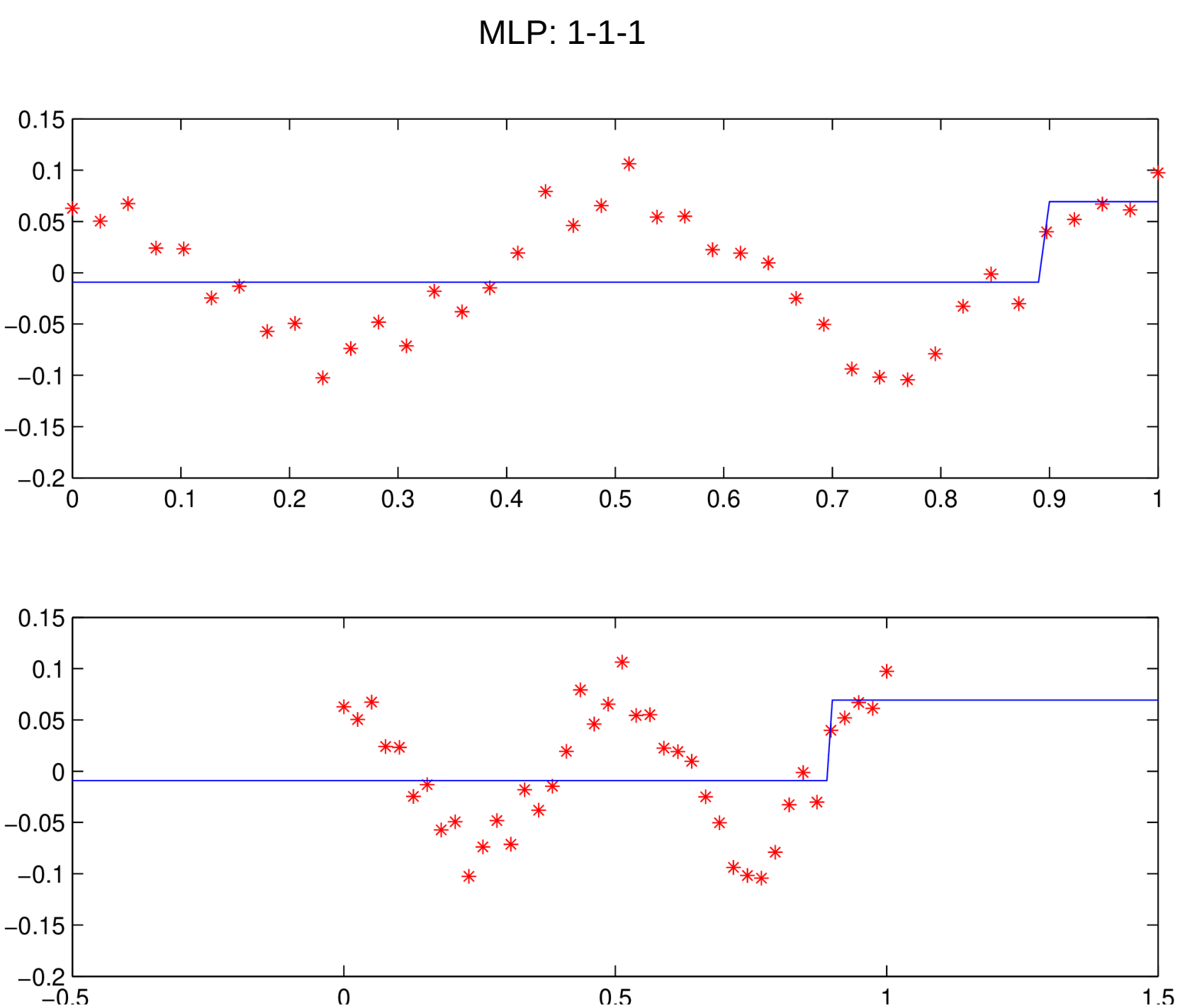
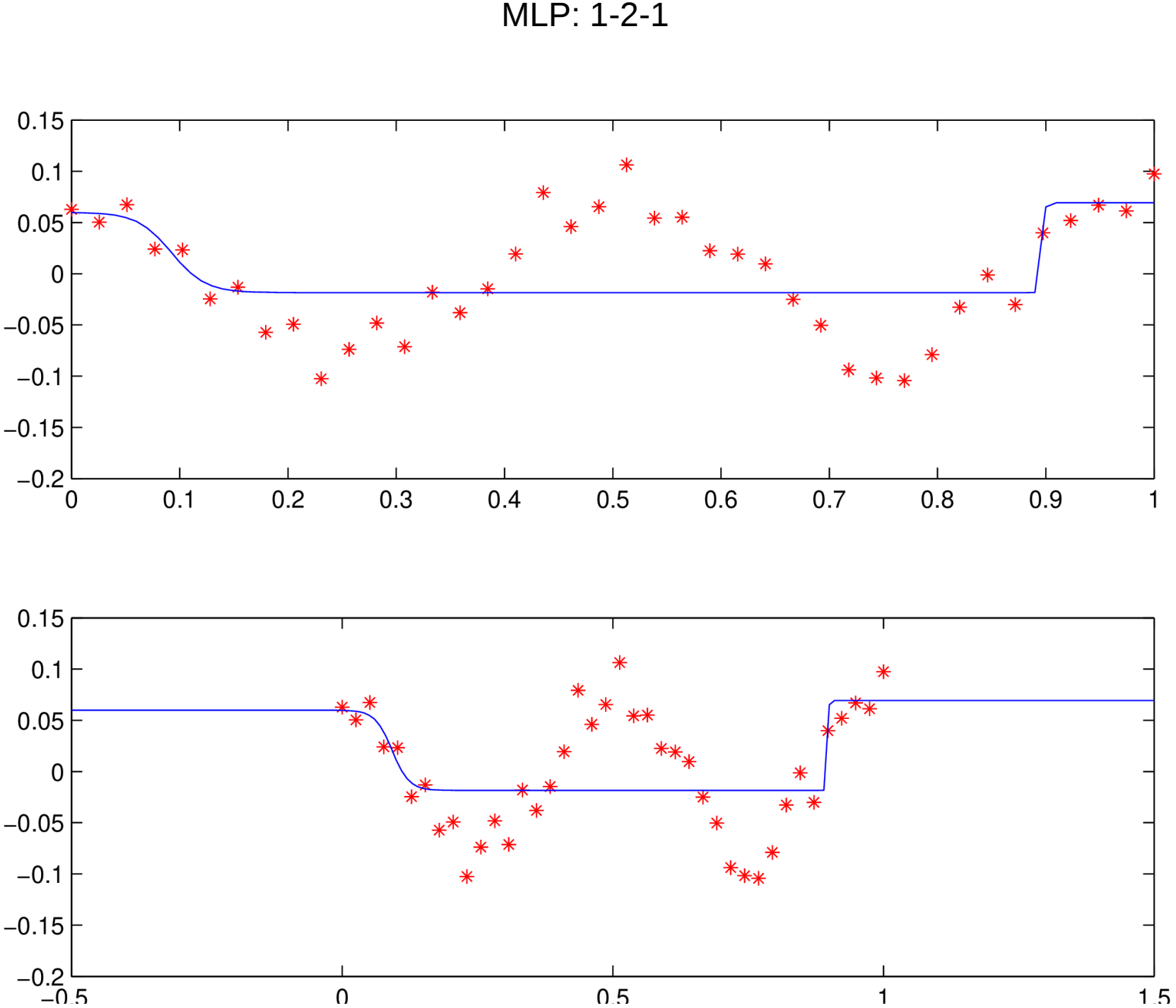
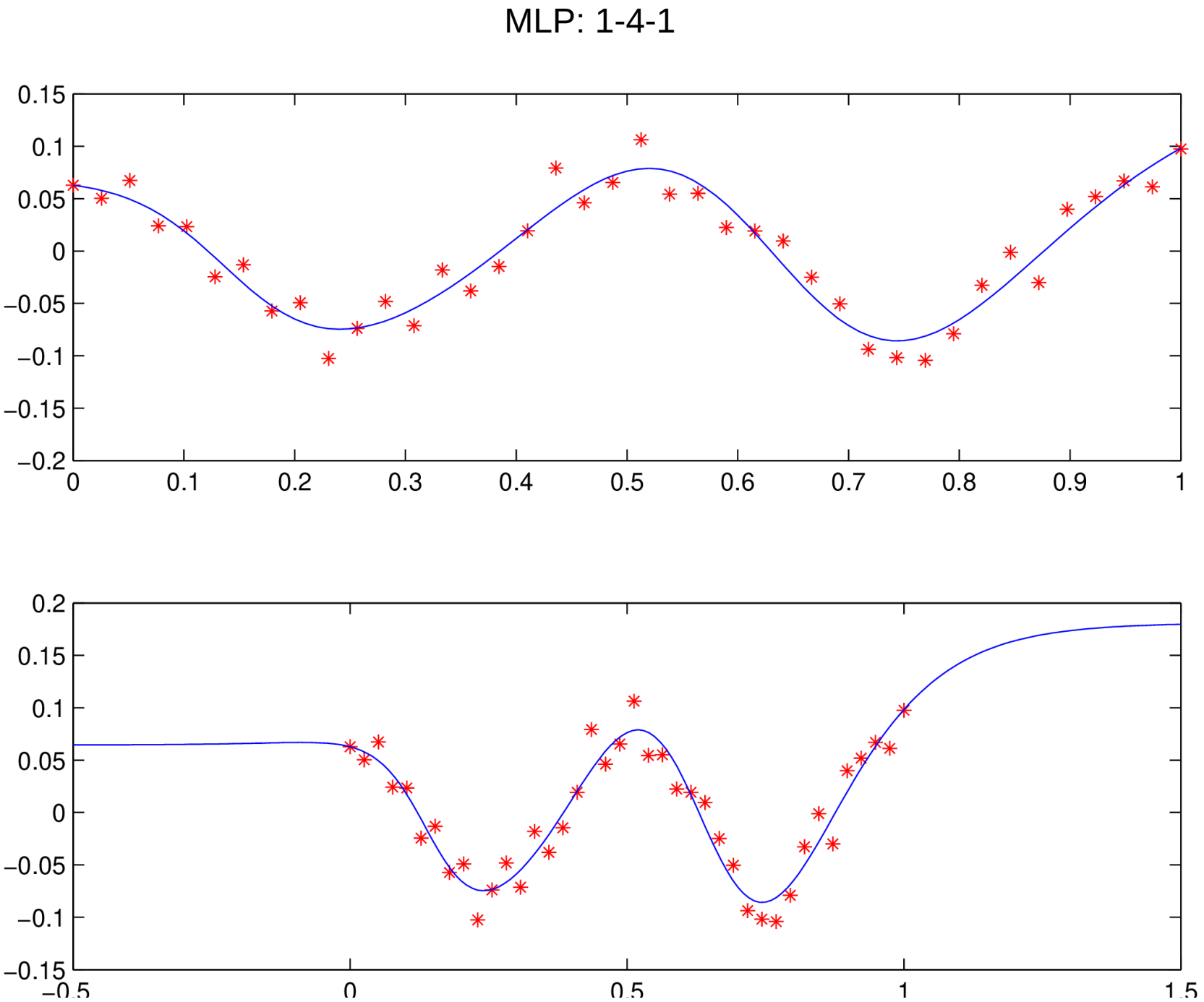
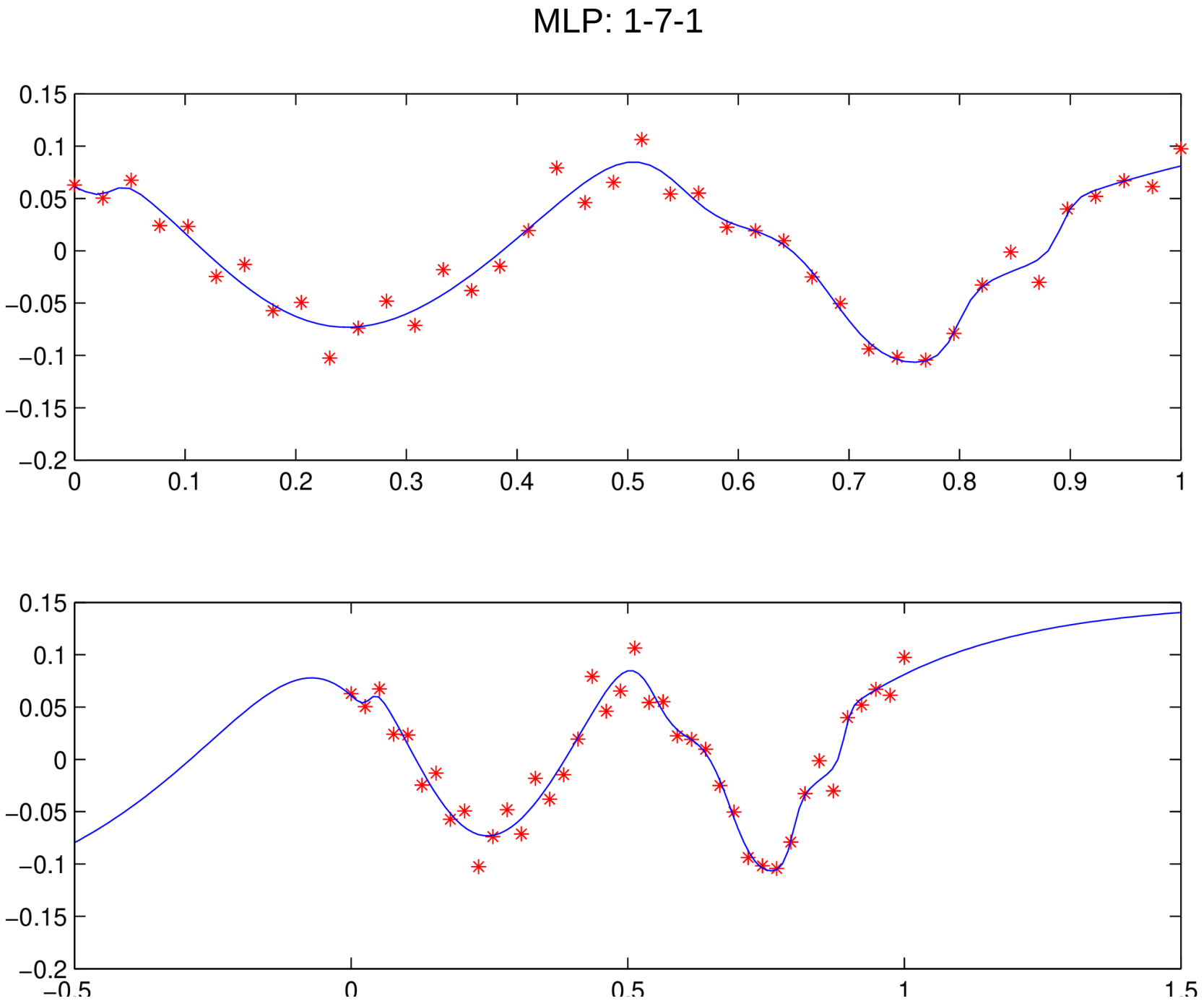
1D regression problem

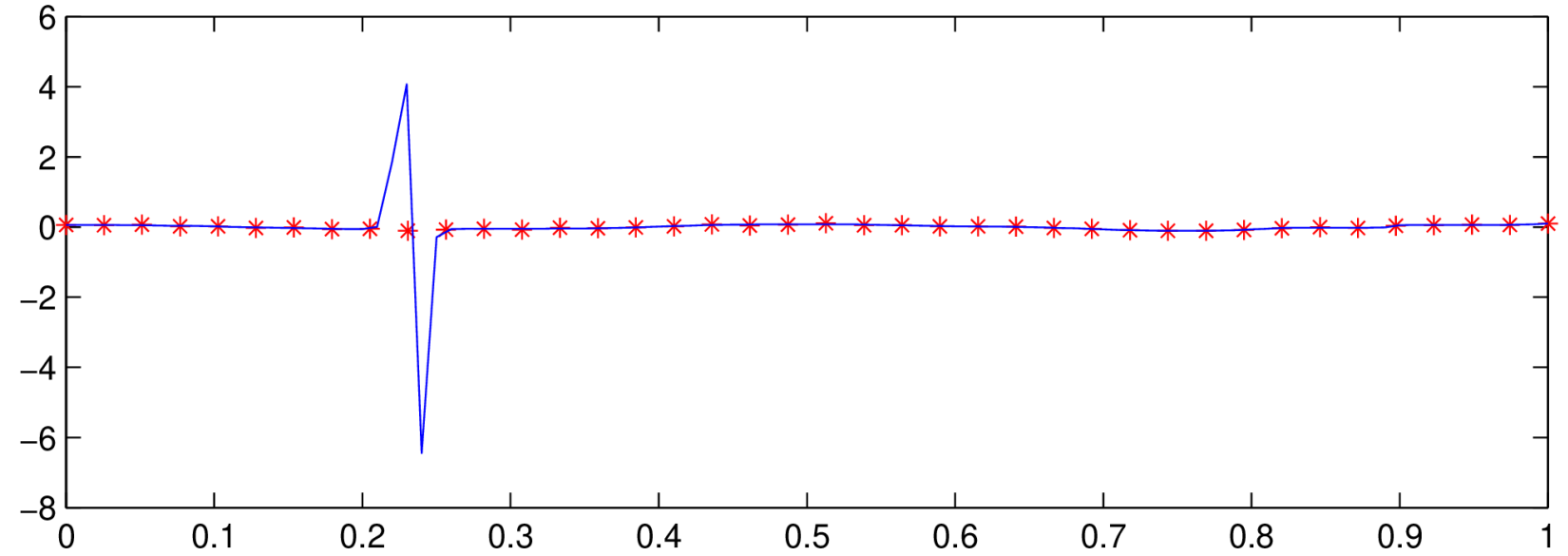

MLP: 1-1-1
0.15
0.1
0.05
0
−0.05
−0.1
−0.15
−0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.15
0.1
0.05
0
−0.05
−0.1
−0.15
−0.2
−0.5
0
0.5
1
1.5

