Machine Learning and Intelligent Systems - Fall 2017

3rd Lab Session's Report

by Berkay KÖKSAL

Part 1: Implementing the computation of entropy

- Trainingdata result screenshot:

```
Running in training mode.
Reading file ../../data/kdd_cup_trainingdata
   86854 samples with 38 attributes
Data set: 86854 samples, 38 attributes
Node 0 depth=0 size=86854 N(Y=0)=83225 N(Y=1)=3629 entropy=0.250407
  ...trying attribute 5
  split node 0 with question X[4]<=18 (entropy improvement=0.136283) to create nodes 1 and 2
Node 1 depth=1 size=9133 N(Y=0)=5606 N(Y=1)=3527 entropy=0.962291
  split node 1 with question X[2]<=5 (entropy improvement=0.868361) to create nodes 3 and 4
Node 3 depth=2 size=5522 N(Y=0)=5503 N(Y=1)=19 entropy=0.0331116
  split node 3 with question X[31]<=0.14 (entropy improvement=0.0157501) to create nodes 5 and 6
Node 5 depth=3 size=5404 N(Y=0)=5402 N(Y=1)=2 entropy=0.00475285
  split node 5 with question X[25]<=0.18 (entropy improvement=0.00424306) to create nodes 7 and 8
Node 7 depth=4 size=3 N(Y=0)=1 N(Y=1)=2 entropy=0.918296
Node 8 depth=4 size=5401 N(Y=0)=5401 N(Y=1)=0 entropy=0
Node 6 depth=3 size=118 N(Y=0)=101 N(Y=1)=17 entropy=0.594794
  split node 6 with question X[36]<=0.05 (entropy improvement=0.333859) to create nodes 9 and 10
Node 9 depth=4 size=87 N(Y=0)=87 N(Y=1)=0 entropy=0
Node 10 depth=4 size=31 N(Y=0)=14 N(Y=1)=17 entropy=0.993234
Node 4 depth=2 size=3611 N(Y=0)=103 N(Y=1)=3508 entropy=0.186935
  split node 4 with question X[2]<=12 (entropy improvement=0.0976333) to create nodes 11 and 12
Node 11 depth=3 size=3419 N(Y=0)=14 N(Y=1)=3405 entropy=0.0383751
  split node 11 with question X[31]<=0.02 (entropy improvement=0.006979) to create nodes 13 and 14
Node 13 depth=4 size=1054 N(Y=0)=14 N(Y=1)=1040 entropy=0.101844
Node 14 depth=4 size=2365 N(Y=0)=0 N(Y=1)=2365 entropy=0
Node 12 depth=3 size=192 N(Y=0)=89 N(Y=1)=103 entropy=0.996161
  split node 12 with question X[24]<=0 (entropy improvement=0.317888) to create nodes 15 and 16
Node 15 depth=4 size=129 N(Y=0)=87 N(Y=1)=42 entropy=0.910348
Node 16 depth=4 size=63 N(Y=0)=2 N(Y=1)=61 entropy=0.203074
Node 2 depth=1 size=77721 N(Y=0)=77619 N(Y=1)=102 entropy=0.0144564
  ...trying attribute 5
  split node 2 with question X[8]<=0 (entropy improvement=0.00581168) to create nodes 17 and 18
Node 17 depth=2 size=77180 N(Y=0)=77152 N(Y=1)=28 entropy=0.00466949
  ...trying attribute 5
  split node 17 with question X[29]<=48 (entropy improvement=0.000904318) to create nodes 19 and 20
Node 19 depth=3 size=9201 N(Y=0)=9175 N(Y=1)=26 entropy=0.0279973
  split node 19 with question X[28]<=6 (entropy improvement=0.00380336) to create nodes 21 and 22
Node 21 depth=4 size=289 N(Y=0)=277 N(Y=1)=12 entropy=0.24923
Node 22 depth=4 size=8912 N(Y=0)=8898 N(Y=1)=14 entropy=0.0168964
Node 20 depth=3 size=67979 N(Y=0)=67977 N(Y=1)=2 entropy=0.000485345
  ...trying attribute 20
  split node 20 with question X[29]<=128 (entropy improvement=0.000101804) to create nodes 23 and 24
Node 23 depth=4 size=6182 N(Y=0)=6180 N(Y=1)=2 entropy=0.00421753
Node 24 depth=4 size=61797 N(Y=0)=61797 N(Y=1)=0 entropy=0
Node 18 depth=2 size=541 N(Y=0)=467 N(Y=1)=74 entropy=0.575753
  split node 18 with question X[4]<=126 (entropy improvement=0.30717) to create nodes 25 and 26
Node 25 depth=3 size=51 N(Y=0)=1 N(Y=1)=50 entropy=0.139233
  split node 25 with question X[0]<=192 (entropy improvement=0.139233) to create nodes 27 and 28
Node 27 depth=4 size=50 N(Y=0)=0 N(Y=1)=50 entropy=0
Node 28 depth=4 size=1 N(Y=0)=1 N(Y=1)=0 entropy=0
Node 26 depth=3 size=490 N(Y=0)=466 N(Y=1)=24 entropy=0.282046
  split node 26 with question X[11]<=0 (entropy improvement=0.146722) to create nodes 29 and 30
Node 29 depth=4 size=456 N(Y=0)=452 N(Y=1)=4 entropy=0.0725372
Node 30 depth=4 size=34 N(Y=0)=14 N(Y=1)=20 entropy=0.977418
The complete tree has 31 nodes (non-splitting nodes are included)
Tree (31 nodes) saved in: ../../data/kdd_cup_tree
Finished, press Enter to exit.
```

