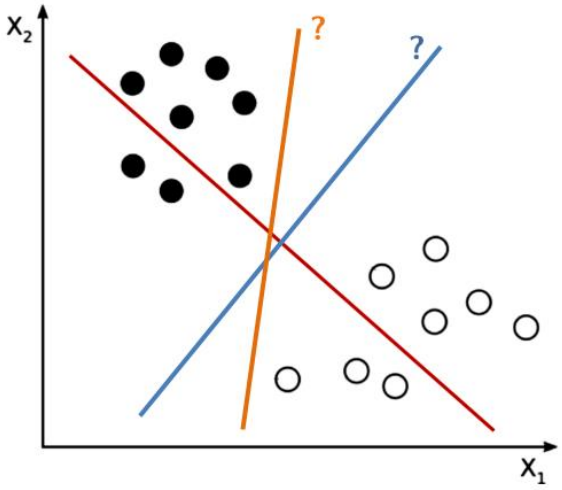
지능정보수학 기말고사대체과제 Part 2

휴먼지능정보공학과 201810825 홍종현

1. 접근

이 과제를 수행하려면 우선 Support Vector Machine(이하 SVM)에 대해 알아야 하므로 SVM이 무엇인지 설명하고자 한다.



위 그림은 error based를 사용하여 classification 하였을 때 주어진 Data들을 표현한 모습이다. 특정한 선형방정식(분류선 : Decision Boundary)을 세워 그것보다 크면(+1) 검은색 원으로, 작으면(-1) 하얀색 원으로 분류 할 수 있는 구역이 있다. 이 때 그 Decision Boundary은 주황색 선이 될 수도 있고, 파란색 선이 될 수도 있다. 하지만 이는 새로운 Input Data를 적절하게 classification 하지 못한다. 미지의 Data에 대해서 더 적절하게 구분하는 방법은 Margin을 사용해서 알 수 있다.

Margin이란 선과 가장 가까운 양 옆 데이터와의 거리이다. 이때 Decision Boundary 과 가장 가까운 포인트를 Support Vector(이하 SV)라고 한다. 즉 Margin은 Decision Boundary와 SV간의 거리를 의미한다. 이때 Decision Boundary가 급격하게 변화된 Input Data가 들어와도 최대한 잘 classification을 하려면 Decision Boundary 양쪽으로 Margin이 최대가 되어야 한다. 이렇게 Margin이 최대가 되는 Decision Boundary를 만드는 것을 추구하는 것이 SVM이다.

만약 새로운 Input Data가 Decision Boundary를 그릴 수 없는 곳에 들어온다면 이러한 Outlier를 무시하고 Margin을 최대화 하는 Decision Boundary을 찾는다. 또한 X,Y축으로 이루어진 좌표에서 두 Data Set을 classification할 수 있는 Decision Boundary을 그릴 수 없다면 Kernel Trick을 이용해서 차원을 바꿔 Decision Boundary을 그릴 수 있게 하는 등의 방식도 있지만 과제풀이에는 큰 연관이 없으므로 간단하게 언급만 하고 지나가도록 하겠다.

과제 코드를 분석하면 이산 푸리에 변환에 대한 언급이 있다. 하지만 심도 있게 설명하기엔 충분한 지식이 갖춰져 있지 않기 때문에 간략하게 설명하자면 하나의 파형은 여러 sin함수의 조합으로 이루어져 있다. 이때 sin함수는 주파수에 따라 변화하므로 앞에서 언급한 하나의 파형의 주파수는 여러 개여 sin함수의 주파수의 합과 같다고 할 수 있다. 이산 푸리에 변환은 이러한 주파수들의 특징을 함수화 하여 나타내는 것을 말한다.

