|  |
| --- |
|  |
| Project No.1 |
|  |
| Learning Iris Data classifition without tools |

**소프트웨어융합대학**

Due data : 2017 March 26

Student ID/Name : 20113337 최영근

Contents

1. 프로젝트 개요 및 환경
   1. 개요
   2. 컴파일 환경, 사용 언어, etc
2. 프로젝트 설명
   1. Project 1-1-a
   2. Project 1-1-b
   3. Project 1-1-c
   4. Project 1-2-a, 1-2-b, 1-2-e
   5. Project 1-2-c
   6. Project 1-2-d
   7. Project 1-2-f

1. 프로젝트 개요 및 환경

1. 프로젝트 개요

주어진 Iris Data를 Machine Learning tool 없이 분류할 수 있는 Classifier를 만든다.

분류기를 만들기 위해 Machine Learing의 기초인 Bayes Classifier이론을 바탕으로 훈련

데이터(Training Data)로부터 객체를 분류하는 **지도 학습** (Supervised Learning) 방법을

사용한다. 훈련 데이터들의 값을 출력하고 주어진 입력 벡터가 어떤 종류의 값인지

[분류](https://ko.wikipedia.org/wiki/%EB%B6%84%EB%A5%98)(Classification) 하고 화면에 출력한다. 훈련 데이터로부터 주어진 데이터가 올바른지

확인 한다.

1. 컴파일 환경 및 사용언어, etc

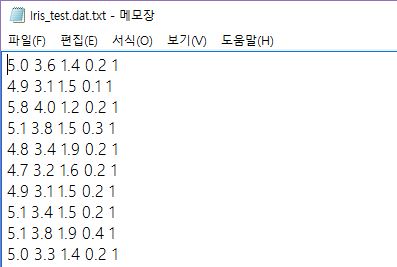
OS : MAC OS

Compile : MAC OS Terminal

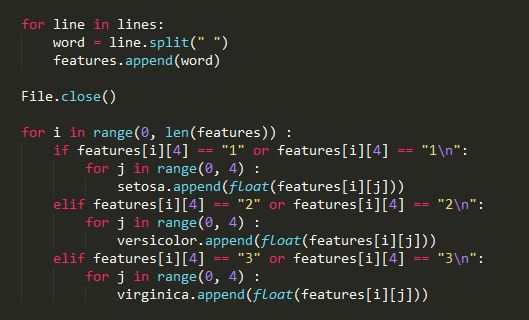
사용 언어 : Python ver2.7.4

사용 라이브러리 : numpy, matplotlib

1. 프로젝트 개요 및 환경



* + - * Data file format

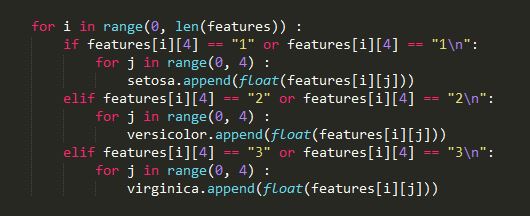


* + - * File read code

주어진 Input data를 space 단위로 재 배열 -> \r, \n 을 제거하여 코드 상에서 space 단위로 쉽게 스플릿하여 저장하기 위함.

