



ADRIAN BEVAN

ATLAS UK MEETING 2019: MACHINE LEARNING SESSION

NEURAL NETWORKS (NN_s): TUTORIAL

(SEE SEPARATE SLIDES ON DECISION TREES FOR DETAILS OF THE ALGORITHM)

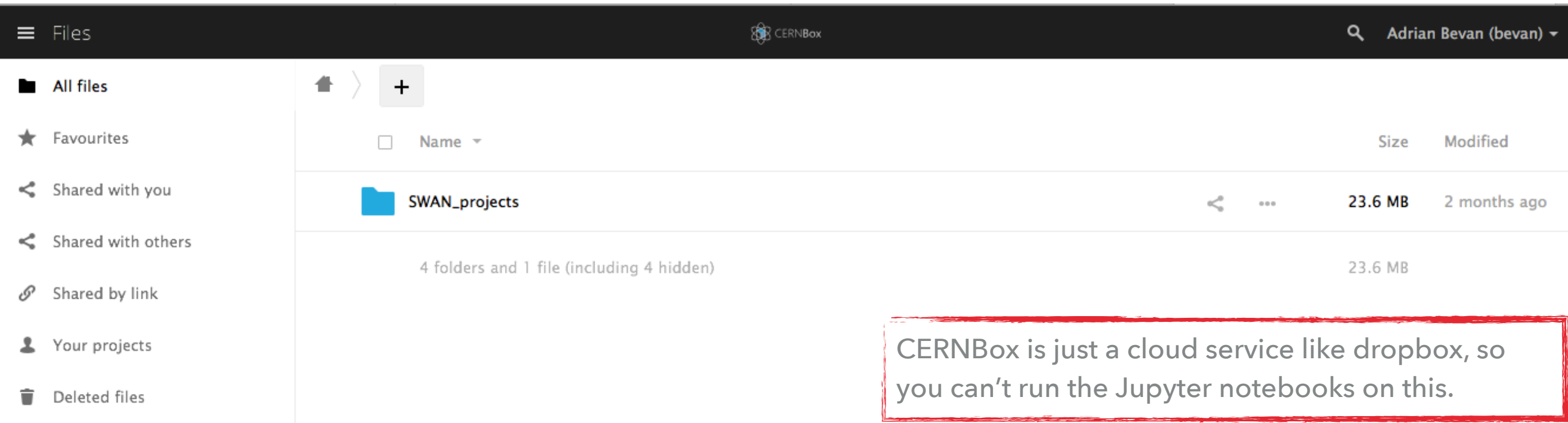
This tutorial has been written for use with CERNBox and the cern SWAN server. The code relies on Python 3 (2 is sufficient) and TensorFlow v1.2 through 1.8. Some API functionality has been changed with respect to the most recent stable API (v1.12).

WORKPLAN

- ▶ 0. Setup.
- ▶ 1. Model building and execution with TensorFlow
- ▶ 2. Linear regression:
 - ▶ Explore hyper-parameter (HP) performance and optimisation convergence for a gradient descent algorithm with a simple model.
- ▶ 3. Function approximation:
 - ▶ Explore HP training and model reliability for a regression problem.
- ▶ 4. Closing remarks

0. SETUP

- ▶ We will use CERNBox and the CERN SWAN server.
- ▶ To start make sure that you have installed CERNBox. Please see the following URL for details and follow the instructions:
<http://information-technology.web.cern.ch/services/CERNBox-Service>
- ▶ With CERNBox setup you should see a directory cernbox in your home area and you should be able to inspect your files online via <https://cernbox.cern.ch/>



Files

CERNBox

Adrian Bevan (bevan)