2. 구현(feature.py 코드 분석)

```
8   from scipy import signal
9   import matplotlib.pyplot as plt
10
11  # read original data file: ecg.txt
12  # then visualise the data
13
14  train = []
15  fp = open('ecg.txt', 'r')
16
17  for line in fp:
18
19      line = line.strip('\n\r')
20      cols = line.split('\t')
21      val = float(cols[1])
22      train.append(val)
23
24  plt.plot(train)
25  plt.show()
26
27  wd = train[0:400]
28  f, pxx_den = signal.welch(wd, 40)
29
30  plt.plot(pxx_den)
31  plt.show()
```

```
32
33  # compute the spectral power of the signal
34  # and, save it into the file
35
36  window = 400
37  dimension = 10
38
39  fout = open('p.train', 'w')
40  for i in range(0, len(train) - window, window):
41
42      wd = train[i : i + window]
43      f, pxx_den = signal.welch(wd, window / 10.0)
44
45      tmp = {}
46      p = 0
47      psum = 0
48
49      for fd in f:
50          binpos = int(fd)
51          if not binpos in tmp:
52              tmp[binpos] = 0
53          tmp[binpos] += pxx_den[p]
54          psum += pxx_den[p]
55          p += 1
56
57      p = 0
58      for binpos in tmp:
59          pavg = tmp[binpos] / psum
60          if p != 0:
61              fout.write(',')
62          fout.write('%.3f' % pavg)
63          if p > dimension:
64              break
65          p += 1
66          if i == 0:
67              plt.bar(binpos, pavg)
68
69      fout.write('\n')
70  fout.close()
71  plt.show()
72
73  # draw plot
74  plt.semilogy(f, pxx_den)
75  plt.xlim([0, 20])
76  plt.xlabel('frequency')
77  plt.ylabel('PSD')
78  plt.show()
```

feature.py는 ecg.txt (원본 Data)에서 feature를 추출하고 SVM 학습에 사용할 수 있도록 Data를 변환하는 코드이다.

line17~22에서 주어진 Data (ecg.txt)가 DataSet에 들어갈 때 불필요한 요소들을 제거한다.

line28에서 7000개의 Data 중 0~399개의 Data만을 추출하여 이산 푸리에 변환을 시행하여 각각의 주파수가 가지고 있는 특성을 함수화 하여 표현한다.

line39~71에서는 이산 푸리에 변환을 한 Data들을(line28은 이산 푸리에 변환을 거친 후를 보여주기 위함 이고 실제 이산 푸리에 변환은 여기서 시행) SVM에 사용하기 위한 feature들로 구분하는 작업을 거친다.

2. 구현(svm_anomaly.py 코드 분석)

```
8   from sklearn.svm import OneClassSVM
9   import numpy as np
10  import matplotlib.pyplot as plt
11
12  def make_svmdata_from_file(fname):
13      dlist = []
14      fp = open(fname,"r")
15      for line in fp:
16          tmp = []
17          line = line.strip("\r\n")
18          psd = line.split(",")
19          for td in psd:
20              if len(tmp) >= 4: break ## valid dimension
21              tmp.append(float(td))
22          dlist.append(tmp)
23      fp.close()
24      return dlist
25
26  #1. read svm input ( train, test)
27  train = make_svmdata_from_file("p.train")
28  test = make_svmdata_from_file("p.train")
29
30  #2. svm test
31  X_train = np.array(train)
32  X_test = np.array(test)
33
34  anomalymodel = OneClassSVM(nu=0.12, kernel='rbf', gamma=0.1)
35  anomalymodel.fit(X_train)
36  p = anomalymodel.predict(X_test)
37
38  pos=0
39  for pd in p:
40      pos += 1
41      if pd == -1:
42          print ("anomaly detection", pos - 1)
43
44  #3. ploting
45  plt.subplot(2,1,1)
46  plt.plot(X_train)
47  plt.subplot(2,1,2)
48  plt.plot(X_test)
49  plt.show()
```

svm_anomaly.py는 SVM을 이용하여 학습하고 예측하는 코드이다. (위에서 구한 feature들을 이용하여 이상 파형을 보이는 심전도를 확인)

