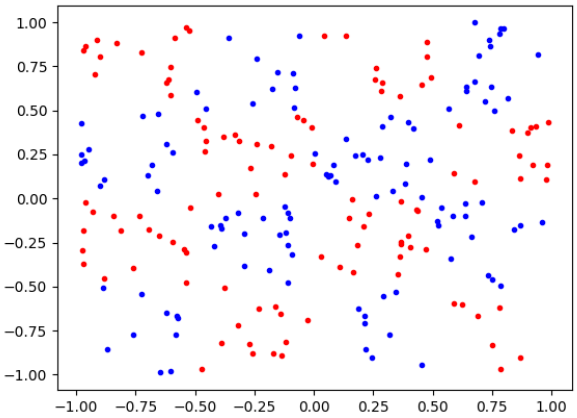
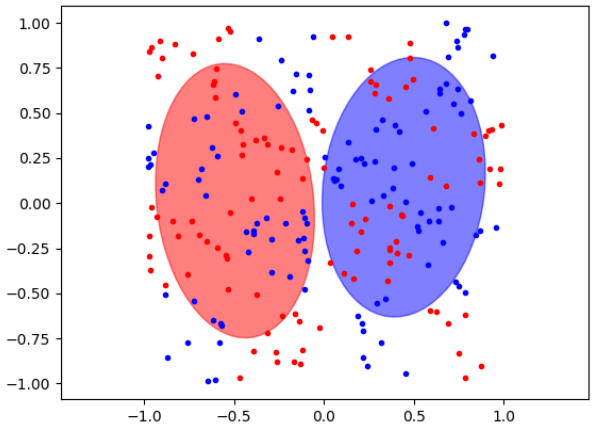
2022311499 고우빈

Figure 1.a 는 p1 train dataset 을 좌표평면에 나타낸 것이며, label 이 0 인 경우 파란 점, 1 인 경우 빨간 점으로 표시했다. 빨간 점과 파란 점이 위치를 특정할 수 없을 정도로 뒤섞인 것을 볼 수 있다. 여기서 예상할 수 있는 것은, 보통의 예시처럼 Gaussian Component 를 label 의 가짓수만큼 설정하는 것(현재의 경우 2 개)으로는 좋은 결과를 얻을 수 없다는 것이다.
Figure 1.b 는 component 의 개수가 2 개이고, covariance_type 을 full 로 설정한 Gaussian Mixture Model 을 fit 시킨 뒤, 2 개의 component 를 Figure1.a 위에 그린 것이다. 예상한 대로 데이터의 분포를 제대로 반영하지 못하고 있는 모습이다.





또한, Train Accuracy 도 53.3%에 불과하다.

Figure1.a (p1_train_dataset)

Figure1.b (p1_train_dataset with gaussian component)

Method

정확성을 높이기 위해 p1 의 데이터들을 같은 크기의 4x4 섹션으로 나누고, 각 섹션마다 다른 label 을 부여할 것이다. 가장 왼쪽 위 섹션부터 가장 오른쪽 아래 섹션까지 0 ~ 15 의 새로운 class 를 새로 부여한다. 0, 2, 5, 7, 8, 10, 13, 15 번 클래스는 빨간 점(원래 label 이 1)이 대부분을 차지하고, 나머지 8 개의 클래스는 파란 점(원래 label 이 0)이 대부분을 차지할 것이다.
Gaussian Component 의 개수를 n 개로 설정하면, predict 했을 때의 결과도 n 개의 class 가 나와 기존 label 인 0, 1 외에도 2, 3, n 과 같은 class 로 예측할 것이기 때문에 이와 같은 방법을 선택했다.
섹션의 개수만큼 16 개의 component 를 생성하고, predict 결과 또한 0~15 범위 내에서 나오도록 한 뒤, predict 결과가 0, 2, 5, 7, 8, 10, 13, 15 에 속한다면 빨간 점(원래 label 1), 1, 3, 4, 6, 9, 11, 12, 14 에 속한다면 파란 점(원래 label 0)으로 간주할 것이다. 예상되는 출력은 Figure 2.a 이고, 실제 출력은 Figure2.b 이다.

