패턴인식 HW3

2018573009
김영준

1. 개요

이번 과제에서는 수업시간에 배운 다양한 Non-linear Classification 알고리즘을 구현하고 이를 이차원 데이터셋과 MNIST 데이터셋으로 테스트해보았다.

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

소스코드는 총 3개로 구성되어있다. 먼저 일반 이차원 데이터셋 테스트를 위한 hw3.py, 그리고 MNIST 데이터셋을 이용한 hw3_mnist.py, 마짐가으로 RBF Network를 구현한 rbfnet.py이다.
이번 과제에서는 테스트 효율성과 그래프 가독성을 위해 한번에 모든 테스트를 진행하지 않고 실행 옵션으로써 알고리즘과 옵션을 입력하도록 설계하였다. 자세한 사용법을 알고싶다면 *python hw3.py(hw3_mnist.py) -h* 명령어를 입력하면 된다.
Hw3.py, hw3_mnist.py는 저번 과제 소스코드와 유사하게 크게 데이터를 위한 클래스 정의, 데이터 생성, 모델 학습 및 사용, 결과 출력으로 나누어진다. 각각의 과정은 가독성과 재사용성을 위해 함수로 분리하였으며 각 함수의 기능은 다음과 같다.

a. **generate_classification_data()**
   Classification 모델 실습을 위한 데이터를 생성한다. -l 옵션을 사용할 경우 기존에 미리 생성되어있는 데이터를 로드해온다.
   Mnist의 경우 인터넷에 업로드되어있는 mnist 데이터셋을 다운로드 받는다.
b. **fit(algorithm)**
   Classification 알고리즘 이름을 입력으로 받아 그에 맞는 모델을 생성하고 이를 학습용 데이터로 학습시킨후 학습 결과와 모델을 리턴해준다.
c. **classification(model)**
   학습시킨 Classification 모델을 입력으로 받아 테스트 데이터를 prediction하고 그 결과를 리턴한다.
d. **plot_class(data, pos, title=None, model=None, boundary=False)**

Classification 결과를 그래프로 출력한다. 각 클래스를 다른 색상으로 구분하고 옵션 값에 따라 decision boundary를 출력해준다. SVM 모델의 경우 support vector들을 출력해준다.
(Mnist 데이터셋의 경우 그래프를 출력하지 않고 confusion matrix를 출력한다.)

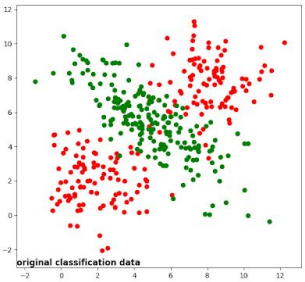
e. main()

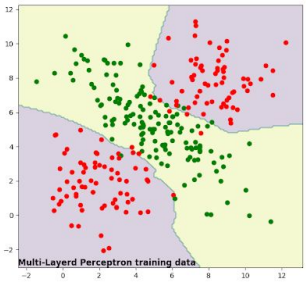
앞서 언급한 함수들을 적절히 호출하면서 전체적인 과정을 진행한다.