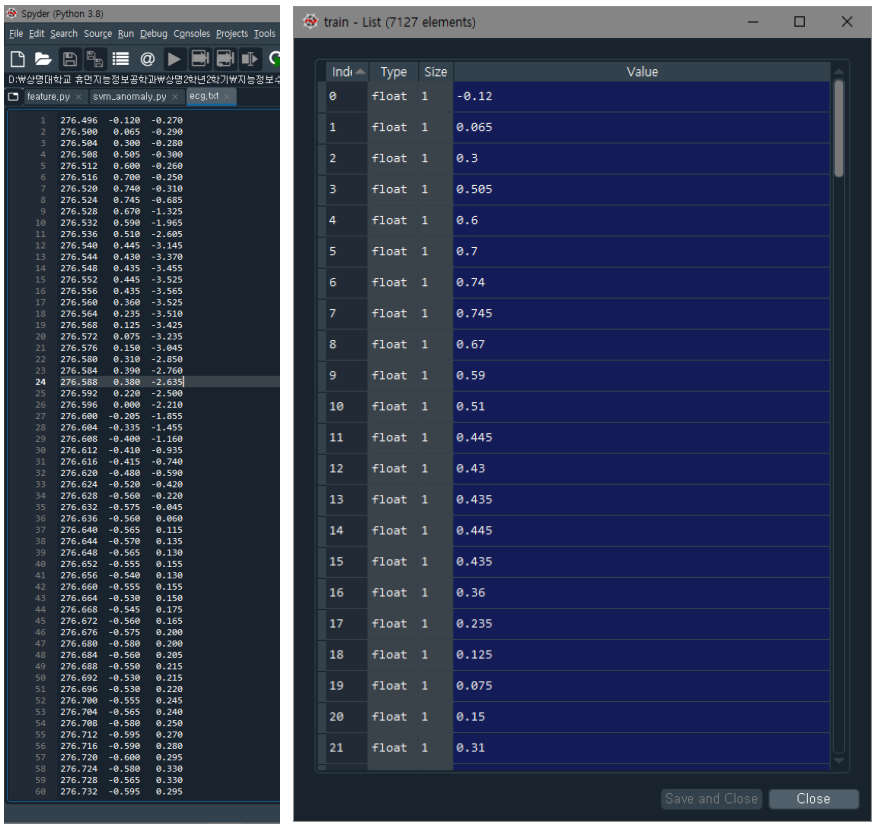
line12의 메소드에서 feature.py를 통해 분류된 feature들을 SVM에서 사용할 때 쓸모없는 값들을 제거하기 위한 과정을 거친다

line27~28에서 train set과 test set을 설정하는데 이때 train ,test set이 같아 overfitting의 우려가 있지만 이는 새로운 Input Data가 들어오는 것을 prediction하는 것이 아니라 주어진 Data중에서 outlier를 이루는 Data를 찾아내는 모델이기 때문에 train과 test의 값이 같아도 무방하다.

line34~36에서 train set으로 모델을 학습시킨 후 test set으로 prediction한다. (이때 predict 함수는 outlier를 -1로 반환하는 특성을 가짐)

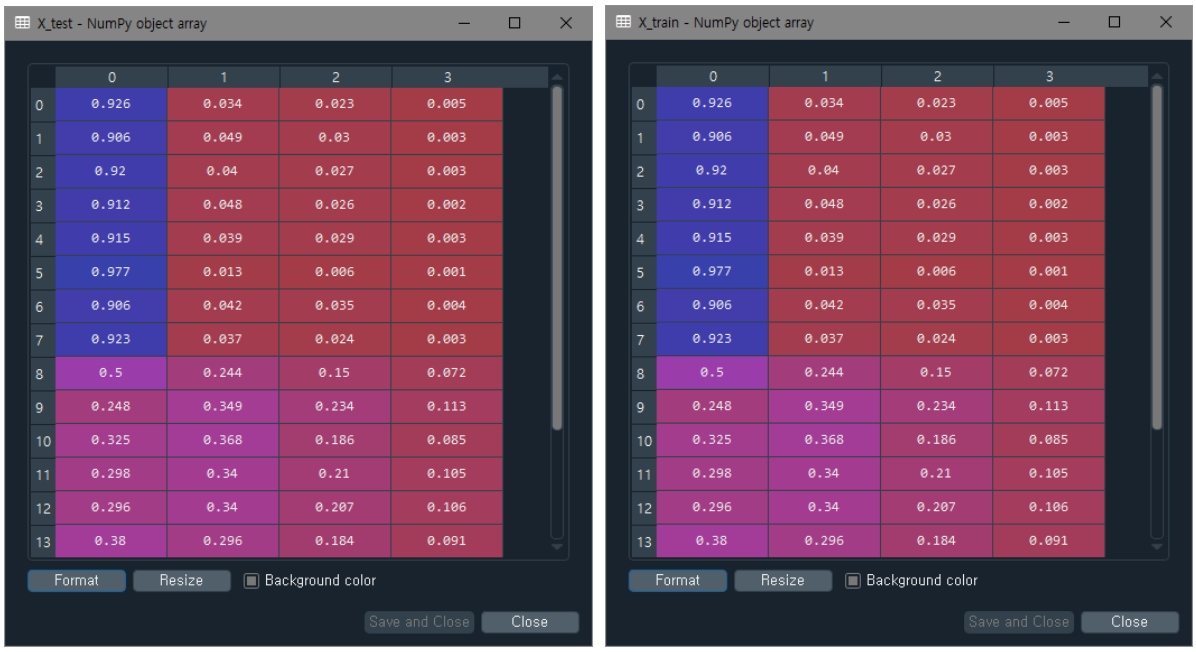
line 38~42에서는 모델에서 classification한 outlier를 -1을 return하게 하였기 때문에 outlier로 classification한 값들에 대해 출력하게 한다.