All files

Favourites

Shared with you

Shared with others

Shared by link

Your projects

Deleted files

SWAN_projects

23.6 MB

2 months ago

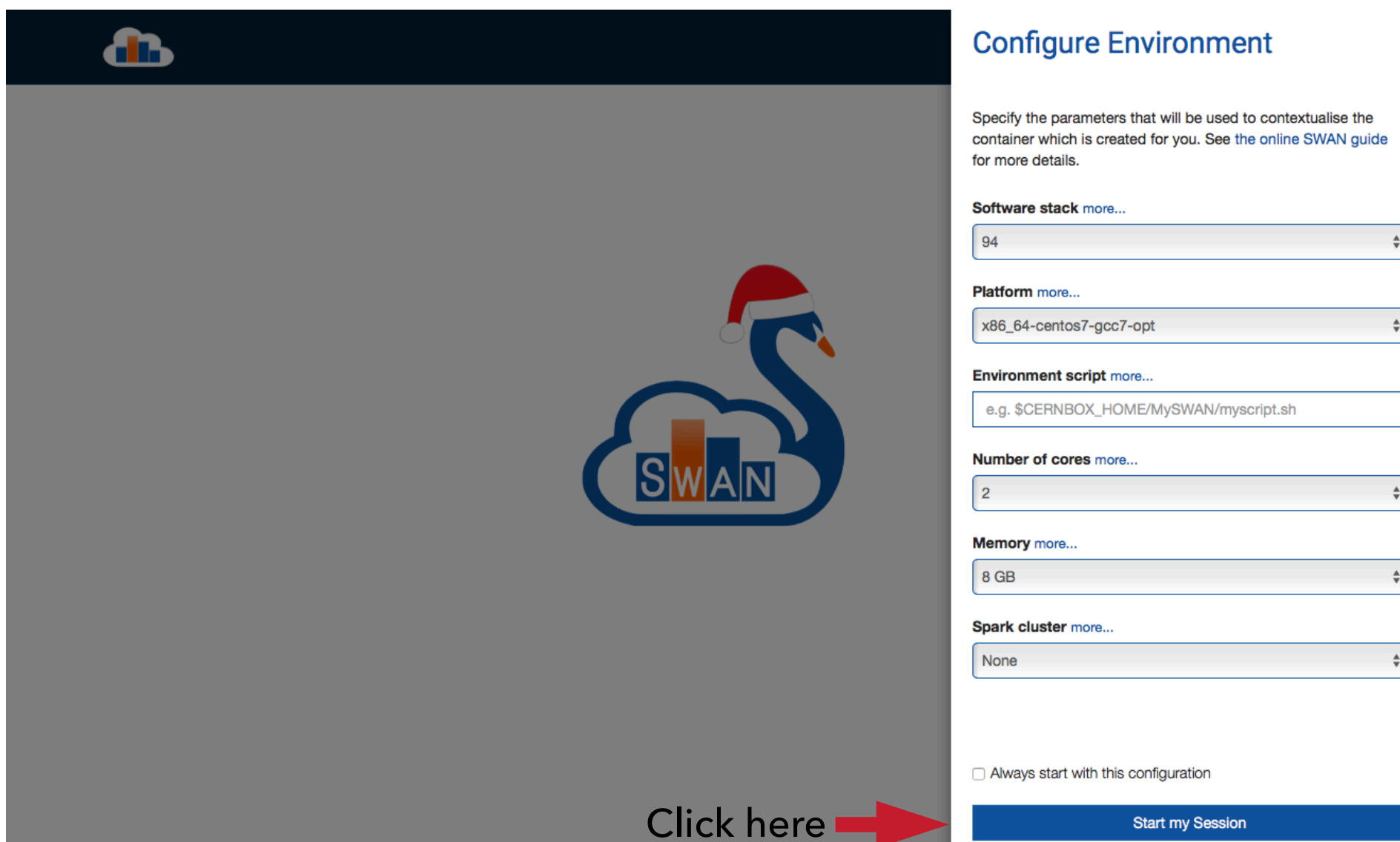
4 folders and 1 file (including 4 hidden)

23.6 MB

CERNBox is just a cloud service like dropbox, so you can't run the Jupyter notebooks on this.

0. SETUP

- ▶ Now we can look at the CERN SWAN server. Navigate to:
<https://swan.cern.ch/>



Configure Environment

Specify the parameters that will be used to contextualise the container which is created for you. See [the online SWAN guide](#) for more details.

Software stack [more...](#)
94

Platform [more...](#)
x86_64-centos7-gcc7-opt

Environment script [more...](#)
e.g. \$CERNBOX_HOME/MySWAN/myscript.sh

Number of cores [more...](#)
2

Memory [more...](#)
8 GB

Spark cluster [more...](#)
None

Always start with this configuration

[Click here](#) → [Start my Session](#)

0. SETUP

- ▶ Having downloaded the example tar file from the indico agenda, you can now copy the content of the NN directory into your cernbox directory.

- ▶ i.e.

```
tar xvf ATLAS-UK.tar  
cp -r ./NN ~/cernbox
```

- ▶ Having done this you should be able to view and run the Jupyter notebooks by browsing to them via the SWAN server web page.