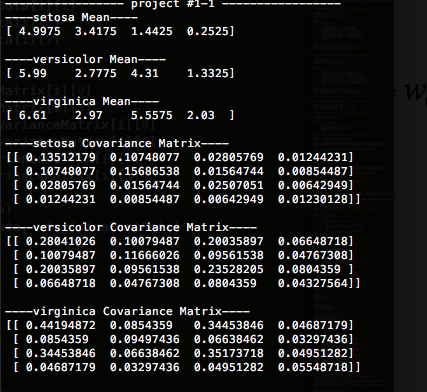
2. 프로젝트 설명

1. Project 1-1-a



1. 각각의 클래스마다 리스트의 형태로 데이터를 저장
2. 리스트로 저장된 각 클래스의 데이터들에 대한 평균과 공분산 행렬을 계산

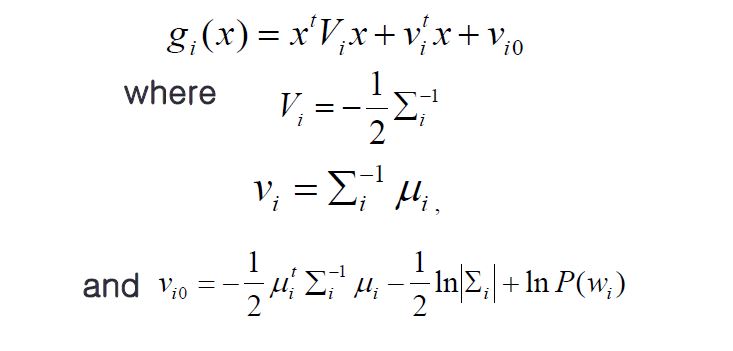
평균 : 1\*4 Matrix

공분산 행렬 : 4\*4 Matrix

* + - * 각 클래스의 평균, 공분산 행렬 출력 결과

2. 프로젝트 설명

1. Project 1-1-b

Discriminant Functions을 이용하여 Decision Boundary을 계산

이 공식을 이용하여 각 클래스의 g(x) 값을 계산



* Code of calculating discriminant function

각 클래스 마다 테스트 데이터에 대한 g값의 list를 구함 (size = 10)

2. 프로젝트 설명

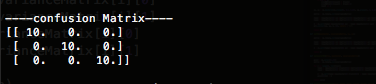
1. Project 1-1-c

W12 -> g1(x) – g2(x) = 0

W23 -> g2(x) – g3(x) = 0

W31 -> g3(x) – g1(x) = 0

위의 공식을 이용하여 각 클래스의 Hyperplane을 분리하여 입력된 테스트 데이터가 잘 분류 되었는지 계산



* Discriminant function을 사용하여 구한 confusion matrix의 출력 결과

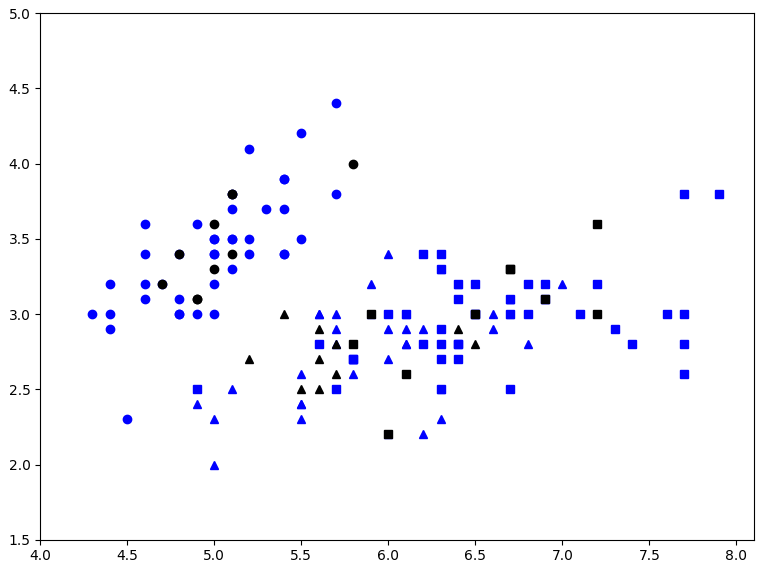
2. 프로젝트 설명

1. Project 1-2-a, 1-2-b, 1-2-e

Training data의 2가지 특징만 가지고 데이터를 분류하기 위해 각 클래스 별로

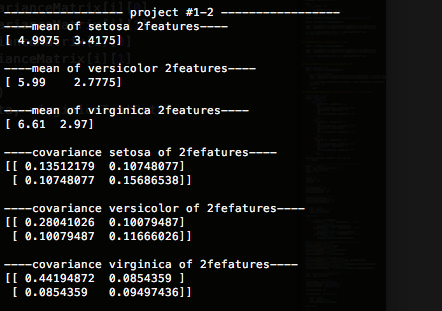
1-1에서 저장했던 데이터, 평균, 공분산 행렬에서 2가지 특징만 가져와서