- on level 0 we have node 0 : question is if X[4] is less than the 18 threshold
- on level 1 we have node 1 and node 2 :

  - node 1 : the chosen question is X[2] < 5 (threshold)

  - node 2 : holds 77721 of the input data, needs to be split once more. So splits with question x[8] < 0 (threshold is zero)

- Comments on the tree created for the complete training set. (which attributes are the most significant in determining a normal network connection, etc...)

For the trainingdata file (traning completed in 5 minutes)

The final tree has 31 nodes. The attributes used for the splitting are X[4], X[2], X[31], X[25], X[36], X[31], X[24], X[8], X[28], X[29], X[0] and X[11]. As can be seen we can choose many different attributes to split the dataset on different levels of the decision tree. The threshold we use to compare the attributes are also different on each level, sometimes zero sometimes even reachs up to 192.

Tree file generated:

```
0 0 4 18 0 1 2
1 1 2 5 0 3 4
2 1 8 0 0 17 18
3 2 31 0.14 0 5 6
4 2 2 12 1 11 12
5 3 25 0.18 0 7 8
6 3 36 0.05 0 9 10
7 4 -1 0 1
8 4 -1 0 0
9 4 -1 0 0
10 4 -1 0 1
11 3 31 0.02 1 13 14
12 3 24 0 1 15 16
13 4 -1 0 1
14 4 -1 0 1
15 4 -1 0 0
16 4 -1 0 1
17 2 29 48 0 19 20
18 2 4 126 0 25 26
19 3 28 6 0 21 22
20 3 29 128 0 23 24
21 4 -1 0 0
22 4 -1 0 0
23 4 -1 0 0
24 4 -1 0 0
25 3 0 192 1 27 28
26 3 11 0 0 29 30
27 4 -1 0 1
28 4 -1 0 0
29 4 -1 0 0
30 4 -1 0 1
```

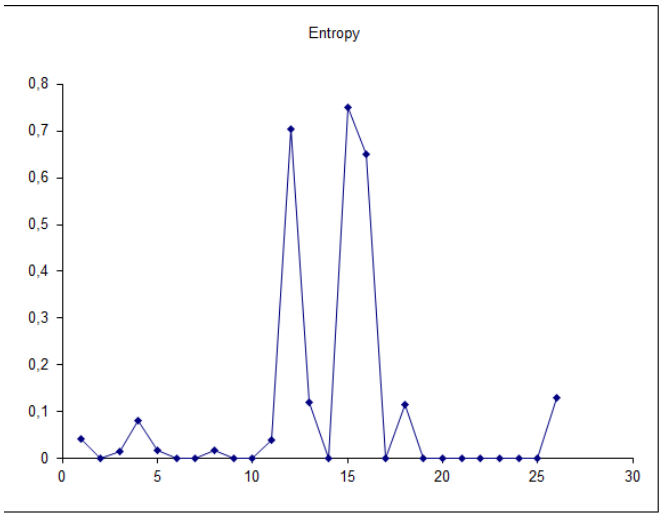
Part 2: Evaluation of the classification