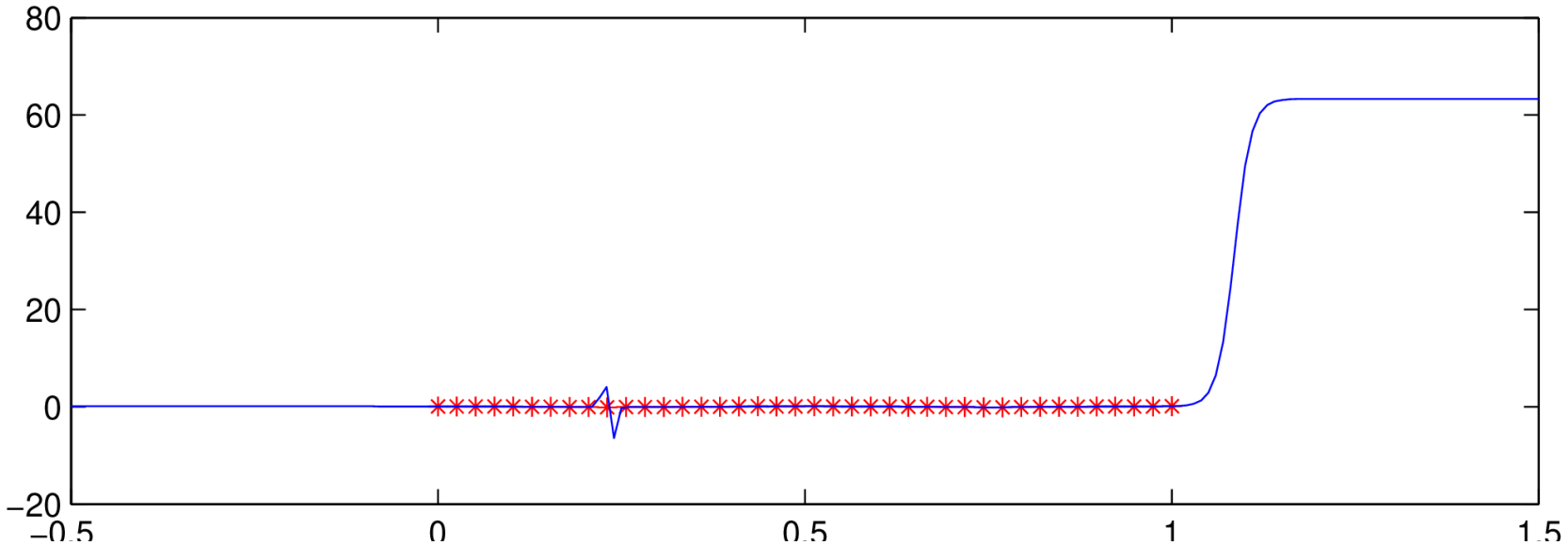

MLP: 1-2-1
0.15
0.1
0.05
0
−0.05
−0.1
−0.15
−0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.15
0.1
0.05
0
−0.05
−0.1
−0.15
−0.2
−0.5
0
0.5
1
1.5


MLP: 1-4-1
0.15
0.1
0.05
0
−0.05
−0.1
−0.15
−0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.2
0.15
0.1
0.05
0
−0.05
−0.1
−0.15
−0.5
0
0.5
1
1.5

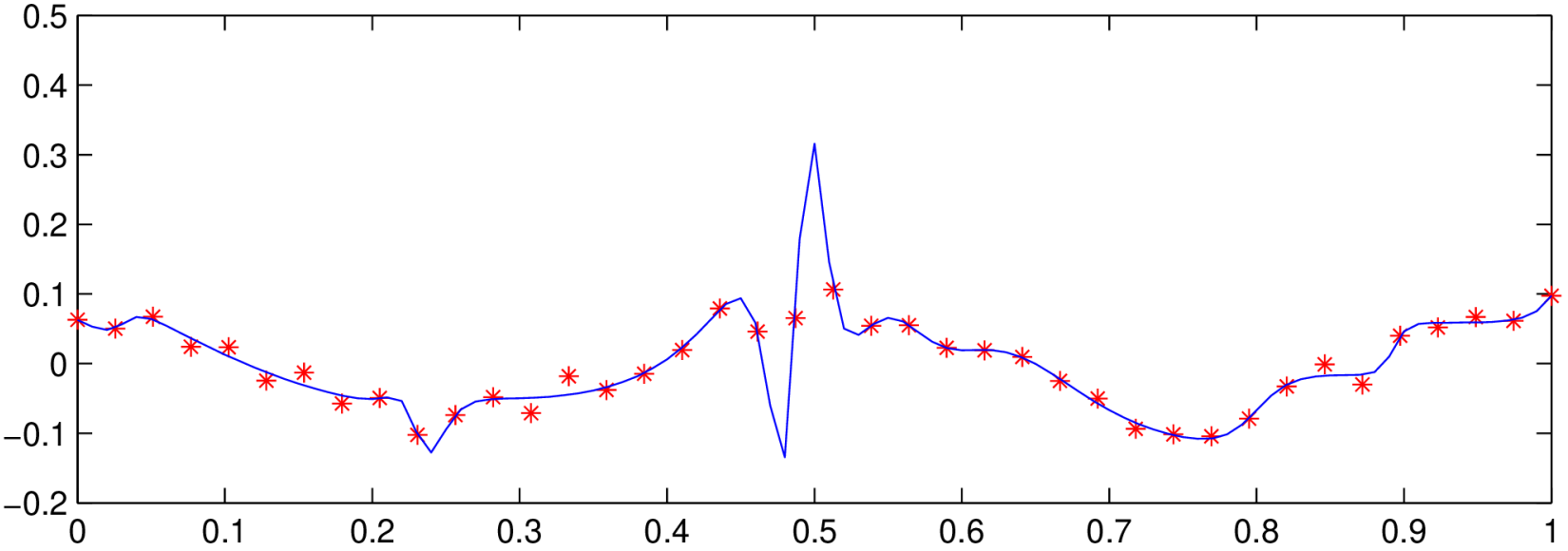

MLP: 1-7-1
0.15
0.1
0.05
0
−0.05
−0.1
−0.15
−0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.15
0.1
0.05
0
−0.05
−0.1
−0.15
−0.2
−0.5
0
0.5
1
1.5

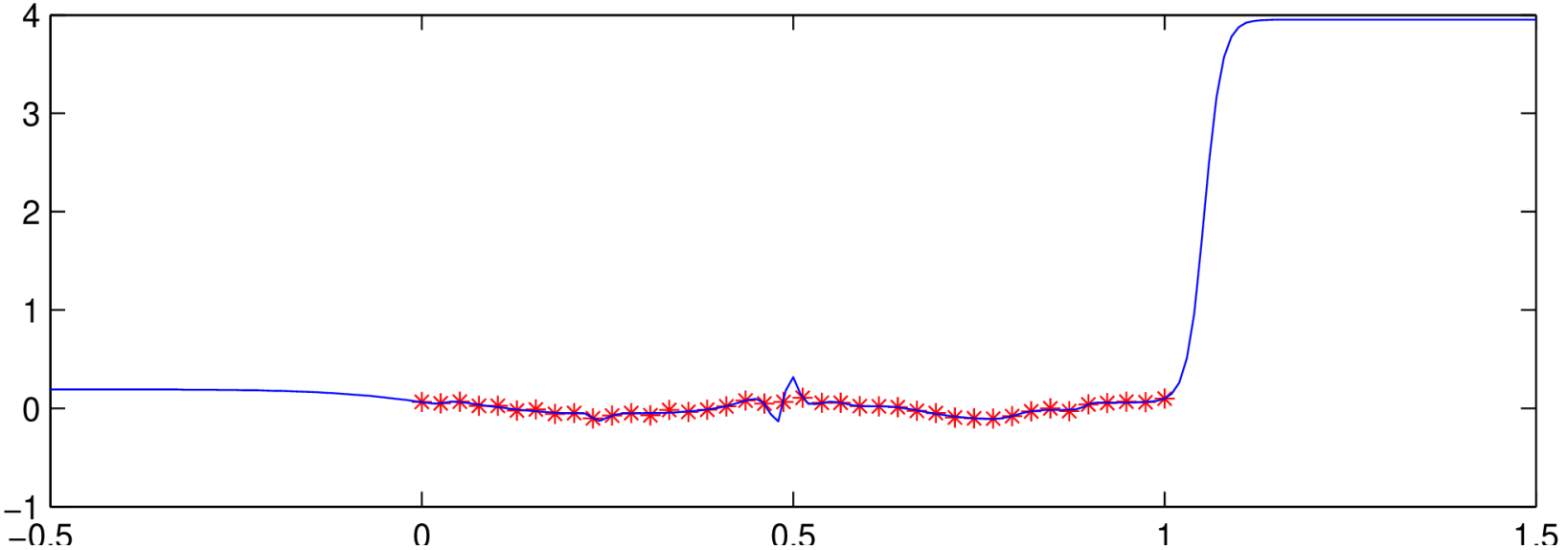
MLP: 1-9-1



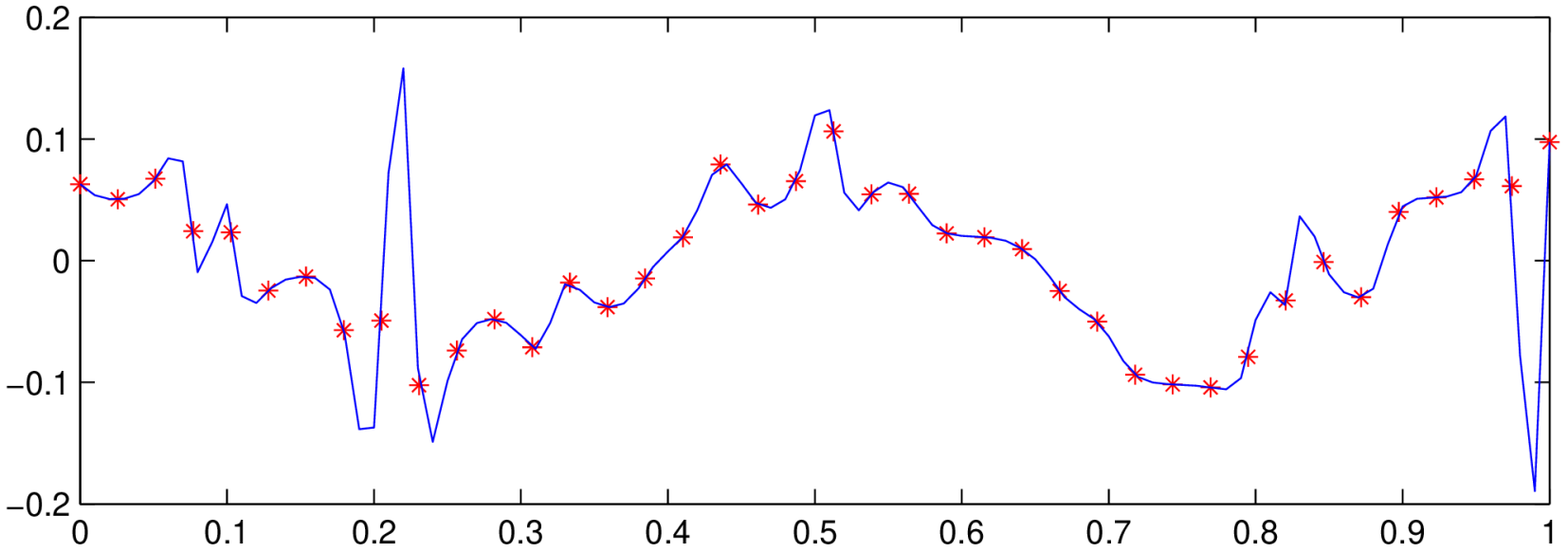


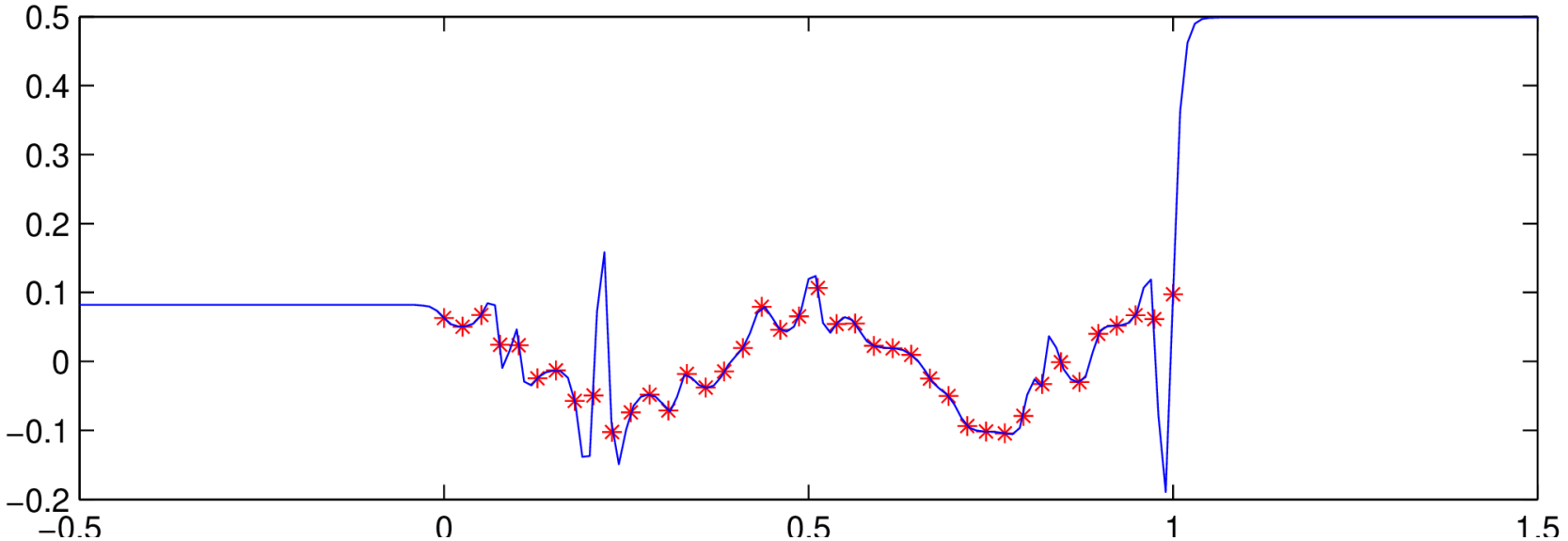
MLP: 1-12-1



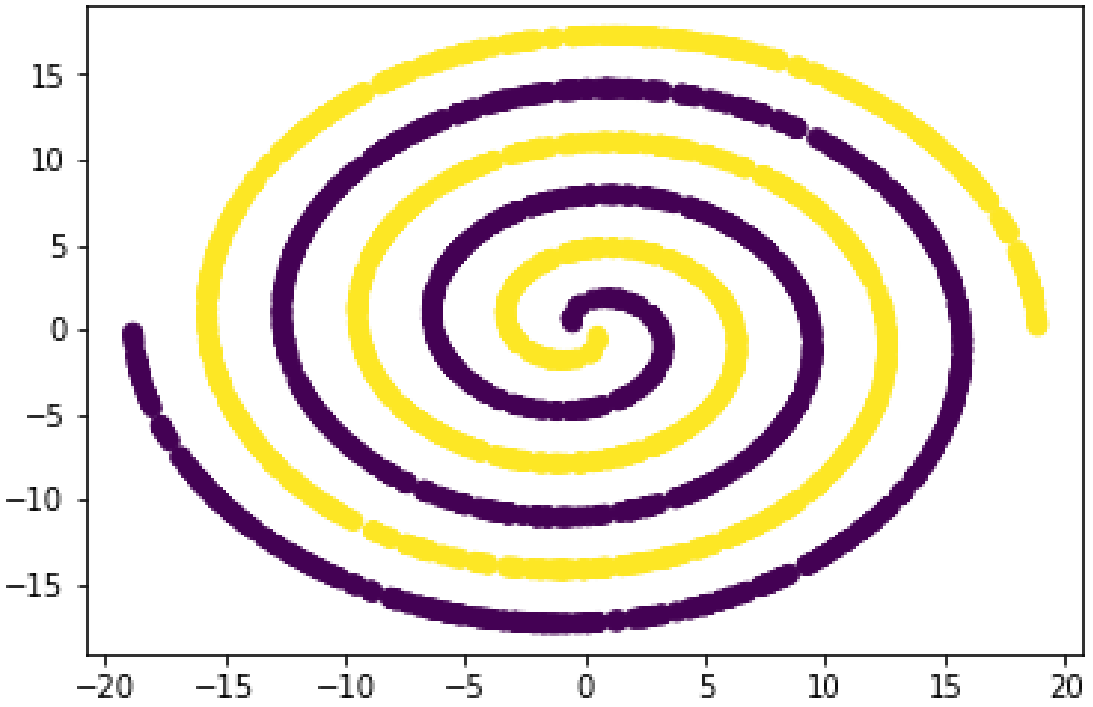


MLP: 1-25-1 (Zero training error)

