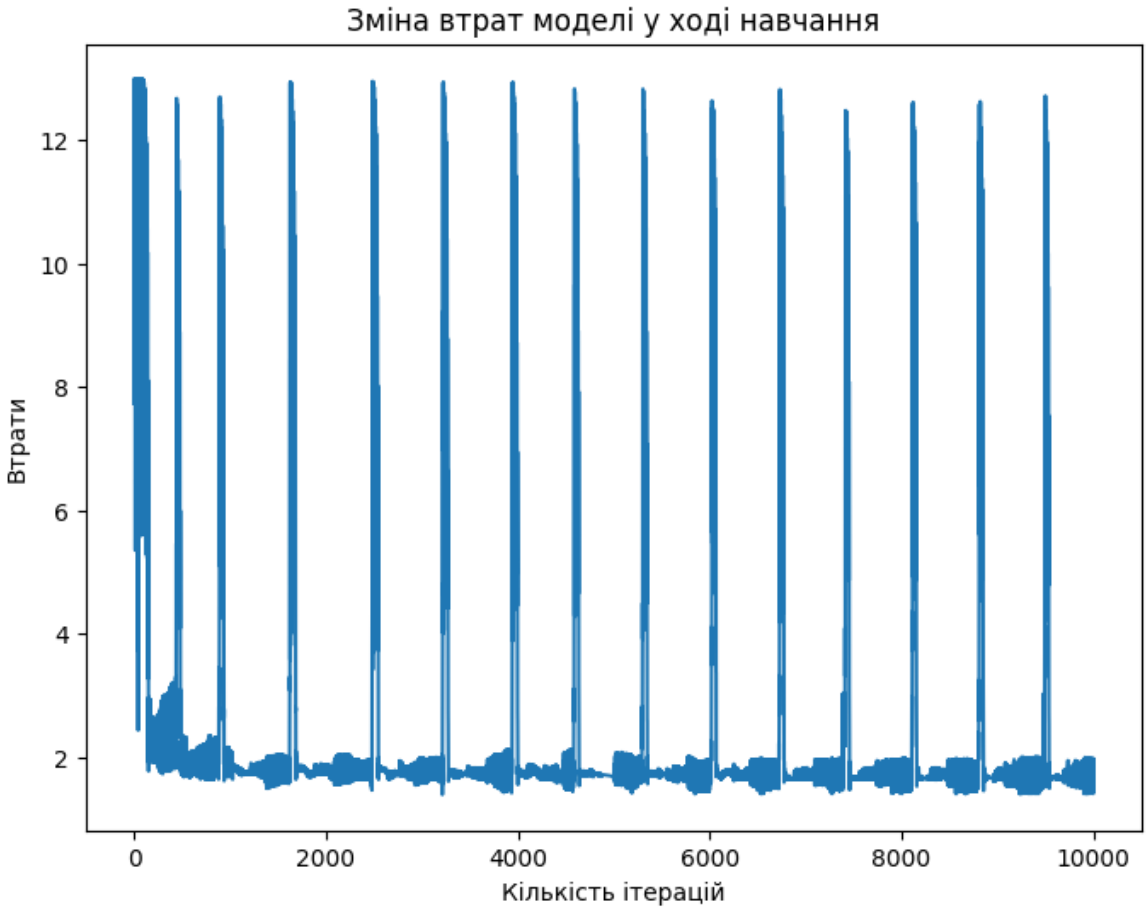

Рис. 8: Усереднена втрата моделі = 1.6360477068827715.

При збільшенні ітерацій на рисунку 8 підтвеждується перетренування - що і очікувано.