3. 실험(feature.py Data)



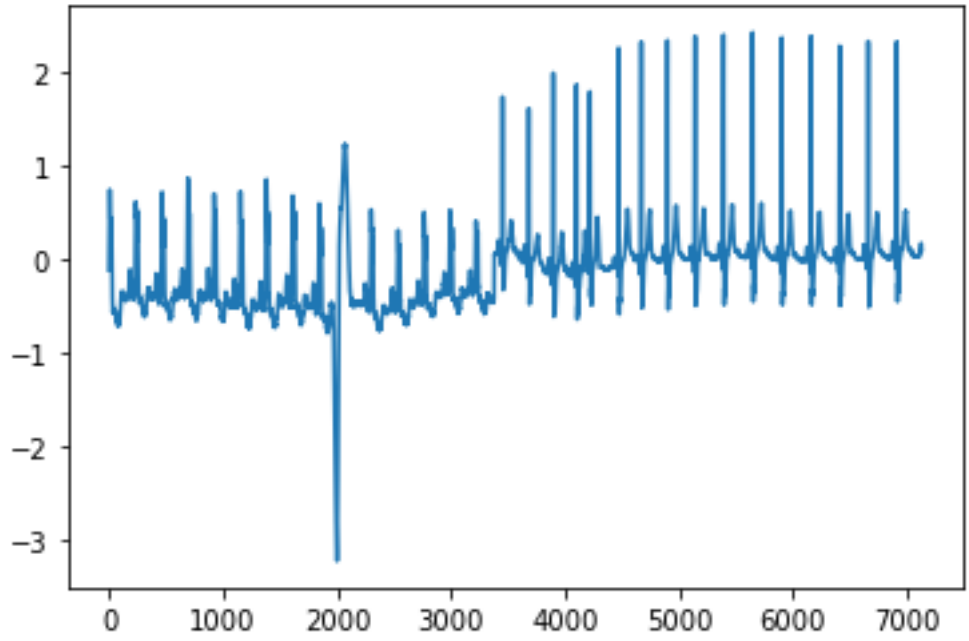
txt파일로 이루어진 7000개의 Data가 train에 입력 되어있음을 알 수 있다.

3. 실험(svm_anomaly.py Data)