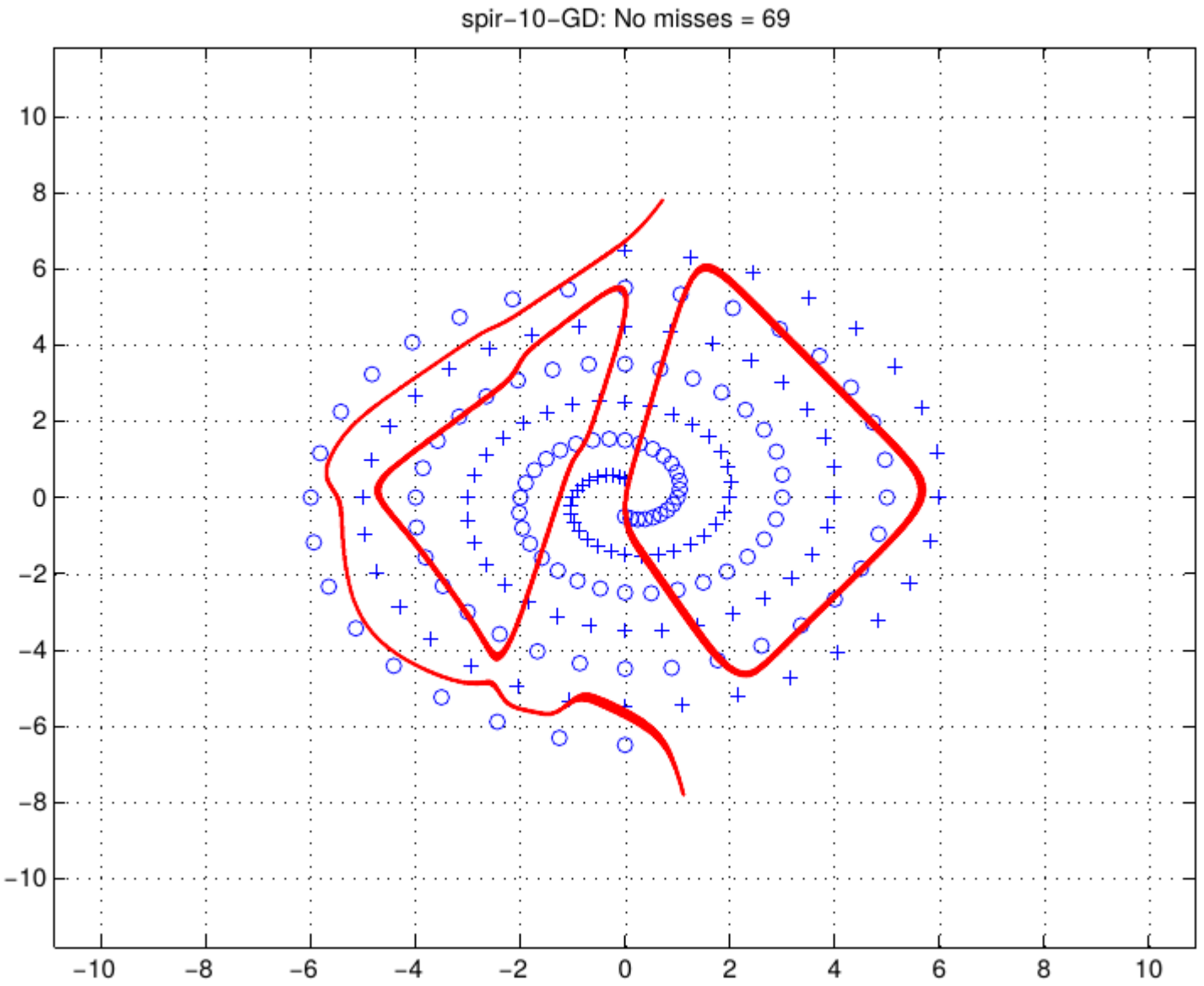




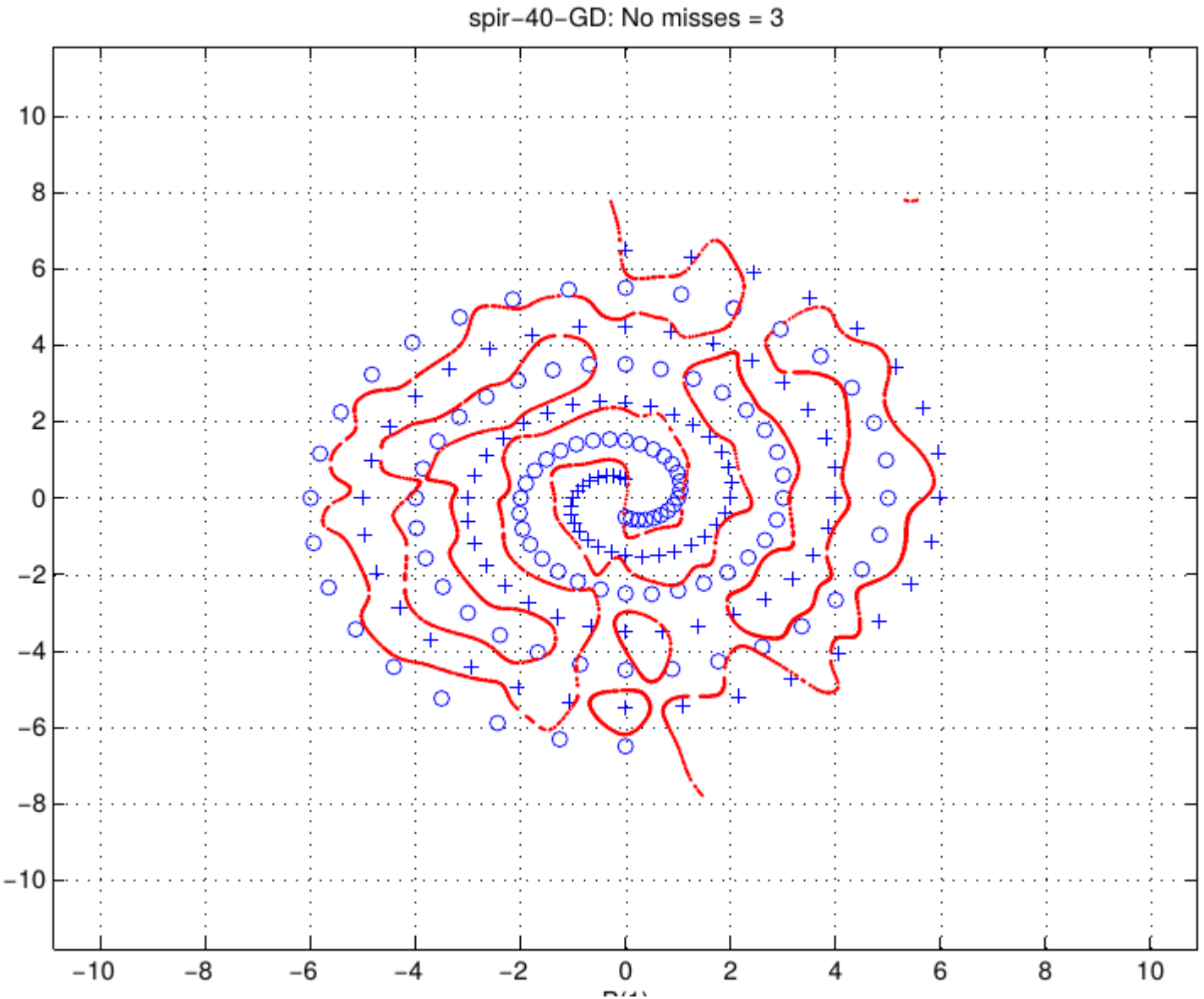
Binary classification problem
The spiral problem