0. SETUP

Click here



The screenshot shows the ATLAS CERNBox interface. At the top, there is a navigation bar with 'Projects', 'Share', and 'CERNBox' (highlighted with an orange underline). Below the navigation bar, the breadcrumb path 'SWAN > CERNBox > NN' is visible. The main content area shows a list of notebooks under the heading 'NN'. The list has columns for 'NAME', 'SIZE', 'STATUS', and 'MODIFIED'. Two notebooks are listed: 'FunctionApproximation.ipynb' (57.3 kB, a minute ago) and 'LinearRegression.ipynb' (36.7 kB, a minute ago).

NAME	SIZE	STATUS	MODIFIED
FunctionApproximation.ipynb	57.3 kB		a minute ago
LinearRegression.ipynb	36.7 kB		a minute ago

Click to browse down to your [SWAN](#) > [CERNBox](#) > [NN](#) level.

It may take a few minutes for CERNBox to synchronise your data so that it becomes visible in the browser.

You can now click on one of the notebooks to view it.

The notebooks have comments and text cells describing the code.

1. MODEL BUILDING AND EXECUTION WITH TENSORFLOW Background

- ▶ Machine Learning with TensorFlow works in the same way you would describe a calculation on the white board.
 - ▶ 1. You define the calculation you want to do (the graph).
 - ▶ 2. Then you evaluate that graph some number of times; each evaluation corresponds to an iteration in the optimisation step.
- ▶ Conceptually this differs from how one would traditionally approach a coding problem like this, and the different approach can take some time to adapt to.

2. LINEAR REGRESSION

Background


LinearRegression.ipynb

36.7 kB

a minute ago

- ▶ We will learn the values of m and c for

$$y = mx + c$$

- ▶ Initial parameter guesses are in the range $[0, 1]$
- ▶ We set the learning rate to be some reasonable number
- ▶ We set the number of epochs to some value (initially 100) to explore if this is enough for the model to converge.
- ▶ Cells in the notebook can be evaluated using `[shift]+[enter]` or by pressing the play button .

2. LINEAR REGRESSION

Background

LinearRegression.ipynb

36.7 kB

a minute ago

- ▶ Suggested things to explore:
 - ▶ Increase the `learning_rate` from 0.0005 to observe how the model agreement and optimisation convergence changes.
 - ▶ At what point does the optimisation stop working?
 - ▶ You may want to change the number of `training_epochs` to facilitate getting the model to converge.
 - ▶ You may want to change the true values of `gradient` and `intercept` to explore how this affects the optimisation (this is equivalent to changing how close the initial guess is for the optimisation to the true parameters).