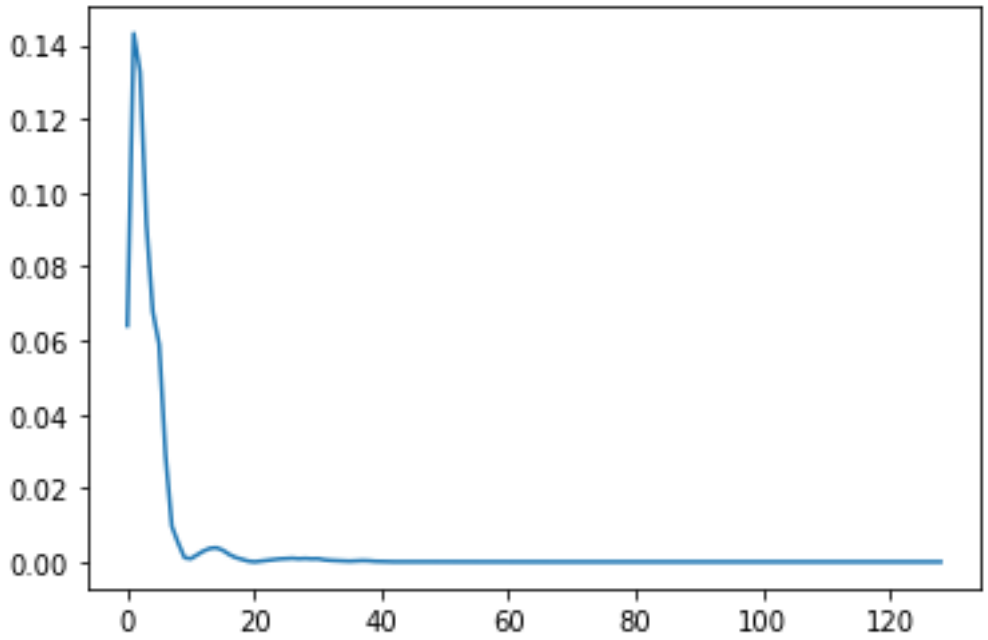


feature.py에서 분류한 feature들이 X_test와 X_train에 제대로 입력되어 있다.

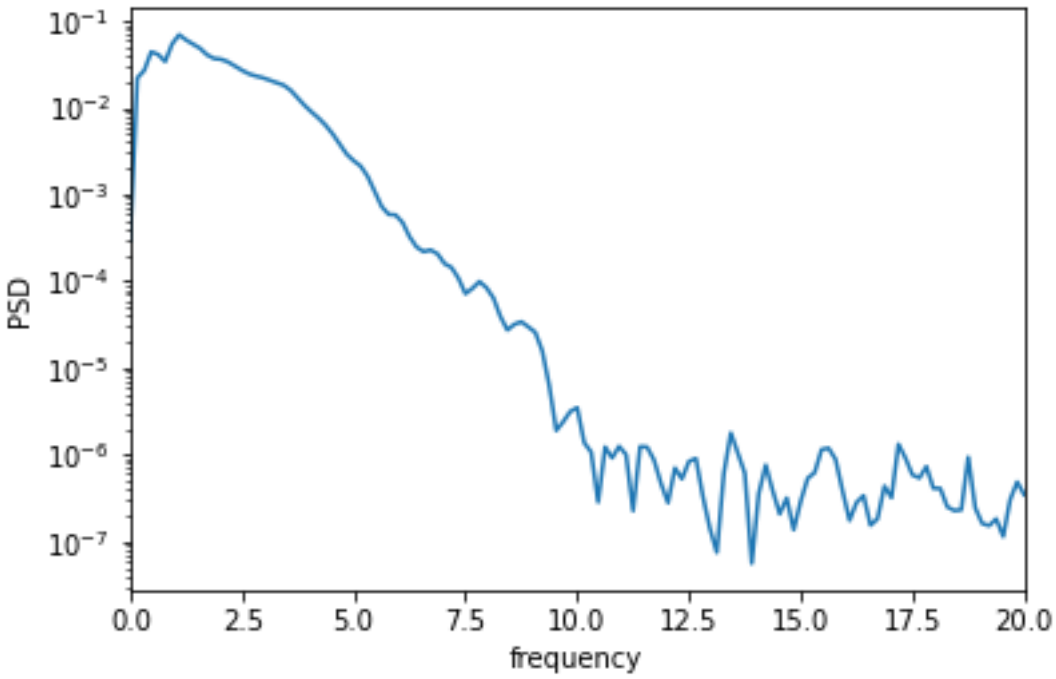
4. 결과(feature.py)



7000개의 Data들을 나타낸 그래프다. 육안으로 확인하였을 때는 2000번째쯤의 Data가 outlier를 나타냄을 확인할 수 있다.