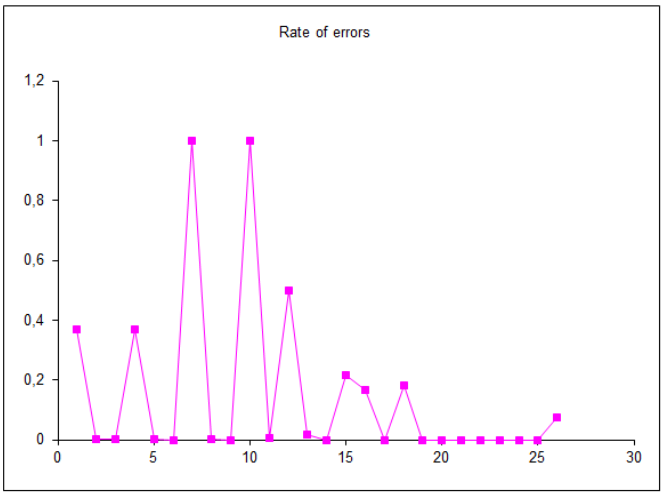
Tree file is created using the trainingdata file. So for this part I will try to predict another dataset.

- Predicting the kdd_cup_testdata

9602 / 10000 Success, 392 instruction missed.

Node	Entropy	error
0	0,00508331	0,0398
1	0,0418891	0,37069
2	0,000792863	0,00122823
3	0,0167006	0,00155521
4	0,0822787	0,37069
5	0,0170452	0,0015873
6	0	0
7	0	1
8	0,0170723	0,00158983
9	0	0
10	0	1
11	0,0391977	0,00533333
12	0,703639	0,5
13	0,12148	0,0165289
14	0	0
15	0,749595	0,214286
16	0,650022	0,166667
17	0	0
18	0,116408	0,180328
19	0	0
20	0	0
21	0	0
22	0	0
23	0	0
24	0	0
25	0	0
26	0,131498	0,0740741
27	0	0
28	0	0
29	0,139233	0,0196078
30	0	0



