**프로젝트 I I**

|  |  |
| --- | --- |
| 프로젝트 명 | *Project2* |
| 이 름 | *송창헌* |
| 문서 제목 | 결과보고서 |

|  |  |
| --- | --- |
| **Version** | 1.0 |
| **Date** | 31 |

|  |  |
| --- | --- |
| **팀원** | 송 창 헌 |
|  |
|  |
|  |
|  |

|  |
| --- |
| **CONFIDENTIALITY/SECURITY WARNING**  이 문서에 포함되어 있는 정보는 국민대학교 전자정보통신대학 컴퓨터공학부 및 컴퓨터공학부 개설 교과목 비주얼 컴퓨팅 최신기술 수강 학생 중 프로젝트 “프로젝트 2”를 수행하는 “송창헌”의 자산입니다. 국민대학교 컴퓨터공학부 및 “송창헌”의 서면 허락없이 사용되거나, 재가공 될 수 없습니다. |

**문서 정보 / 수정 내역**

|  |  |
| --- | --- |
| **Filename** | Project\_2.docx |
| **원안작성자** | 송창헌 |
| **수정작업자** | 송창헌 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 수정날짜 | 대표수정자 | Revision | 추가/수정 항목 | 내 용 |
| 2017-3-28 | 송창헌 | 1.0 | 작성 | 전체 내용 작성 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**목 차**

[1 개요(서론) 4](#_Toc478923839)

[1.1 프로젝트 개요 4](#_Toc478923840)

[2 개발 내용 및 결과물(본론) 5](#_Toc478923841)

[2.1 목표 5](#_Toc478923842)

[2.2 연구/개발 내용 및 결과물 5](#_Toc478923843)

[2.2.1 연구/개발 내용 5](#_Toc478923844)

[3 결과(결론) 15](#_Toc478923845)

# 개요(서론)

## 프로젝트 개요

본 프로젝트는 비주얼 컴퓨팅 최신기술 두번째 프로젝트로, 주어진 2D sample pattern 데이터를 이용하여 3가지 클래스 중 두가지를 클래스들을 각각 분류하는 선형분류기를 작성하는 것을 목표로 한다.

1) Perceptron algorithm 을 사용하여 w1, w2 를 구분하는 선형분류기를 작성하고 초기 weight vector 와 학습율을 다양하게 시도하여 결과를 확인한다.

2) Batch relaxation algorithm 을 사용하여 w1 과 w3 를 구분하는 선형분류기를 작성하고 Margin b 를 0.1 과 0.5, 초기 weight vector 를 (0, 0, 0)로 하였을 경우의 결과들을 확인한다.

3) Widrow-Hoff (LMS) algorithm 을 사용하여 w1 과 w3 를 구분하는 선형분류기를 작성하고 다양한 initial weight vector a, margin vector b, threshold, learning rate 를 시도해 각각 결과를 확인한다.

# 개발 내용 및 결과물(본론)

## 목표

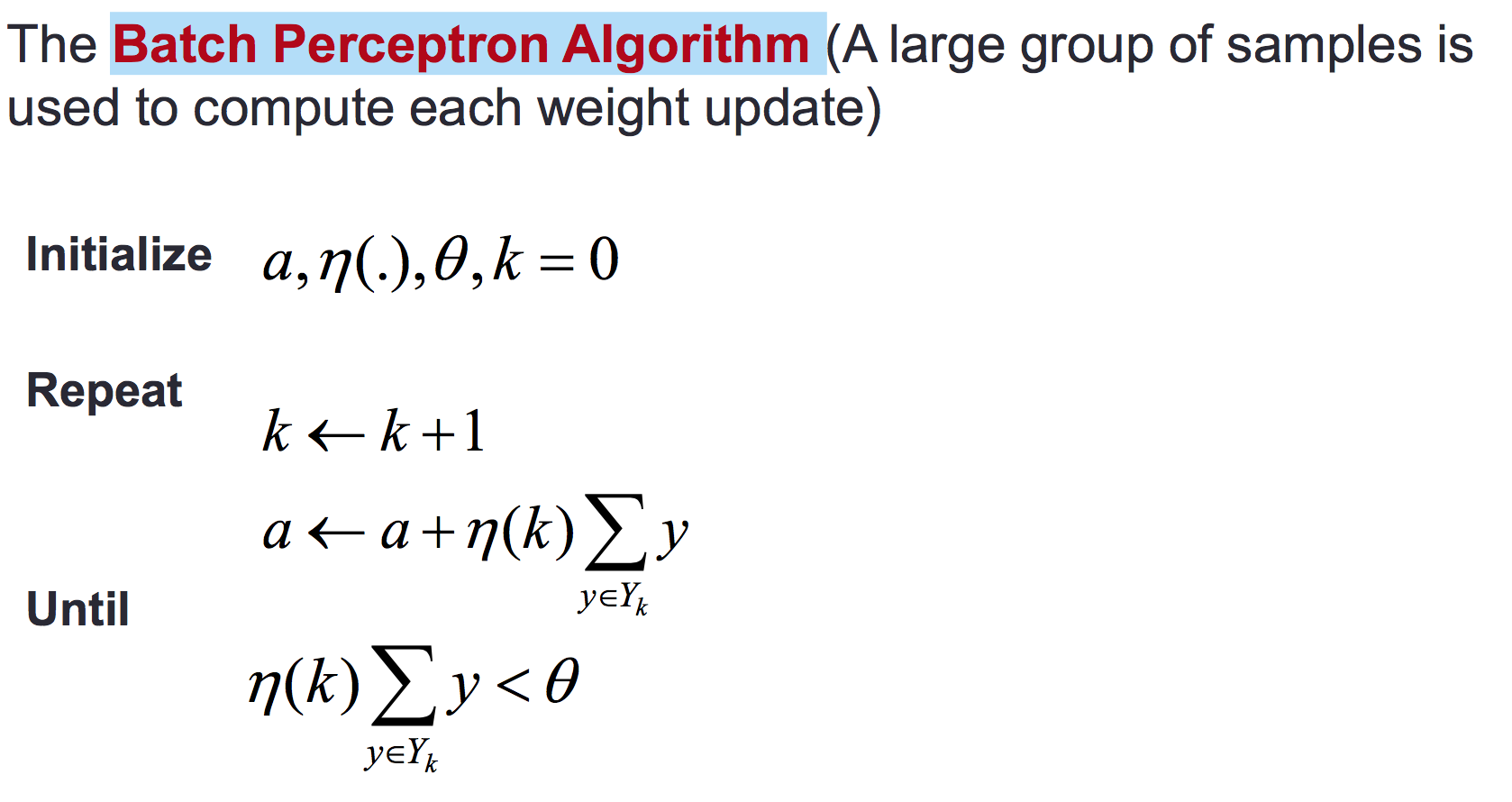
2D sample pattern을 사용하여 Perceptron algorithm, Batch relaxation algorithm, 그리고 Widrow-Hoff(LMS) algorithm 을 사용하여 각 클래스를 구분하는 선형 분류기를 작성한다.

