N. Bertschinger
M. Kaschube
V. Ramesh

Machine Learning I
8. Exercise Sheet
Due on Jan. 13, midnight

Competition: On this exercise you can win **bonus points** by scoring high in a small competition. Here are the instructions:

- Download the training data from the course web site:
  - *trainX.dat*: 750 samples × 20 input dimensions
  - *traint.dat*: target output for 750 samples
- Send your program within four weeks (deadline: Sunday, Jan. 13, midnight) via email to `bertschinger@fias.uni-frankfurt.de` with subject "#MLCompetition2019".

  Your program must be *executable* without arguments and do the following:
  - Read test inputs from standard input `stdin` (same format as training data, i.e. 20 numeric values per line)
  - Write predictions to standard output (same format as training data, i.e. one numeric value per line)

  Your program is allowed to use additional files, e.g. containing the weights of your model . . . just hand them in along with your program. Your executable can assume that the files can be found in the working directory, i.e. where your program is run.
- *Scoring*:
  - You get **5 points** for participation. Your program should run without error though.
  - The performance of your program is evaluated on testing data — which are different from your training data — in terms of its classification loss. The following loss function will be used:

		True class: 0	True class: 1
Predicted class	0	0	5
	1	1	0

  Thus, a false positive has a higher cost than a false negative. Your goal is to minimize the loss on the test set.

  You get $\frac{1}{2}$ **point** per 5% (rounded up) of 1− loss, e.g. for a loss of 0.42, i.e. 1 - 0.42 = 0.58, you would get $12\frac{1}{2}$ points.

N. Bertschinger
M. Kaschube
V. Ramesh

Machine Learning I
8. Exercise Sheet
Due on Jan. 13, midnight

Hints:

- Some of the input variables are actually categorical variables. The following table list their minimum, maximum and number of different values (categories):

column	min	max	categories
0	1	4	4
2	0	4	5
3	0	10	10
5	1	5	5
6	1	5	5
7	1	4	4
8	1	4	4
9	1	3	3
10	1	4	4
11	1	4	4
13	1	3	3
14	1	3	3
15	1	4	4
16	1	4	4
17	1	2	2
18	1	2	2

  You might want to recode all or some of these columns using the `sklearn.preprocessing.OneHotEncoder`. As an example, a column containing three possible values $1, 2, 3$ would be recoded into three binary columns as follows:

Original	Recoded		
1	1	0	0
2	0	1	0
3	0	0	1
2	0	1	0
1	1	0	0
2	0	1	0
⋮		⋮	

- Plot and inspect the data to see if non-linear transformations would make sense. Experiment with different basis functions, e.g. polynomial, squared exponential ....
- Beware of overfitting and consider adding regularization. This is particularly useful if you have added many basis functions above.

  You may want to use cross-validation to adjust your hyper-parameters, e.g. for regularization.
- Consider the loss function and choose your decision threshold accordingly.

If you have any further questions, don't hesitate to contact me.
Good luck ☺