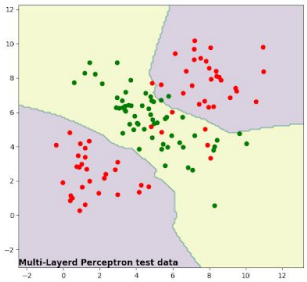
4. 결과

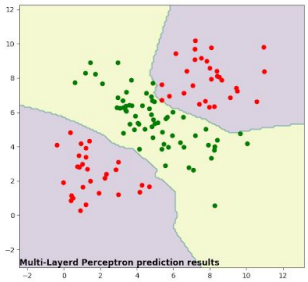
a. 2차원 데이터셋

i. MLP



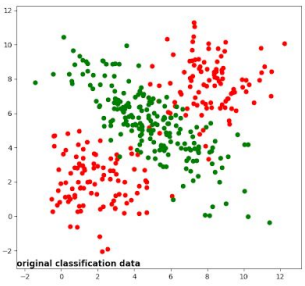


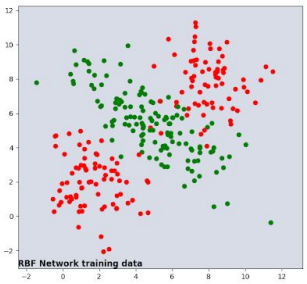


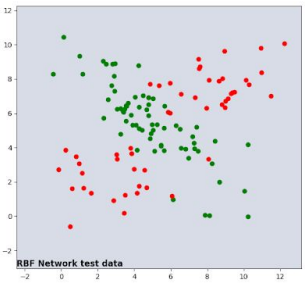


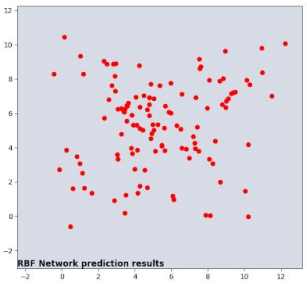
ii. RBF Network

- Cluster = 1

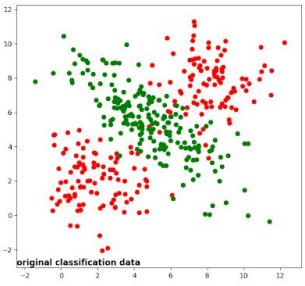


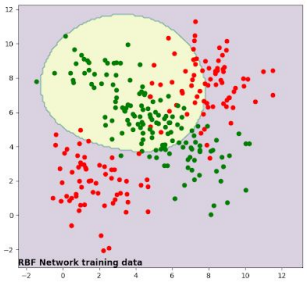


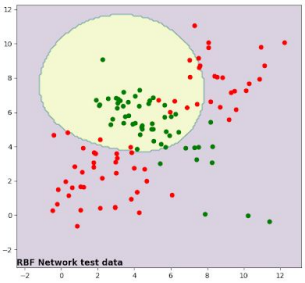


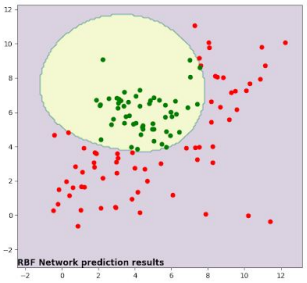


- Cluster = 3

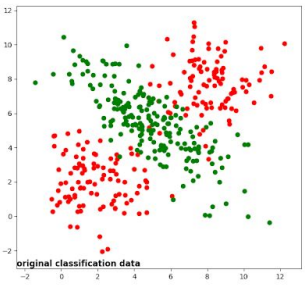


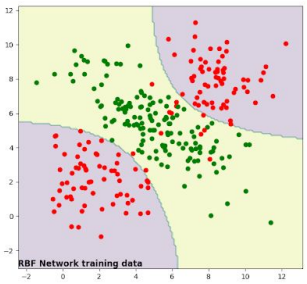


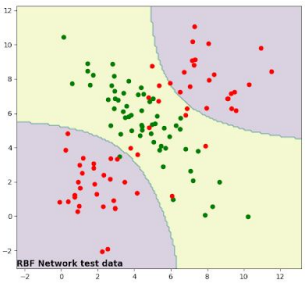


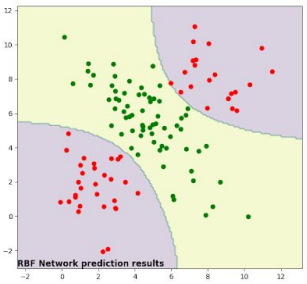


- Cluster = 5