Weight vector와 학습율, margin, threshold를 변형하며 각 결과를 보고 결과를 비교한다.

## 연구/개발 내용 및 결과물

### 연구/개발 내용

1. Perceptron algorithm 을 사용하여 w1, w2 를 구분하는 선형분류기를 작성하고 초기 weight vector 와 학습율을 다양하게 시도하여 결과를 확인한다.



[그림 1] Perceptron Algorithm

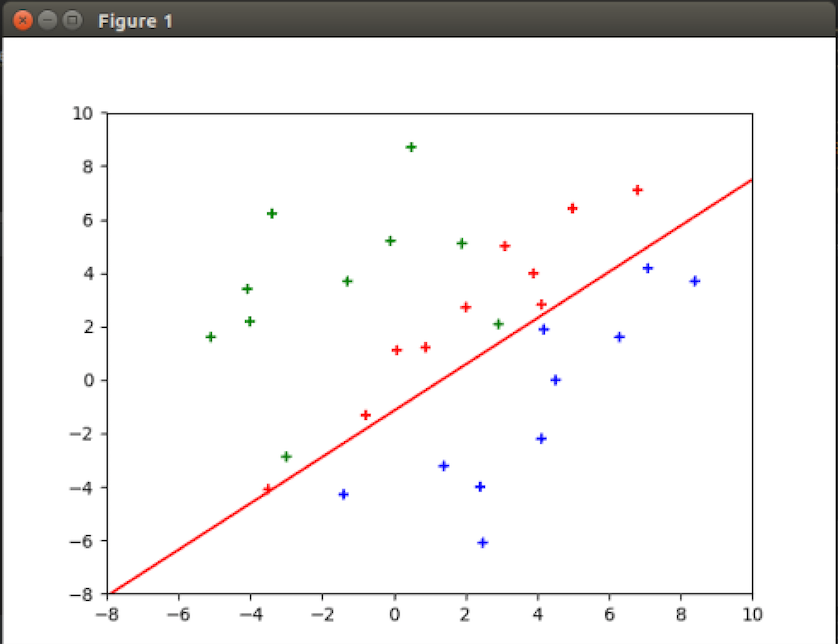
Perceoptron algorithm을 구현하기 위해 python3와 모듈인 numpy를 사용하였고, 결과를 쉽게 보기 위해 pyplot을 사용하여 화면에 나타내었다.

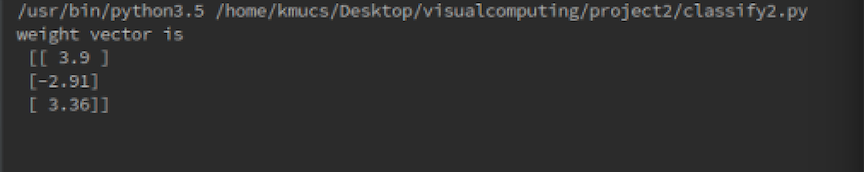


[그림 ]Percep 함수 구현

주어진 2D sample pattern을 “data.txt”에 작성하여 두고 이를 파일입출력을 통해서 받아와 각각의 list에 저장한다. 그 후 클래스 w1과 w2를 분류하는 분류기의 weight vector를 구하기 위해 bias를 포함하고 있는 데이터 셋으로 만들어 3X20 행렬로 만들어 ‘Percep’ 함수에 인풋으로 넣어준다. 이 후 함수 내부의 계산을 통해 결과가 음수가 나오는 열을 대상으로 indexing list를 만들어 가중치를 계산해준다. 계산을 통해 음수가 나오는 열이 없을 때까지 반복하여 실행하고 결과 weight vector를 통해 화면에 나타내었다.

