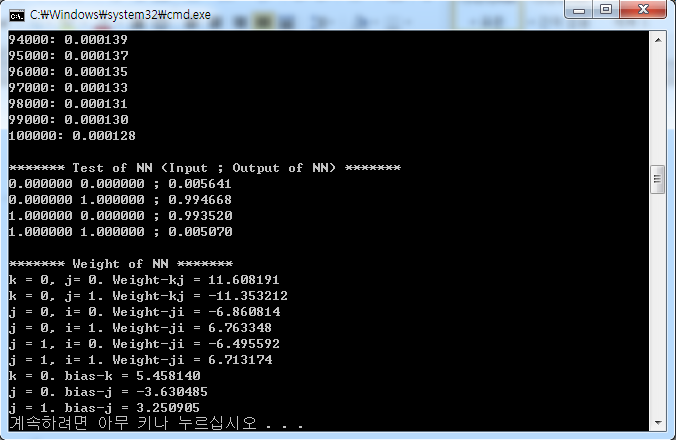
**인공지능 Programming Homework #1**

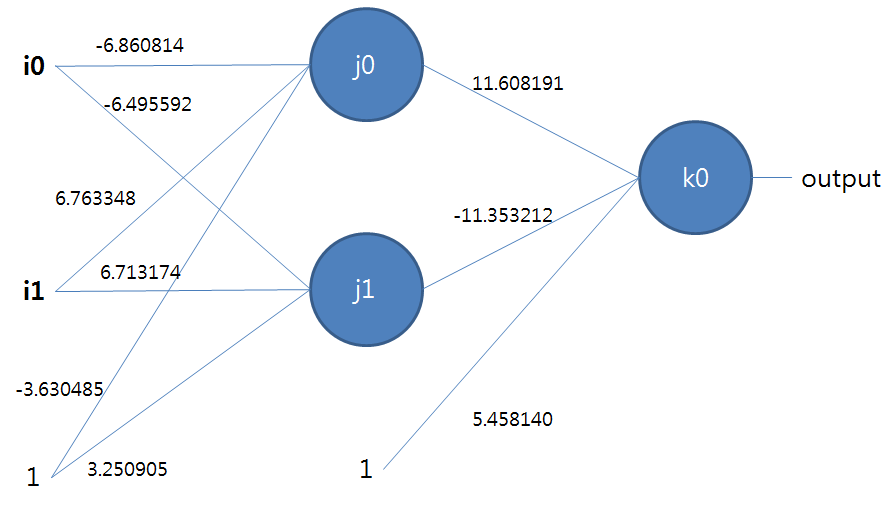
2014313366 홍기원

**#1**

Example 1의 코드를 돌려본 결과는 다음과 같다.



이 결과(각 connection weight)를 가지고 neural network를 그려보면 아래와 같다.

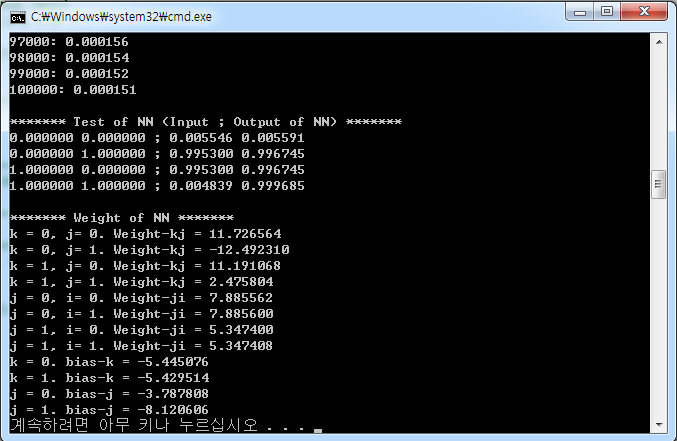


학습한 neural network에서 주어진 input에 대한 output값은 아래 표와 같다. (위의 스크린샷의 내용과 동일)

