3-6. 그레디언트 벡터와 기울기: Gradient Descent 최적화와 관련 지어 학습

1. 미분과 기울기

   미분은 함수의 기울기를 구하는 과정입니다. 함수의 변화율을 계산하는데, 이는 입력 값의 변화에따라 출력값이 얼마나 변하는지를 알려줍니다. 기울기는 특정 지점에서 함수의 변화 방향과 크기를 나타냅니다.

   예를 들어, 함수 $f(x) = x^2$의 기울기는 $2x$입니다. 이는 입력 값이 증가할 때 함수 값이 어떻게 변하는지 나타냅니다.

2. 그레디언트 벡터 (Gradient Vector)

   다변수 함수 $f(x, y)$의 경우, 미분을 각각의 변수에 대해 해야 합니다. 각 변수에 대한 편미분을 모아 놓은 것이 그레디언트 벡터입니다.

   - 함수가 $f(x, y) = x^2 + y^2$일 때,
   - $\frac{\partial f}{\partial x} = 2x$
   - $\frac{\partial f}{\partial y} = 2y$

   따라서 그레디언트 벡터는 $\nabla f(x, y) = (2x, 2y)$입니다. 이는 함수가 가장 가파르게 증가하는 방향을 나타냅니다.

3. Gradient Descent (경사 하강법)

   딥러닝에서 Gradient Descent는 모델의 파라미터를 업데이트하여 손실 함수를 최소화하는데 사용됩니다. 이는 손실함수의 그레디언트를 계산하고, 그레디언트의 반대 방향으로 파라미터를 이동시키는 과정입니다.

   파라미터 업데이트 공식은 다음과 같습니다.

$$\theta = \theta - \alpha \cdot \nabla \theta J(\theta)$$

- θ는 모델의 파라미터
- α는 학습률 (learning rate)
- $\nabla\theta J(\theta)$는 파라미터 θ에 대한 손실함수 J의 그레디언트입니다.

이 과정을 반복하면서 손실 함수의 값을 줄이는 것이 Gradient Descent의 목표입니다.

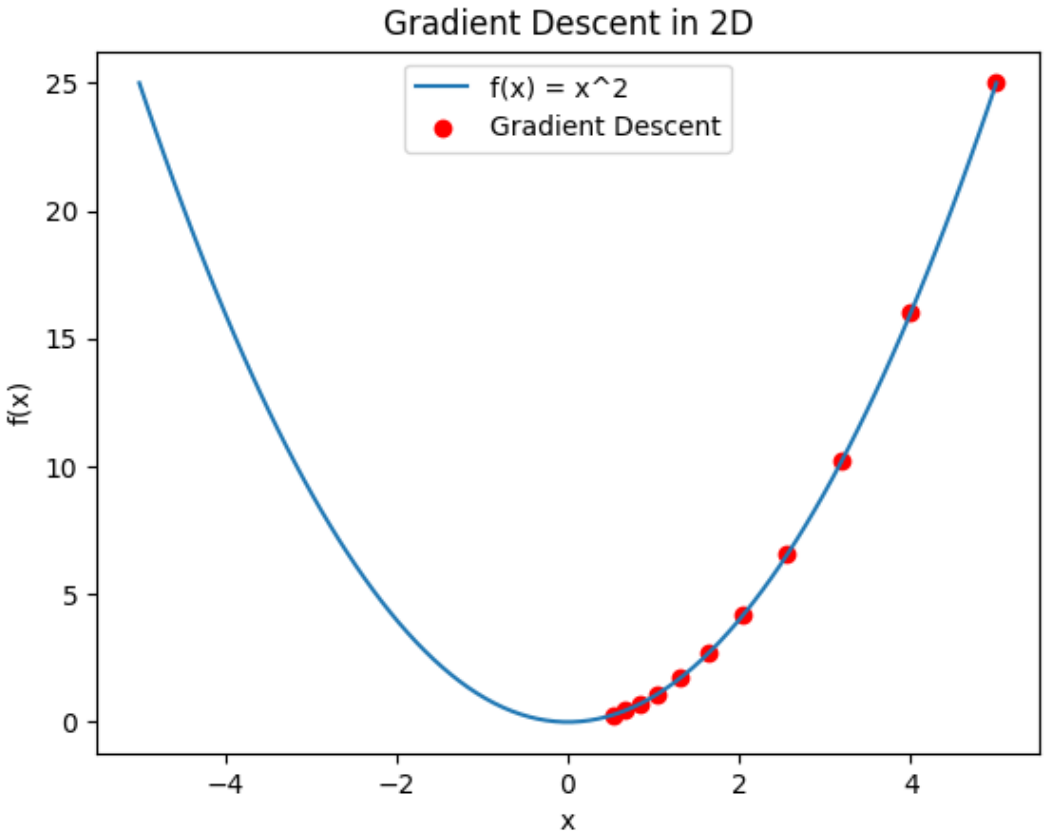
4. 2D 및 3D 그래프 시각화

   Gradient Descent가 어떻게 동작하는지 시각적으로 보기 위해 2D 및 3D 그래프를 통해 그레디언트의 이동 경로를 보여드리겠습니다.