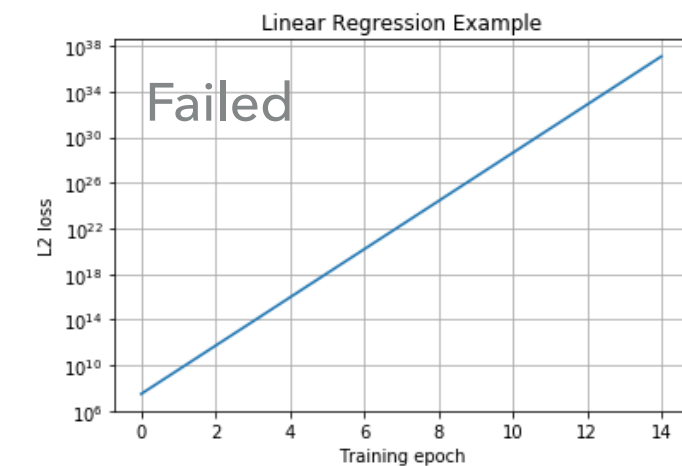
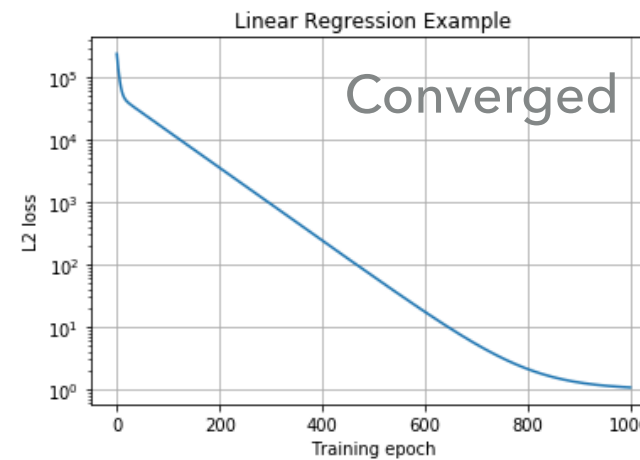
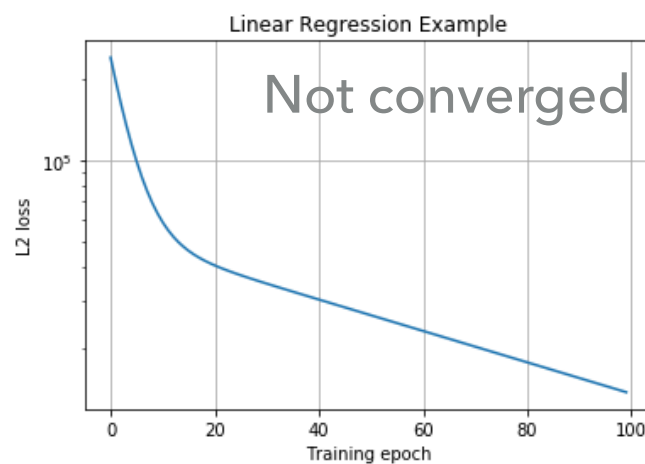
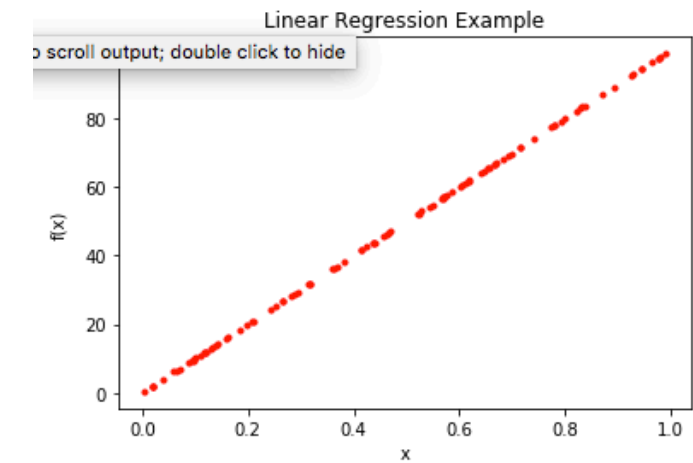
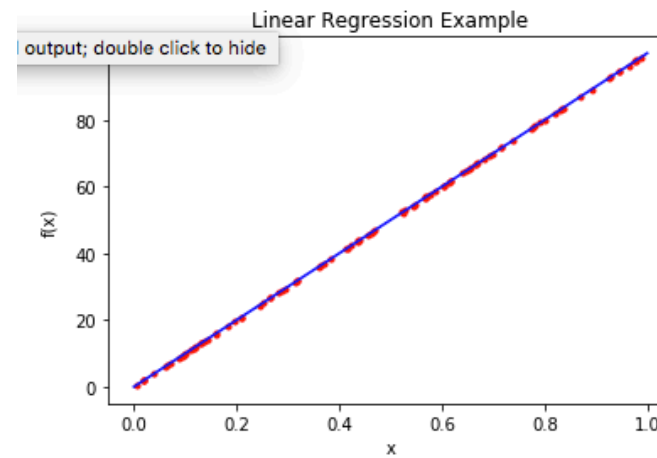
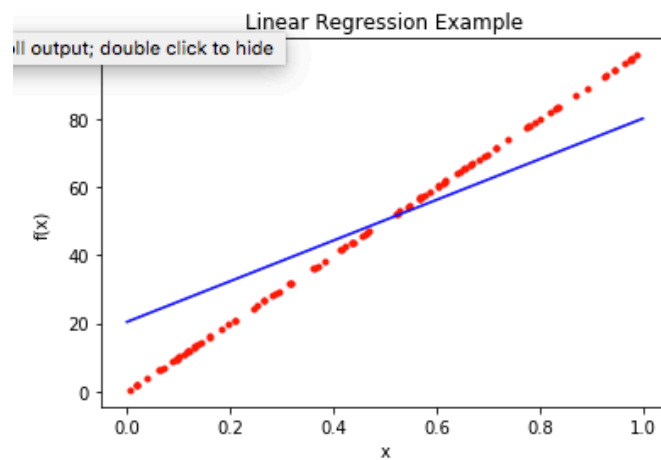
2. LINEAR REGRESSION

LinearRegression.ipynb

36.7 kB

a minute ago

- ▶ Having done this please pause and reflect.
- ▶ This is a simple model with 1 input feature, 1 output and 2 HPs. It takes some effort to ensure that the model converges to a sensible result.