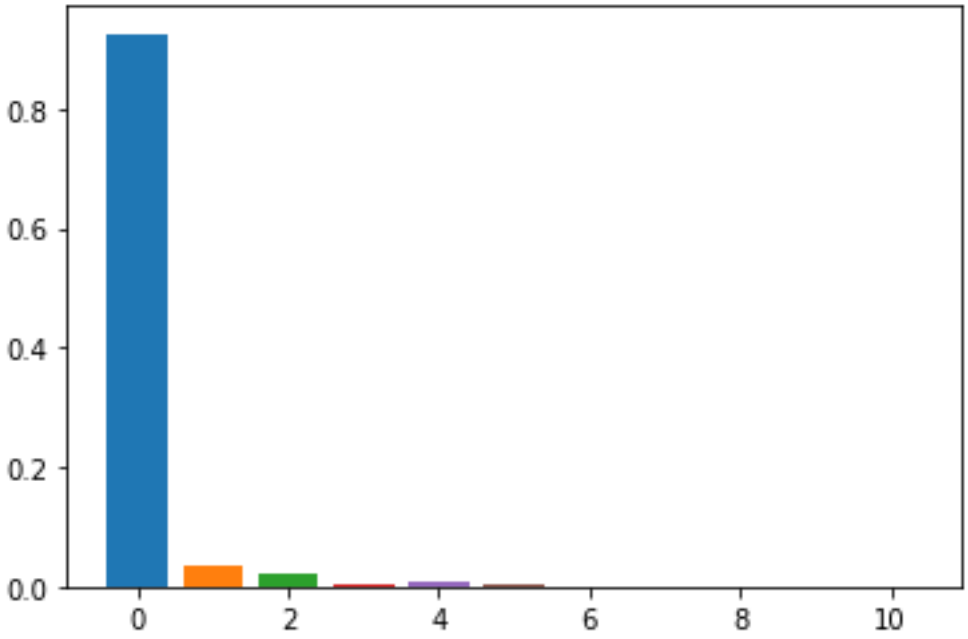


0~399까지의 400개 Data만 시범적으로 뽑아 이산 푸리에 변환을 거친 후 나타낸 그래프이다. 처음 7000개의 Data들을 보았을 때 400개의 Data들을 하나로 묶어서 보는 것이 그래프가 가진 특성을 잘 파악할 수 있다고 판단하여 묶었다. 앞서 설명했듯이 하나의 파형의 주파수는 여러 개의 sin함수의 주파수의 합으로 볼 수 있기 때문에 이산 푸리에 변환을 거친 후 그래프의 주파수 특성을 잘 파악할 수 있다. 따라서 7000개의 심장박동 Data를 여러 sin함수의 주파수의 합으로 나타냄을 알 수 있고 이 특성들이 추후 SVM의 Input 값(train)으로 사용된다.



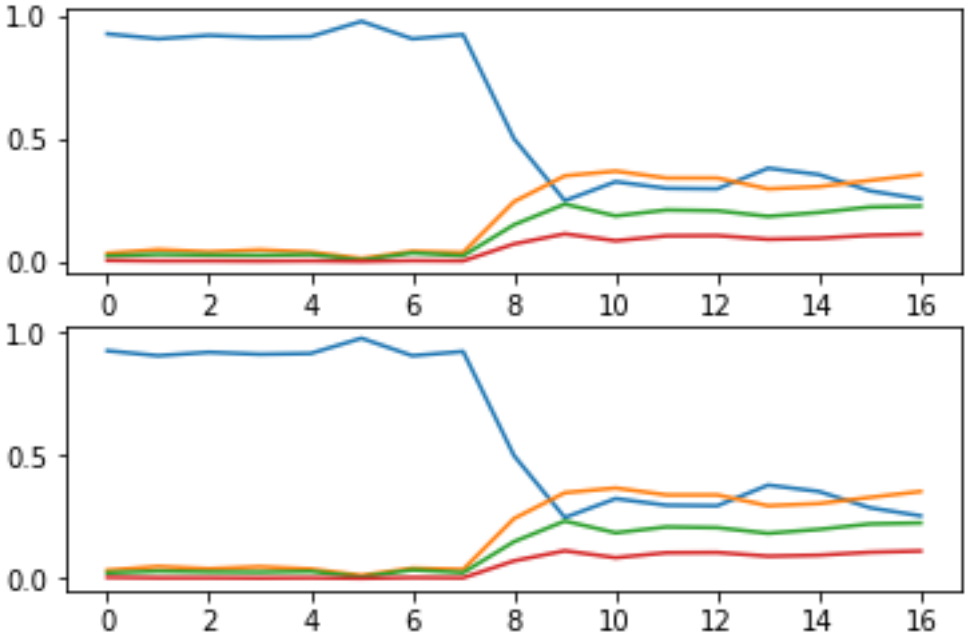
***실제 출력상으로는 이 그래프는 마지막에 출력되고 다음에 소개될 그래프가 3번째로 출력되지만
순서를 바꿔 설명하는 것이 더 용이하다고 판단되어 출력순서를 바꿈***

앞서서는 400개의 data만 뽑아서 시범적으로 이산 푸리에 변환을 거치면 어떤 식으로 출력되는지 보았다면 이 그래프는 line39에서 만든 list를 바탕으로 feature들을 뽑아 쓸 데이터를 구축하여 그 Data들에 대해 이산 푸리에 변환을 거친 후를 나타낸 그래프다. 하지만 이산 푸리에 변환을 하더라도 연속함수로 도출되어 SVM의 Input Data로 쓰기에는 적합하지 않다. 때문에 이를 디지털화 시킬 필요성이 있어 나온 그림이 다음 그래프이다.