   - 2D에서의 경사하강법

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

# 함수 정의
def f(x):
    return x**2

# 함수의 미분 (기울기)
def grad_f(x):
    return 2 * x

# 초기값 설정
x = 5
learning_rate = 0.1
iterations = 10
x_values = [x]

# 경사 하강법 반복
for _ in range(iterations):
    x = x - learning_rate * grad_f(x)
    x_values.append(x)

# 그래프 그리기
x_range = np.linspace(-5, 5, 100)
y_range = f(x_range)
```

```
plt.plot(x_range, y_range, label="f(x) = x^2")
plt.scatter(x_values, [f(x) for x in x_values], color='
plt.title("Gradient Descent in 2D")
plt.xlabel("x")
plt.ylabel("f(x)")
plt.legend()
plt.show()
```

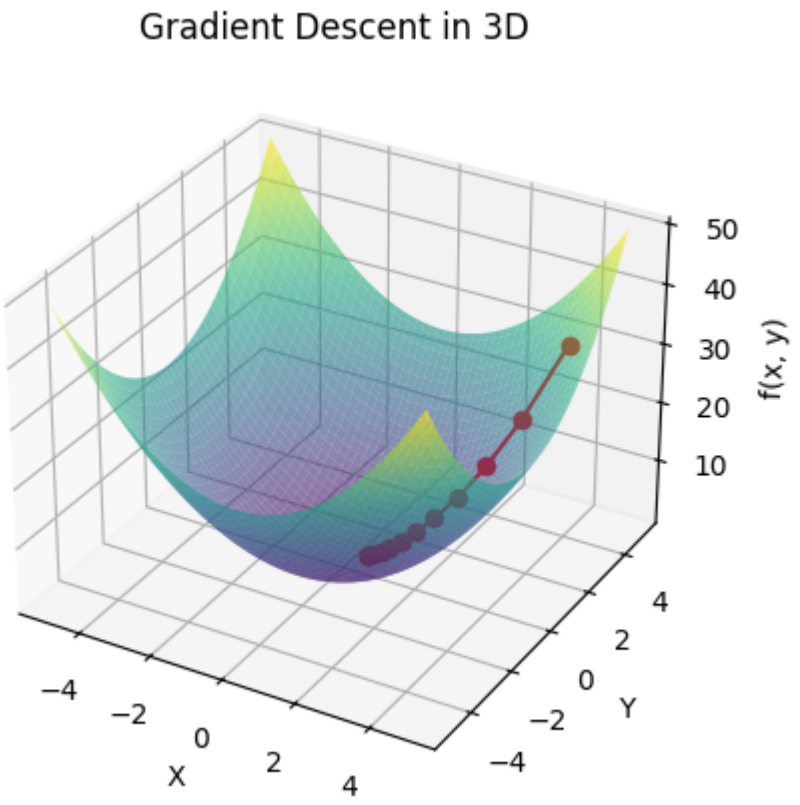


이 코드는 1차원 함수 $f(x) = x^2$에서 경사 하강법을 수행하는 과정입니다. 초기값을 설정하고, 각 반복마다 학습률만큼 파라미터를 업데이터하여 최소값에 수렴하는 것을 보여줍니다.

- 3D에서의 경사 하강법

```
from mpl_toolkits.mplot3d import Axes3D

# 2변수 함수 정의
def f(x, y):
    return x**2 + y**2
```

```
# 그레디언트 정의
def grad_f(x, y):
    return np.array([2*x, 2*y])

# 초기값 설정
xy = np.array([4, 4])
learning_rate = 0.1
iterations = 10
xy_values = [xy]

# 경사 하강법 반복
for _ in range(iterations):
    xy = xy - learning_rate * grad_f(xy[0], xy[1])
    xy_values.append(xy)

xy_values = np.array(xy_values)

# 3D 그래프 그리기
x = np.linspace(-5, 5, 100)
y = np.linspace(-5, 5, 100)
X, Y = np.meshgrid(x, y)
Z = f(X, Y)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X, Y, Z, cmap='viridis', alpha=0.6)
ax.plot(xy_values[:, 0], xy_values[:, 1], f(xy_values[:
ax.set_title('Gradient Descent in 3D')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('f(x, y)')
plt.show()
```



이 코드는 2변수 함수 $f(x, y) = x^2 + y^2$에서 경사 하강법을 통해 최소값으로 수렴하는 경로를 3D로 보여줍니다.

핵심요약

- 미분은 함수의 기울기를 구하는 방법이며, 딥러닝에서는 모델 파라미터의 업데이트에 사용됩니다.
- 그레디언트 벡터는 함수의 변화 방향을 나타내며, Gradient Descent는 이 벡터의 반대 방향으로 파라미터를 업데이트해 최적의 모델을 학습합니다.
- 학습률(learning rate)에 따라 경로가 달라질 수 있으며, 이를 시각화하는 방법으로 2D/3D 그래프를 활용할 수 있습니다.