3. FUNCTION APPROXIMATION

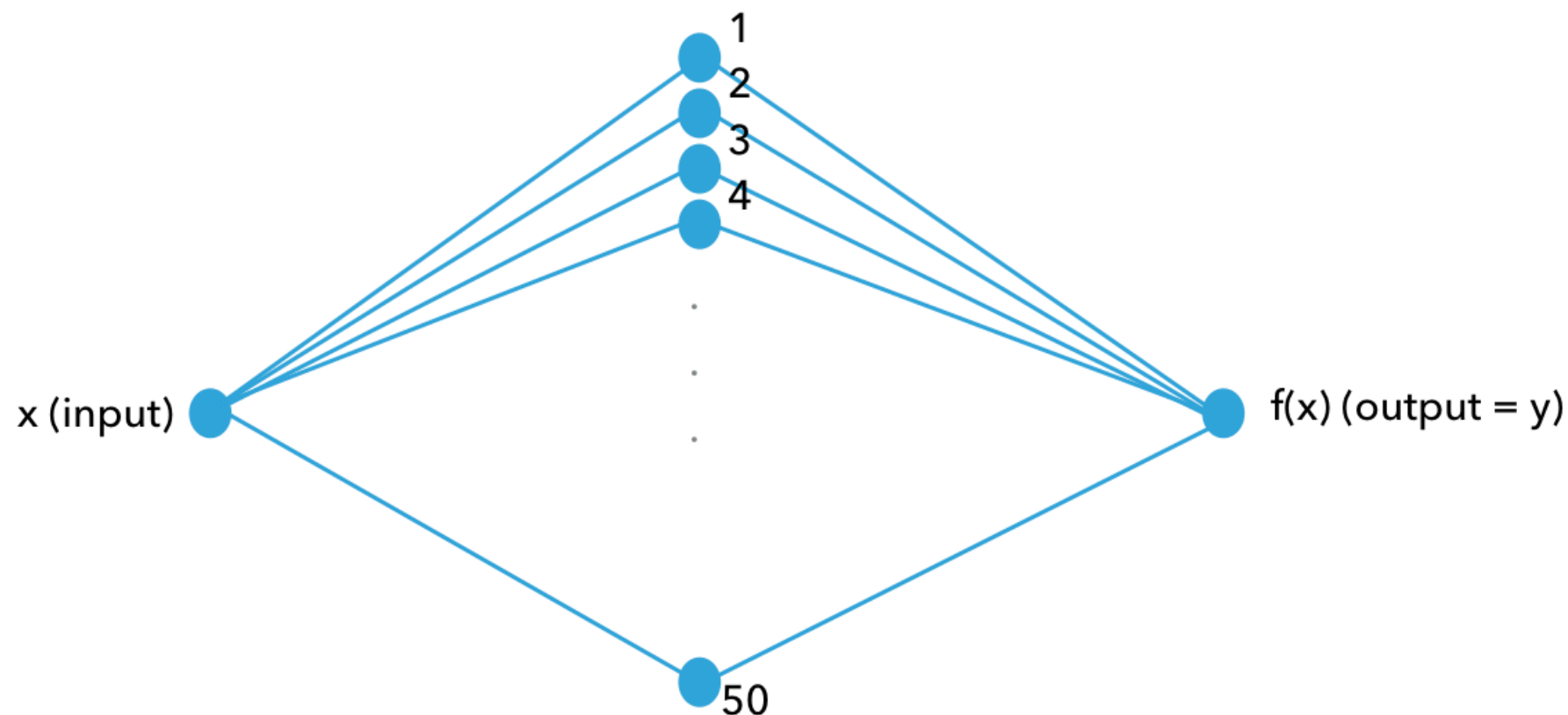
Background

FunctionApproximation.ipynb

57.3 kB

a minute ago

- ▶ We now consider a more complicated model - this will be a neural network with the configuration 1 input feature, 50 nodes in a single layer and one output feature.