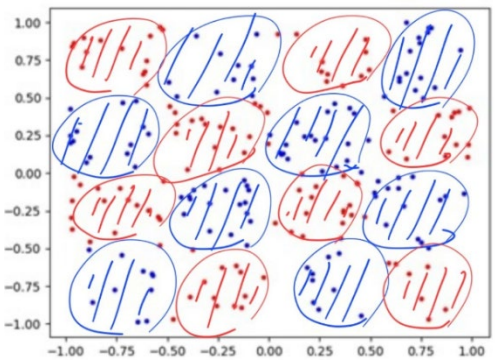


Figure2.a

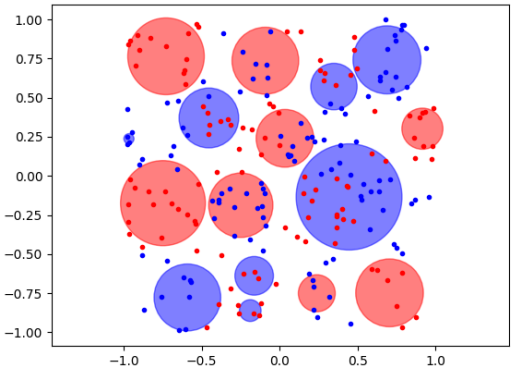


Figure2.b

Result

초기 파라미터들은 다음과 같다.
n_components=16, tol=0.0001, covariance_type="spherical", max_iter=100
실행한 결과, 바랐던 결과와는 다르게 각 covariance 들의 위치가 제각각이고, 각 클래스에 일치하는 data 들을 포함하지 못하는 모습을 보인다. 이 내용은 뒤에서 다시 다룬다.
Train Accuracy 는 55.1%, Test dataset 에 대한 Accuracy 는 49.8%에 불과하고, iteration 의 최댓값을 100 으로 설정했음에도 불구하고 40 번의 iteration 만을 수행했다. (convergence threshold = 0.0001 보다 작은 변화를 보였기 때문)

Figure 3.a ~ d 는 spherical, diagonal, tied, full 4 개의 covariance 타입에 따른 출력, Table1 은 그에 따른 train/test accuracy 이다.

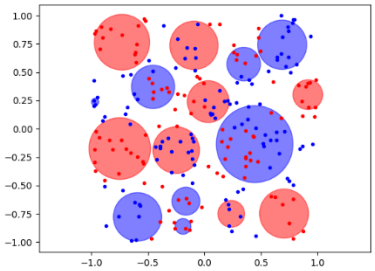


Figure3.a (Spherical)

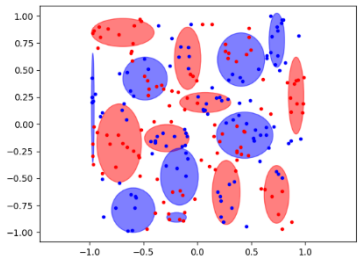


Figure3.b (Diagonal)

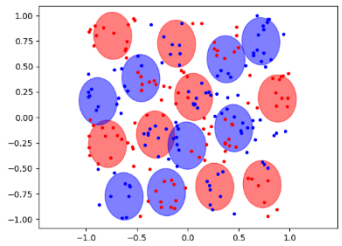


Figure3.c (tied)

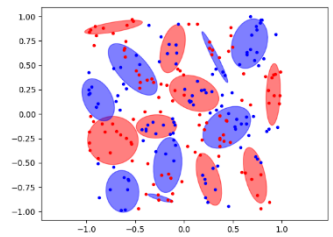


Figure3.d (full)

covariance_type	Train Accuracy	Test Accuracy
spherical	55.1%	49.8%
diagonal	50.2%	48.9%
tied	54.7%	49.8%
full	51.1%	48.9%

Table1

4 가지 경우 모두 test accuracy 가 50%를 넘기지 못했다. 별개로, 각 covariance type 에 따른 iteration 횟수는 spherical 40 회, diagonal 30 회, tied 20 회, full 20 회에 그쳤다. 더 많은 iteration 을 거치며 covariance 를 수정해도 유의미한 개선점을 얻을 수 없다는 의미로 해석할 수 있다.