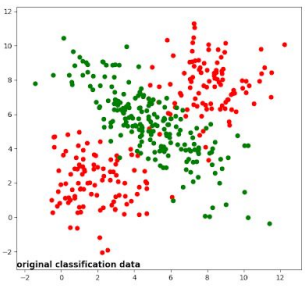


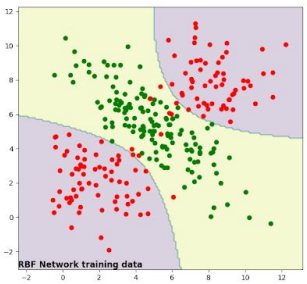


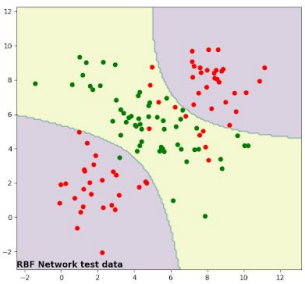


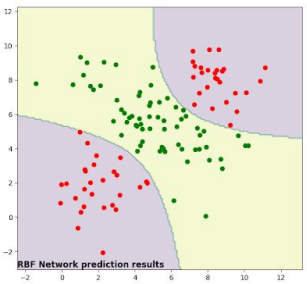


- Cluster = 7



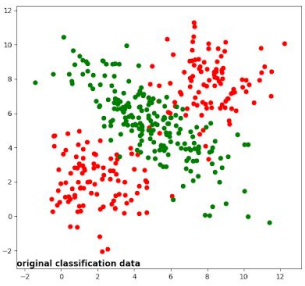


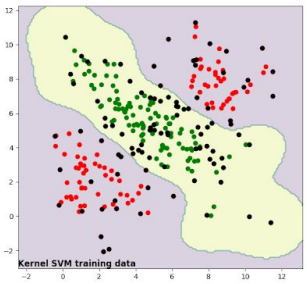


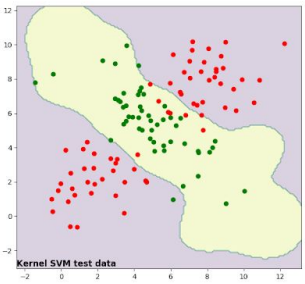


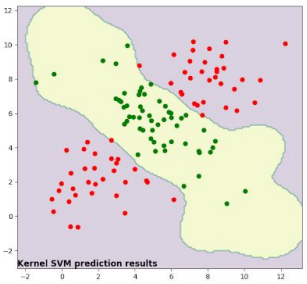
iii. SVM

- RBF Kernel

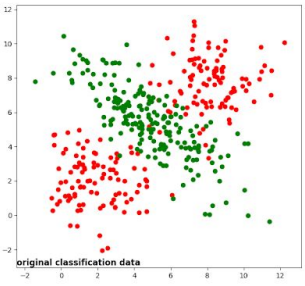


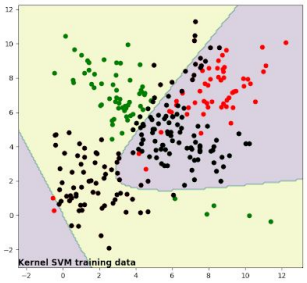


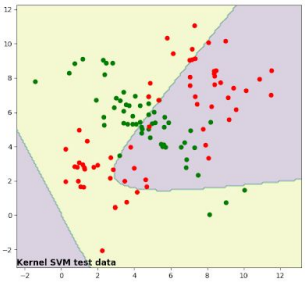


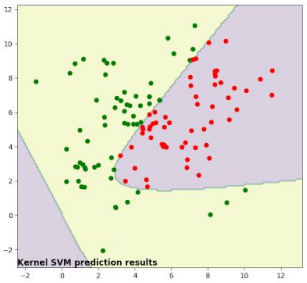


- Sigmoid kernel

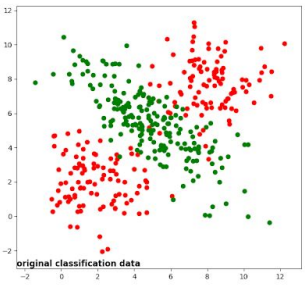


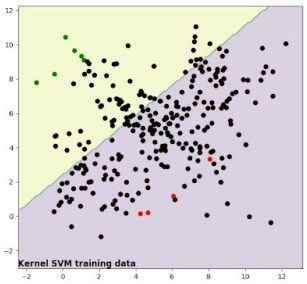


