패턴인식 HW4

2018573009
김영준

1. 개요

이번 과제에서는 수업시간에 배운 다양한 Feature Generation 알고리즘을 구현하고 이를 이차원 데이터셋을 이용하여 테스트해 보았다.

2. 개발환경

이번 과제도 마찬가지로 Python3를 기반으로 구현하였다. 사용한 주요 모듈은 matplotlib 3.0.3, Numpy 1.6.2, scikit-learn 0.20.3, scipy 1.2.1이 있다.

3. 소스코드 설명

이번 과제는 hw4.py 하나의 소스코드에 모든 내용을 구현하였다. 지난 hw3에서 채용하였던 커맨드 명령 옵션을 그대로 채택하였다. 자세한 커맨드 명령 옵션은 *python hw4.py -h* 명령어를 입력하면 알 수 있다.

```
(venv) (base) Youngjoon:HW4 kimyoungjoon$ python hw4.py -h
usage: hw4 [-h] [-n] [-s] [-l] algo

Homework4 of Pattern Recognition Spring 2019

positional arguments:
  algo            Feature Generation algorithm{'Kernel PCA' : 0, 'SVD' : 1,
                  'Kernel FDA' : 2)

optional arguments:
  -h, --help      show this help message and exit
  -n, --nofigure  do not show figure
  -s, --save      save figure to png file
  -l, --load      load exist dataset
```

Hw4.py 소스코드 역시 앞선 과제의 구조와 매우 유사하게 구성하였다. 크게 데이터를 위한 클래스 정의, 데이터 생성, feature generation 알고리즘 실행, 결과 출력으로 나누어진다. 각각의 과정은 가독성과 재사용성을 위해 함수로 분리하였으며 각 함수의 기능은 다음과 같다.

a. **generate_data()**
Classification 모델 실습을 위한 데이터를 생성한다. -l 옵션을 사용할 경우 기존에 미리 생성되어있는 데이터를 로드해온다.
이번 과제의 경우 Kernel FDA의 데이터셋은 클래스가 3종류이기 때문에 SVD, Kernel PCA 데이터셋과 구분하여 생성하였다. 과제 설명에 제시된 대로 sklearn.datasets.make_classification 함수를 사용하였다.

b. **kernelFDA(data, labels, n_class, gamma=10)**

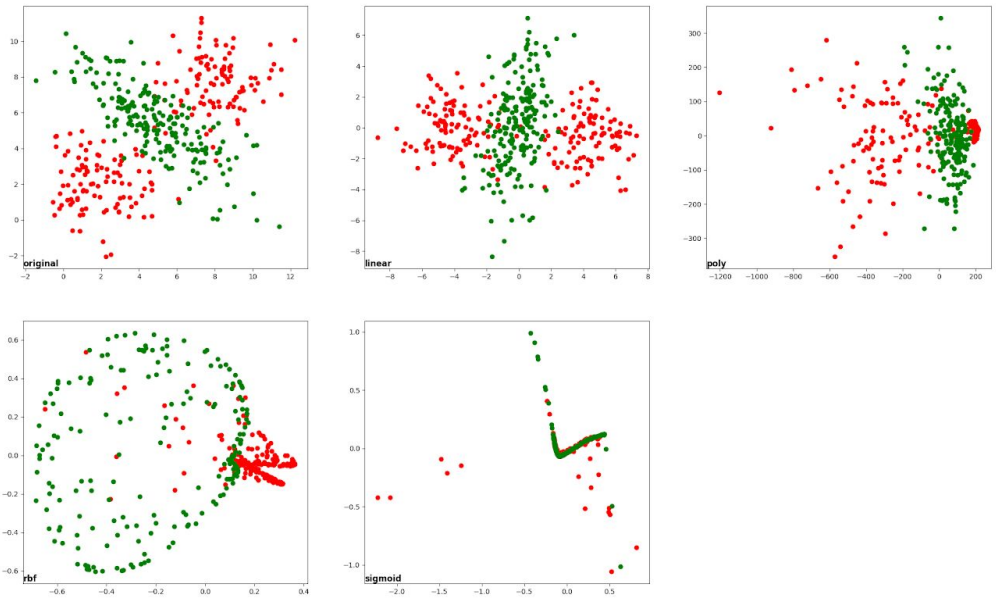
Kernel FDA를 구현한 함수이다. Kernel FDA는 sklearn에 구현체가 없어 직접 구현하였다. 커널은 RBF 커널을 사용하였으며 위키피디아 Kernel Fisher discriminant analysis 페이지의 Multi-class KFD 수식을 그대로 구현하였다.
(참고 : https://en.wikipedia.org/wiki/Kernel_Fisher_discriminant_analysis)
2차원 데이터셋을 data, 각 데이터의 라벨을 labels, 클래스의 개수를 n_class(본 과제에서는 3으로 고정), RBF 커널의 감마값을 gamma로 전달받아 생성한 Feature를 결과로 반환한다.

c. **plot_class(data, size, pos, title=None)**
생성된 데이터셋과 Feature Generation 결과를 그래프로 출력한다. 각 클래스는 다른 색상으로 구분하고 알고리즘에 따라 가독성이 뛰어나도록 서브그래프를 구성하였다.