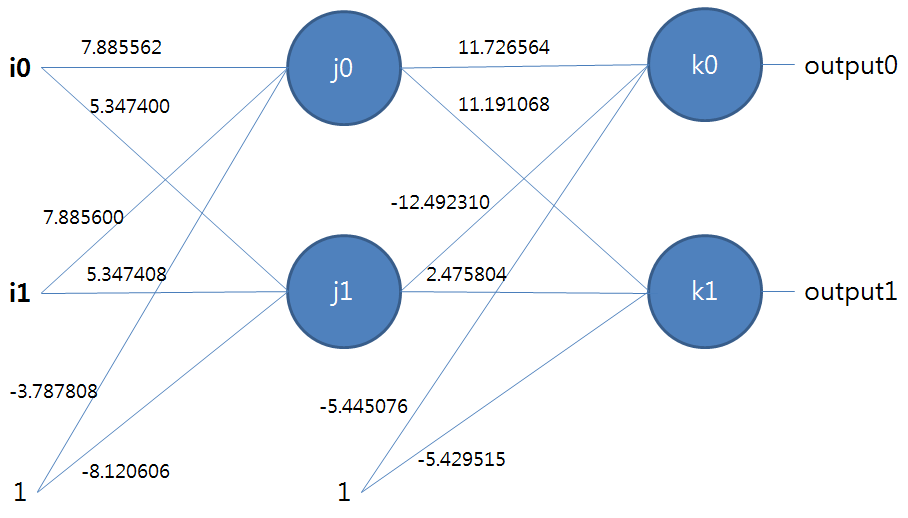
|  |  |  |  |
| --- | --- | --- | --- |
| input 1 | input 2 | target | output |
| 0 | 0 | 0 | 0.005641 |
| 0 | 1 | 1 | 0.994668 |
| 1 | 0 | 1 | 0.993520 |
| 1 | 1 | 0 | 0.005070 |

**#2**

Example 2의 코드를 돌려본 결과는 다음과 같다.



이 결과(각 connection weight)를 가지고 neural network를 그려보면 아래와 같다.

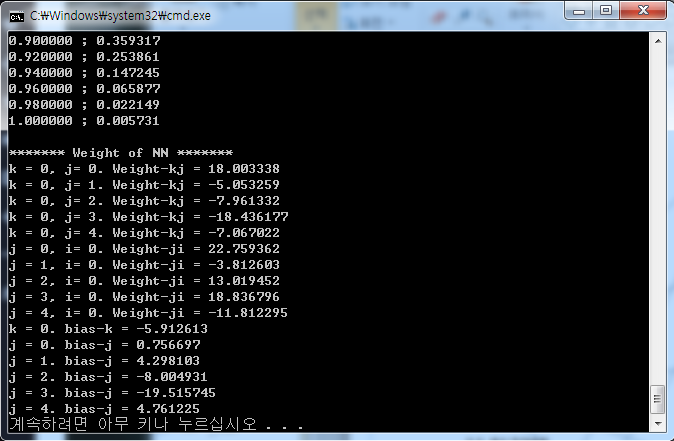


학습한 neural network에서 주어진 input에 대한 output값은 아래 표와 같다. (위의 스크린샷의 내용과 동일)

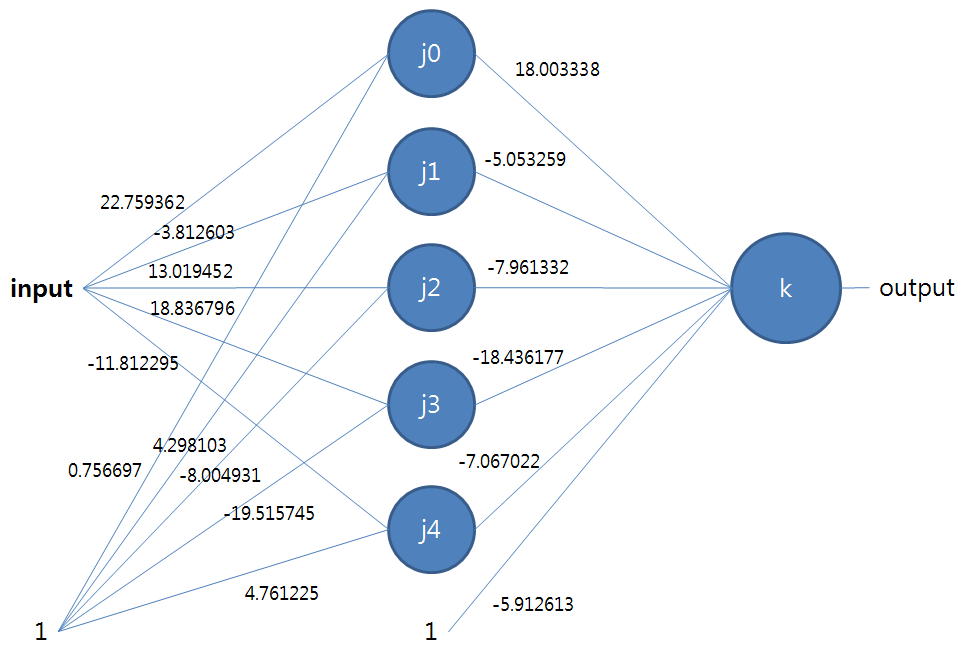
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| input 1 | input 2 | target1 | target2 | output1 | output2 |
| 0 | 0 | 0 | 0 | 0.005546 | 0.005591 |
| 0 | 1 | 1 | 1 | 0.995300 | 0.996745 |
| 1 | 0 | 1 | 1 | 0.995300 | 0.996745 |
| 1 | 1 | 0 | 1 | 0.004839 | 0.999685 |