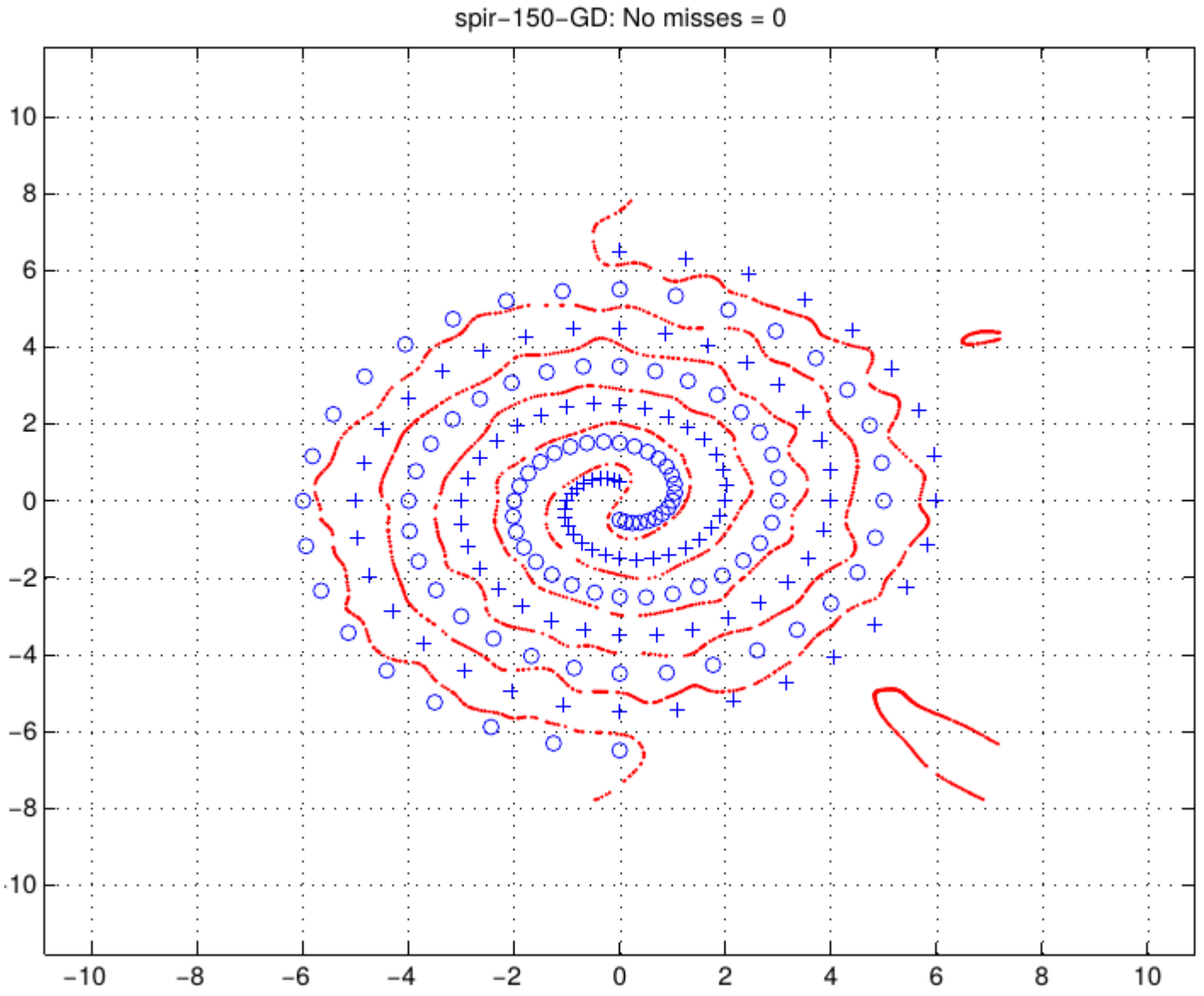
MLP: 2 – 10 – 1



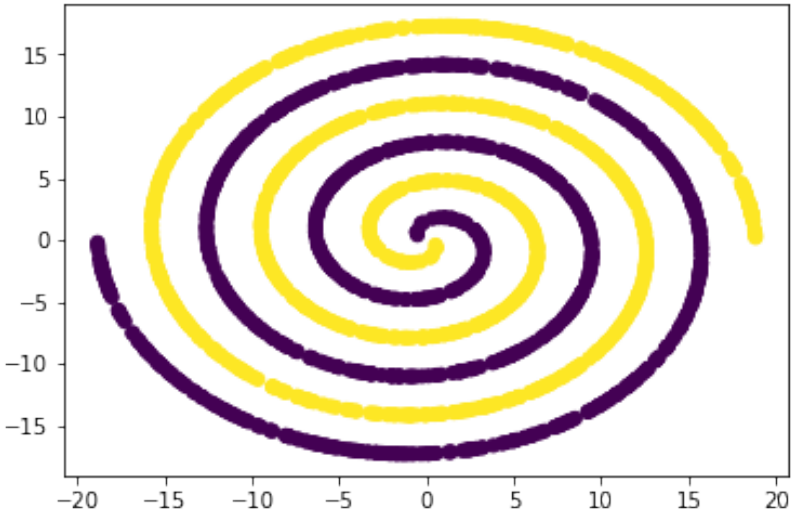
MLP: 2 – 40 – 1

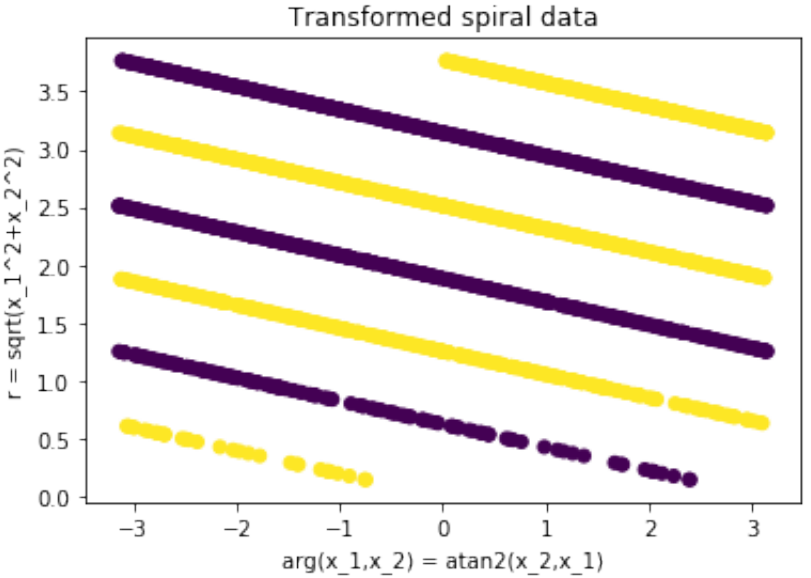


MLP: 2 – 150 – 1



Pre-processing helps

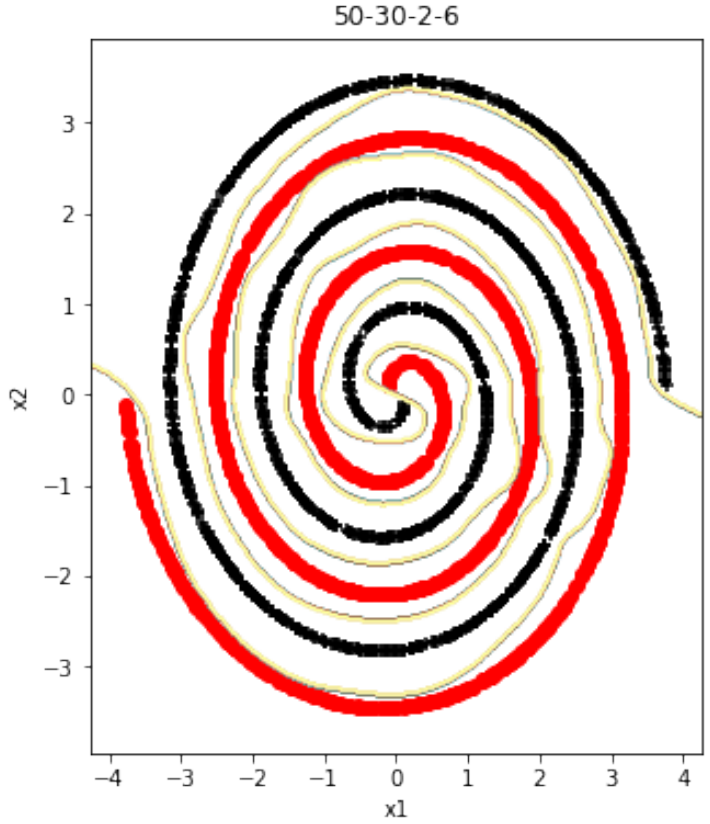


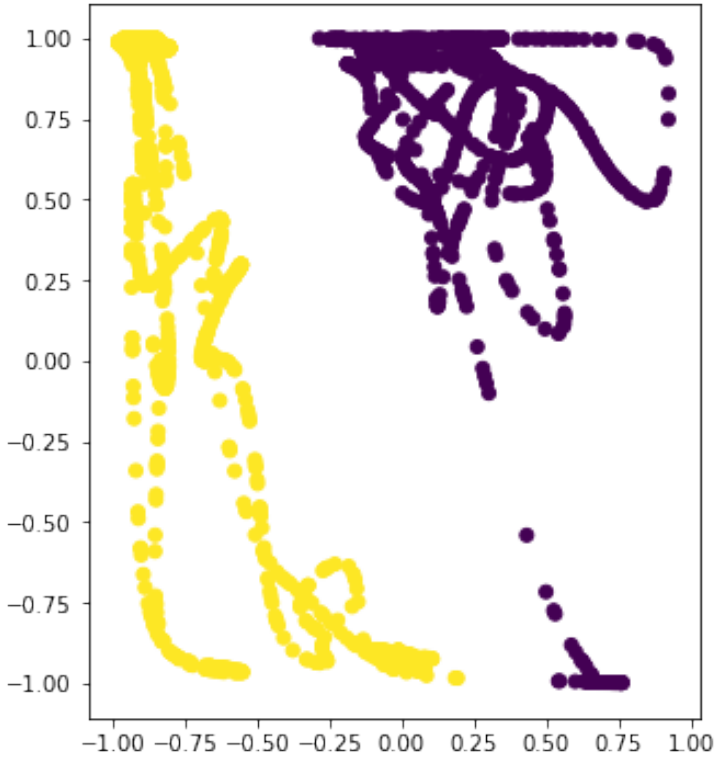


This can be solved by an MLP: 2 – 6 – 1

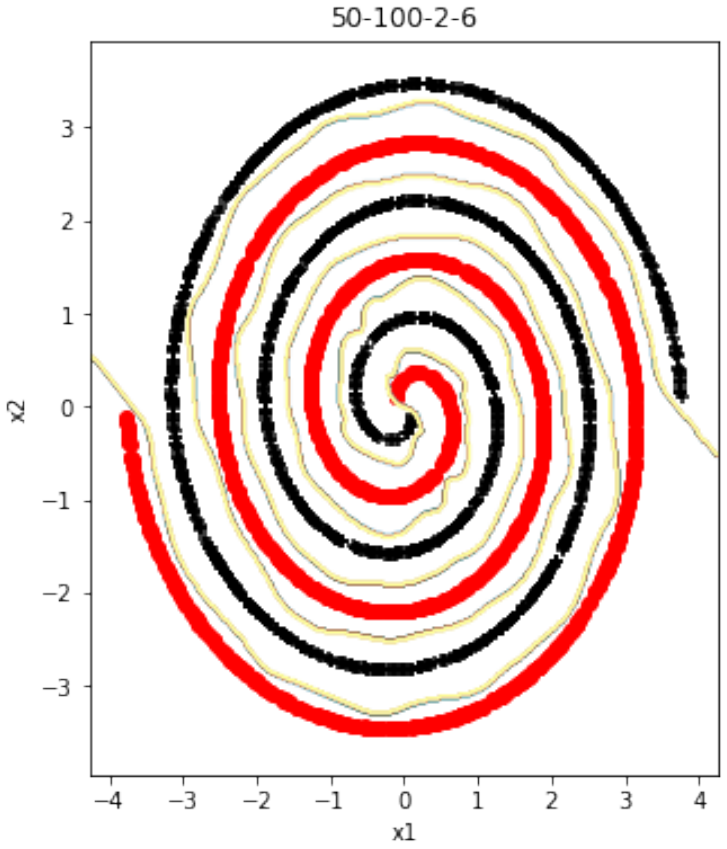
MLP: 2 – 50 – 30 – 2 – 6 – 1

Third hidden layer values

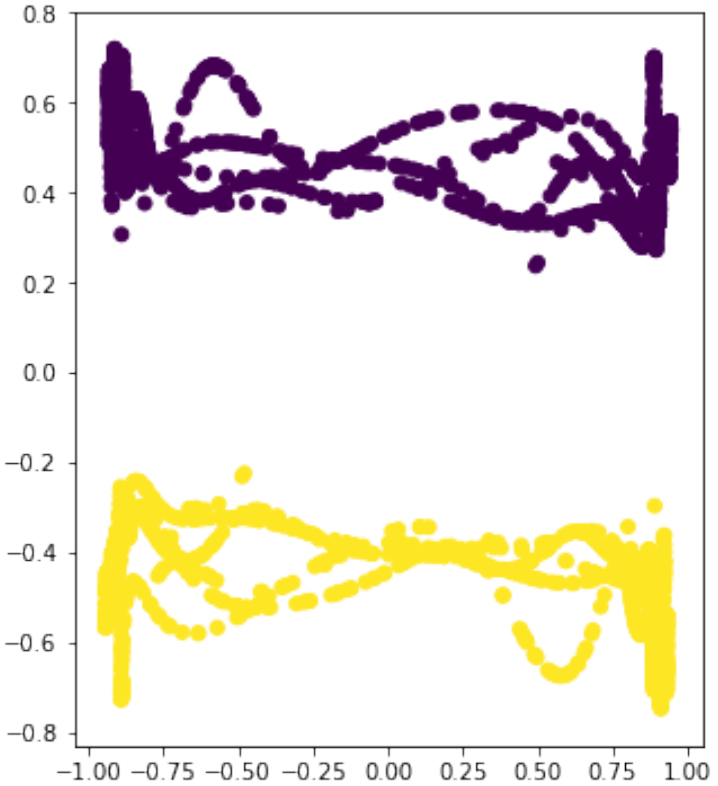




MLP: 2 – 50 – 100 – 2 – 6 – 1

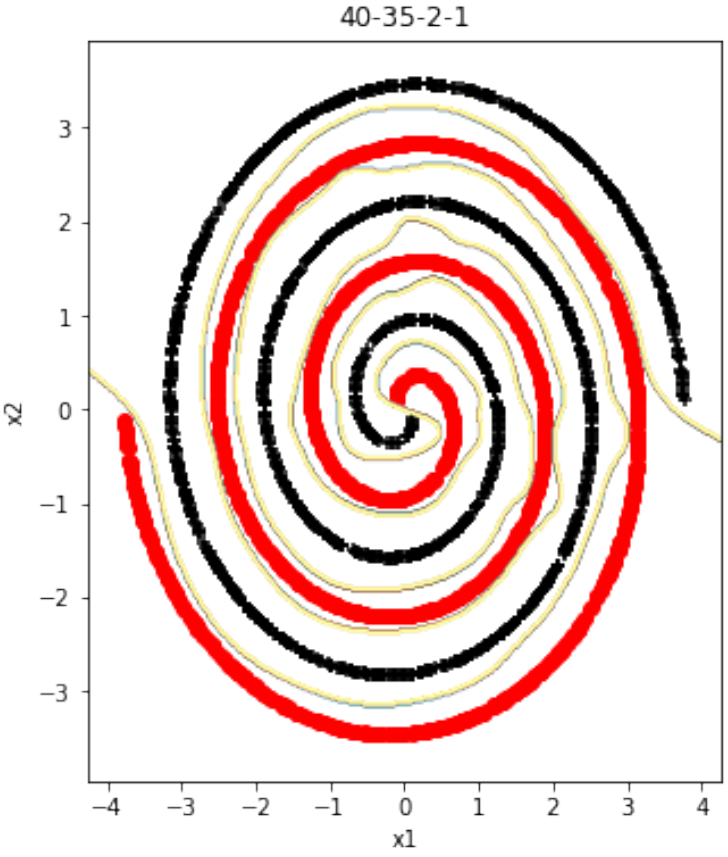


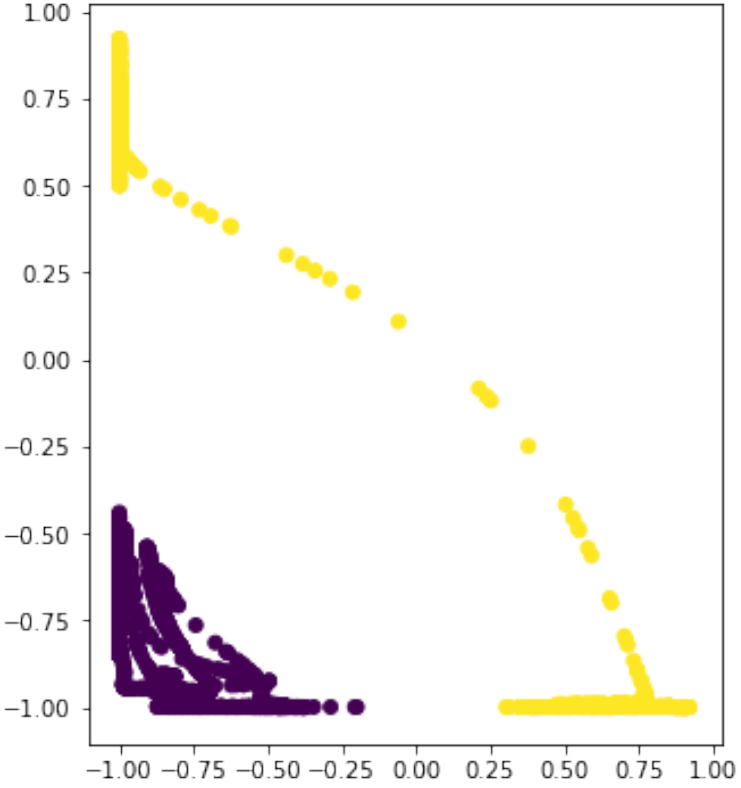
Third hidden layer values



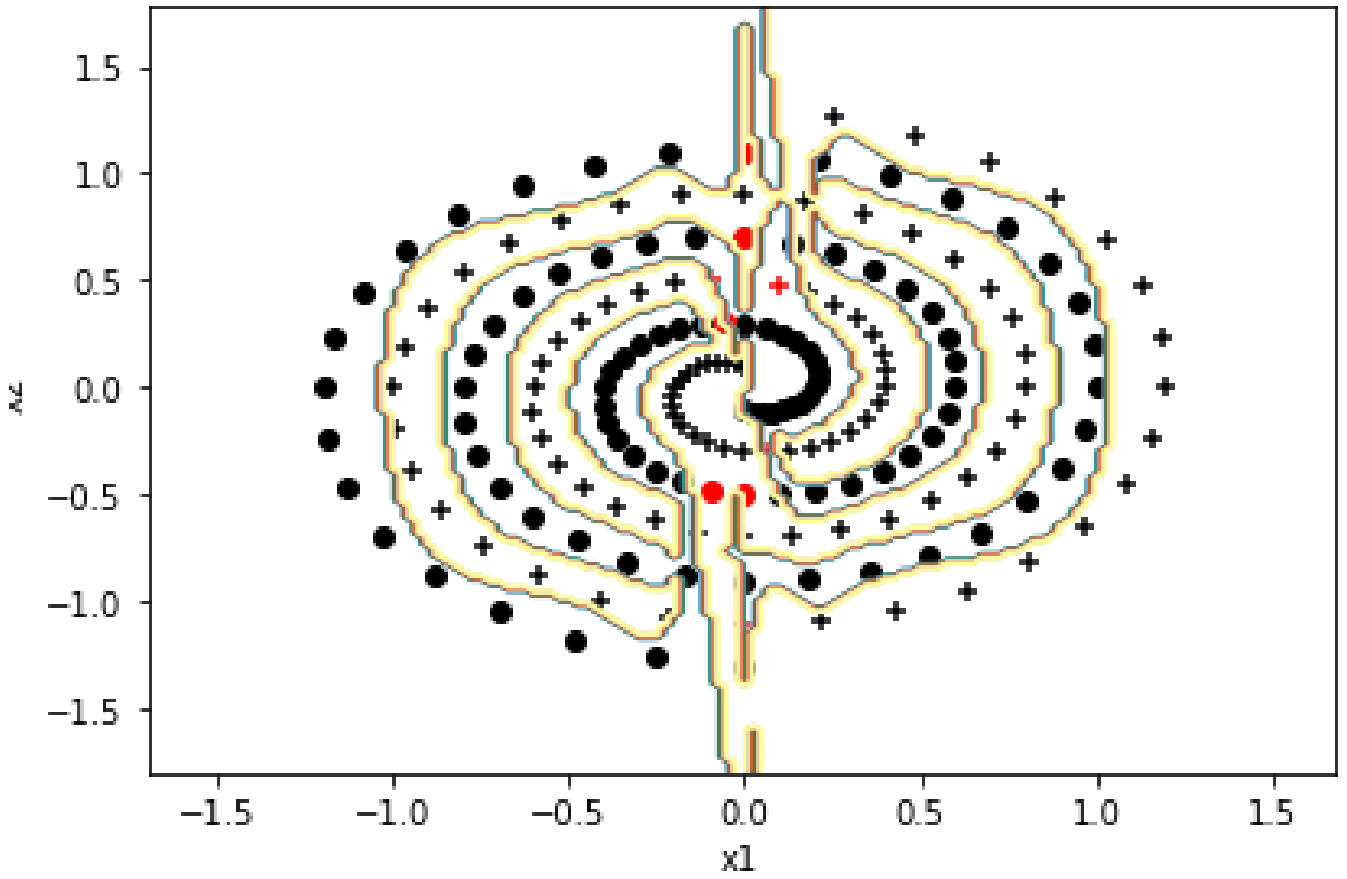
MLP: 2 – 40 – 35 – 2 – 1 – 1

Third hidden layer values



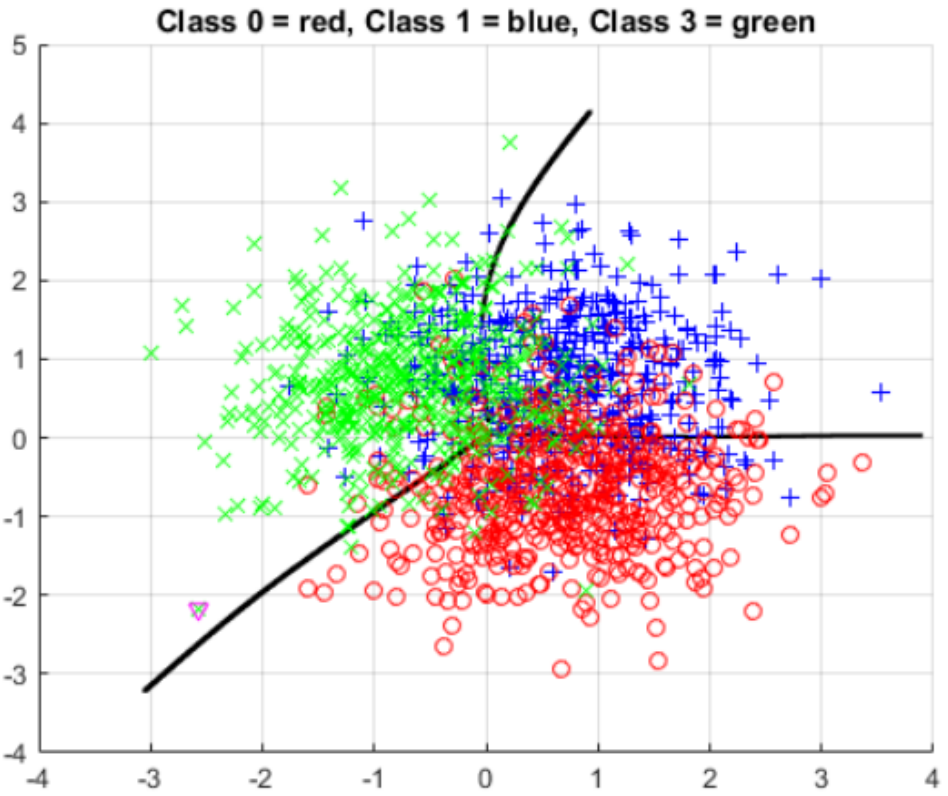


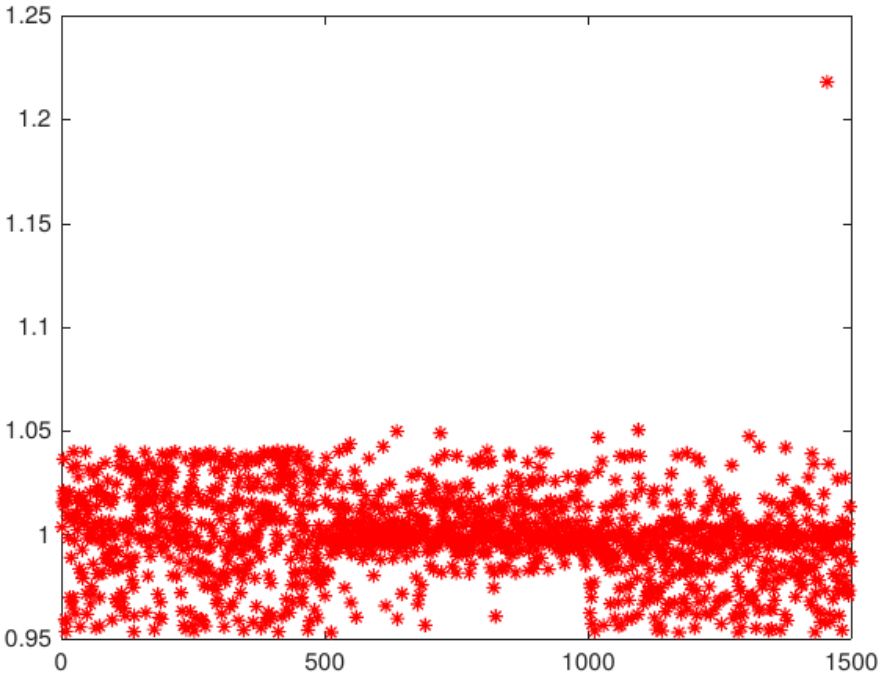
MLP: 2 – 5 – 5 – 5 – 1



Sum of output nodes

MLP: 2 – 4 – 3, 3-class problem

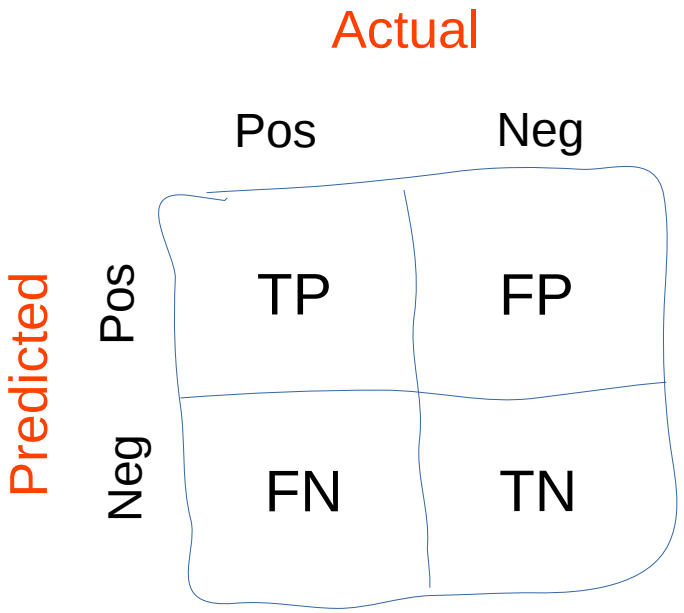




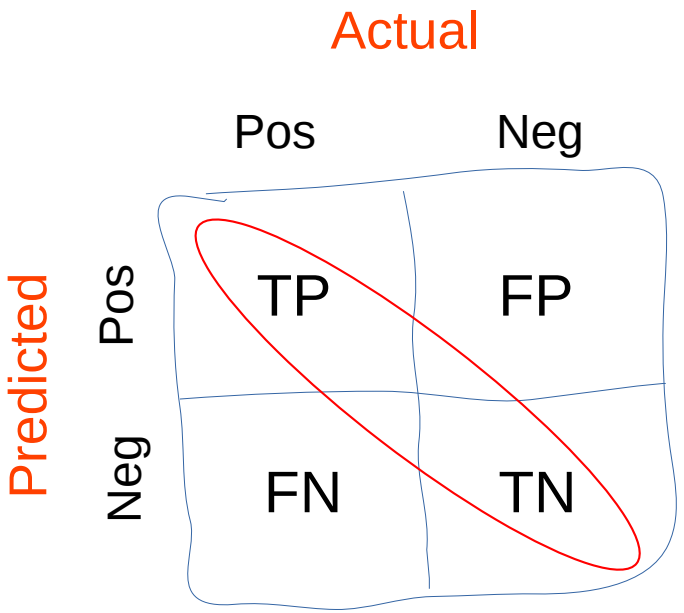
How to measure performance?