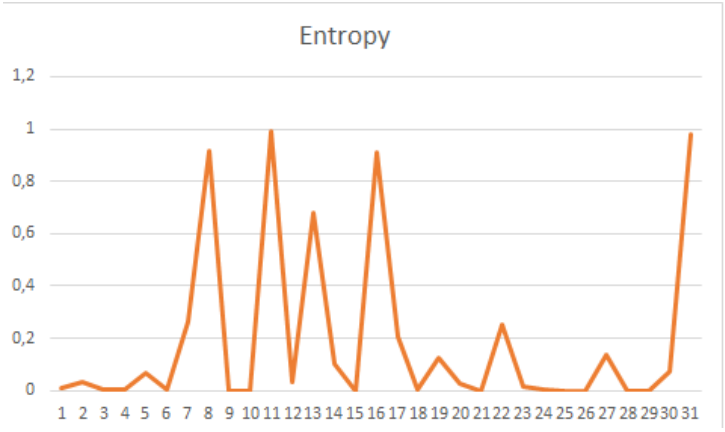


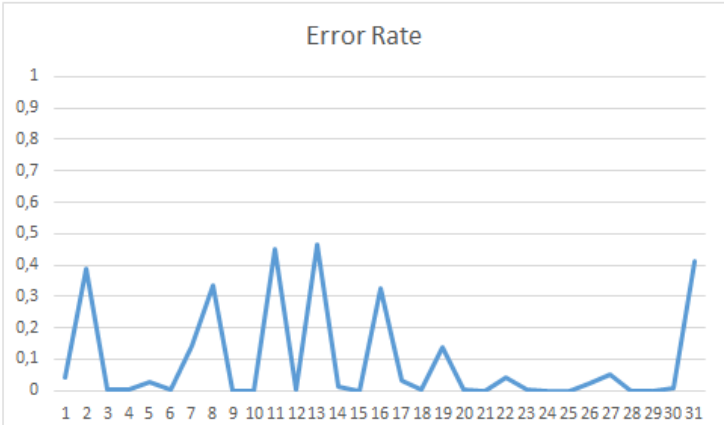

```
Test data: 10000 samples, 398 errors
  9602 ok
  0 intrusions correctly detected
  398 intrusions missed
  0 false alarms
node    entropy error rate      nb of errors
7       0       1       1
8       0.0170723       0.00158983      1
5       0.0170452       0.0015873       1
9       0       0       0
10      0       1       2
6       0       0       0
3       0.0167006       0.00155521      1
13      0.12148 0.0165289       2
14      0       0       0
11      0.0391977       0.00533333      2
15      0.749595        0.214286        3
16      0.650022        0.166667        2
12      0.703639        0.5     13
4       0.0822787       0.0374065       15
1       0.0418891       0.37069 387
21      0       0       0
22      0       0       0
19      0       0       0
23      0       0       0
24      0       0       0
20      0       0       0
17      0       0       0
27      0       0       0
28      0       0       0
25      0       0       0
29      0.139233        0.0196078       1
30      0       0       0
26      0.131498        0.0740741       4
18      0.116408        0.180328        11
2       0.000792863     0.00122823      11
0       0.00508331      0.0398  398
```

- Predicting the trainingdata
  Let us try to predict the dataset that we used to generate our tree just to see what will happen.

83225 / 86854 Success, 3629 instruction missed.

Node	Entropy	Error
0	0,00674819	0,0417828
1	0,0296854	0,386182
2	0,00405283	0,00131239
3	0,00607482	0,00344078
4	0,0657911	0,028524
5	0,000509787	0,000370096
6	0,260934	0,144068
7	0,918296	0,333333
8	0	0
9	0	0
10	0,993234	0,451613
11	0,0313961	0,00409476
12	0,678274	0,463542
13	0,101844	0,0132827
14	0	0
15	0,910348	0,325581
16	0,203074	0,031746
17	0,00322209	0,000362788
18	0,122568	0,136784
19	0,0241939	0,00282578
20	0,000383541	0
21	0,24923	0,0415225
22	0,0168964	0,00157092
23	0,00421753	0,00032352
24	0	0
25	0	0,0196078
26	0,135325	0,0489796
27	0	0
28	0	0
29	0,0725372	0,00877193
30	0,977418	0,411765






```
Test data: 86854 samples, 3629 errors
  83225 ok
  0 intrusions correctly detected
  3629 intrusions missed
  0 false alarms
node    entropy error rate      nb of errors
7       0.918296        0.333333        1
8       0       0       0
5       0.000509787     0.000370096     2
9       0       0       0
10      0.993234        0.451613        14
6       0.260934        0.144068        17
3       0.00607482      0.00344078      19
13      0.101844        0.0132827       14
14      0       0       0
11      0.0313961       0.00409476      14
15      0.910348        0.325581        42
16      0.203074        0.031746        2
12      0.678274        0.463542        89
4       0.0657911       0.028524        103
1       0.0296854       0.386182        3527
21      0.24923 0.0415225       12
22      0.0168964       0.00157092      14
19      0.0241939       0.00282578      26
23      0.00421753      0.00032352      2
24      0       0       0
20      0.000383541     2.94209e-05     2
17      0.00322209      0.000362788     28
27      0       0       0
28      0       0       0
25      0       0.0196078       1
29      0.0725372       0.00877193      4
30      0.977418        0.411765        14
26      0.135325        0.0489796       24
18      0.122568        0.136784        74
2       0.00405283      0.00131239      102
0       0.00674819      0.0417828       3629
```

When we are using our decision tree to predict a dataset, our success relies on our threshold. If we use a very high threshold our decisions will not be correct and it will result in a terrible predictions with very wrong classification. Similarly, if we use a very low threshold then we will over fit the dataset and we will never give correct decisions because even a small error will result in a rejection on our decision algorithm. There is no such thing as the best threshold unless we 100% know what the dataset is, which is impossible since as it is named this is a ‘’prediction’’. Therefore, we need to try our best to define a threshold that will give us the best solution in overall.

Please notice that even the tree file we generated on the trainingset gave better result on another dataset (%96) but now on its own data that generated it (%95) and the reason is because we over fit the values on the trainingset, resulting in rejection of most on the attributes to be decided as success.