연속함수로 도출된 결과를 디지털화 시키기 위해서 feature.py의 line40에서 [0:1), [1:2), [2:3) … 등으로 구별하여 재 표현한다. 이때 연속함수로 나온 이산 푸리에 변환 결과를 누적 덧셈하여 그 수치를 표현한다.

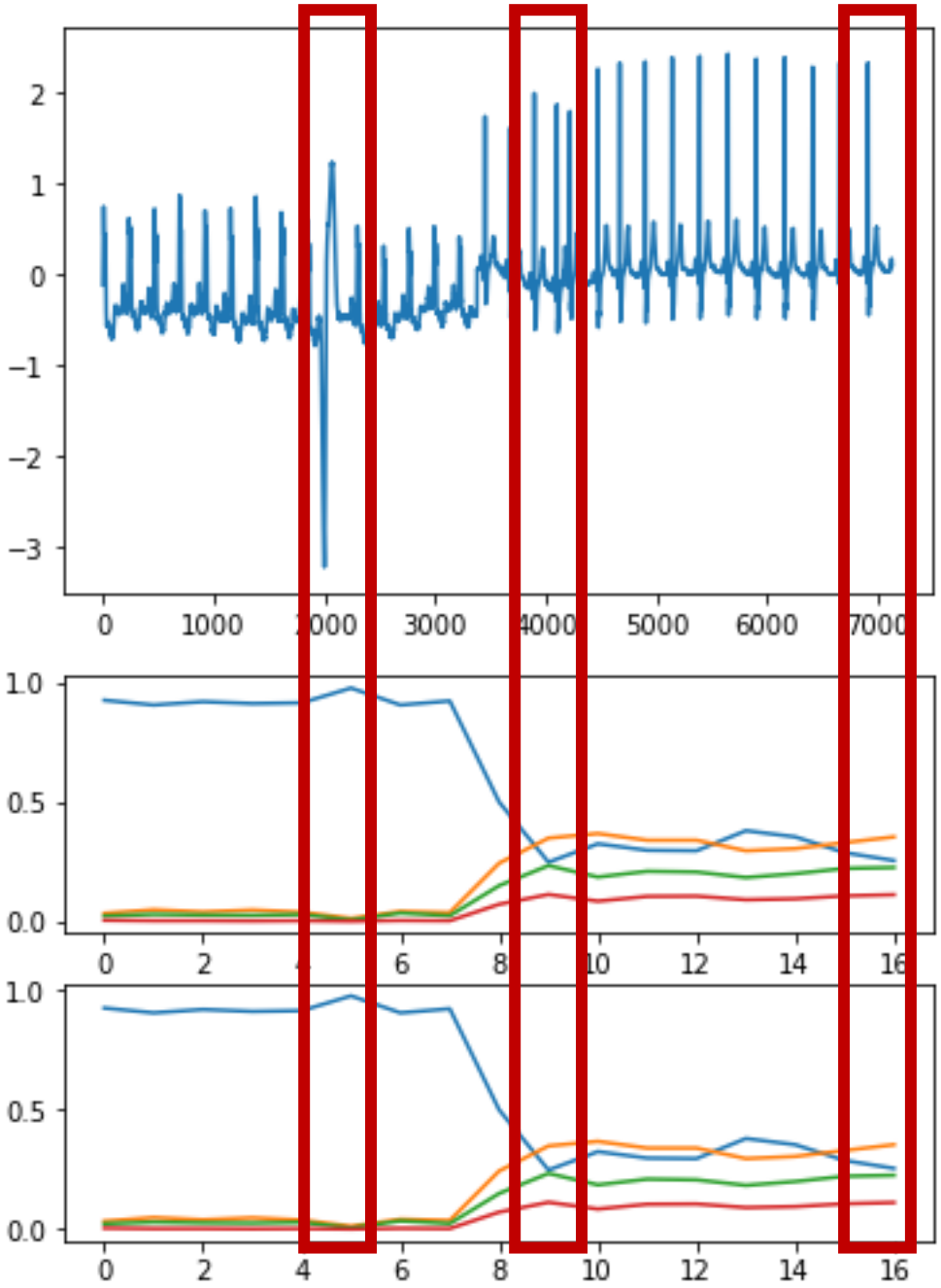
4. 결과(svm_anomaly.py)