How to measure performance – binary classification problems

Confusion matrix

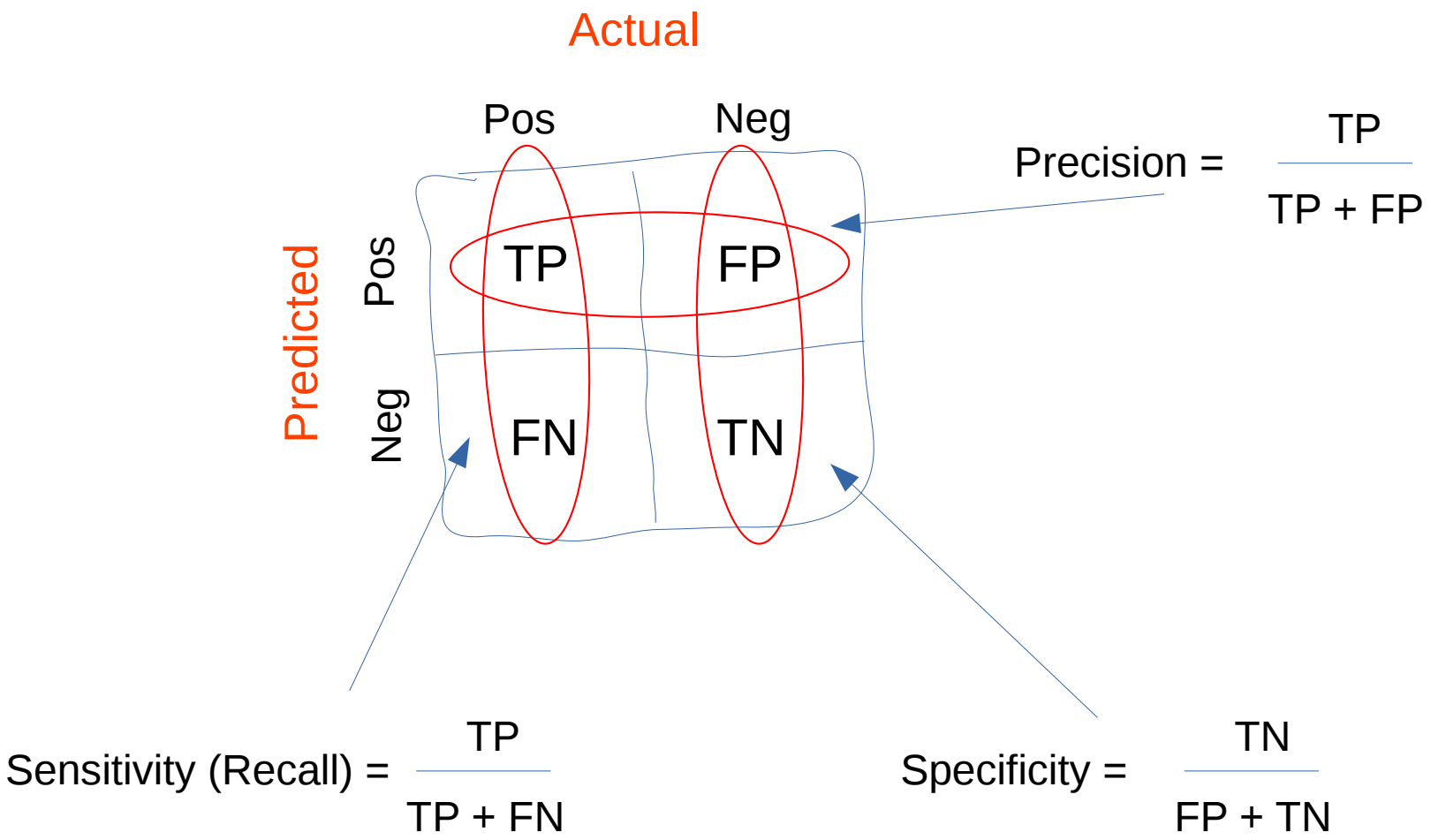


TP = True Positives
TN = True Negatives
FP = False Positives
FN = False Negatives

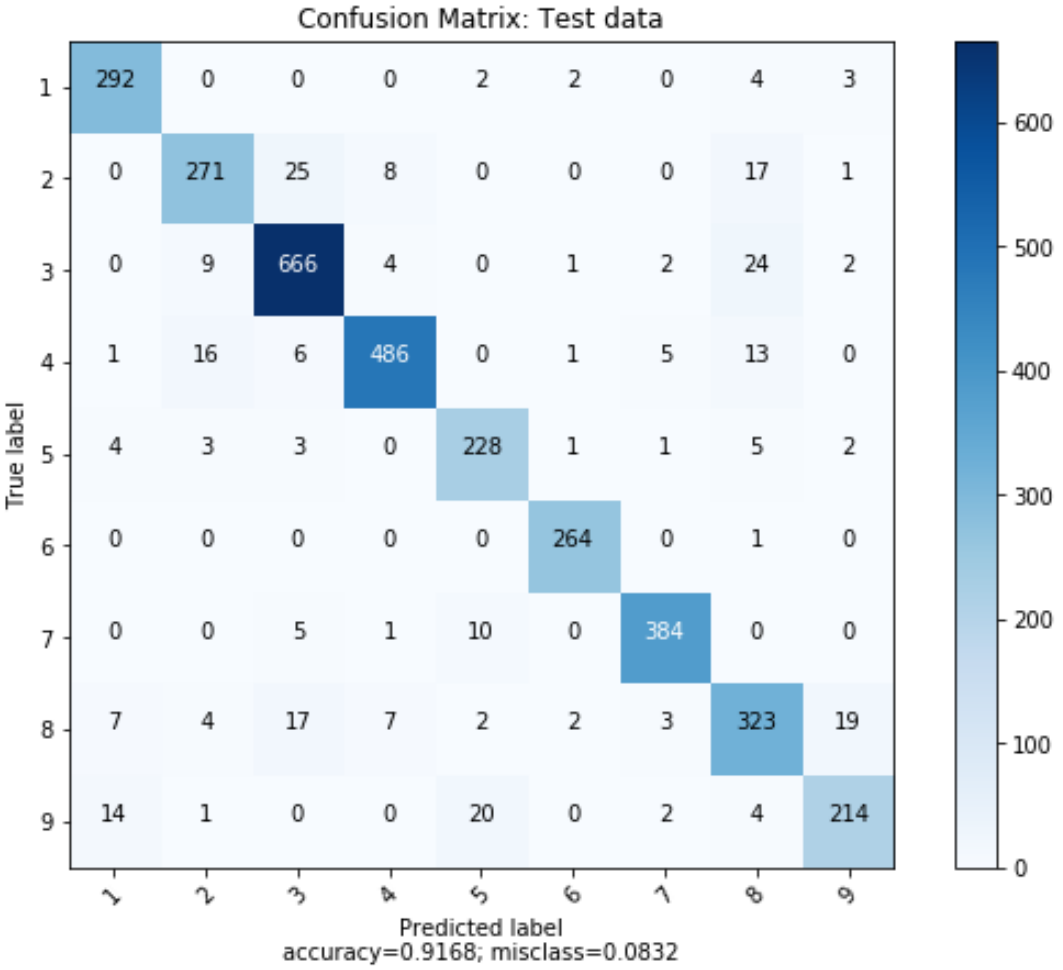


$$\text{Accuracy} = \frac{TP + TN}{TP+FP+FN+TN}$$

Why can accuracy be misleading?


Actual
Pos
Neg
Predicted
Pos
Neg
TP
FP
FN
TN
Precision = TP / TP + FP
Sensitivity (Recall) = TP / TP + FN
Specificity = TN / FP + TN

Confusion matrix, also for many classes



For binary classification problems it is common to use the Receiver Operating Characteristics (ROC) curve and the area under it (AUC)

To make a decision for a class, we need a cut value (C)

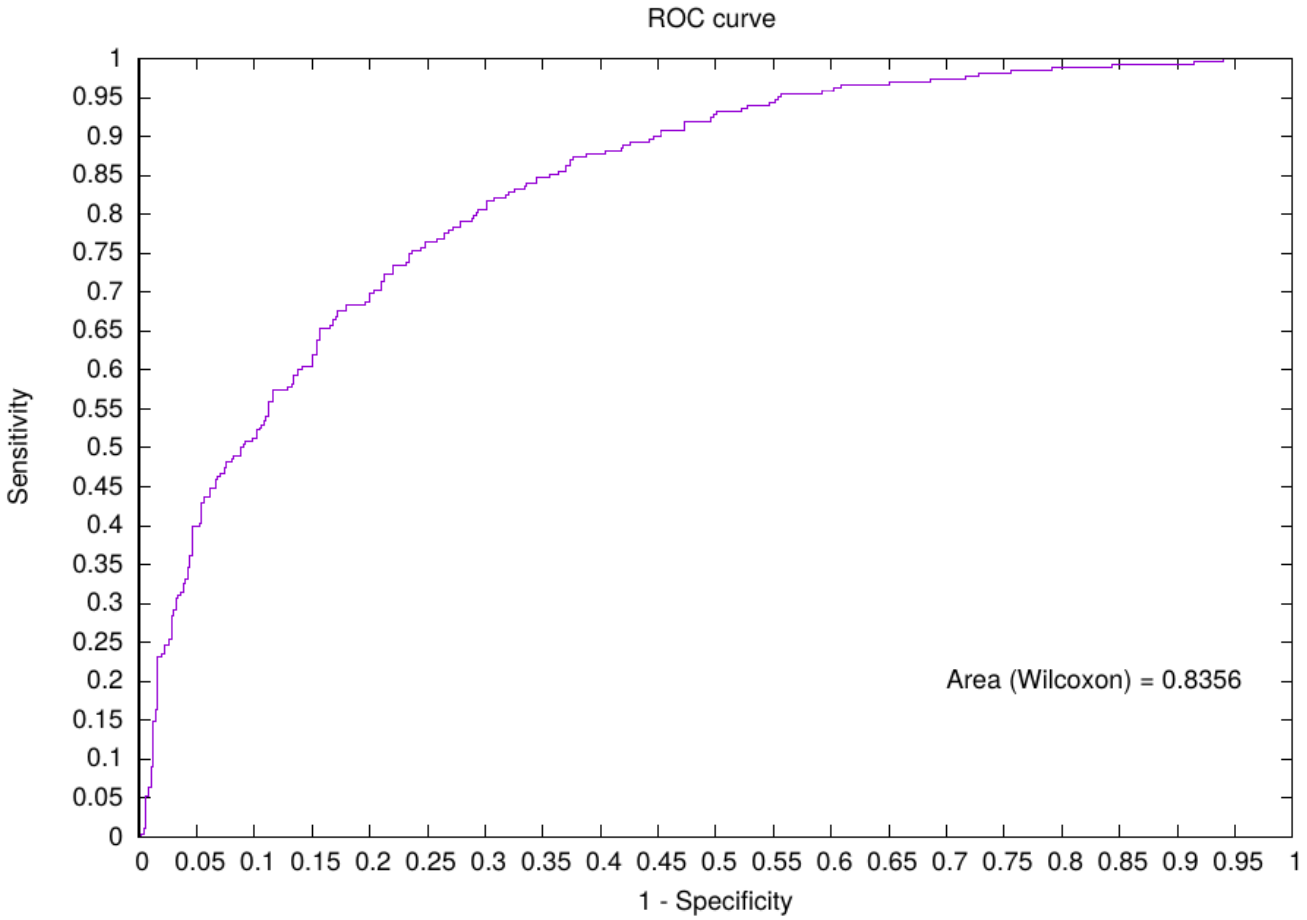
$$\left\{ \begin{array}{lll} \text{class 1 (pos)} & \text{if} & y(\mathbf{x}) > \mathrm{C} \\ \text{class 0 (neg)} & \text{if} & y(\mathbf{x}) \leq \mathrm{C} \end{array} \right\}$$

For each C we get a Sensitivity / Specificity pair

Vary C between [0,1] and plot all Sens vs (1-Spec)

This is the ROC curve!

An example



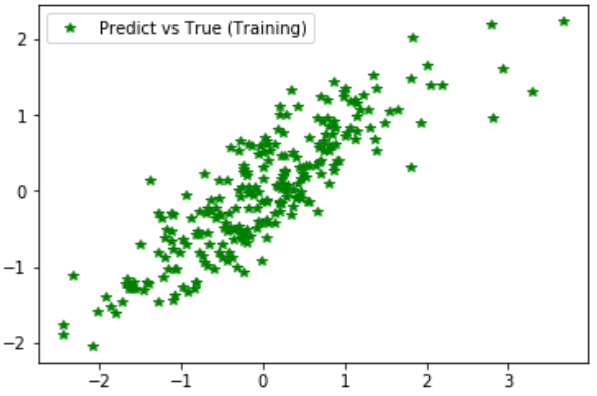
What does the area mean?

How to measure performance – regression problems

RMSE $$E = \sqrt{\frac{1}{N}\sum_{n}^{N}\left\|\mathbf{d}_n - \mathbf{y}(\mathbf{x}_n, \boldsymbol{\omega}^*)\right\|^2}$$

Normalized MSE $$E = \frac{\sum_n \left\|\mathbf{d}_n - \mathbf{y}(\mathbf{x}_n, \boldsymbol{\omega}^*)\right\|^2}{\sum_n \left\|\mathbf{d}_n - < \mathbf{d} >\right\|^2}$$

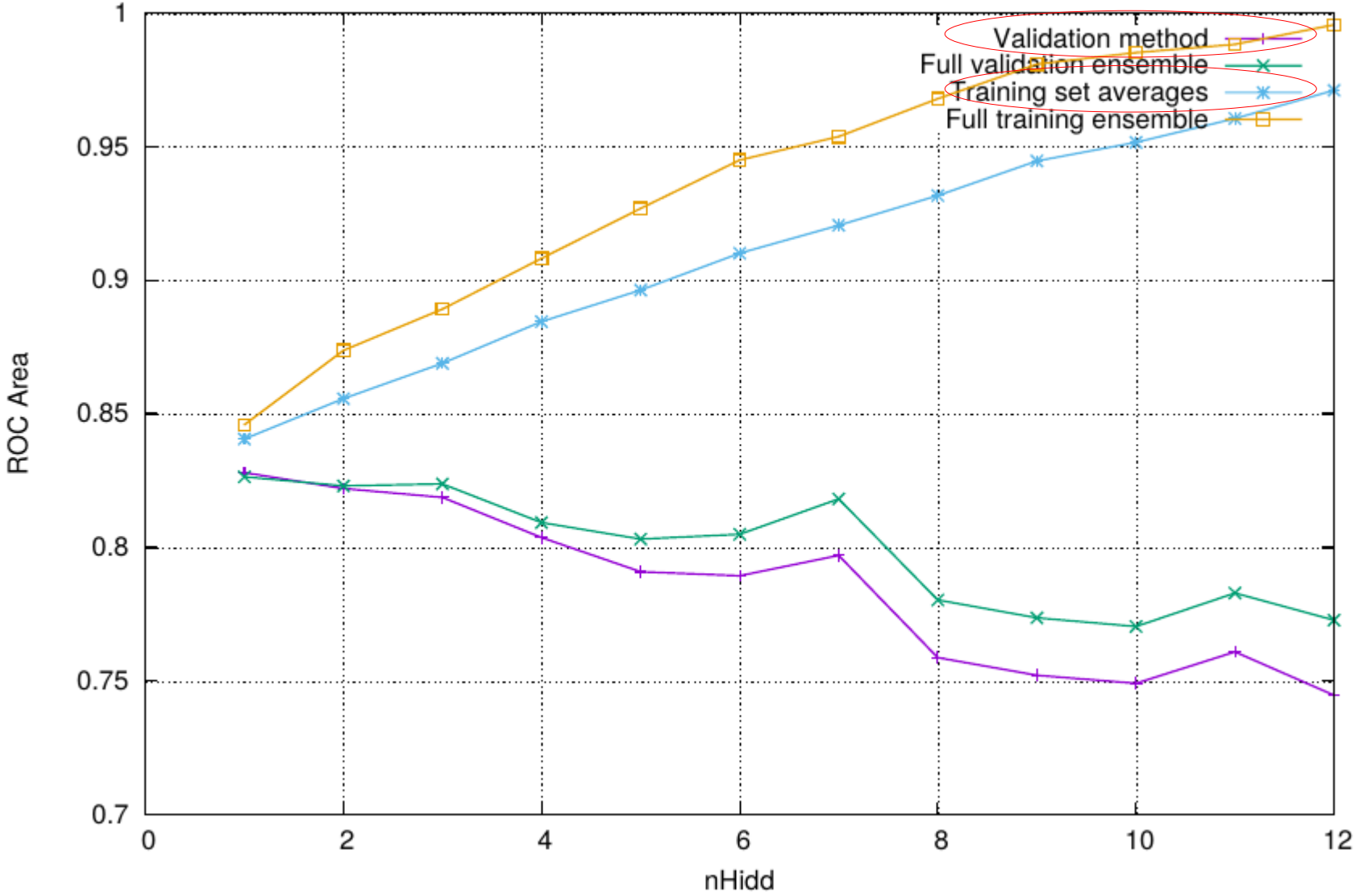
Scatterplot
True vs. predicted



Compute correlation!

Model selection, examples

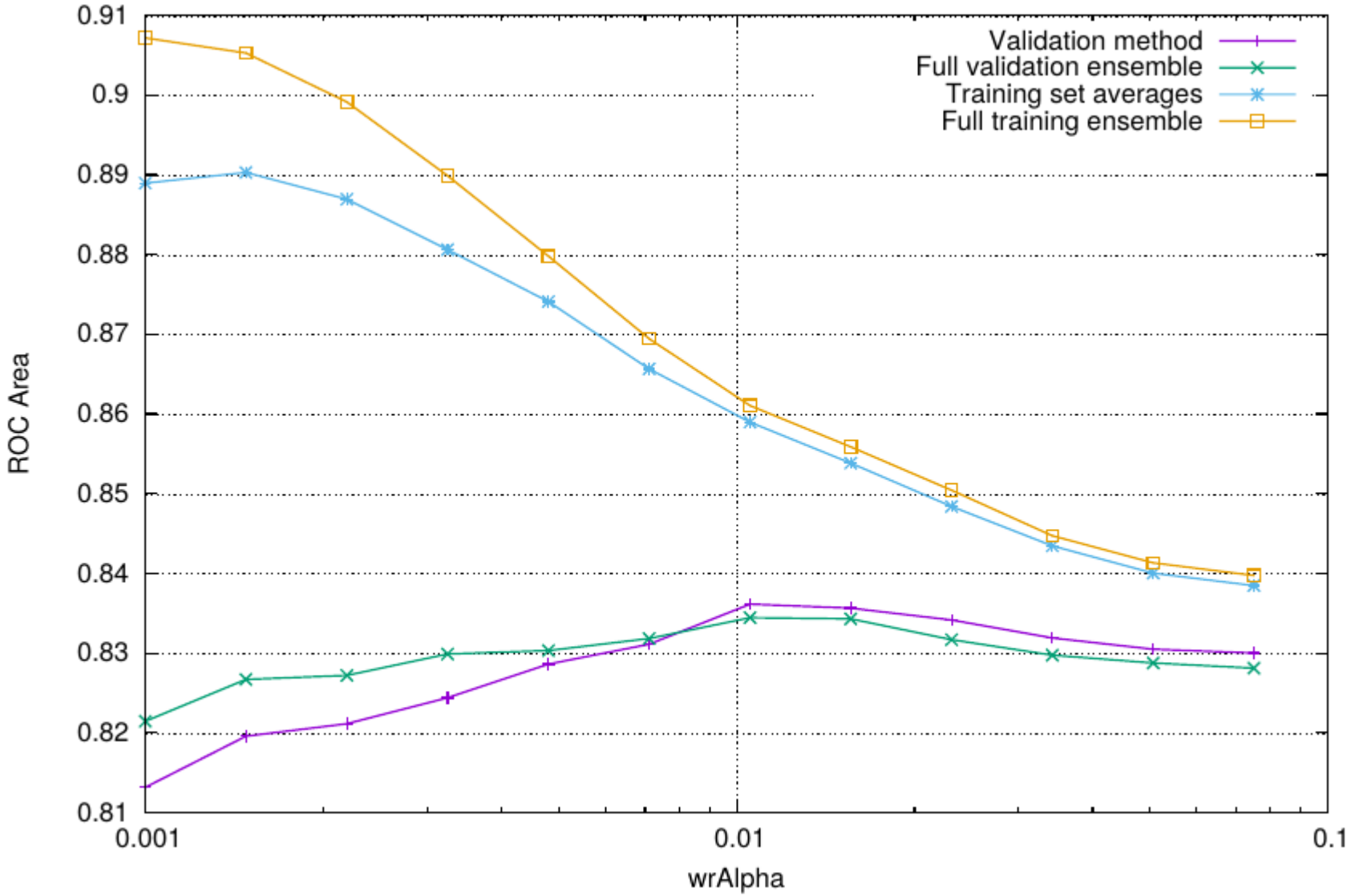
Binary classification problem:
Y-axis: ROC area
X-axis: Number of hidden nodes



Best validation area: 0.83

MLP: 5 hidden nodes!