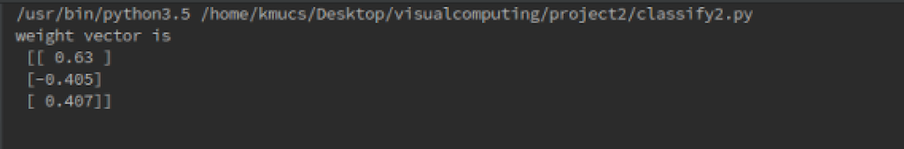
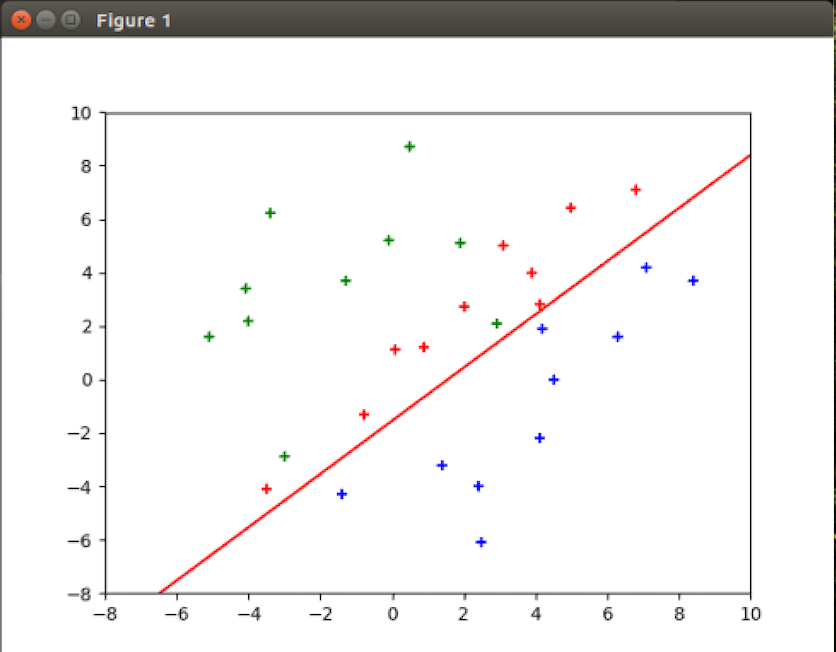
구분을 쉽게 하기 위하여 처음 weight vector를 (1,1,-1) 로 하고, 초기 학습율을 0.1로 시작하고 계산한 결과,





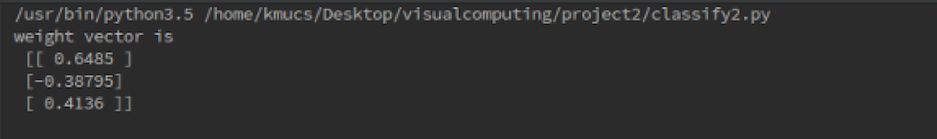
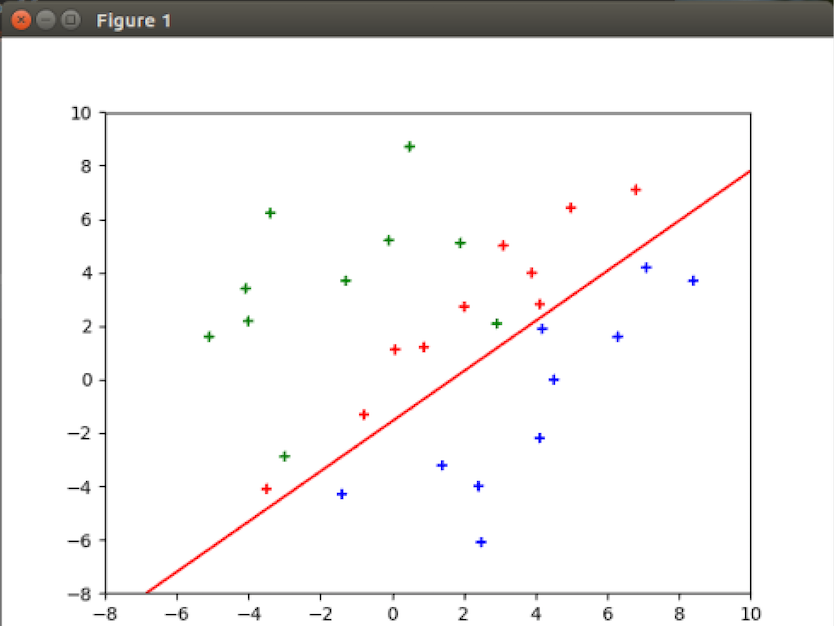
[그림 ] 학습율 : 0.1 / weight vector : (1,1,-1)

위와 같은 결과를 보였다. w1 을 나타내는 붉은색 ‘+’와 w2를 나타내는 파란색 ‘+’를 어느정도 구분하고 있음을 볼 수 있다. 다음으로 반복을 많이 하지만 조금 더 정교한 변화를 주기 위해 학습율을 0.01로 변경하고 실행한 결과,



[그림 ] 학습율 : 0.01 / weight vector : (1,1,-1)

