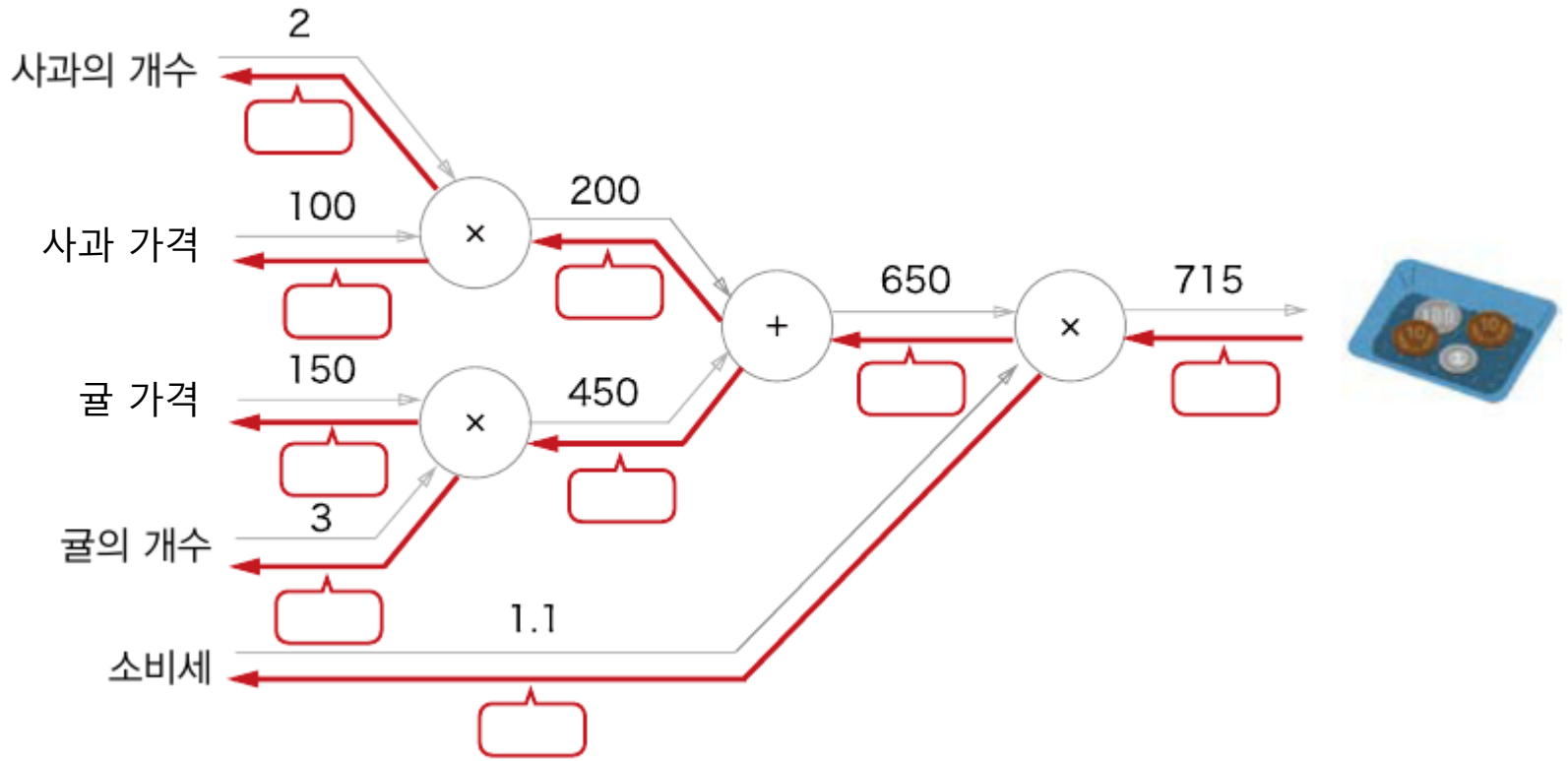
5.4 단순한 계층 구현하기

'사과 쇼핑 ' 의 예를 파이썬으로 구현하자.

곱셉 노드를 MulLayer, 덧셈 노드를 AddLayer라고 이름한다.