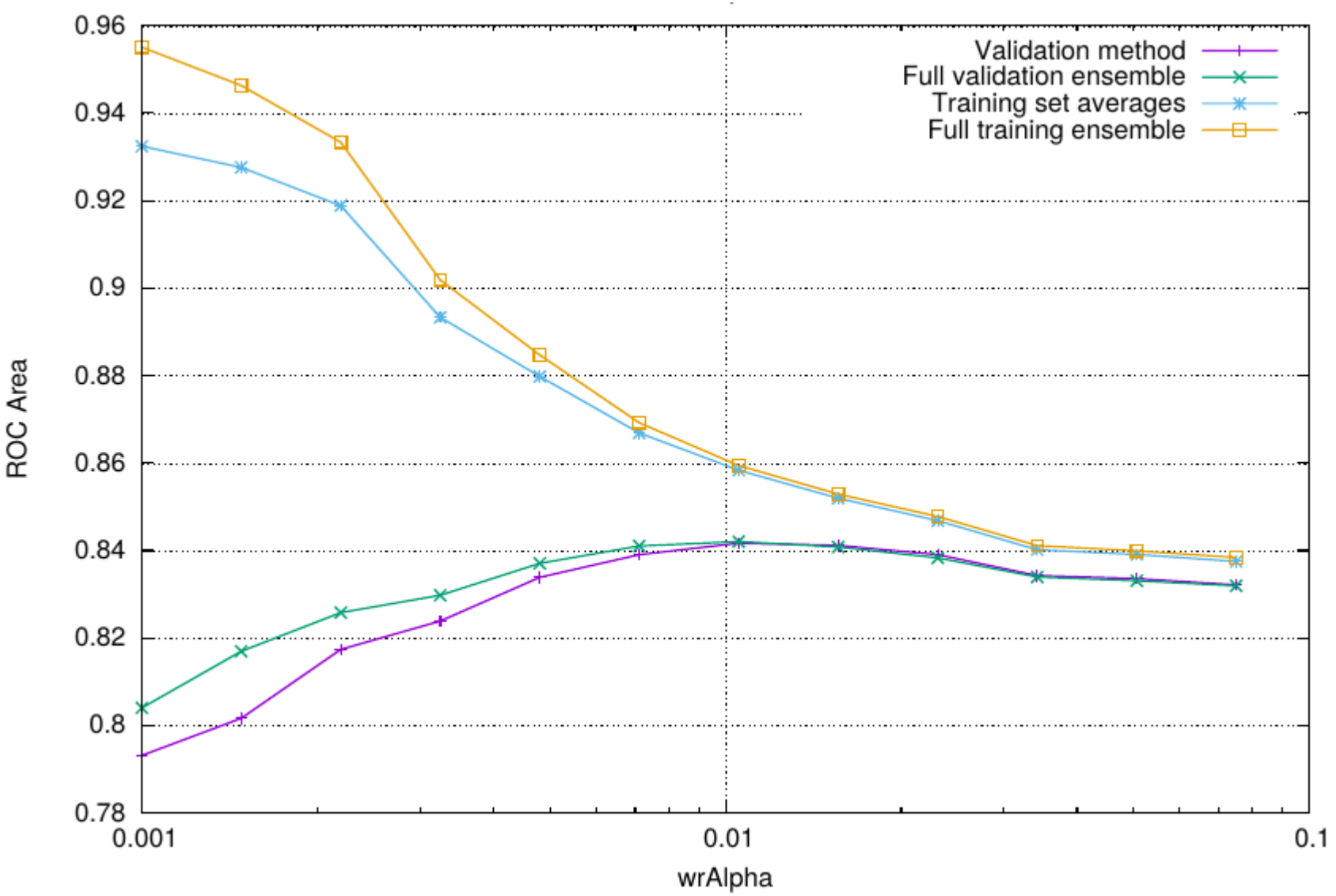
Same problem:
Y-axis: ROC area
X-axis: L2 regularization parameter



Best validation area: 0.835

MLP: 10 hidden nodes!

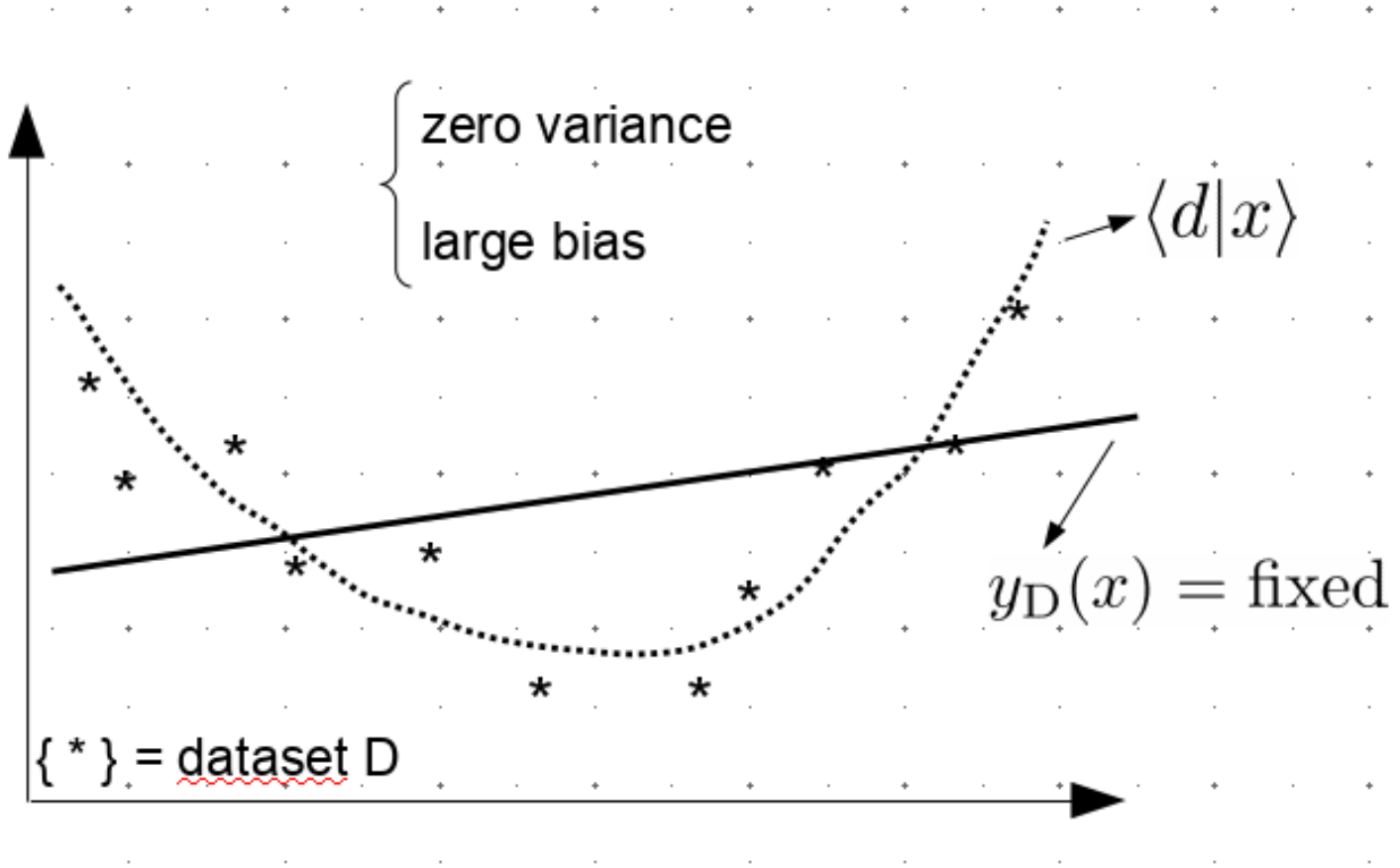
Same problem:
Y-axis: ROC area
X-axis: L2 regularization parameter

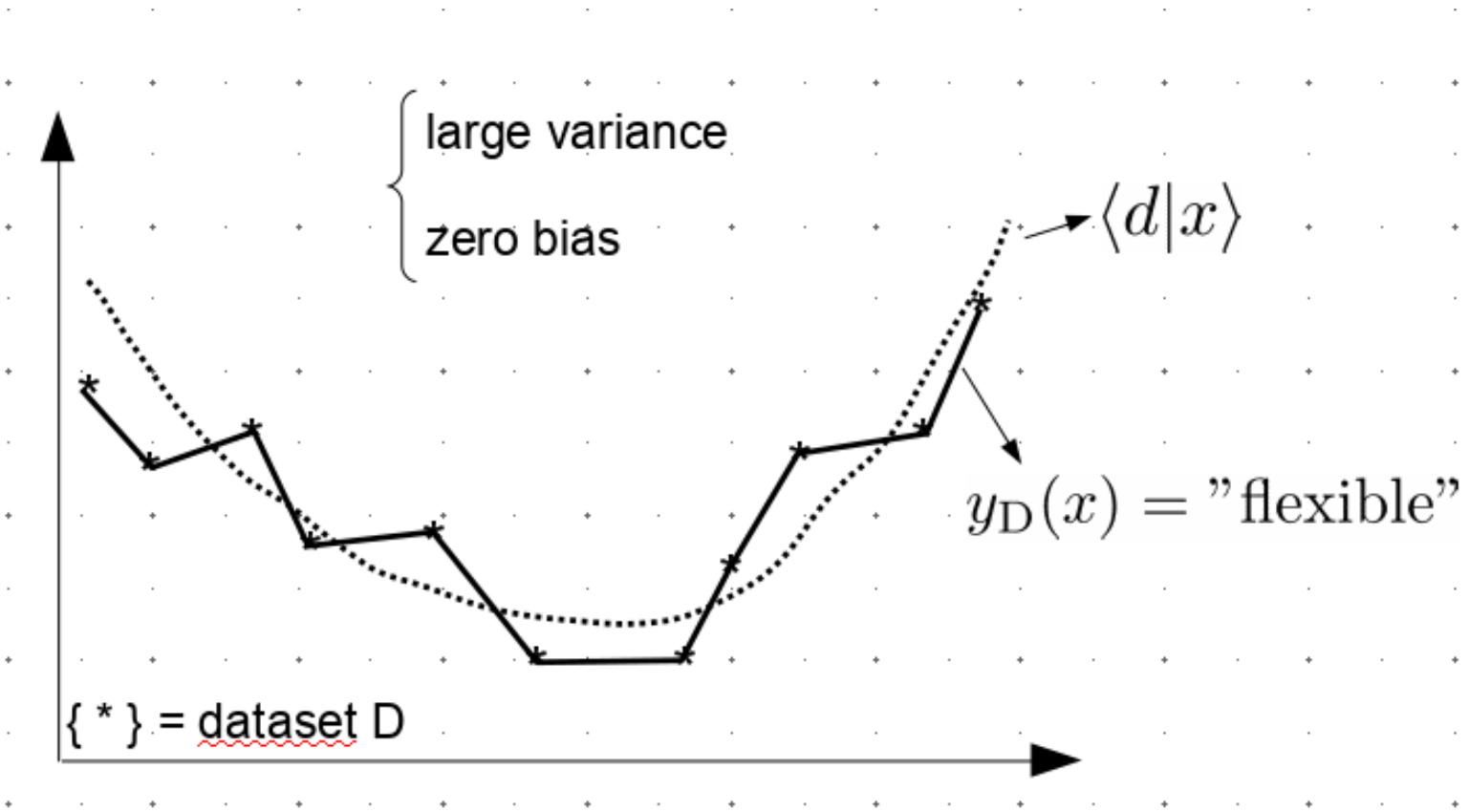


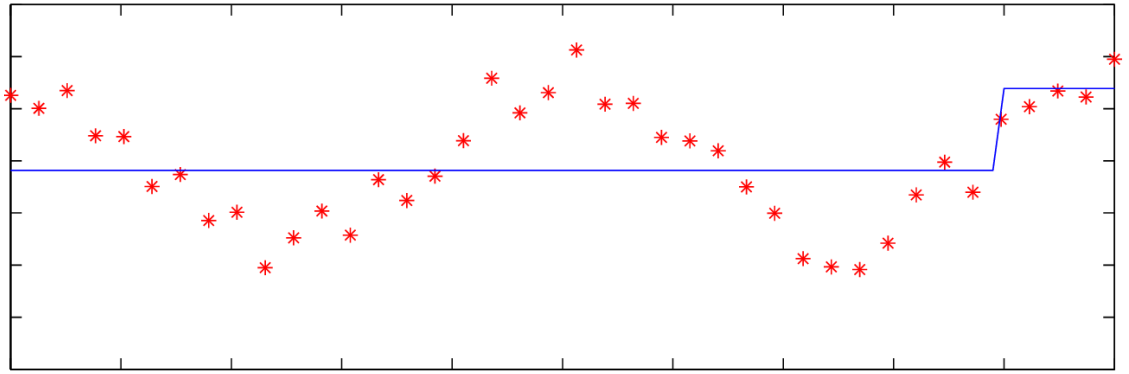
Best validation area: 0.84

Remember the bias variance tradeoff

$$\mathrm{E}\left[(y_{\mathcal{D}}(\mathbf{x}) - \langle d|\mathbf{x}\rangle)^2\right] = \underbrace{\left(\mathrm{E}\left[y_{\mathcal{D}}(\mathbf{x})\right] - \langle d|\mathbf{x}\rangle\right)^2}_{\mathrm{bias}^2} + \mathrm{Var}\left[y_{\mathcal{D}}(\mathbf{x})\right].$$

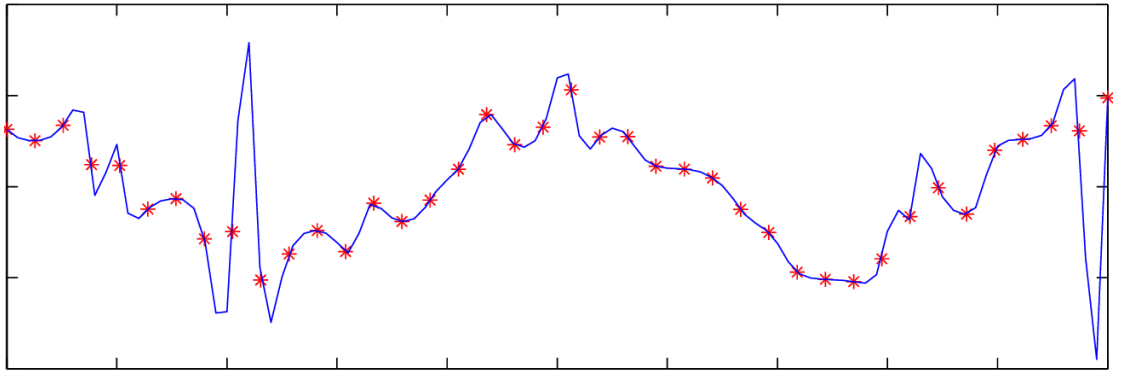

zero variance
large bias
$\langle d|x \rangle$
$y_{\mathrm{D}}(x) = \mathrm{fixed}$
{ * } = dataset D


large variance
zero bias
$\langle d|x \rangle$
$y_{\mathrm{D}}(x) = \text{"flexible"}$
{ * } = dataset D