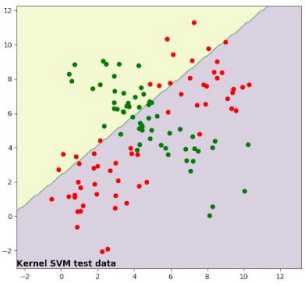


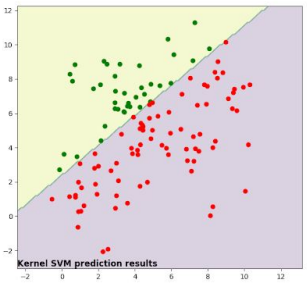


- Poly kernel

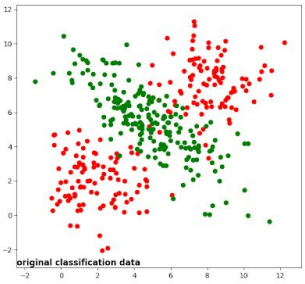


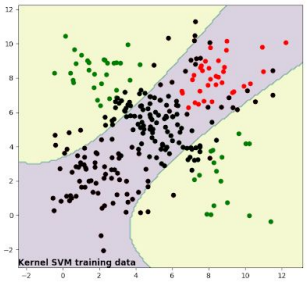


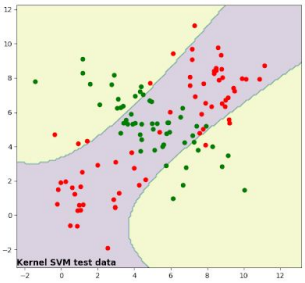


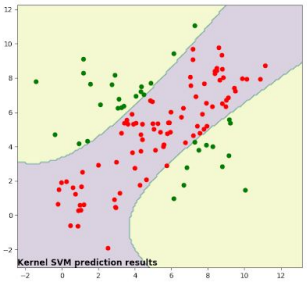


- Linear kernel



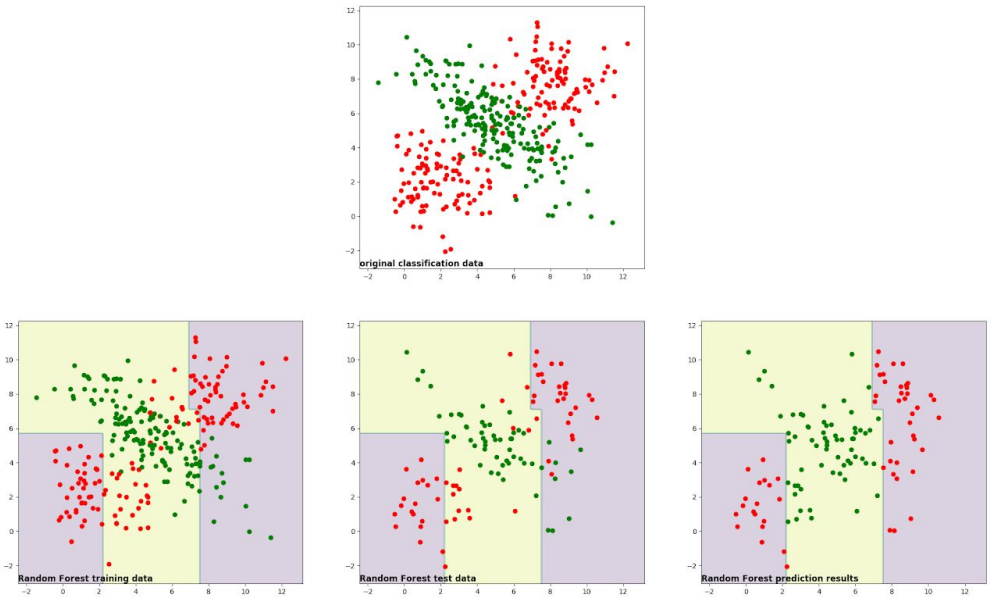




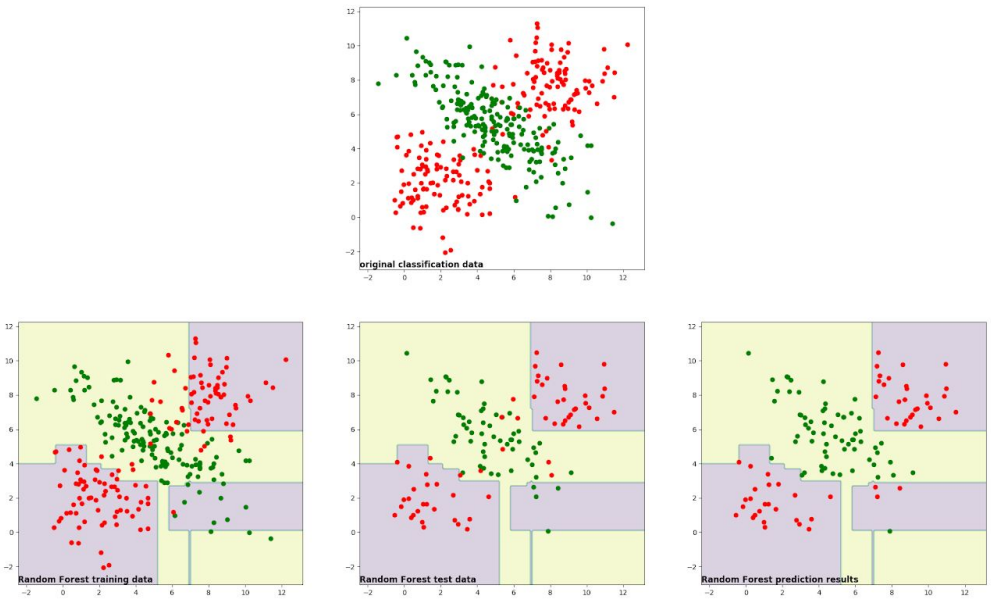


iv. Decision Tree

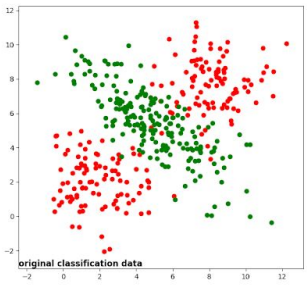
- 5 trees

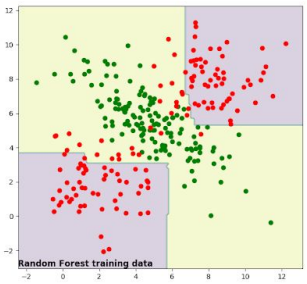


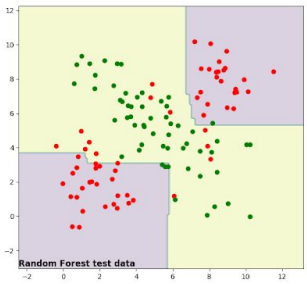
- 10 trees

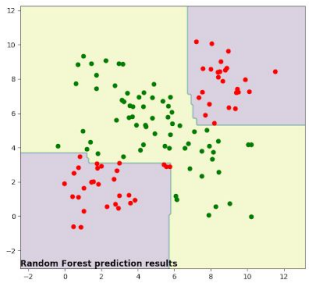


- 15 trees









b. MNIST

i. MLP

Classification report for classifier MLPClassifier(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9, beta_2=0.999, early_stopping=False, epsilon=1e-08, hidden_layer_sizes=(100,), learning_rate='constant', learning_rate_init=0.001, max_iter=200, momentum=0.9, n_iter_no_change=10, nesterovs_momentum=True, power_t=0.5, random_state=None, shuffle=True, solver='lbfgs', tol=0.0001, validation_fraction=0.1, verbose=False, warm_start=False):

	precision	recall	f1-score	support
0.0	0.97	0.96	0.96	2082
1.0	0.98	0.97	0.98	2359
2.0	0.93	0.94	0.93	2099
3.0	0.94	0.92	0.93	2198
4.0	0.93	0.95	0.94	2023
5.0	0.94	0.93	0.93	1889
6.0	0.96	0.96	0.96	1991
7.0	0.96	0.95	0.96	2203
8.0	0.91	0.93	0.92	2056
9.0	0.92	0.92	0.92	2100
micro avg	0.94	0.94	0.94	21000