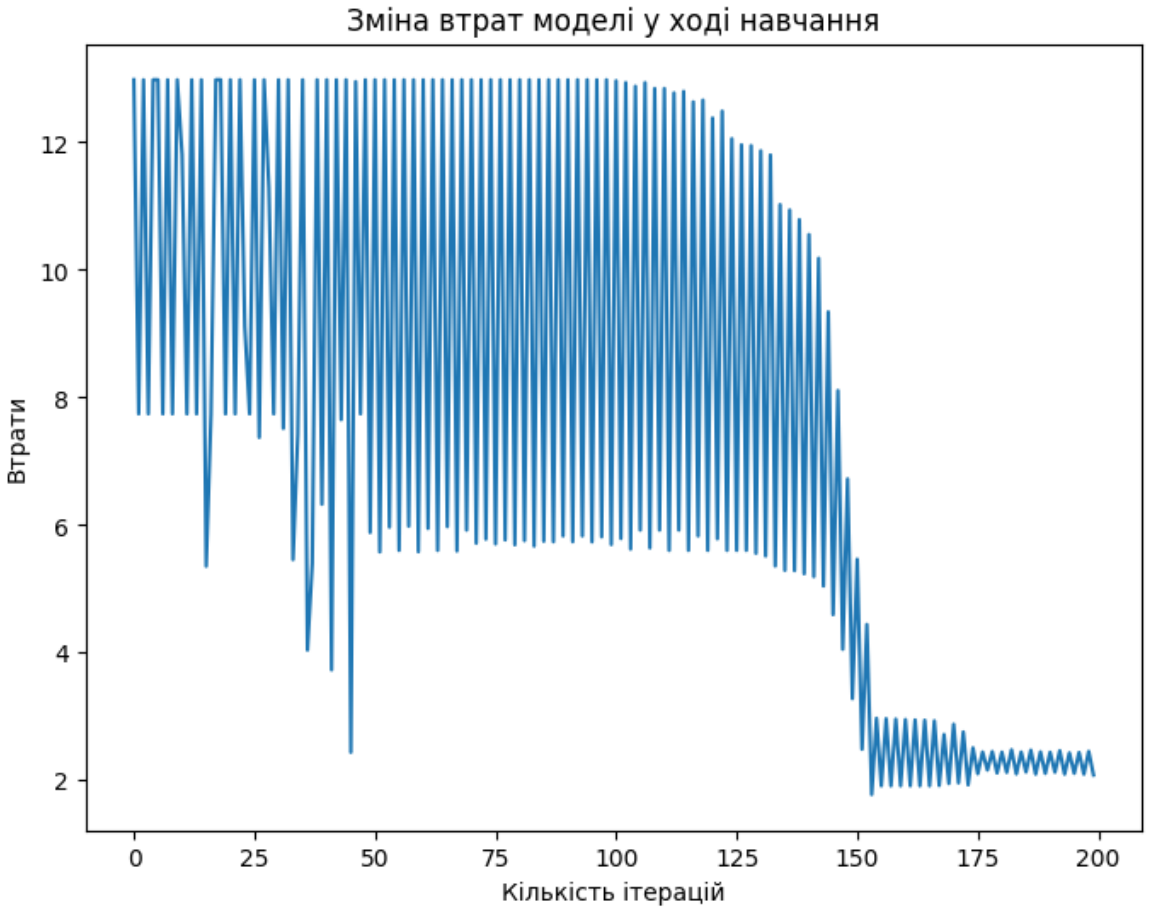

Рис. 9: Усереднена втрата моделі = 1.6360577568443848.

При зменшенні ітерацій на рисунку 9 бачимо наскільки сильно стрибають втрати від ітерації до ітерації. А також бачимо що дуже погано сходиться

1.3 Допомога

Ну, по більшій частині задавав питання ChatGPT (якщо потрібно по буде по особистому запису можу дати посилання на відповідну сессію з питаннями) та трохи подивився вже існуючий аналіз цього ж датасету від HamnaKhalid [1], де використовується не тільки логістична регресія. Не використовував чужі матеріали (код / графіки / текст).

1.4 Висновки

Судячи с проведених тестів, можна сказати що най оптимальнішим буде вибір певного компромісу з повільності швидкості навчання та відповідній до неї кількості ітерацій, щоб модель не перенавчалась та ці значення - швидкість навчання 0.0001 та кількість ітерацій 6000. Проте варто зауважити що це саме такі результати для саме цієї моделі, данних та задачі.

Література

[1] HamnaKhalid. Breast_cancer. Kaggle. [Online]. Available: https://www.kaggle.com/code/hkhamnakhalid/breast-cancer