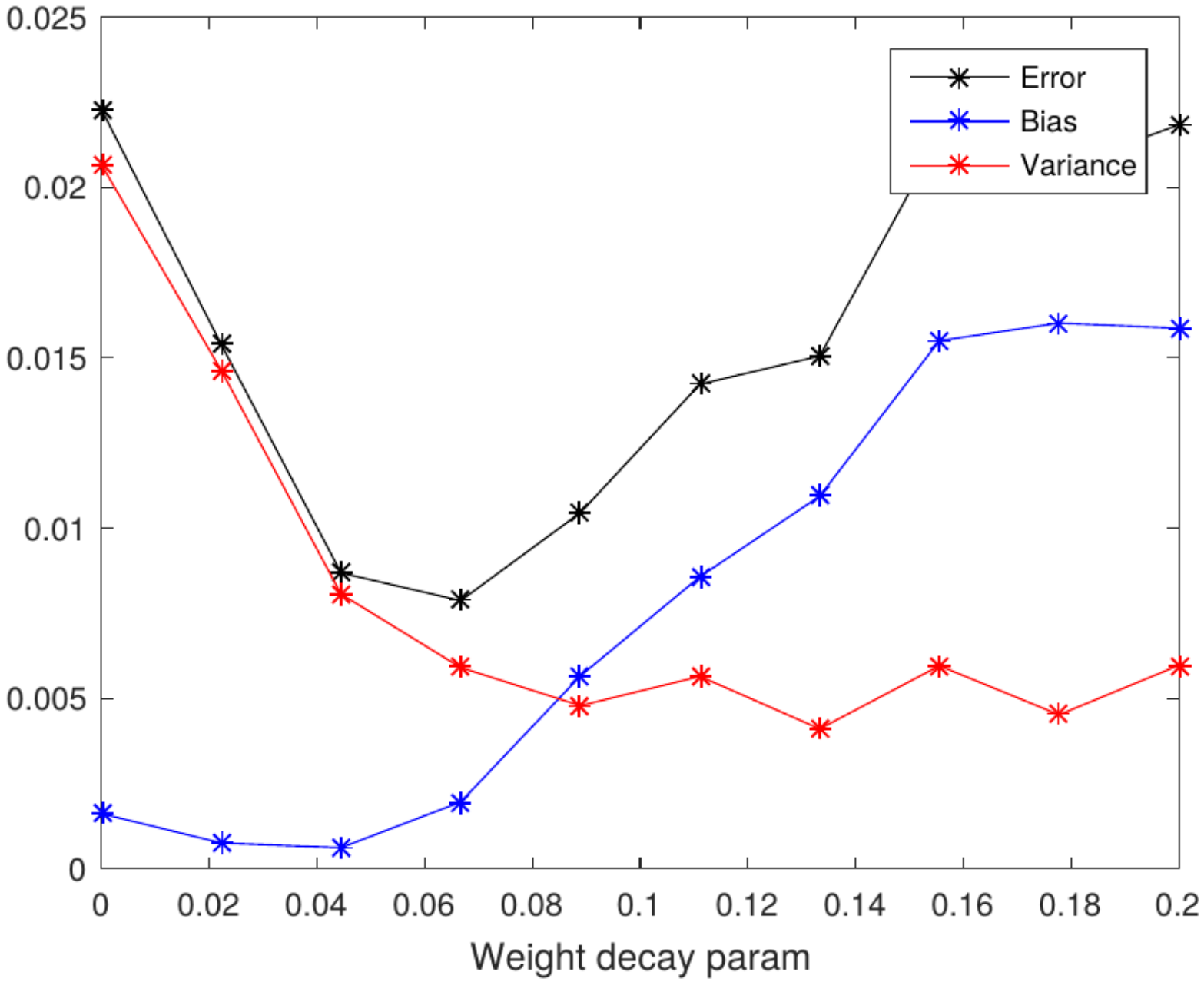
Underfitting

Large bias
Small variance



Overfitting

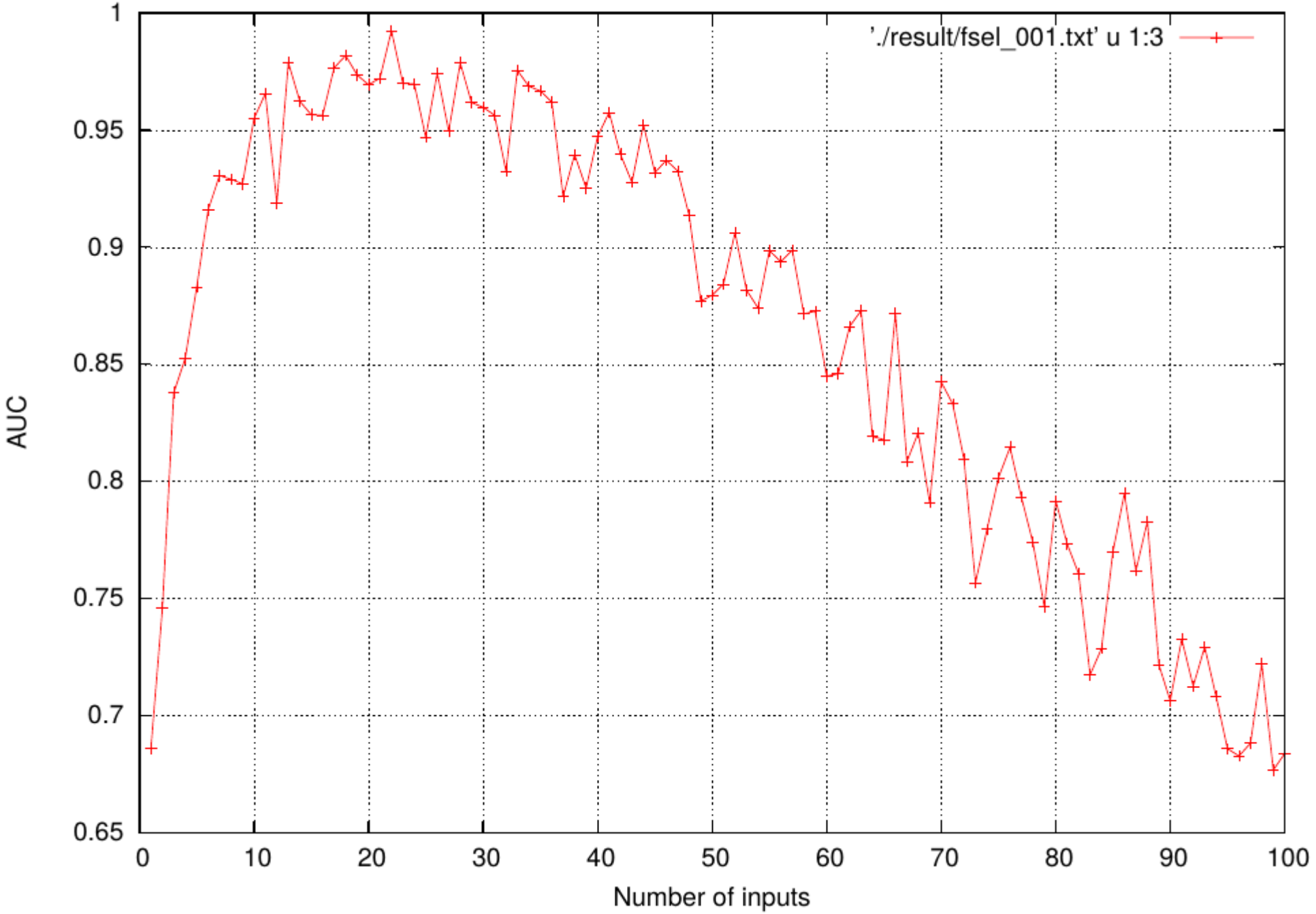
Small bias
Large variance


0.025
0.02
0.015
0.01
0.005
0
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2
Weight decay param
Error
Bias
Variance

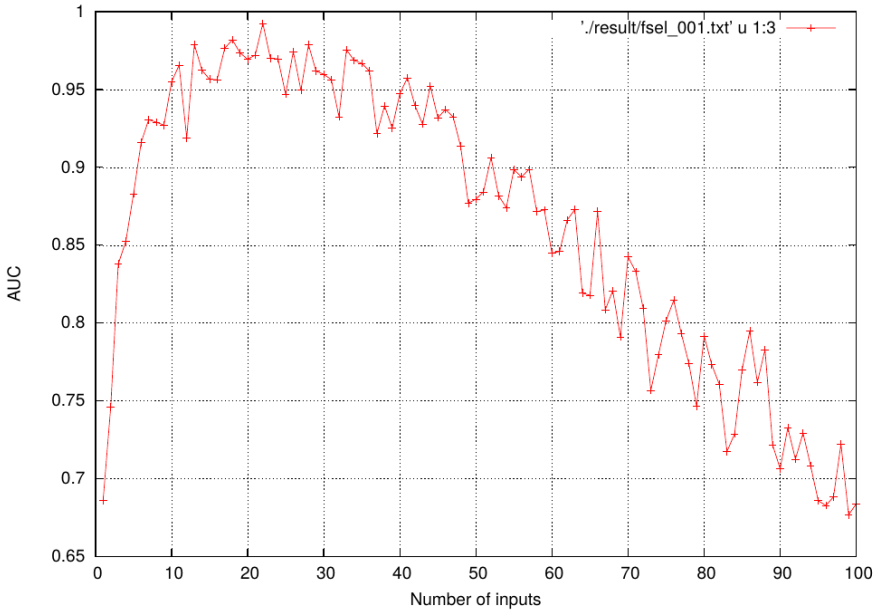
Model selection can sometimes be tricky!!

Example:

- Binary classification problem
- Small dataset (50)
- Large number of inputs (100)
- Use 5-fold cross-validation to find optimal number of inputs (backward elimination)
- Small model (5 hidden nodes)


'./result/fsel_001.txt' u 1:3
AUC
Number of inputs
1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0
10
20
30
40
50
60
70
80
90
100

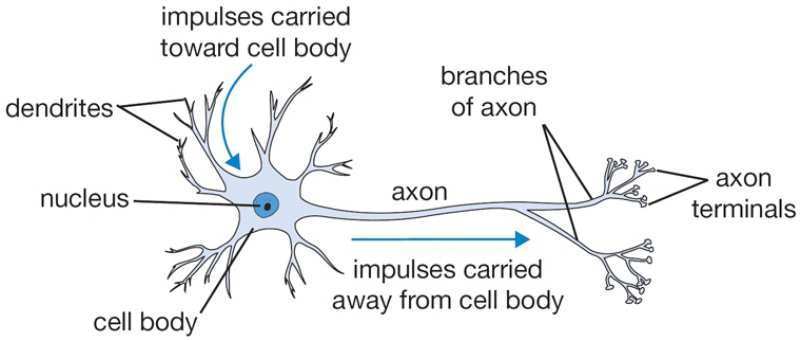
20 inputs is optimal!

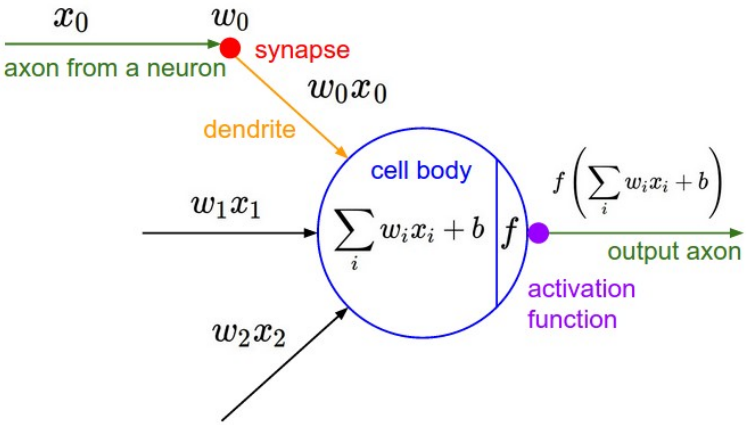


The problem is that all variables
in this problem was random numbers!!

Inputs were random numbers
Target values were random numbers (0/1)

A bit of neural networks history

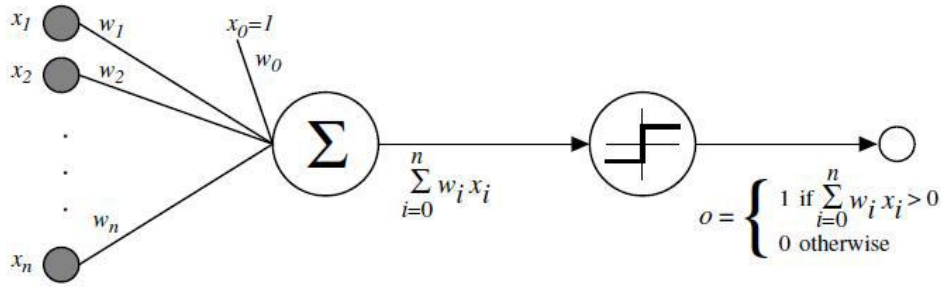




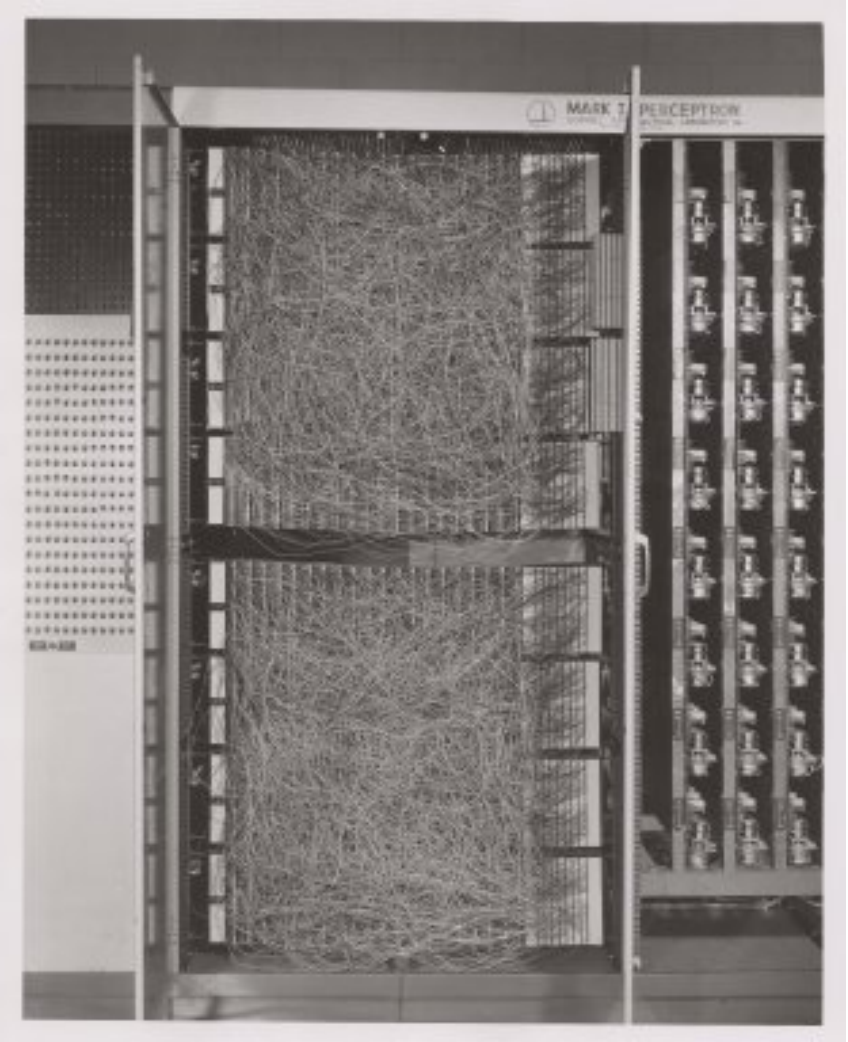
1943 McCulloch-Pitts model

1949 Hebb's rule. Learning occurs by formation and changing of synapses (weights).

1957 Roseblatt's perceptron learning algorithm.



http://cs231n.github.io/neural-networks-1/



The Mark I Perceptron (1957-58)

- 20x20 photocells
- weights were encoded in potentiometers
- weight updates during learning were performed by electric motors

By Source (WP:NFCC#4), Fair use, https://en.wikipedia.org/w/index.php?curid=47541432

The first winter!

1969: Perceptrons by Minsky and Papert showed limitations of the perceptron

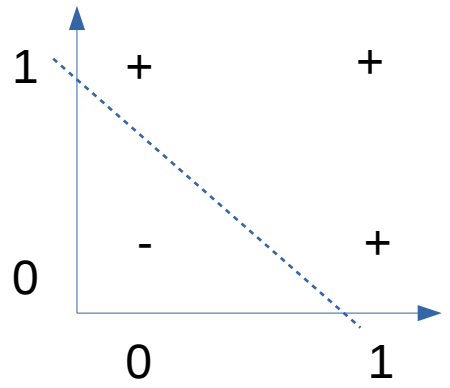
$$y = f\left(\sum_i x_i w_i + b\right)$$

y changes class when

$$\sum_i x_i w_i + b = 0$$

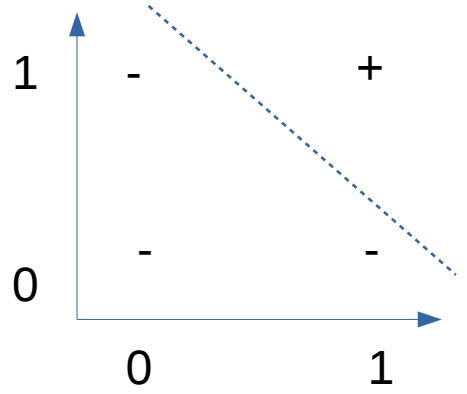
→ decision boundary is a hyperplane

OR



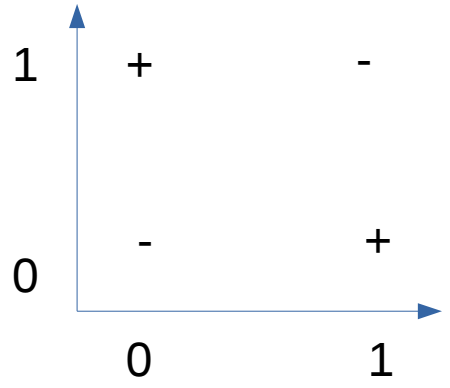
YES

AND



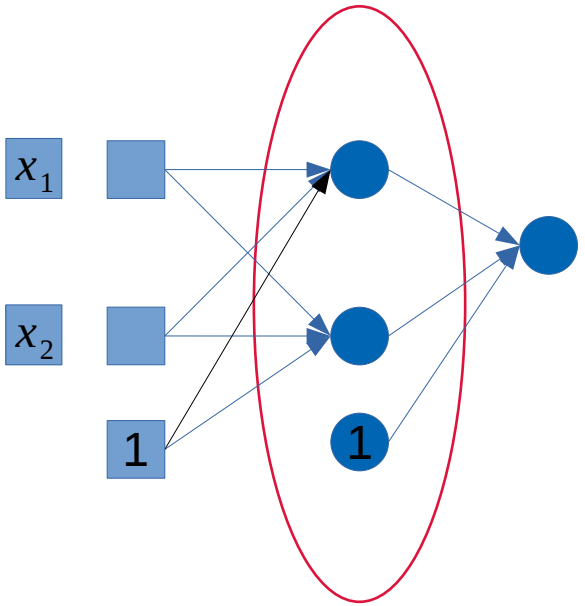
YES

XOR



NOPE

The solution to the XOR problem

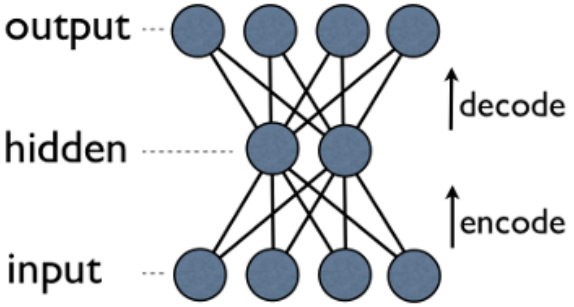


The first return of neural networks

1982: Hopfield. “Hopfield model”

1985: Hinton: Learning in Boltzmann machines

1986: Rumelhart, Hinton, Williams, backpropagation to train networks with several layers. (Already 1974 by Werbos)

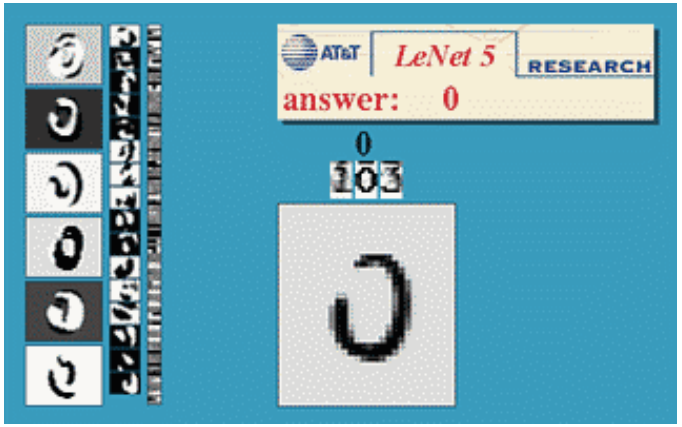


1986: Unsupervised, The autoencoder

1989: Hornik & White: Neural networks are universal approximators

1989: Lecun et al.: LeNet, the first convolutional neural network. Example of classifying handwritten digits.