새로운 변수에 저장하고 2차원 데이터들을 출력



* Test data(black color), Training data(blue color)를 모두 출력한 결과

(원 : setosa data, 삼각형 : versicolor data, 사각형 : virginica data)



* Mean matrix & covariance Matrix of 2features training data

2. 프로젝트 설명

1. Project 1-2-c

마할라노비스 거리 (Mahalanobis distance) : 평균과의 거리가 표준편차의 몇 배인지 나타내는 값

C:\Users\choey\AppData\Local\Microsoft\Windows\INetCache\Content.Word\마할공식.jpg

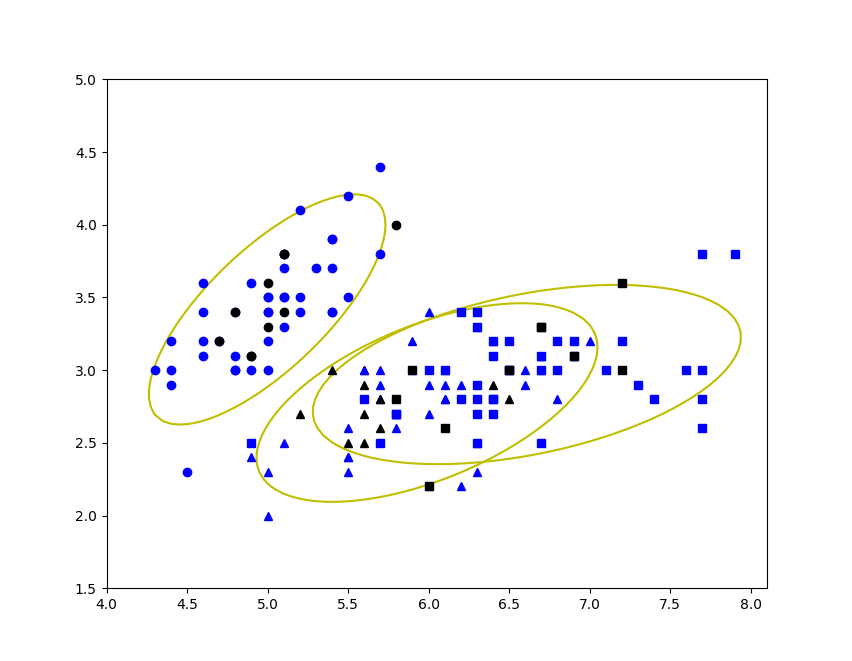
* Mahalanobis distance formula



* Code of calculating & ploting Mahalanobis distance = 2

주어진 트레이닝 데이터가 어느 일정범위안에 존재하기 때문에 x와 y의 범위를 지정하고 그 사이 데이터들을 0.05의 단위로 쪼개어 마할라노비스 거리 2를 갖는 데이터(Z) 들을 모두 출력

2. 프로젝트 설명



* 세 클래스의 마할라노비스 거리가 2인 데이터들을 출력한 결과

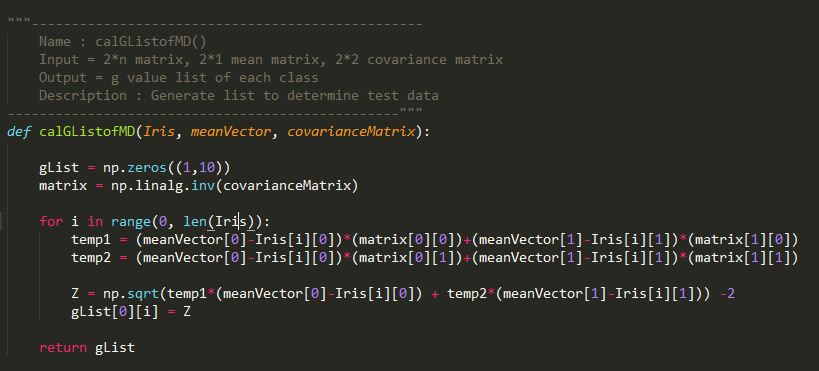
(원 : setosa data, 삼각형 : versicolor data, 사각형 : virginica data)