SVM을 통해 주어진 7000개의 Data들이 가지고 있는 여러 sin함수의 주파수 특성을 표현한 그래프이다. 상단과 하단은 train, test 에 대한 결과인데 둘이 같은 이유는 이 모델은 새로운 Input Data에 대한 prediction을 구하는 모델이 아니라 outlier를 찾는 모델이기 때문이다. 이 그래프를 보면 주어진 7000개의 심장박동 Data에서 일반적인 심장박동을 나타날 때의 주파수와 차이가 있는 부분을 판단하여 outlier가 있다고 볼 수 있다.

코드를 실행하면 위 그래프에서 5, 9 ,16번째에서 outlier가 존재함을 알 수 있다. 이는 prediction한 값들을 저장한 p에서도 확인 할 수 있다. 이는 이산 푸리에 변환을 거친 심장박동 Data의 sin함수의 주파수의 합이 어떤 일반적인 특징을 가지게끔 그려져 있는데 이 일반적인 특징에서 벗어난 outlier가 발생하였기 때문이다. Outlier는 SVM을 통해 학습하여 적절한 Decision Boundary가 설정되었기 때문에 알 수 있다.

4. 결과(비교분석)



심장박동 Data에서 육안으로 확인할 수 있는 2000번대 Data외에도 3600(9 x 400), 6400(16 x 400)번 에서도 outlier가 있음을 알 수 있다. 이는 심장박동 Data의 일반적인 특성을 이산 푸리에 변환을 통해 classification하였을 때 기준으로 일반적이지 않은 sin함수의 주파수의 합이 나타난다면 이때 이 값을 outlier로 판단하기 때문이다.

5. 결론 및 고찰

4. 결과들을 보면 7000개의 복잡한 심장박동 Data를 이산 푸리에 변환을 거친 후 SVM을 통해 분류하고 outlier를 찾아 이상징후 여부와 그 위치를 판단할 수 있었다. 이산 푸리에 변환을 거치는 이유는 7000개의 심장박동 Data들중 일반적인 심장박동 형태를 찾아 일반적이지 않은 outlier를 찾을 수 있는데 이때 일반적인 형태. 즉 파형을 찾기 위해서는 파형이 여러 sin함수들의 조합으로 이루어져 있기 때문에 이때의 여러 sin 함수들의 주파수의 합을 구하기 위해서 이산 푸리에 변환을 거친다. 그 후 연속함수로 도출된 변환 결과를 0,1,2 … 등으로 디지털화 시켜 SVM의 train Data로 쓸 수 있게 바꾼 후 SVM을 거쳐 나온 결과를 통해 outlier를 찾는다.

이때 여러가지 색들의 sin함수의 합을 볼 수 있는데 일련의 색들 간의 관계는 SVM이 학습하여 일반적인 sin함수의 주파수의 합을 구한 상태에서 Decision Boundary을 설정하여 일반적이지 않은 결과를 찾아내는 방식으로 설명할 수 있다.

이번 과제를 진행하면서 이산 푸리에 변환이 정확히 무엇을 의미하는지 알게 되었다. 연속적인 심장박동 Data에서 어떤 방식으로 일반화를 할 것인지 궁금하였는데 파형의 특성을 활용하여 sin함수의 주파수의 합으로 나타내는 것으로 주파수의 특성을 잡아 SVM의 train Data로 활용하는 것을 보고 일련의 코드들과 실행 결과들이 무엇을 의미하는지 알 수 있었다. 또한 SVM을 통해 확인한 결과로는 육안으로 확인가능한 부분 외에도 유심히 봐야 알 수 있는 부분들을 알 수 있다. 이러한 특성들을 이용하면 육안으로는 구분이 불가능한 여러 연속, 비연속적인 Data들에 대해서도 충분한 train Data만 있다면 classification하여 유의미한 결과를 낼 것으로 보인다.