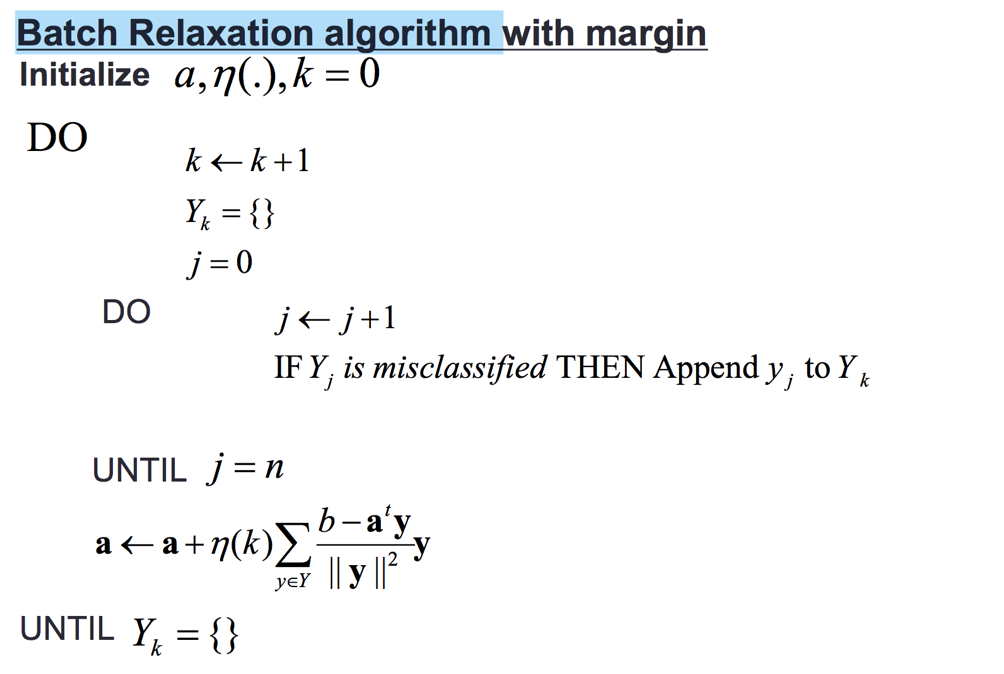
위와 같은 결과를 보였다. 마지막으로 학습율을 0.0095로 변경하여 시험해본 결과는 다음과 같다.



[그림 ] 학습율 : 0.0095 / weight vector : (1,1,-1)

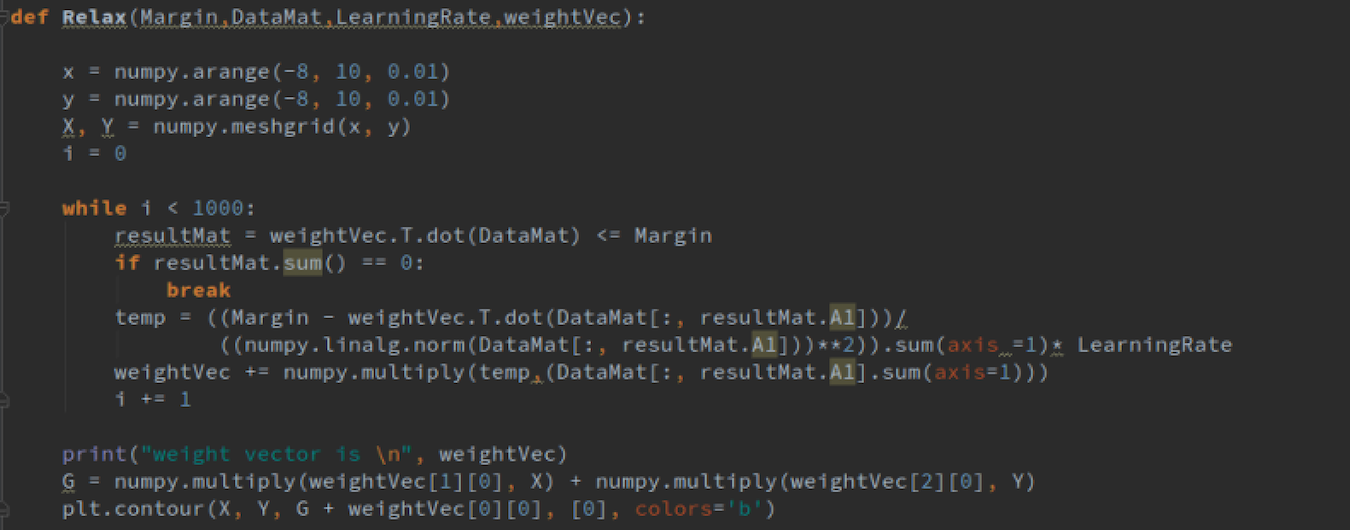
이처럼 학습율이 변할 수록 결과로 보여지는 선형 분류기는 변형됨을 알 수 있고 이로인해 0.0095의 학습율이 주어진 data를 기반으로 두 클래스를 분류하기에 적합함을 알 수 있다.

2) Batch relaxation algorithm 을 사용하여 w1 과 w3 를 구분하는 선형분류기를 작성하고 Margin b 를 0.1 과 0.5, 초기 weight vector 를 (0, 0, 0)로 하였을 경우의 결과들을 확인한다.



[그림 ] Relaxation Algorithm

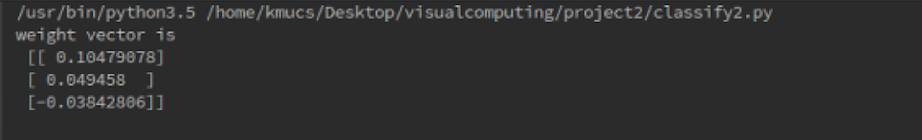
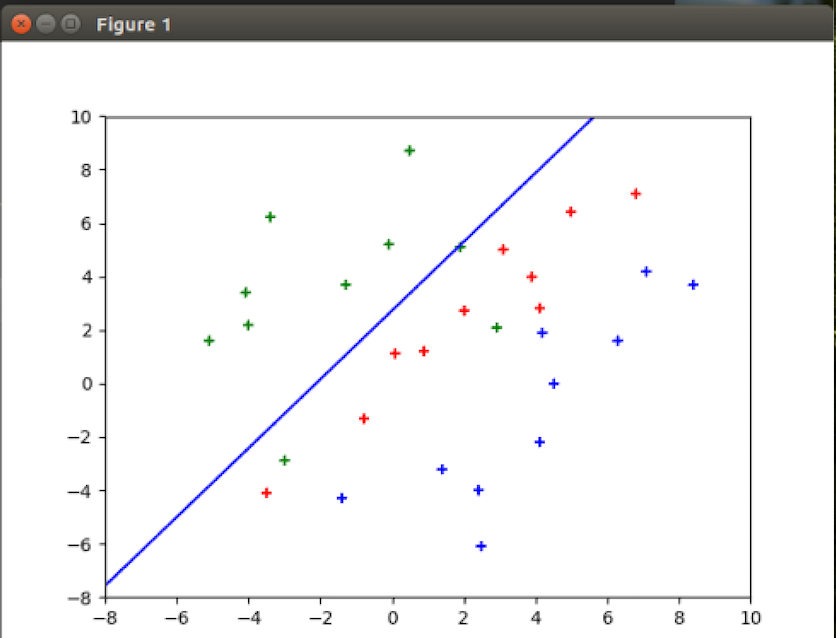
Batch relaxation algorithm을 구현하기 위해 python3와 모듈인 numpy를 사용하였고, 결과를 쉽게 보기 위해 pyplot을 사용하여 화면에 나타내었다.