**#3**

Example 3의 코드를 돌려본 결과는 다음과 같다.



이 결과(각 connection weight)를 가지고 neural network를 그려보면 아래와 같다.

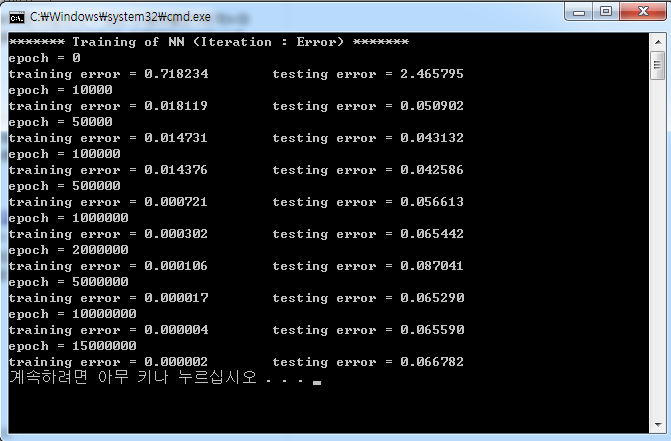


학습한 neural network에서 주어진 input에 대한 output값은 아래 표와 같다.

|  |  |  |
| --- | --- | --- |
| input | target | output |
| 0.0 | 0.0 | 0.003491 |
| 0.1 | 0.36 | 0.359689 |
| 0.2 | 0.64 | 0.640506 |
| 0.3 | 0.84 | 0.839169 |
| 0.4 | 0.96 | 0.963676 |
| 0.5 | 1.00 | 0.986559 |
| 0.6 | 0.96 | 0.967805 |
| 0.7 | 0.84 | 0.837722 |
| 0.8 | 0.64 | 0.641285 |
| 0.9 | 0.36 | 0.359317 |
| 1.0 | 0.00 | 0.005731 |

**#4**

Training error와 test error를 구해보면 다음과 같다.



Training error와 test error의 값의 크기가 처음부터 차이가 나는 것은 training data는 11개이고 test data는 51개이기 때문으로 보인다. (error들을 누적하여 합하므로 data의 개수가 많을수록 값이 커질 것이다)

처음에는 iteration 횟수를 5,000,000번으로 잡았지만, 이렇게 할 경우 overfit이 어디서 발생하는 지를 정확히 잡기 힘들어서 (위 사진에서 보다시피 testing error가 오십만 번째에서 증가하여 overfit이 생기는 듯 보이지만 오백만 번째에서 다시 testing error가 감소한다) 횟수를 천오백만 번으로 늘렸다.

위 결과를 이용해 표를 채워보면 다음과 같다. (표에는 문제에서 요구한 대로 5,000,000번째까지만 적었다)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Iteration** | 1,000 | 10,000 | 50,000 | 100,000 | 500,000 | 1,000,000 | 2,000,000 | 5,000,000 |
| **Error of training** | 0.718234 | 0.018119 | 0.014731 | 0.014376 | 0.000721 | 0.000302 | 0.000106 | 0.000017 |
| **Error of testing** | 2.465795 | 0.050902 | 0.043132 | 0.042586 | 0.056613 | 0.065442 | 0.087041 | 0.065290 |

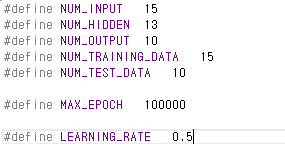
위 표를 통해 보면, Training error는 줄어들지만 testing error는 감소하는 시점은 500,000 ~5,000,000 iteration 사이에서 있는 것으로 보인다. Iteration이 500,000번 일 때 testing error가 이전에 비해 오히려 증가하므로 overfit이 일어난다고 추측해 볼 수 있다. 하지만 iteration이 5,000,000인 시점에서는 오히려 testing error가 감소하기 때문에 어느 지점부터 overfit이 생기는지 정확히 알아보기 위해 iteration을 15,000,000번까지 늘려야 했다 (testing error가 계속 감소하여 iteration이 500,000일 때보다 더 작아질 가능성이 있으므로). 그 결과, iteration이 10,000,000번인 시점에서는 testing error가 다시 증가한다는 것을 볼 수 있다.