결과로 나온 좌표평면을 보면, 16 개의 Component 들이 의도한 바(4x4)와는 다르게 제각각 아무데나 위치하고 있으며, 파란 점이 분포한 지역에 빨간 covariance 가 보이기도 한다.
이에 대한 원인을 생각해봤는데, GMM 은 unsupervised learning technique 이다. model 을 fit 시키는 과정에서 각 data 들에 대한 label 을 피드백으로서 제공해주지 않기 때문에, 우리가 원하는 label 기준으로 covariance 가 이동하지 않는다. (우리에게는 빨간 점/파란 점으로 구분되지만 GMM 입장에서는 모두 같은 검정색 점으로 보일 것이다.)
또한, 우리가 원하는 대로 covariance 가 4x4 의 형태로 잘 위치하고 있다고 해도, 우리가 임의로 설정했던 16 가지의 클래스와 일치할 것이라는 보장이 없다. 우리가 정한 기준은 왼쪽 위부터 [0, 1, 2, …, 15] 지만, GMM predict 가 내놓는 클래스의 기준은 [3, 5, 12, 8, …, 2] 와 같이 뒤죽박죽일 수 있다.

내가 GMM 을 제대로 이해했는가에 의심이 많이 생기지만, 서로 다른 label 을 가진 data 들이 골고루 퍼져있는 경우 GMM 으로 이를 classify 하기 적절하지 않다는 결론에 도달했다.

이후 내용은 p2 dataset 에 대해 component 개수를 2 로 고정하고(개수를 늘리면 예측 결과의 가짓수도 늘어나기 때문에), 그 외에 다른 parameter 값들을 바꿔서 실행했을 때 나오는 결과들의 차이에 집중했다.

Figure 4.a~d 는 p2 dataset 에 대해 4 가지 covariance type 을 주었을 때의 결과이고, Table2 는 각 결과의 accuracy 이다.

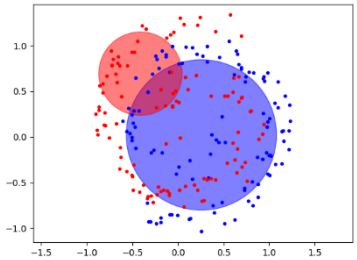


Figure4.a (Spherical)

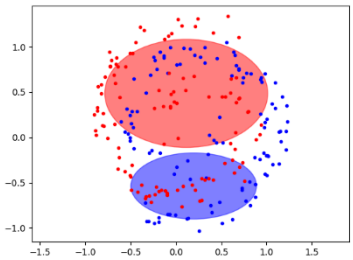


Figure4.b (Diagonal)

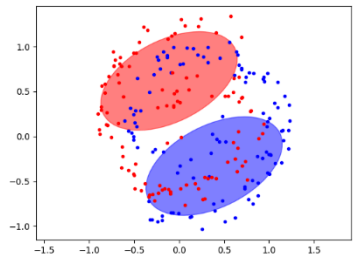


Figure4.c (Tied)

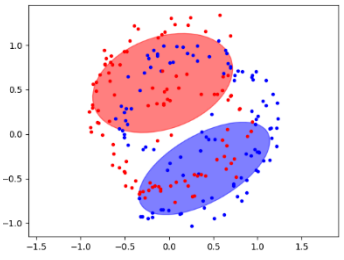


Figure4.d (Full)

covariance_type	Train Accuracy	Test Accuracy
spherical	54.2%	50.2%
diagonal	50.2%	56.4%
tied	55.1%	61.3%
full	56.0%	62.2%