[그림 ] Relax 함수 구현

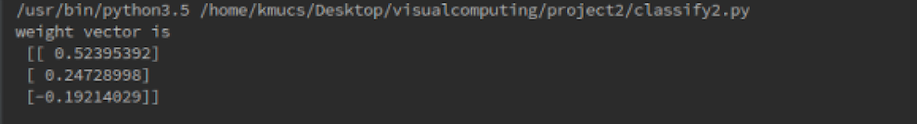
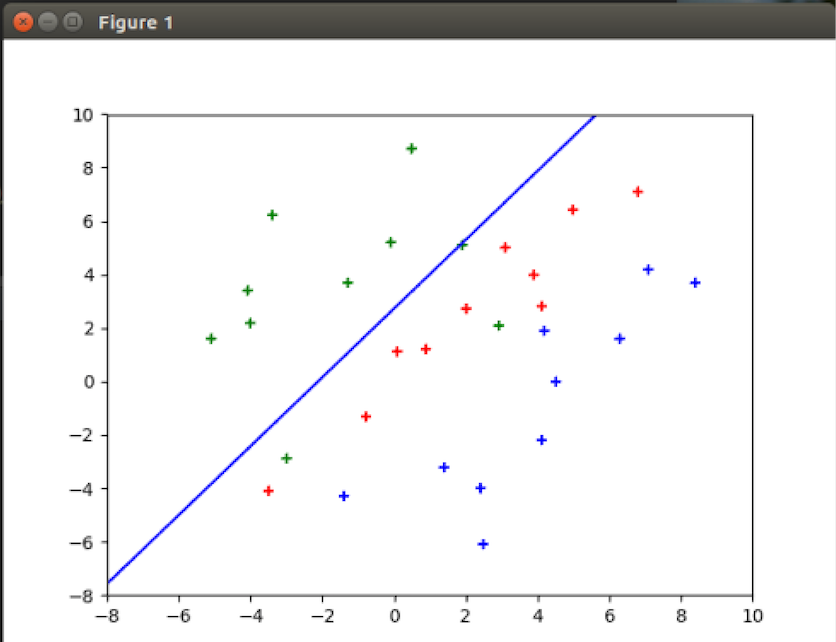
주어진 2D sample pattern을 “data.txt”에 작성하여 두고 이를 파일입출력을 통해서 받아와 각각의 list에 저장한다. 그 후 클래스 w1과 w3를 분류하는 분류기의 weight vector를 구하기 위해 bias를 포함하고 있는 데이터 셋으로 만들어 3X20 행렬로 만들어 ‘Relax’ 함수에 인풋으로 넣어준다. 이 후 함수 내부의 계산을 통해 결과가 Margin 이하의 계산값이 나오는 열을 대상으로 indexing list를 만들어 가중치를 계산해준다. 계산을 통해 Margin 이하의 계산값이 나오지 않을 때까지 반복하여 실행하고 결과 weight vector를 통해 화면에 나타내었다.

학습율을 0.01로 고정하고 Margin을 0.1로 하고 초기 weight vector를 (0, 0, 0)으로 계산한 결과, 반복이 종료되지 않고 계속 진행이 되기 때문에 반복을 1000번으로 지정하고 다시 계산을 하였고 다음과 같은 결과를 얻었다.