macro avg	0.94	0.94	0.94	21000
weighted avg	0.94	0.94	0.94	21000

Confusion matrix:

```
[[2003    1   15    2   11    4    9    3   26    8]
 [   0 2297   15   10    2    2    2    5   23    3]
 [  11    7 1965   26   20    6   11   22   24    7]
 [   1    3   38 2033    3   52    3   11   39   15]
 [   5    6   13    1 1913    0    9    2   13   61]
 [  13    2    4   38    9 1756   24    2   25   16]
 [  15    5   20    1   19   14 1904    0   13    0]
 [   5    9   25   11   10    7    2 2093    6   35]
 [   9   13   17   17   11   18   19    0 1919   33]
 [  10    7    3   17   55   15    2   36   22 1933]]
```

ii. RBF Network

- Cluster = 9

<rbfnet.RBFNet object at 0x7f489cfac390>:

	precision	recall	f1-score	support
-2.0	0.00	0.00	0.00	0
-1.0	0.00	0.00	0.00	0
0.0	0.00	0.00	0.00	2096
1.0	1.00	0.21	0.35	2349
2.0	0.00	0.00	0.00	2116
3.0	0.00	0.00	0.00	2183
4.0	0.05	0.01	0.02	2045
5.0	0.10	0.99	0.19	1890
6.0	0.00	0.00	0.00	2059
7.0	0.00	0.00	0.00	2174
8.0	0.00	0.00	0.00	2018
9.0	0.00	0.00	0.00	2070
micro avg	0.11	0.11	0.11	21000
macro avg	0.10	0.10	0.05	21000
weighted avg	0.13	0.11	0.06	21000

Confusion matrix:

```
[[   0    0    0    0    0    0    0    0    0    0    0    0]
 [   0    0    0    0    0    0    0    0    0    0    0    0]
 [   0    0    0    0    0    0    0 2096    0    0    0    0]
```

```
[   1   61  218  499  638  455  328  149    0    0    0    0]
 [   0    0    0    0    0    0   10 2106    0    0    0    0]
 [   0    0    0    0    0    1   21 2160    1    0    0    0]
 [   0    0    0    0    0    1   25 2017    2    0    0    0]
 [   0    0    0    0    0    0   16 1874    0    0    0    0]
 [   0    0    0    0    0    1   21 2037    0    0    0    0]
 [   0    0    0    0    2    6   49 1988  129    0    0    0]
 [   0    0    0    0    0    2   34 1982    0    0    0    0]
 [   0    0    0    0    0    3   38 1949   80    0    0    0]]
```

iii. SVM

iv. Decision Tree

- 500 trees

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                       max_depth=2, max_features='auto', max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=500, n_jobs=-1,
                       oob_score=False, random_state=None, verbose=0,
                       warm_start=False):
```

	precision	recall	f1-score	support
0.0	0.66	0.97	0.78	2144
1.0	0.51	0.99	0.67	2353
2.0	0.84	0.59	0.69	2107
3.0	0.62	0.69	0.65	2102
4.0	0.62	0.72	0.67	2034
5.0	1.00	0.08	0.14	1879
6.0	0.86	0.63	0.73	2033
7.0	0.72	0.84	0.77	2199
8.0	0.86	0.48	0.61	2067
9.0	0.62	0.49	0.55	2082
micro avg	0.66	0.66	0.66	21000
macro avg	0.73	0.65	0.63	21000
weighted avg	0.72	0.66	0.63	21000

Confusion matrix:

```
[[2081    5    4   11    1    0   14   16   12    0]
 [   2 2331    2    5    1    0    1    8    2    1]
 [ 130  509 1241   45   25    0   37   96   21    3]
 [ 126  307   48 1455   17    0    5   71   22   51]
```