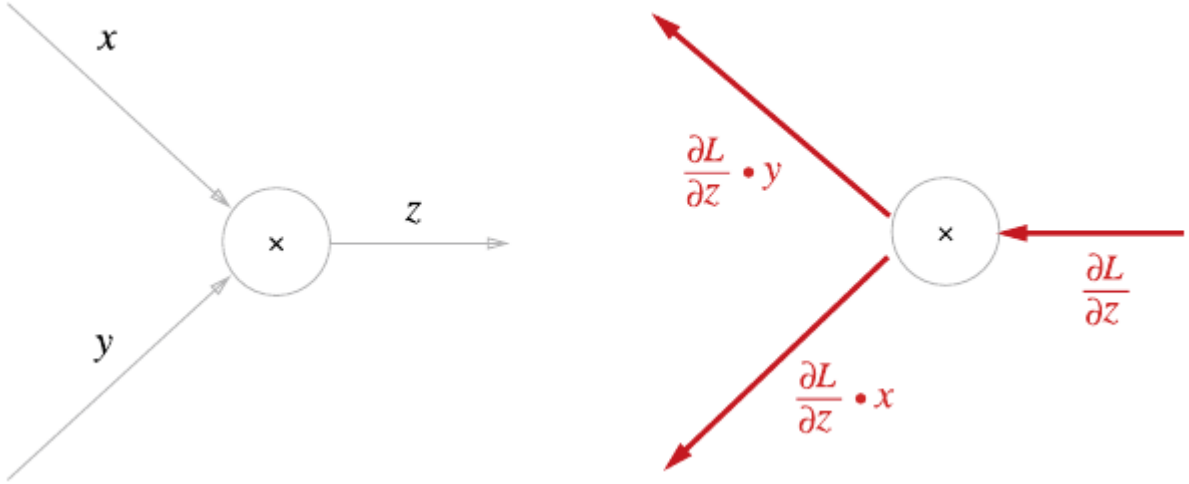
그림 5-15 사과와 귤 쇼핑의 역전파 예 : 빈 상자 안에 적절한 숫자를 넣어 역전파를 완성하시오.



5.4.1 곱셈 계층

```
class MulLayer:
    def __init__(self):
        self.x = None
        self.y = None

    def forward(self, x, y):
        self.x = x
        self.y = y
        out = x * y

        return out

    def backward(self, dout):
        dx = dout * self.y # x와 y를 바꾼다.
        dy = dout * self.x

        return dx, dy
```

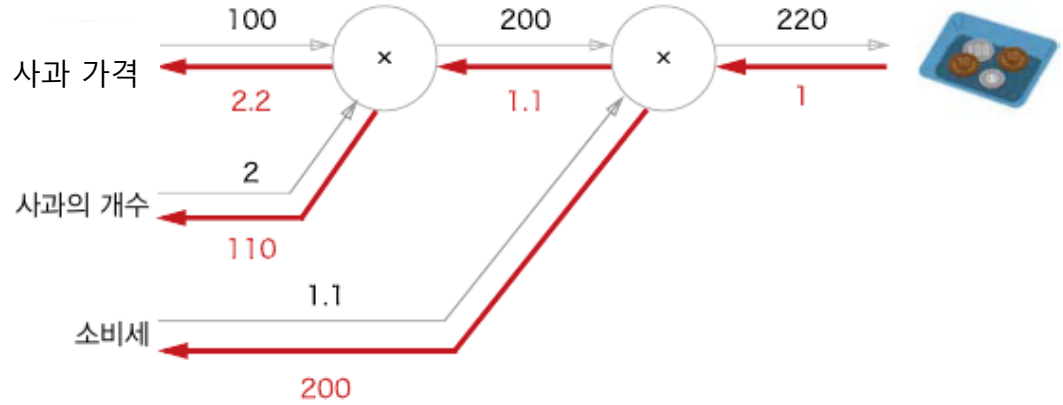
그림 5-12 곱셈 노드의 역전파 : 왼쪽이 순전파, 오른쪽이 역전파다.



backward 계산을 위해 forward시 멤버변수로 저장 필요

5.4.1 곱셈 계층

그림 5-14 사과 쇼핑의 역전파 예



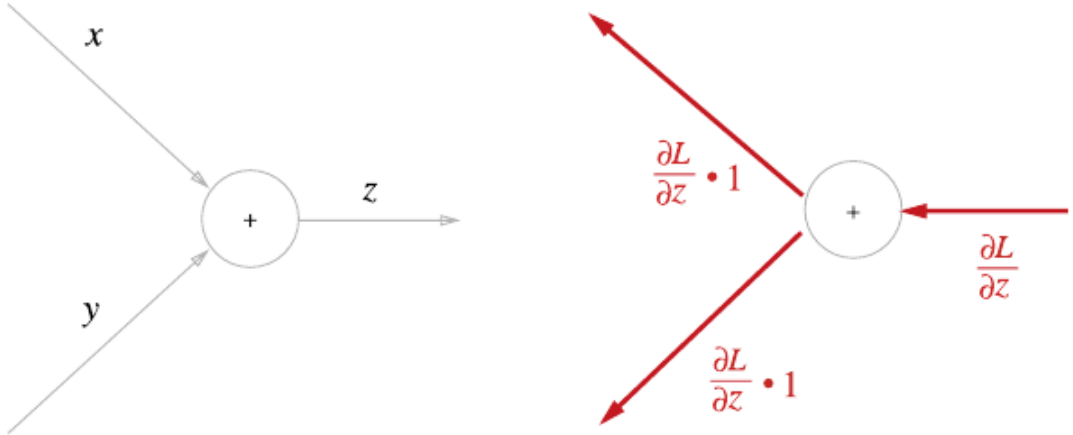

```
apple = 100
apple_num = 2
tax = 1.1

# 계층들
mul_apple_layer = MulLayer()
mul_tax_layer = MulLayer()

# 순전파
apple_price = mul_apple_layer.forward(apple, apple_num)
price = mul_tax_layer.forward(apple_price, tax)

print(price) # 220
```

```
# 역전파
dprice=1
dapple_price, dtax = mul_tax_layer.backward(dprice)
dapple, dapple_num = mul_apple_layer.backward(dapple_price)

print(dapple, dapple_num, dtax) # 2.2 110 200
```