[그림 ] 학습율 : 0.01 / weight vector : (0,0,0) / margin : 0.1 / iter : 1000번

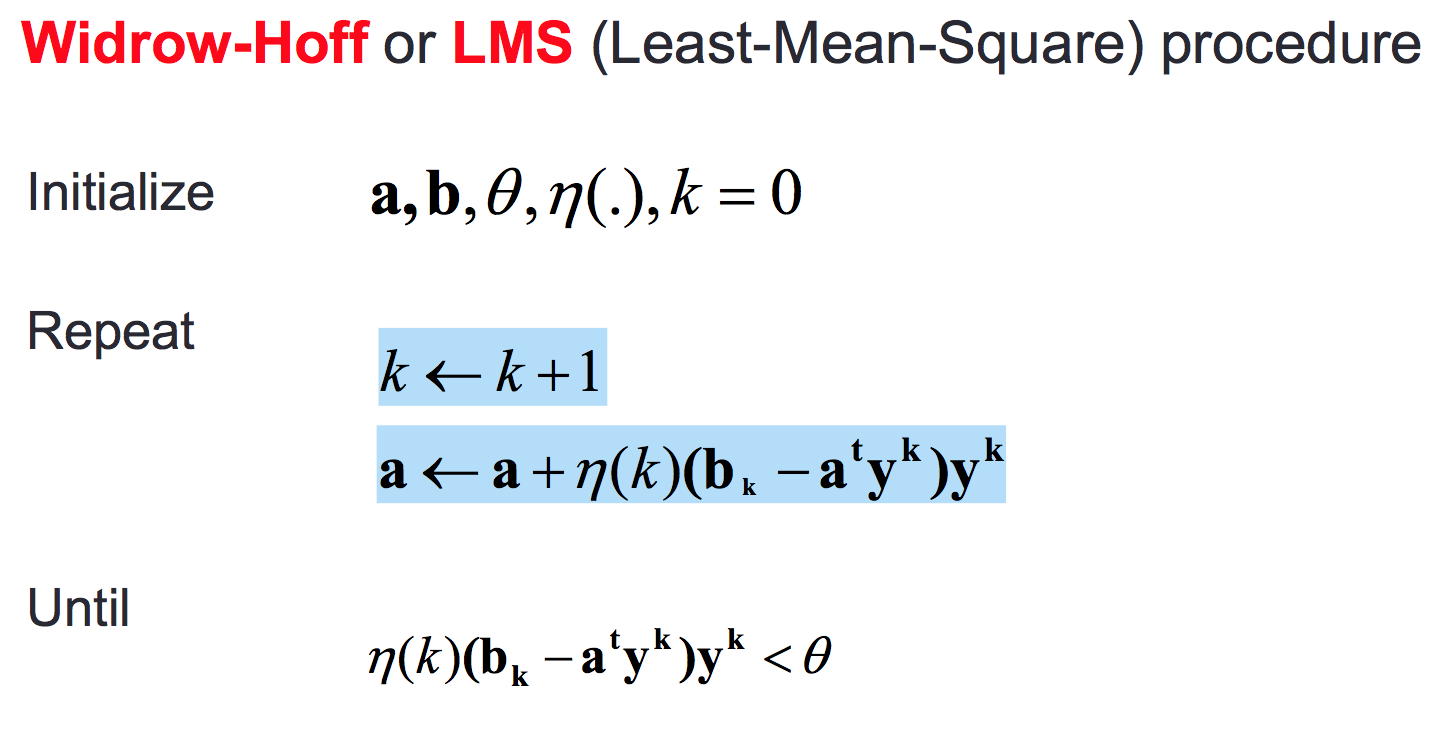
w3인 녹색 ‘+’와 w1인 붉은색 ‘+’가 어느정도 구분이 되고 있음을 볼 수 있다. 다음으로 margin을 0.5로 바꾸어 확인한 결과,



[그림 ] 학습율 : 0.01 / weight vector : (0,0,0) / margin : 0.5 / iter : 1000번

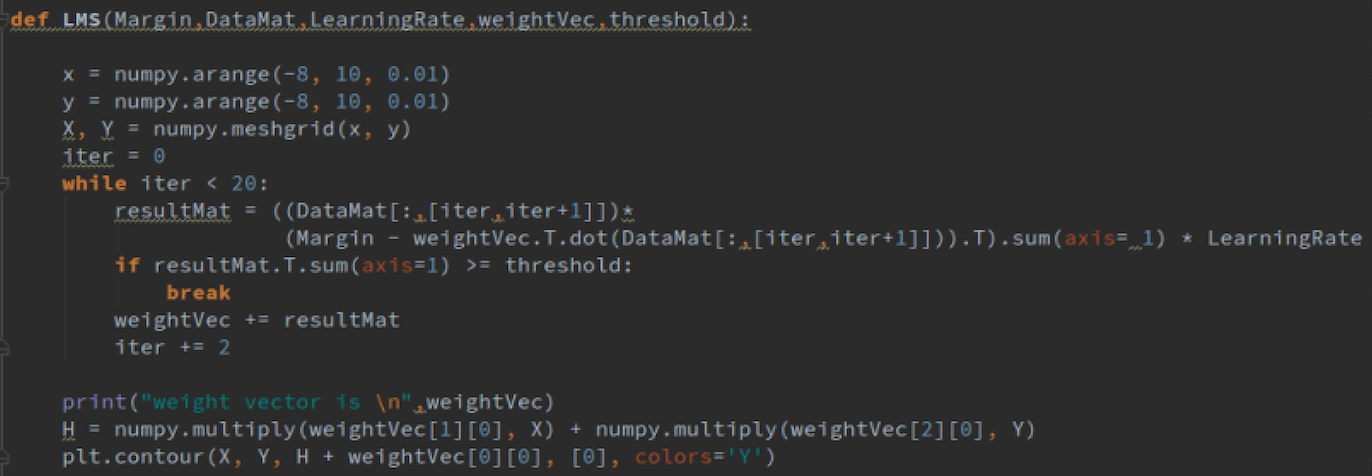
위의 실험과 같은 결과를 보이고 있다. 따라서 계속해서 반복이 진행되지만, 반복이 진행 될 수록 일정한 비율을 유지 하고 있음을 볼 수 있다.

3) Widrow-Hoff (LMS) algorithm 을 사용하여 w1 과 w3 를 구분하는 선형분류기를 작성하고 다양한 initial weight vector a, margin vector b, threshold, learning rate 를 시도해 각각 결과를 확인한다.