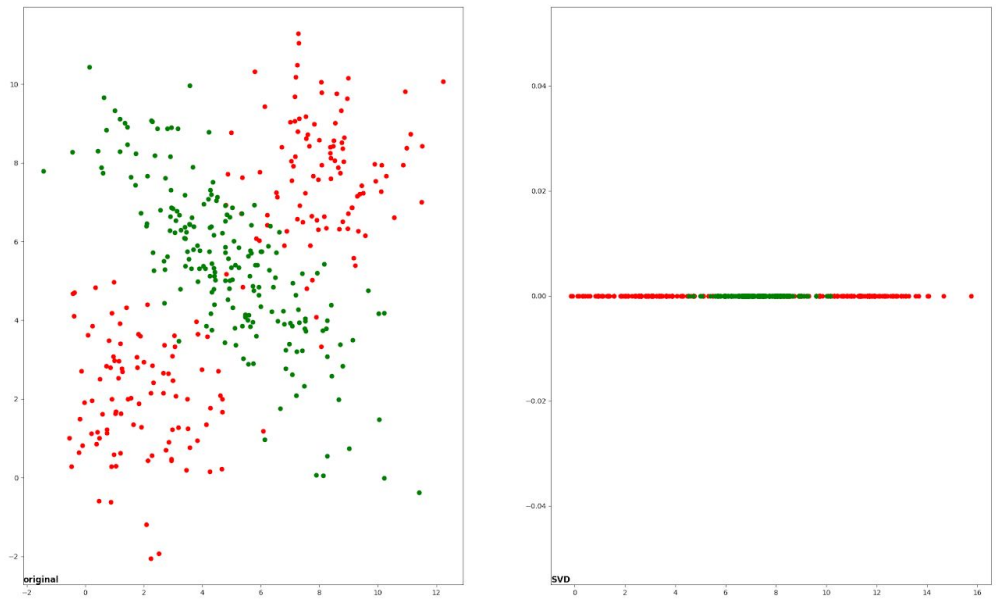
d. **main()**
사용자가 선택한 알고리즘 별로 적절한 함수를 호출하여 전체 과정을 진행한다.

4. 결과

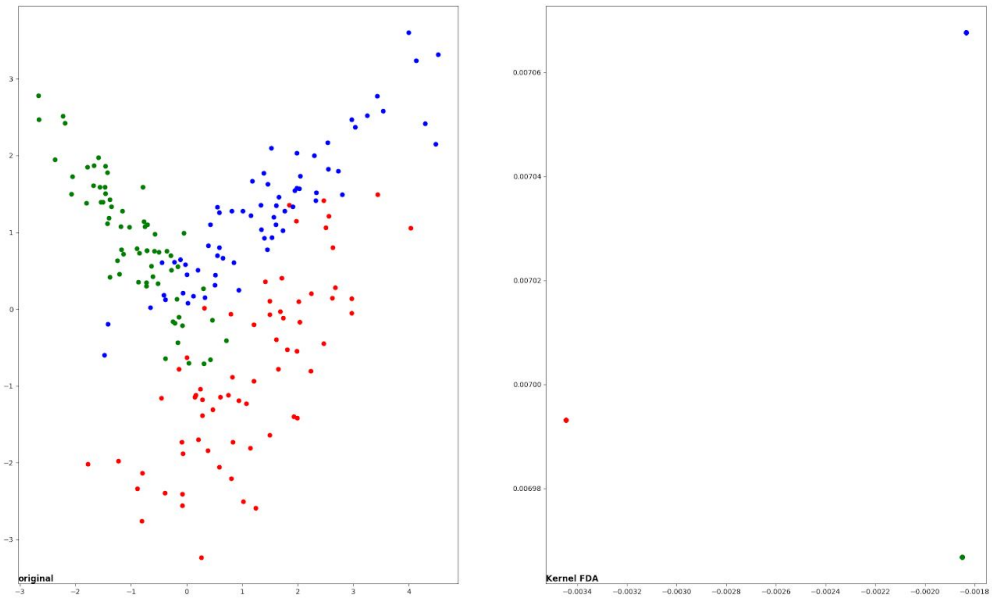
a. Kernel PCA



b. SVD



c. Kernel FDA



5. 결론

이번 과제에서는 세가지 종류의 Feature Generation 알고리즘을 구현해보았다. 학습과정이 크게 복잡하지 않고 조금의 행렬연산만 진행하면 되기 때문에 대체적으로 알고리즘이 기존 과제 알고리즘보다 매우 빠름을 알 수 있었다. 성능 또한 Kernel FDA는 잘못 구현한 것 아닌가 느낄정도로 클래스 구별이 너무 잘될정도로 훌륭하였다. 다만 Kernel PCA의 경우는 커널의 종류에 따라 분포양상이 매우 상이하기 때문에 절절한 커널을 선택하는 것이 중요할 것으로 보인다.