기계학습의기초및응용 HW1 보고서

과목: 기계학습의기초및응용
교수명: 이규행 교수님
학과: 모바일시스템공학과
학번: 32224020
이름: 전은지

목 차

I. 서론

이 보고서에서는 두 가지 다른 모델을 사용하여 주어진 데이터셋에 선형 회귀를 수행하는 두 가지 접근 방법을 탐구합니다. 'data_hw1.csv'을 통해 데이터가 주어지며 데이터는 두 열, 'x' 및 'y'를 포함하고 있습니다. 이 분석의 주요 목표는 선형 모델에 대한 경사 하강법 및 이차 모델에 대한 정규 방정식을 구현하고 결과를 비교하는 것입니다.

II. 본론

1. Gradient Descent

사용 언어는 Python이며 데이터 분석을 위해 numpy, pandas, matplotlib 라이브러리를 불러옵니다.

```
# Gradient Descent
def gradientDescent(x, y, learning_rate, num_iterations):
    m = len(x)
    a, b = 0, 0

    for i in range(num_iterations):
        y_pred = a * x + b
        gradient_a = (1/m) * np.sum(x * (y_pred - y))
        gradient_b = (1/m) * np.sum(y_pred - y)

        a -= learning_rate * gradient_a
        b -= learning_rate * gradient_b

        if (i + 1) % 100 == 0:
            cost = (1/(2*m)) * np.sum((a * x + b - y)**2)
            print(f'Iteration {i+1}, Cost: {cost:.4f}, a {a}, b {b}')

    return a, b
```

주어진 데이터셋에 대하여 경사 하강법을 이용하여 모델을 구현하기 위한 함수를 작성합니다. 평균 제곱 오차를 각각의 파라미터에 대하여 미분하고, 학습률을 곱해 업데이트합니다. 반복이 진행되는 동안 학습률에 따른 비용의 변화를 보기 위하여 if문을 작성하였습니다. 반복이 100번 진행될 때마다 비용과 파라미터가 출력됩니다.

```
# Task 1: Linear Regression with Gradient Descent
learning_rate = 0.01
num_iterations = 1000
a, b = gradientDescent(x, y, learning_rate, num_iterations)
```

학습률을 0.01로 설정하고 반복을 1000번 진행합니다.

```
Iteration 100, Cost: 42.4741, a -7.949068949504677, b -1.4831322163006524
Iteration 200, Cost: 32.0738, a -9.821012182367955, b -4.059933737945941
Iteration 300, Cost: 28.2171, a -10.781049102341528, b -5.754103583668346
Iteration 400, Cost: 26.7531, a -11.360267566722454, b -6.804870686363662
Iteration 500, Cost: 26.1971, a -11.716399109594803, b -7.452802242242487
Iteration 600, Cost: 25.9860, a -11.935801677011758, b -7.85209430008812
Iteration 700, Cost: 25.9059, a -12.070997305266085, b -8.098145298718224
Iteration 800, Cost: 25.8754, a -12.154306492328251, b -8.249765400109018
Iteration 900, Cost: 25.8639, a -12.20564273867739, b -8.343195785863658
Iteration 1000, Cost: 25.8595, a -12.237276834387151, b -8.400768866836302
```

이때 위 사진과 같은 결과를 확인할 수 있습니다. 반복이 진행될수록 비용이 약 25로 최소화되는 것을 확인하였고 0.01을 적절한 학습률이라고 판단하였습니다.

2. Normal Equation

```python
# Normal Equation
def normalEquation(x, y):
    X = np.column_stack((np.ones_like(x), x, x**2))
    params = np.linalg.inv(X.T @ X) @ X.T @ y
    return params[2], params[1], params[0]
```

정규방정식을 이용하여 이차 회귀를 수행하기 위한 함수 또한 작성합니다. 입력값 x에 대하여 파라미터 벡터를 만들고 정규 방정식을 계산하여 최적의 파라미터를 도출합니다.

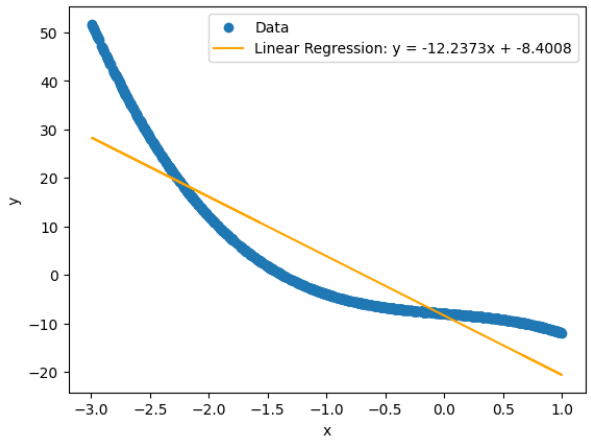
III. 결론

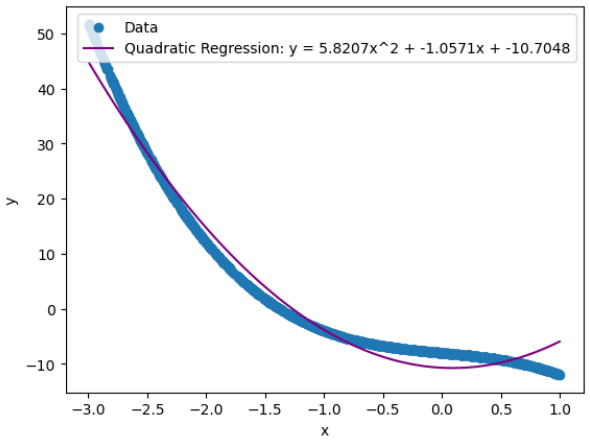
각각의 방식에 대한 결과는 다음과 같습니다.

A. Table of parameters

	a	b	c
Task1	-12.2373	-8.4008	
Task2	5.8207	-1.0571	-10.7048

B. Plots of the models





종합적으로 경사 하강법을 이용한 선형 모델과 정규 방정식을 이용한 이차 모델은 효과적으로 구현되었습니다. 수행된 실험 및 분석 결과, 이차 모델이 선형 모델보다 주어진 데이터셋과 더 일치하는 형태를 보여주었습니다. 이로써 주어진 데이터셋에 대한 분석에서는 정규 방정식을 사용한 이차 회귀 모델이 더 적합하다는 결론을 도출할 수 있습니다.