[그림 ] Widrow-Hoff / LMS Algorithm

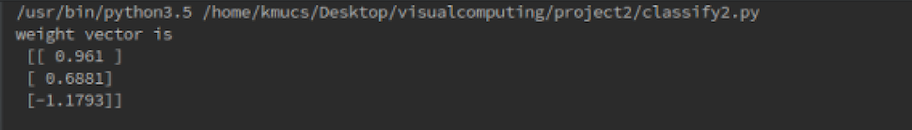
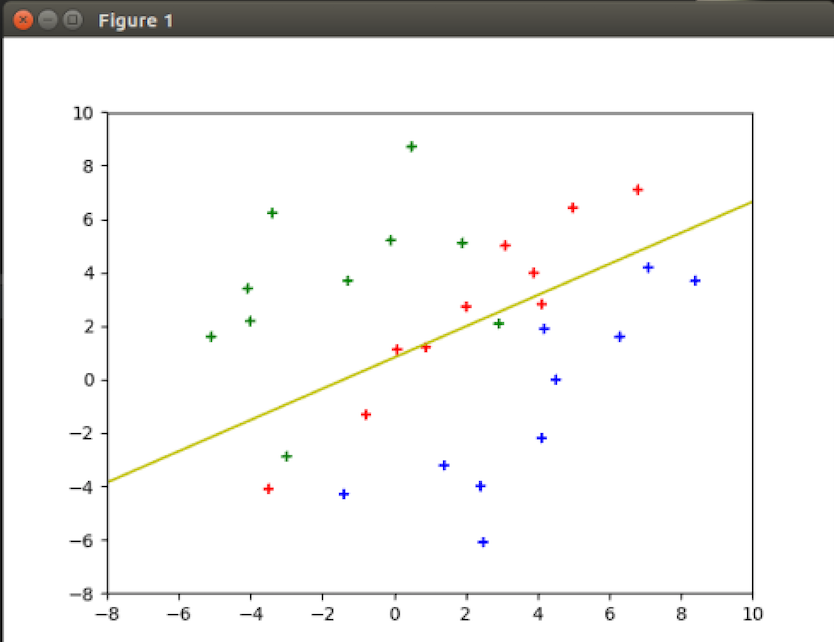
Widrow-Hoff (LMS) algorithm을 구현하기 위해 python3와 모듈인 numpy를 사용하였고, 결과를 쉽게 보기 위해 pyplot을 사용하여 화면에 나타내었다.



[그림 ] LMS 함수 구현

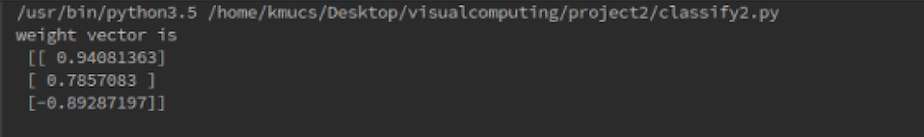
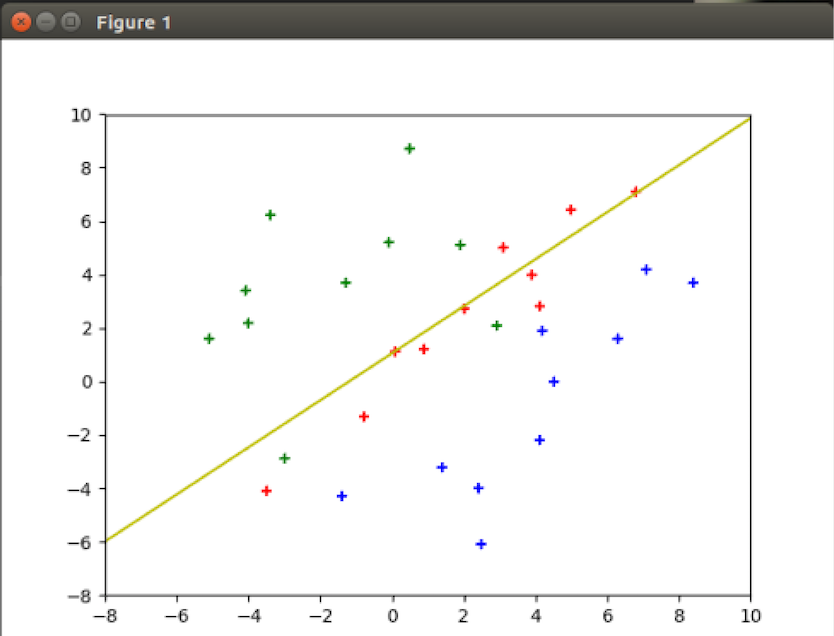
주어진 2D sample pattern을 “data.txt”에 작성하여 두고 이를 파일입출력을 통해서 받아와 각각의 list에 저장한다. 그 후 클래스 w1과 w3를 분류하는 분류기의 weight vector를 구하기 위해 bias를 포함하고 있는 데이터 셋으로 만들어 3X20 행렬로 만들어 ‘LMS’ 함수에 인풋으로 넣어준다. 각 반복마다 각클래스에서 하나의 쌍을 식에 맞게 계산하고,계산을 통해 threshold를 넘는 값이 나오지 않을 때까지만 반복하여 실행한다. 이 후 결과 weight vector를 통해 화면에 나타내었다.

초기 weight vector를 (1,1,-1)로 하고, margin을 (0.5,0.5), threshold를 0.1, 학습율을 0.01로 하고 결과를 구하면,다음과 같은 결과를 보인다.