```
[ 35  67  13  10 1466   0  57 102   7 277]
[ 375 315   6 589 107 146  39  75  58 169]
[ 263 237  50  49  70   0 1288  56  13   7]
[ 63 168  48   2  32   0   0 1852   8  26]
[ 54 591  65 150  58   0  33  30 984 102]
[ 39  64   7  37 595   0  17 277  17 1029]]
```

- 1000 trees

Classification report for classifier RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini', max_depth=2, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, n_estimators=1000, n_jobs=-1, oob_score=False, random_state=None, verbose=0, warm_start=False):

	precision	recall	f1-score	support
0.0	0.61	0.97	0.75	2059
1.0	0.49	0.99	0.65	2358
2.0	0.82	0.59	0.69	2097
3.0	0.63	0.64	0.64	2169
4.0	0.60	0.79	0.68	2023
5.0	0.98	0.06	0.11	1920
6.0	0.88	0.59	0.71	2085
7.0	0.68	0.86	0.76	2119
8.0	0.89	0.40	0.56	2082
9.0	0.64	0.46	0.54	2088
micro avg	0.64	0.64	0.64	21000
macro avg	0.72	0.63	0.61	21000
weighted avg	0.72	0.64	0.61	21000

Confusion matrix:

```
[[1996    7    5   11    3    0   11   21    4    1]
 [   1 2332    3    2    0    0    0   16    2    2]
 [ 158  493 1234   26   35    0   46   89   13    3]
 [ 160  385   44 1391   20    2    7   91   19   50]
 [  28   52   17    3 1589    0   35  122    5  172]
 [ 495  299   12  538  155  112   20   97   35  157]
 [ 279  243   71   62  115    0 1230   72    6    7]
 [  43  158   29    2   36    0    0 1818    7   26]
```

```
[ 69 708  83 122  63   0  38  36 843 120]
 [ 31  94   6  38 625   0   9 312  12 961]]
```

- 1500 trees

Classification report for classifier RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini', max_depth=2, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, n_estimators=1500, n_jobs=-1, oob_score=False, random_state=None, verbose=0, warm_start=False):

	precision	recall	f1-score	support
0.0	0.63	0.98	0.76	2061
1.0	0.50	0.99	0.66	2374
2.0	0.81	0.56	0.66	2106
3.0	0.64	0.65	0.64	2171
4.0	0.60	0.71	0.65	2074
5.0	0.99	0.08	0.15	1874
6.0	0.81	0.60	0.69	2001
7.0	0.74	0.85	0.80	2222
8.0	0.85	0.44	0.58	2026
9.0	0.61	0.48	0.54	2091
micro avg	0.65	0.65	0.65	21000
macro avg	0.72	0.64	0.61	21000
weighted avg	0.71	0.65	0.62	21000

Confusion matrix:

```
[[2017    3    4    8    2    0    9   11    7    0]
 [   1 2355    4    2    1    0    0    9    1    1]
 [ 151  525 1186   32   36    0   73   76   23    4]
 [ 152  393   57 1419   17    2    5   61   24   41]
 [  38   57   14    4 1476    0   63  114    9  299]
 [ 433  275    9  571  105  150   59   57   57  158]
 [ 288  221   73   47   98    0 1210   48    5   11]
 [  51  160   31    1   36    0    1 1899   13   30]
 [  55  633   76  114   60    0   64   24  890  110]
 [  33   92   11   36  630    0   13  255   13 1008]]
```

5. 결론

2차원 데이터셋의 경우 linear kernel을 사용한 SVM을 제외하고는 어느정도 만족할 만한 결과를 얻을 수 있었다. 하지만 MNIST 데이터셋의 경우 상당히 많은 모델들에서 결과가 좋지 않았다. 데이터의 차원이 784까지 증가하고 클래스의 개수 또한 9개로 늘어서 정확도가 상당히 저하된 것으로 판단된다. 그래도 각 알고리즘마다 파라미터를 증가시킬수록 성능의 개선됨을 보아 충분히 좋은 컴퓨팅 파워를 사용할 수 있다면 꽤나 좋은 결과를 얻을 수 있을 것으로 판단된다. 이번 과제에서는 컴퓨터 성능이 좋지 않아 복잡한 모델을 테스트하지 못하였지만 기회가 된다면 파라미터를 대폭 증가시켜 실험해보고 싶다.