따라서 확신할 수는 없지만, 적어도 iteration을 15,000,000번 실행한 결과, iteration이 약 100,000인 시점이 가장 error가 작았고 이후에는 약간의 변동은 있지만 전체적으로 error가 증가한다고 할 수 있다. 따라서 500,000 ~ 5,000,000번 iteration한 시점에서부터 overfit이 발생한다는 결론을 내릴 수 있다.

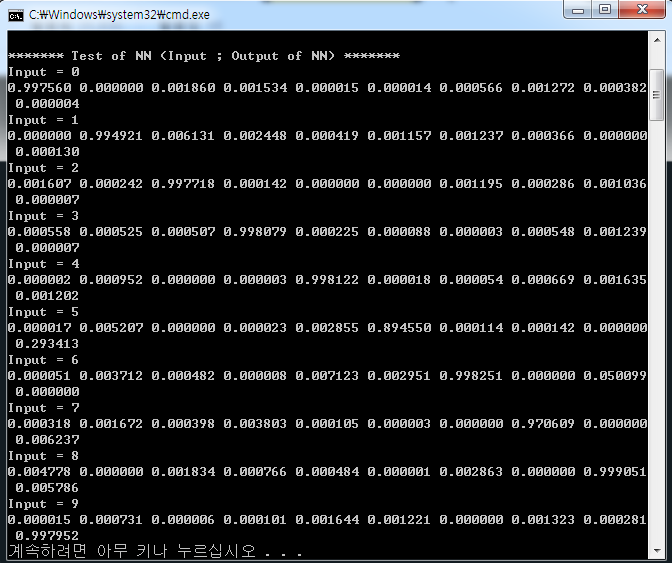
**#5**

Neural network를 주어진 training data로만 학습시켰을 때, test data 중 0과 2와 같이 training data와 구별하기 힘든 경우 정확도가 상당히 떨어지는 문제가 있었다. 때문에 이러한 몇몇 경우에 대해서 neural network가 더 잘 구분할 수 있도록 training data를 추가로 만들어 학습시켰다.

사용한 neural network의 parameter들은 아래와 같다.



학습이 끝난 neural network를 test data로 테스트 해보았을 때의 결과는 아래와 같다.



위 결과를 표로 간략하게 나타내보면 다음과 같다.

|  |  |
| --- | --- |
| Input | Output 중 해당 input에 대한 값 |
| 0 | 0.997560 |
| 1 | 0.994921 |
| 2 | 0.997718 |
| 3 | 0.998079 |
| 4 | 0.998122 |
| 5 | 0.894550 |
| 6 | 0.998251 |
| 7 | 0.970609 |
| 8 | 0.999051 |
| 9 | 0.997952 |

표에서 보면 대부분의 input에 대해 neural network이 정확도가 높지만, 5의 경우 다른 input들과 비교하였을 때 정확도가 상대적으로 낮다. 위의 결과값에서 보면 input이 5일 때 9의 값도 상당히 높은데, 이는 5의 test data가 9의 training data와 유사하기 때문으로 보인다. 하지만 이를 수정하기 위해서 5와 9를 구분해주는 training data을 넣을 경우 다른 input들에 대해서 정확도가 떨어질 수 있다고 생각하여 넣지는 않았다. (input을 추가로 만들어 넣은 경우는 정확도가 너무 낮아 사실상 구분이 불가능하거나 틀린 output이 나왔을 때였다. 5의 경우 값이 0.894550으로 비교적 정확했다고 보아 추가로 구분을 위한 training data를 만들어 학습시키지는 않았다)

표를 통해 학습한 neural network가 학습한 training data를 잘 구분할 뿐만 아니라, 약간의 오차가 있는 test data도 잘 구분함으로써 일반화하는 능력이 있다는 것을 알 수 있다.