2. 프로젝트 설명

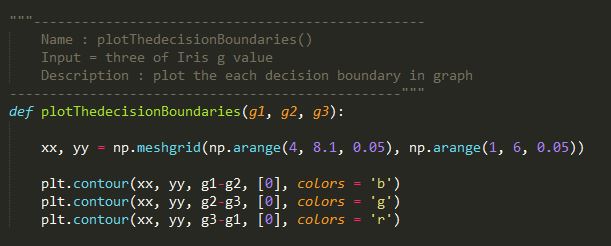
1. Project 1-2-d

1-1-b의 방법과 동일하게 계산

이번엔 마할라노비스 거리가 2인 것들의 집합인 g(x) 리스트를 가지고 Hyperplane을 분리

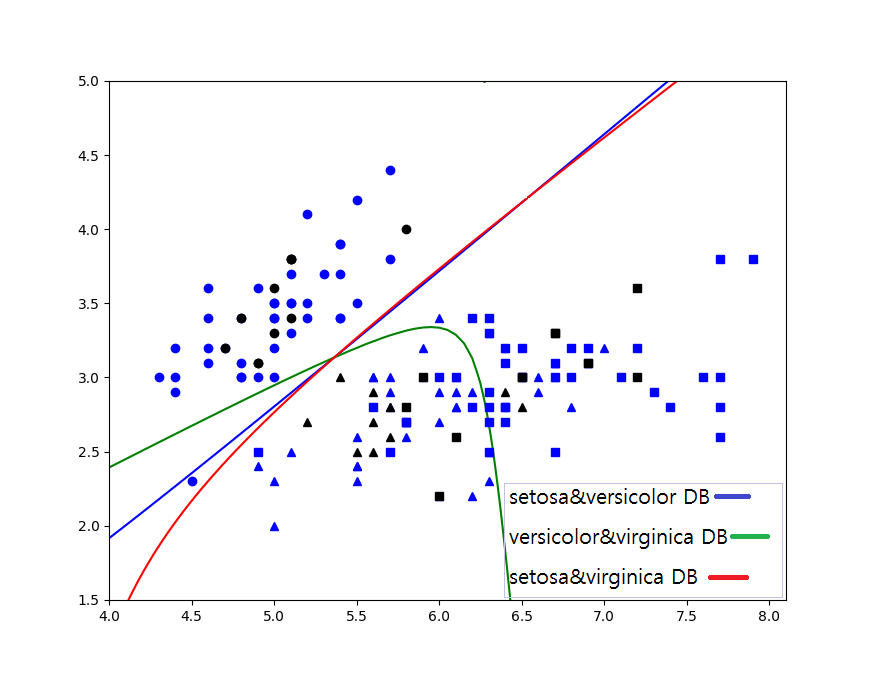


* 각 클래스마다 마할라노비스 거리가 2인 데이터들을 리스트에 넣고 반환하는 함수



* 위에서 구한 각 클래스의 g값을 가지고 decision boundary를 계산하고 화면에 출력

2. 프로젝트 설명



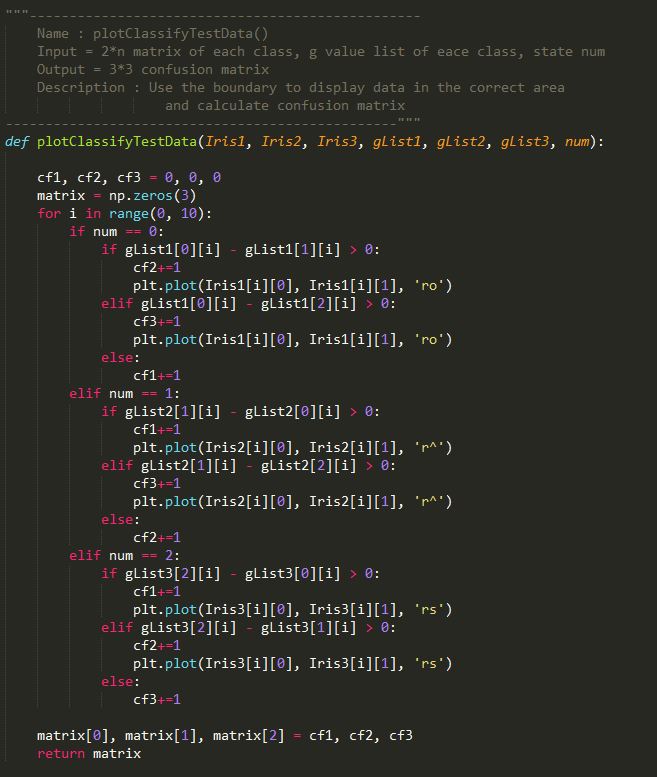
* Decision boundary을 출력한 결과

2. 프로젝트 설명

1. Project 1-2-f

위에서 구한 Decision boundary를 가지고 test data를 분류하고

confusion matrix를 계산한다. 예측과 다른 데이터들은 다른 색으로 화면에 출력 한다.



* Code of classify & plot Testdata and calculating confusion matrix