5.4.2 덧셈 계층

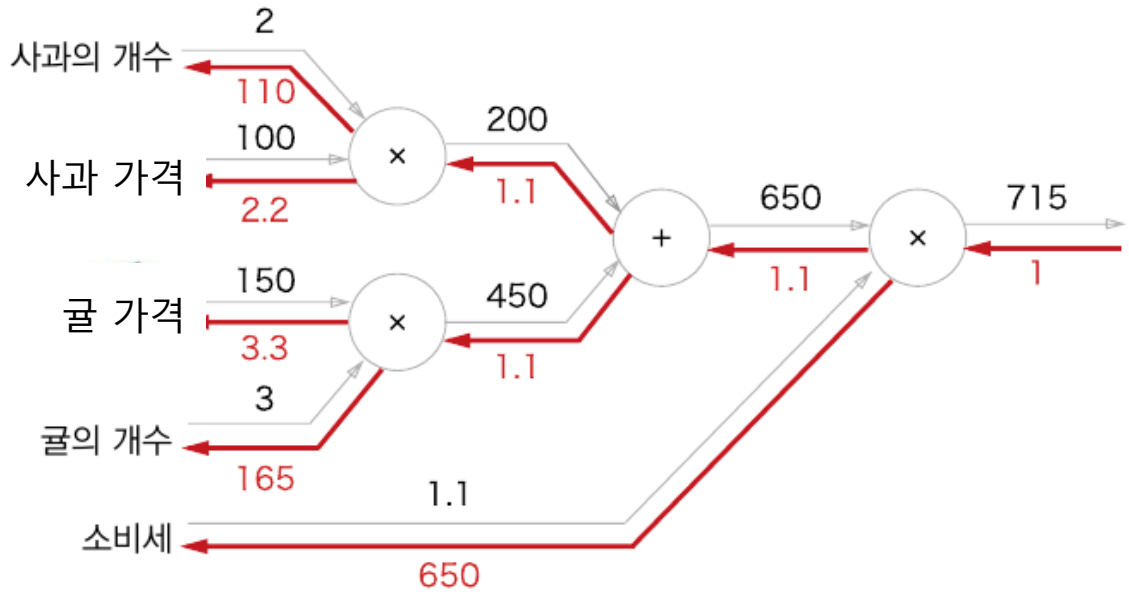
```
class AddLayer:
    def __init__(self):
        pass

    def forward(self, x, y):
        out = x + y
        return out

    def backward(self, dout):
        dx = dout * 1
        dy = dout * 1
        return dx, dy
```

그림 5-9 덧셈 노드의 역전파 : 왼쪽이 순전파, 오른쪽이 역전파다. 덧셈 노드의 역전파는 입력 값을 그대로 흘려보낸다.



5.4.2 덧셈 계층

그림 5-17 사과 2개와 귤 3개 구입




```
apple = 100
apple_num = 2
orange = 150
orange_num = 3
tax = 1.1

# 계층들
mul_apple_layer = MulLayer()
mul_orange_layer = MulLayer()
add_apple_orange_layer = AddLayer()
mul_tax_layer = MulLayer()

# 순전파
apple_price = mul_apple_layer.forward(apple, apple_num) #(1)
orange_price = mul_orange_layer.forward(orange, orange_num) #(2)
all_price = add_apple_orange_layer.forward(apple_price, orange_price) #(3)
price = mul_tax_layer.forward(all_price, tax) #(4)

# 역전파
dprice = 1
dall_price, dtax = mul_tax_layer.backward(dprice) #(4)
dapple_price, dorange_price = add_apple_orange_layer.backward(dall_price) #(3)
dorange, dorange_num = mul_orange_layer.backward(dorange_price) #(2)
dapple, dapple_num = mul_apple_layer.backward(dapple_price) #(1)

print(price) # 715
print(dapple_num, dapple, dorange, dorange_num, dtax) # 110 2.2 3.3 165 650
```

다음 시간에~

5.5 활성화 함수 계층 구현하기

계산 그래프를 신경망에 적용하고자 한다.

먼저 활성화 함수 계층에 대한 계산 그래프를 만들어 보자.

먼저 Relu와 Sigmoid 활성화 함수를 계산 그래프로 구현한다. 그 다음, Affine 계층과 Softmax 계층을 구현한다.