- ▶ For this example we are going to model $y = x^2$.

3. FUNCTION APPROXIMATION

Background

FunctionApproximation.ipynb

57.3 kB


a minute ago

- ▶ Each node in this model is an activation function that has two parameters, takes one input and gives one output (just like the linear regression example).

$$y_i = f(w_i x + \beta_i)$$

- ▶ The chosen activation function for this example is the relu activation function.
- ▶ The output node is just the matrix operation $w^T x + \beta$.

3. FUNCTION APPROXIMATION

[Background](#) FunctionApproximation.ipynb

57.3 kB

a minute ago

- ▶ After running the example out of the box try the following:
 - ▶ Change the `learning_rate` by a factor of 10 either way to observe how that affects the training performance.
 - ▶ Change the number of `training_epochs` with a reasonable learning rate to get a good predictive model.
 - ▶ Change the noise level from 0.1 to 1.0 to see how this affects the learning process.

3. FUNCTION APPROXIMATION

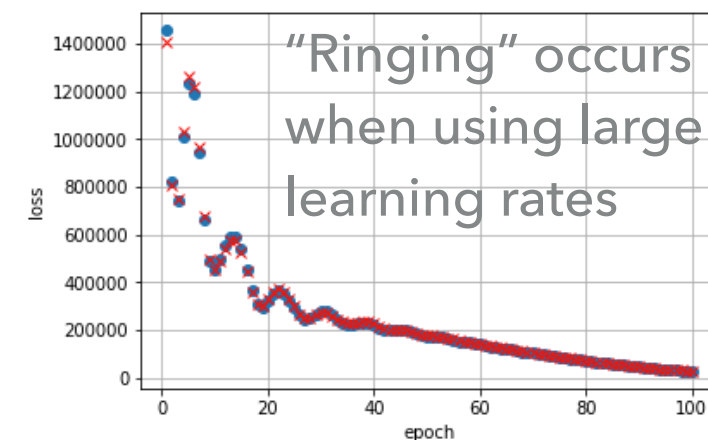
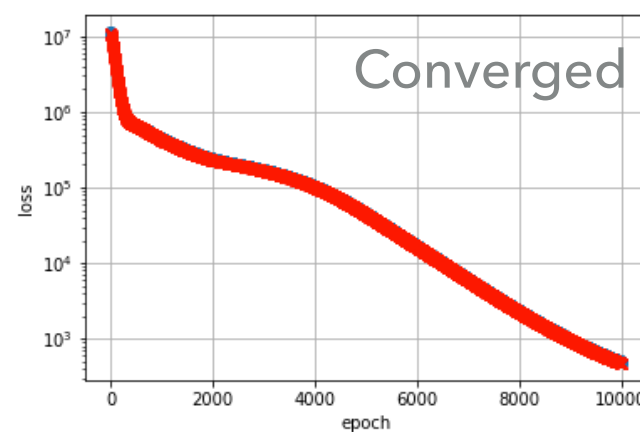
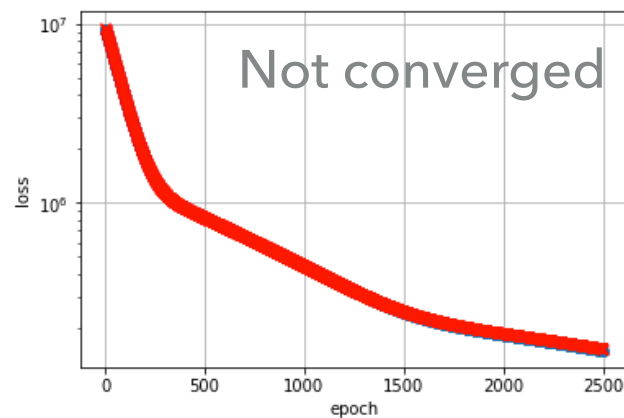
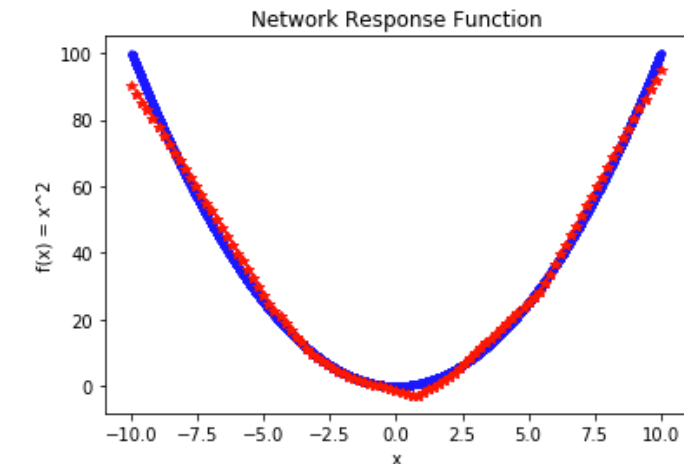
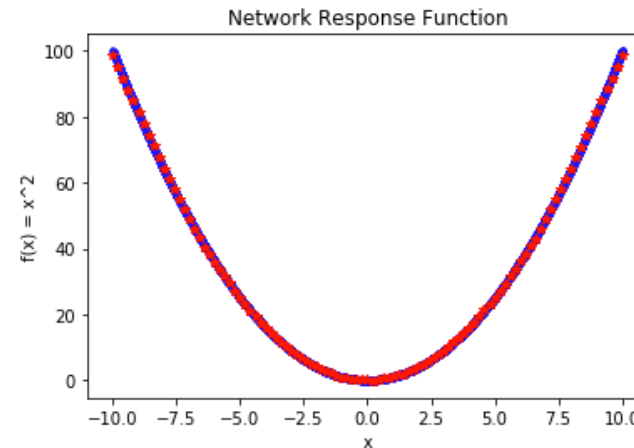
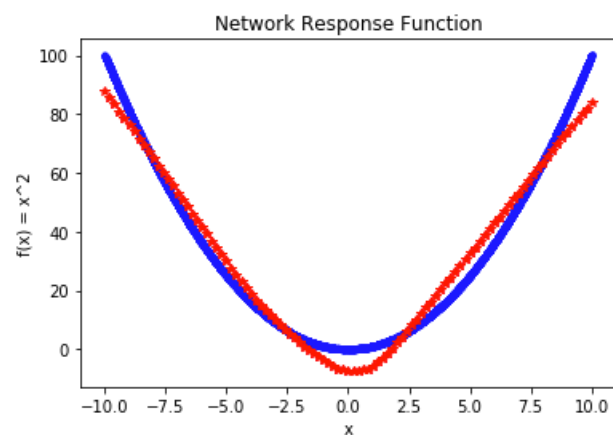
Review Results