[그림 ] 학습율 : 0.01 / weight vector : (1,1,-1) / margin : (0.5,0.5) / threshold 0.1

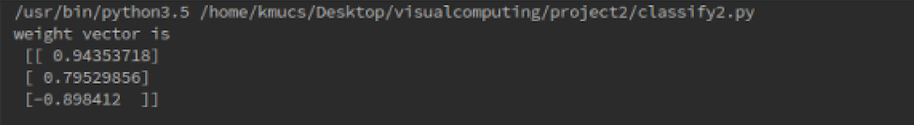
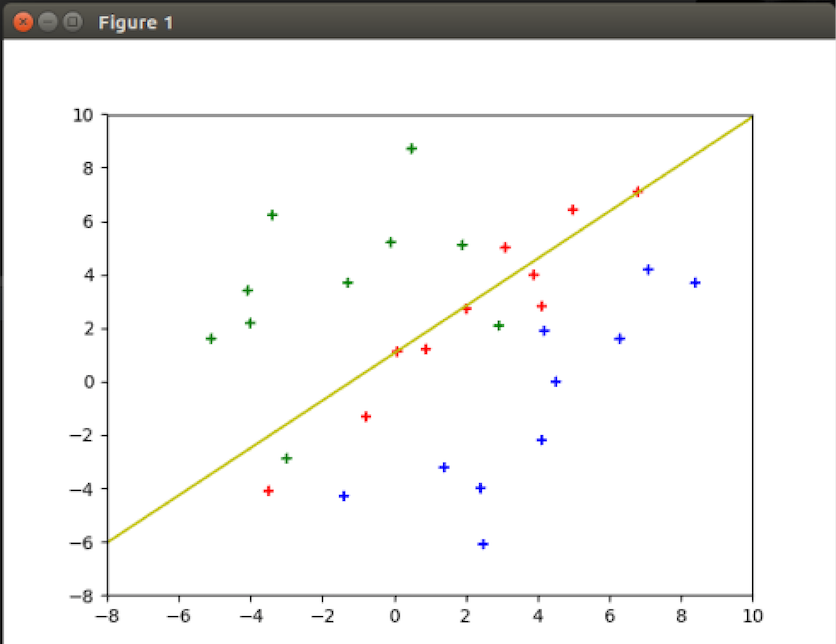
위와 같이 w1을 나타내는 붉은색 ‘+’와 w3를 나타내는 녹색 ‘+’를 제대로 구분하지 못하고 있음을 알 수 있다. margin 을 바꾸어 실험한 결과 별다른 영향을 못 주어, 학습율을 0.001로 변경하고 실험한 결과는 다음과 같았다.



[그림 ] 학습율 : 0.001 / weight vector : (1,1,-1) / margin : (0.5,0.5) / threshold 0.1

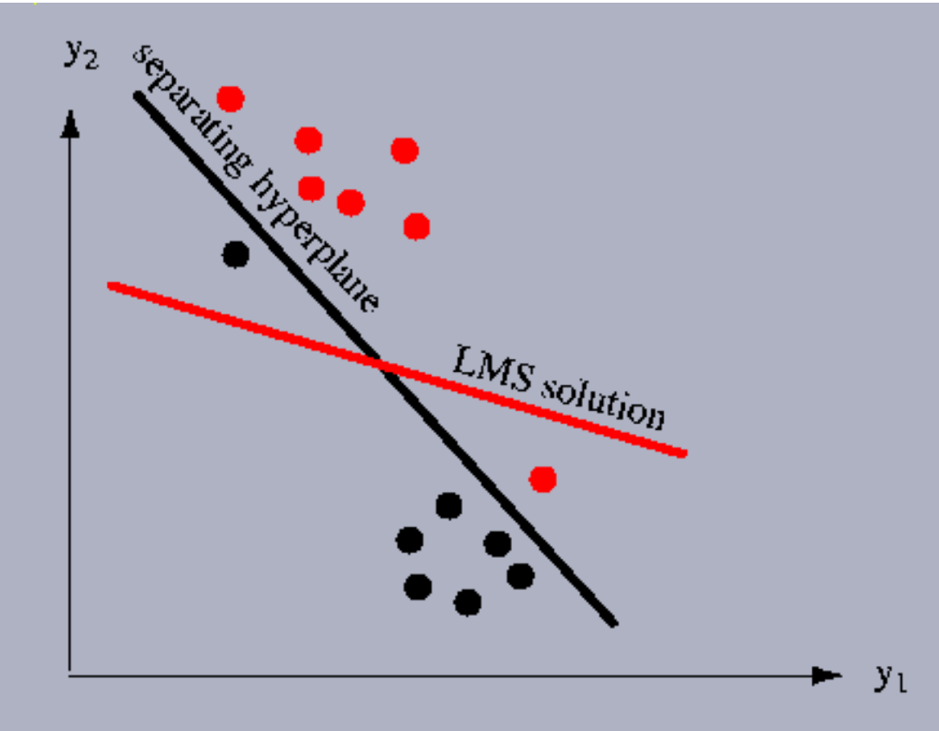
이에 더해 threshold를 0.2로 변화시켜 결과를 확인하면 위와 같은 결과를 확인할 수 있다. 따라서 threshold는 다른 조건을 일치 시킨 상태에서 실제 변경에 크게 영향을 주지 못함을 확인할 수 있다.

마지막으로 weight vector를 (1,1,-1)로 하고, margin을 (0.5,0.5), threshold를 0.1, 학습율을 0.0095로 하고 결과를 구하면,



[그림 ] 학습율 : 0.0095 / weight vector : (1,1,-1) / margin : (0.5,0.5) / threshold 0.1

이처럼 각각의 클래스의 두 데이터를 제외하고 분류가 되는 모습을 확인 할 수 있다. 이는 [그림15]의 선형 분류 중 LMS에서 볼 수 있는 문제와 같이 데이터 쌍에 따라 오분류 되는 경우가 발생할 수 있음을 확인 할 수 있다.



[그림 ] LMS 솔루션의 오분류

# 결과(결론)

각 선형분류기를 작성하고 초기값을 변경하면서 최적화된 선형분류기를 만드는 조건을 찾으면서, 데이터 셋에 따라 초기값 설정, 학습율 설정, Threshold, margin의 설정이 결과에 영향을 미치는 것에 대해 파악할 수 있었다.