: Recurrent networks, Backpropagation through time.



http://www.andreykurenkov.com/writing/a-brief-history-of-neural-nets-and-deep-learning-part-2/

The second winter, mid 90's!

- Backpropagation does not work well for networks with many layers. E.g. reccurent networks were difficult to train.
- They were seen as a hassle to work with
- Not fast enough computers
- Some bad papers …
- New algorithms that in some comparisons worked better. Support vector machines, random forest etc.
- Machine learning community "abandoned" neural networks.

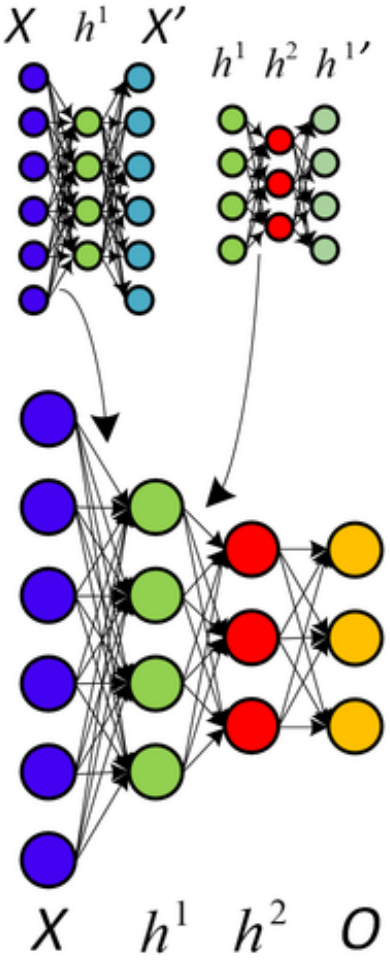
Schmidhuber, 1997, LSTM = "Long short term memory".

Very popular today!!

The second return of neural networks (with a new name, “deep learning”)

2006: Hinton et al., “A fast learning algorithm for deep belief nets”.

2007: Bengio et al. “Greedy layer-wise training of deep networks”



The Deep Learning Explosion...

2012: AlexNet wins ImageNet

2014: GANs

2017: AlphaGo, DeepMind

Very large momentum right now!

A few important factors!

GPU’s

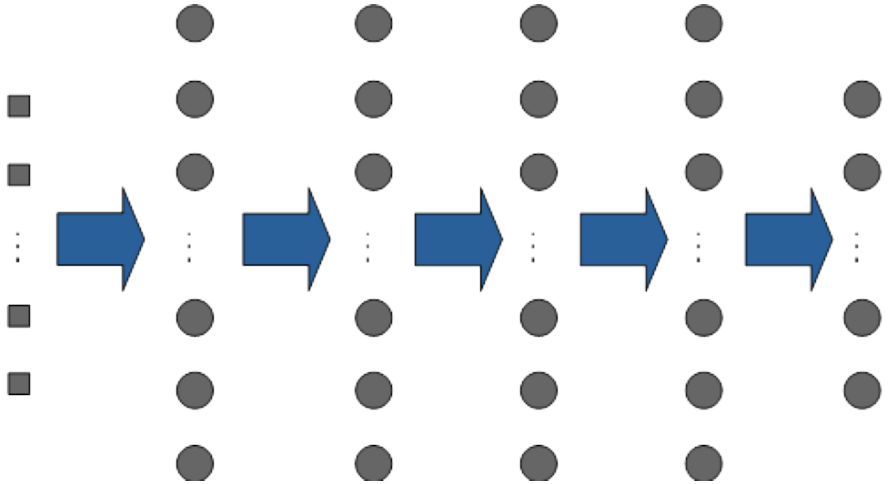
A lot of data!

Google
Facebook
Microsoft
…

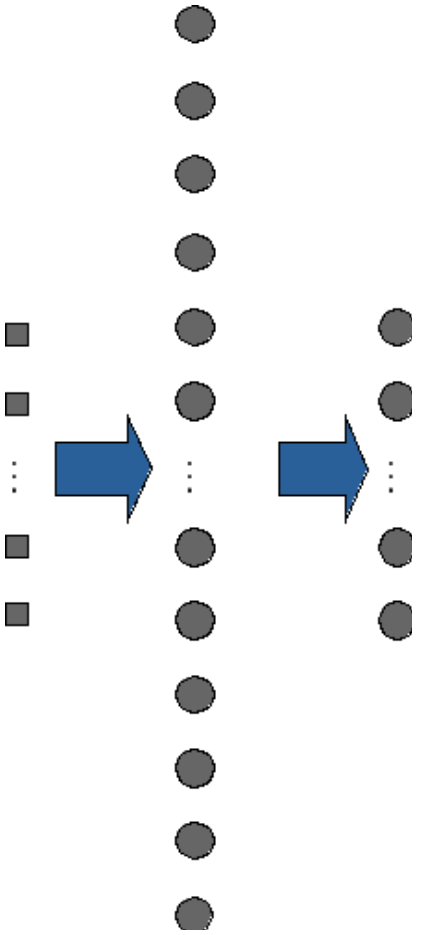
A number of good papers

Deep MLPs

Why this



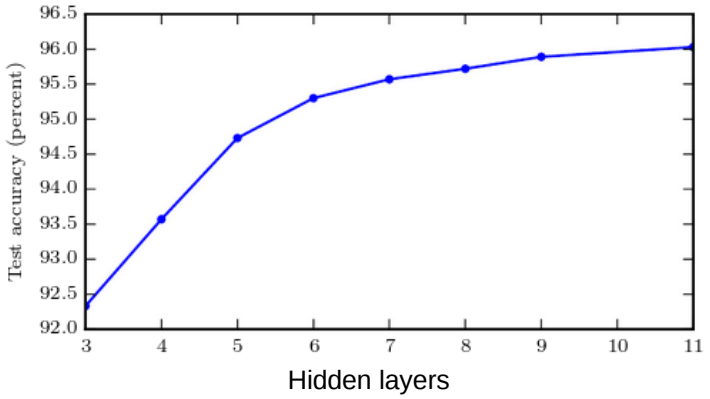
rather than this?

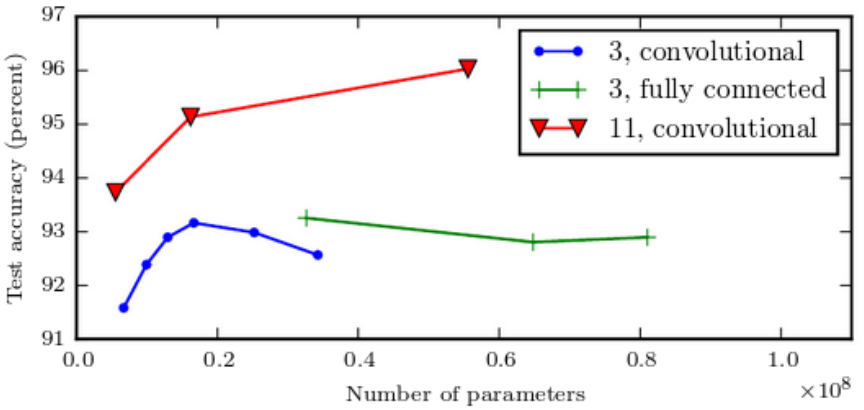


Deep MLPs

Empirically, deeper seems to give better generalization

(Image classification experiments)



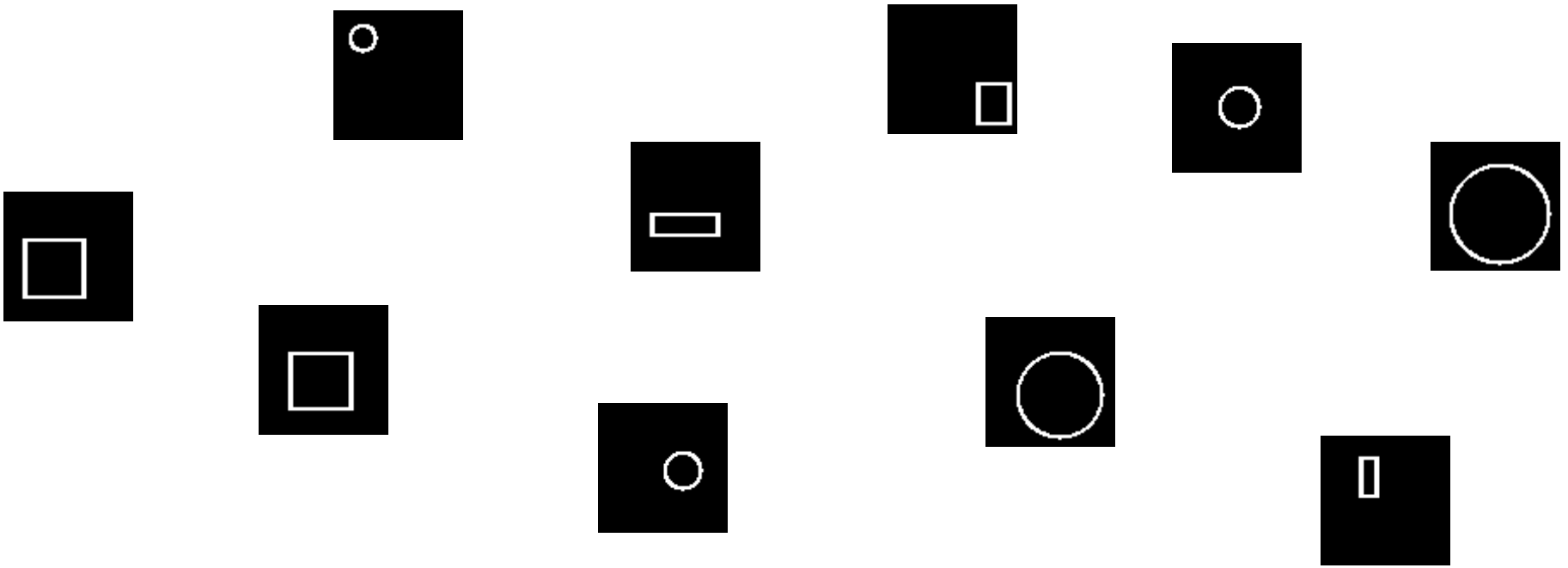


(From the deep learning book:
https://www.deeplearningbook.org/contents/mlp.html)

Deep learning = Representation learning

Rectangle or circle?

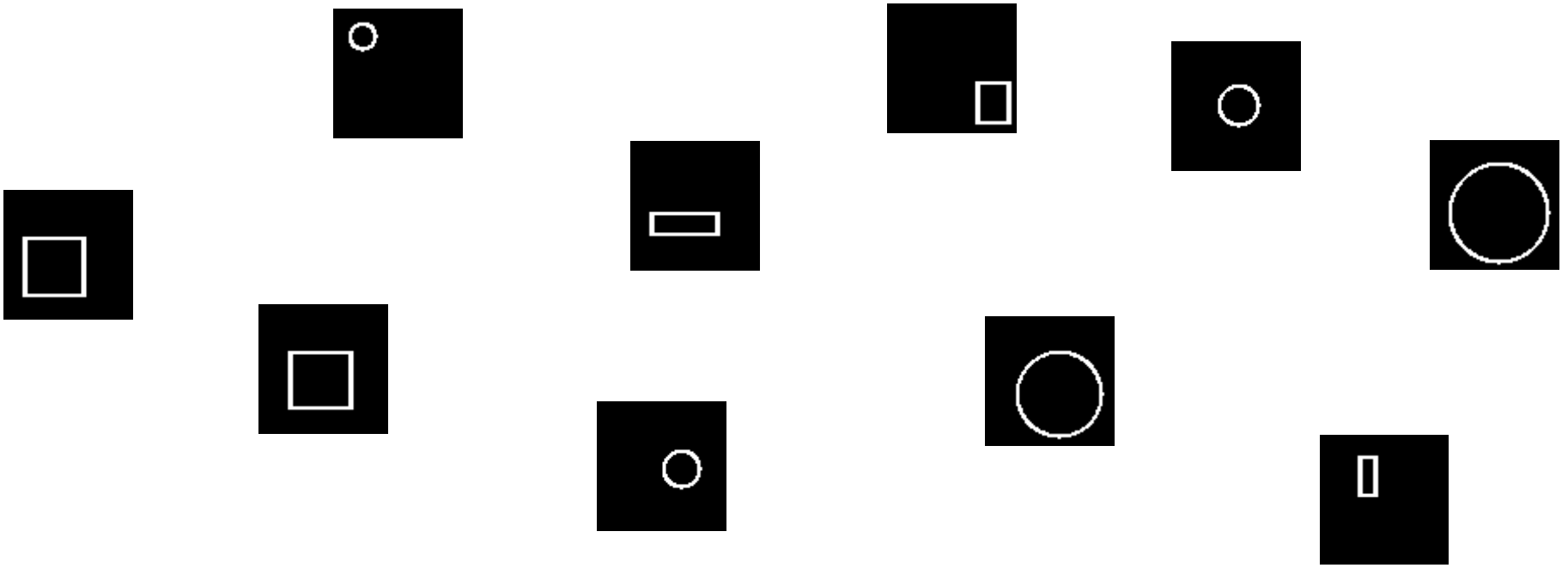
Which representation to use?

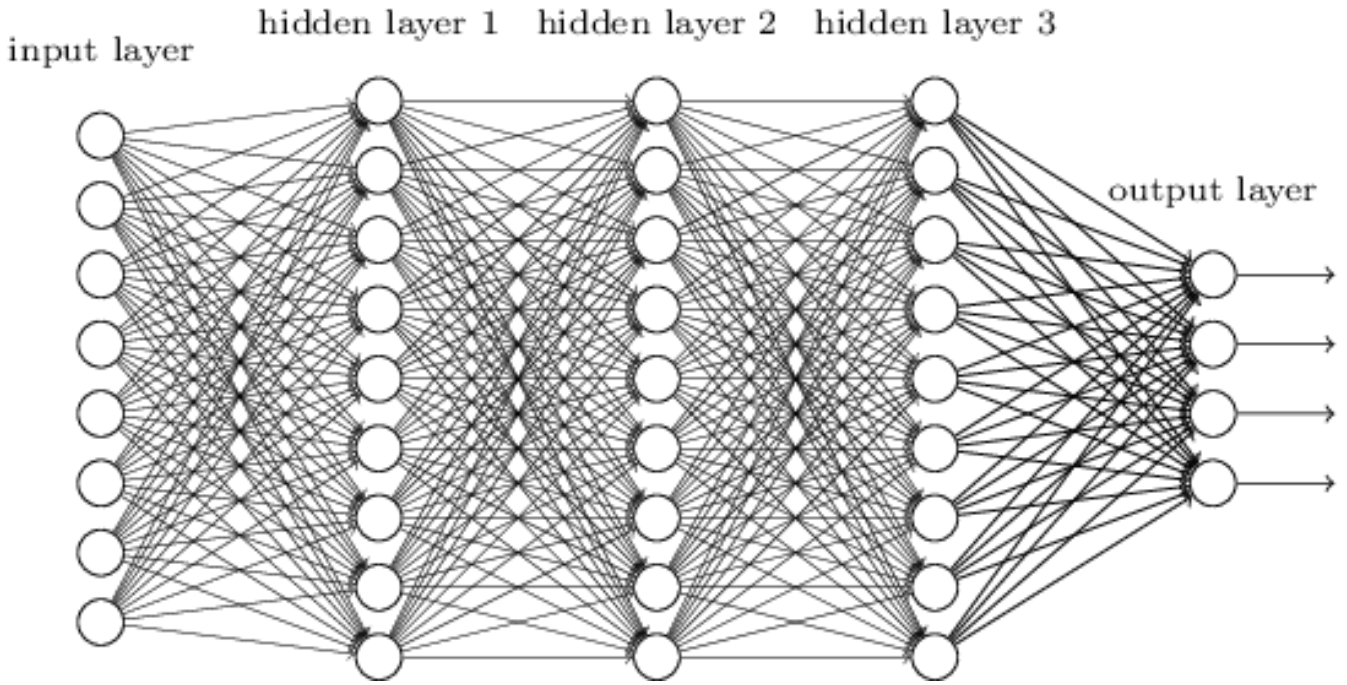


Deep learning = Representation learning

Small or large area?

Which representation to use?





Layer 1 | Layer2 | ... | Layer N

Simple features

.. building more complex features

.. building the “optimal” features for this specific task

Feature learning

Feature learning