Table2

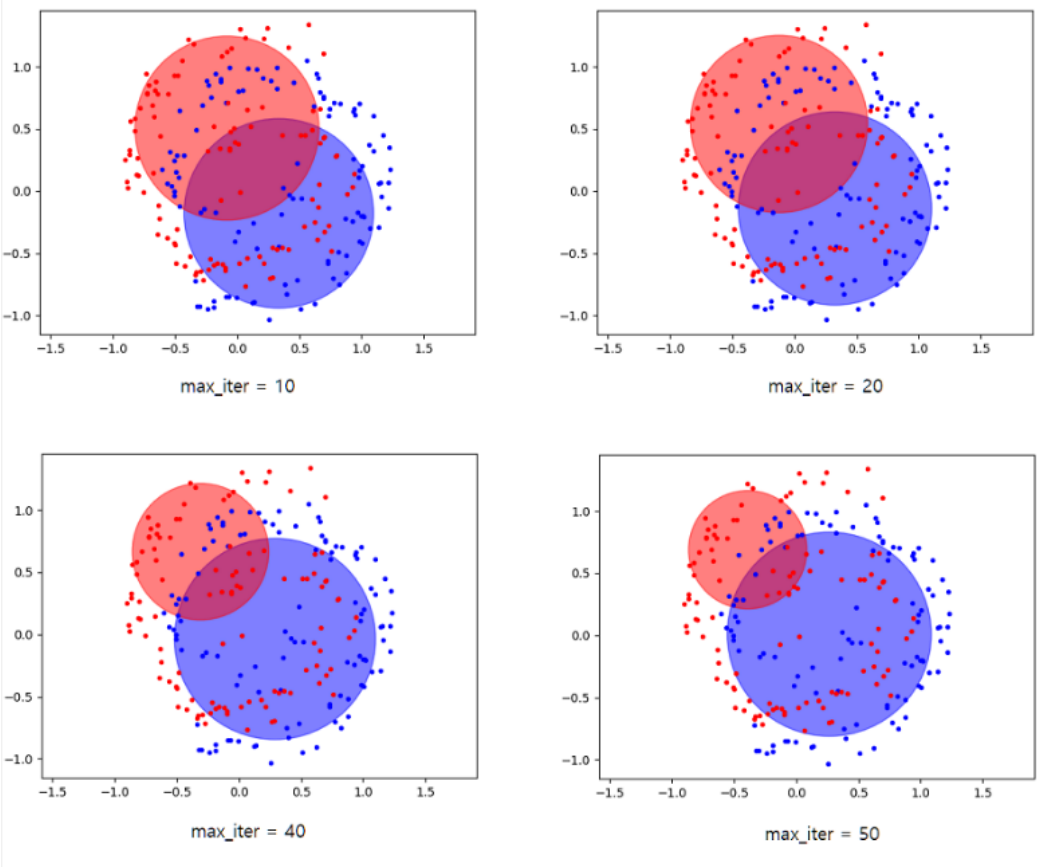
이번에는 covariance type 에 따른 결과 차이가 약간 보인다.
p2 dataset 은 나선 모양을 띄고 있는 것을 볼 수 있는데, tied 와 full type 의 covariance 처럼 각도가 기울어진 타원이 왼쪽 위의 빨간 점들이 모인 지점과, 오른쪽 아래의 파란 점들이 모인 지점을 찾아가 조금 더 나은 Accuracy 를 보일 수 있다고 생각했다.

tol 값을 default 값의 10 분의 1 인 0.0001 로 설정했음에도 불구하고, default max_iter(100)까지 가는 경우가 한 번도 없었다.spherical 과 diagonal 의 경우 50 번, tied 는 10 번 이하, full 은 10 번 실행됐다.
따라서, 그나마 수행횟수가 높은 diagonal 을 선택해 max_iter 를 바꿔가며 실행해봤다.

각각 max _iter 를 10, 20, 40, 50 을 주고 실행한 결과이다. 빨간 원이 점점 작아지는 것을 볼 수 있다.
각 경우에 대해 accuracy 를 출력해봤는데, 아래와 같은 결과가 나왔다.

max_iter=10 -〉 train 55.1%, test 61.3%
max_iter=20 -〉 train 56.9%, test 60.0%
max_iter=40 -〉 train 55.1%, test 55.6%
max_iter=50 -〉 train 55.1%, test 50.2%

위 수치에서 볼 수 있다시피, test accuracy 가 iteration 이 증가함에 따라 오히려 감소하는 모습을 보인다.
overfitting 이 일어났다고 하기에는 train accuracy 가 그대로인데, 각 epoch 마다 피드백이 제공되지 않는 unsupervised learning 이기 때문이라고 추측된다.
(다음 장에 위 내용에 대한 자료 있습니다.)



References

1. https://towardsdatascience.com/gaussian-mixture-models-with-python-36dabed6212a
2. https://scikit-learn.org/stable/modules/generated/sklearn.mixture.GaussianMixture.html#sklearn.mixture.GaussianMixture.fit
3. https://scikit-learn.org/stable/auto_examples/mixture/plot_gmm_covariances.html
4. https://jakevdp.github.io/PythonDataScienceHandbook/05.12-gaussian-mixtures.html

Environment

python 3.9.0
scikit-learn 1.0.2
matplotlib 3.5.1
numpy 1.21.4