2. 프로젝트 설명

1-1-b 에서 구했던 것처럼 Discriminant Functions을 이용하여 Decision Boundary을 계산.

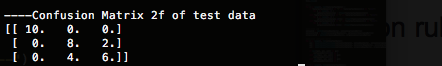
인자로 각각의 테스트 데이터, g(x) 리스트 값, 테스트 데이터의 예측 값이 들어간다.

테스트 데이터의 예측 값에 따라 각각의 if문 안으로 들어가게 되고 해당 테스트

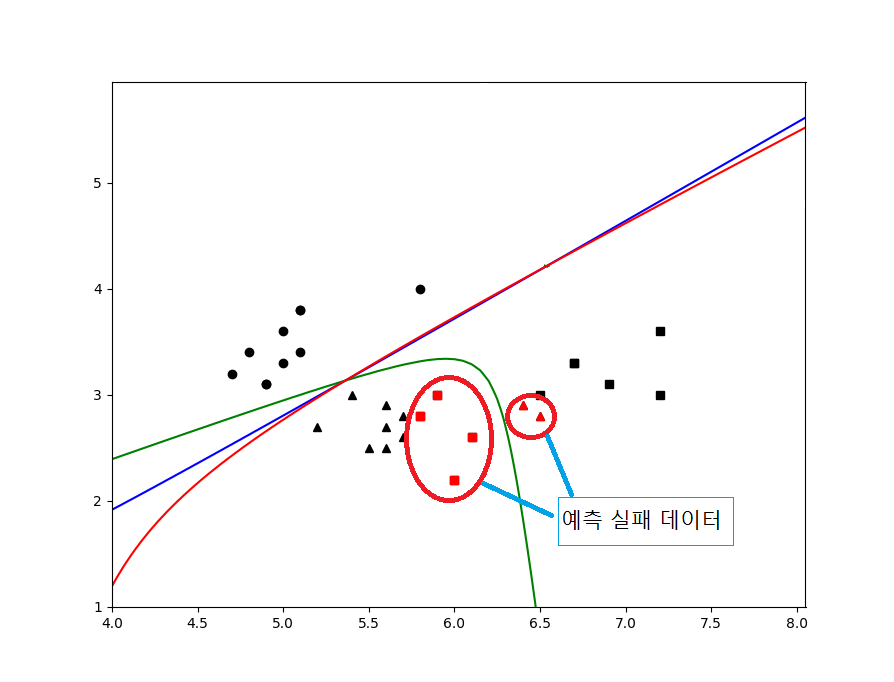
데이터가 decision boundary 안에 있는지 판별하여 다른 클래스에 들어가 있다면

다른 색(빨간색)으로 출력한다.

마지막으로 클래스 마다 confusion matrix를 계산하여 반환한다.



* 최종적인 confusion matrix 출력 결과



* 분류한 테스트 데이터들의 최종적인 출력 결과

Setosa data -> 모두 정상

Versicolor data -> 2개 오류 (virginica 데이터에 2개 포함됨)

Virginica data -> 4개 오류 (versicolor 데이터에 4개 포함됨)

의 결과를 도출하였습니다.