FunctionApproximation.ipynb

57.3 kB

a minute ago

- ▶ As with the linear regression example, one needs to tune HPs related to training to get a good model.



- ▶ HEP problems have larger input feature spaces and would generally result in more layers for a NN. This means care needs to be taken with training to ensure a robust model is derived.

4. CLOSING REMARKS

FunctionApproximation.ipynb

57.3 kB

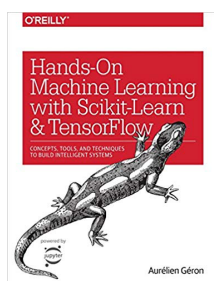
a minute ago

- ▶ Why didn't we discuss deep learning?
 - ▶ These tutorials are intended to give a short practical introduction, that's not compatible with a deep learning approach.
 - ▶ For deep learning you need lots of training examples, and compute resource in order to develop and train a model for a large number of epochs (doesn't fit into an hour).
- ▶ Interested in deep learning?
 - ▶ See the books by Goodfellow and Géron. You may wish to review some of my lectures/tutorials that discuss deep learning, including in the HEP context (see CINVESTAV 2018):
 - ▶ <https://pprc.qmul.ac.uk/~bevan/teaching.html>

REFERENCES & NOTES



- ▶ There are many useful tutorials on the tensorflow web page: <https://www.tensorflow.org> (check the API version number when browsing this site as the version changes frequently).



- ▶ The O'Reilly book by Géron, "Hands-On Machine Learning with Scikit-Learn & TensorFlow", is another good starting point.



- ▶ CERN has an interexperiment machine learning group: <https://iml.web.cern.ch>



- ▶ ATLAS has a Machine Learning Forum: <https://twiki.cern.ch/twiki/bin/viewauth/AtlasComputing/MachineLearningForum>



- ▶ I have more material on ML and TensorFlow available at:
 - ▶ <https://pprc.qmul.ac.uk/~bevan/teaching.html>
 - ▶ <https://www.qmul.ac.uk/summer-school/>