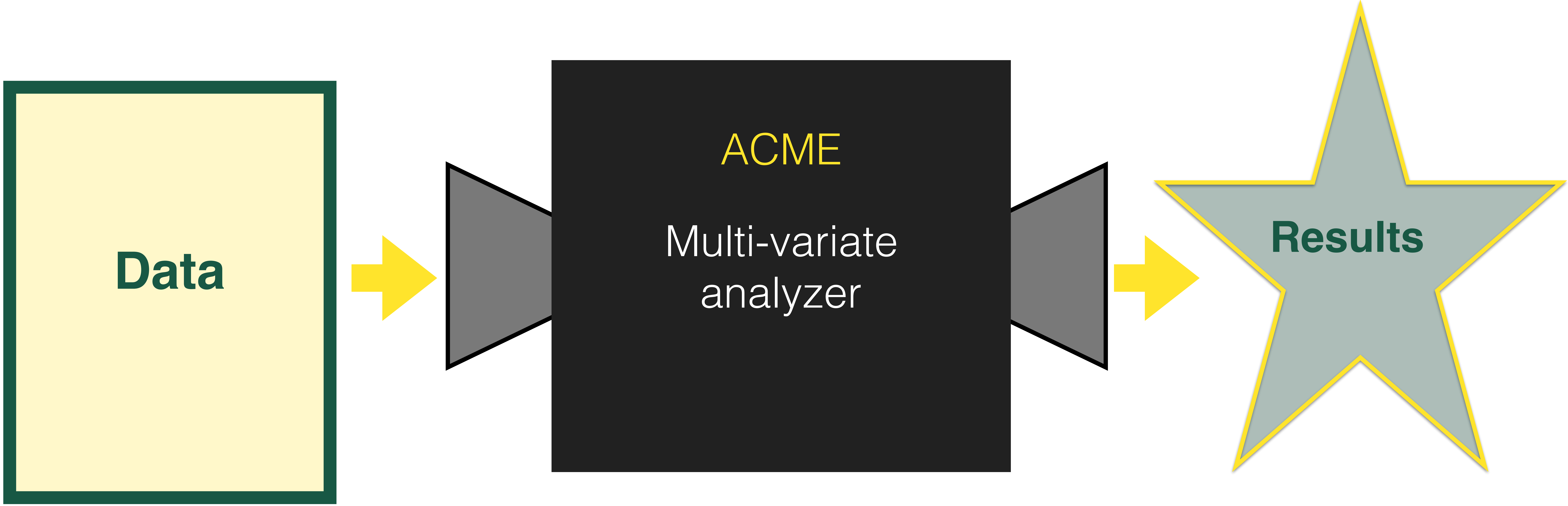
What is the machine learning?



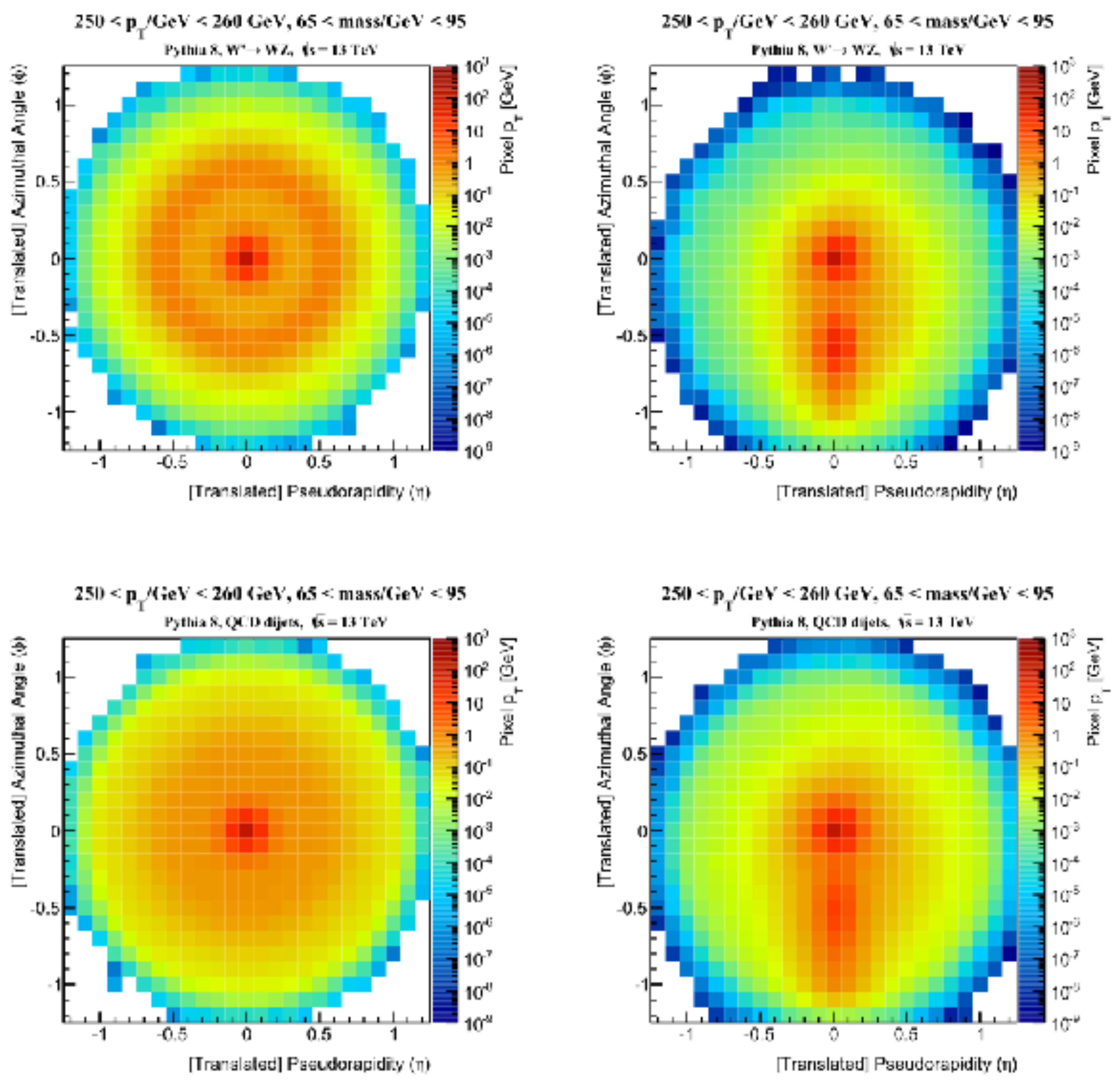
Bryan Ostdiek

Machine learning for phenomenology workshop. IPPP, Durham

April 3, 2018

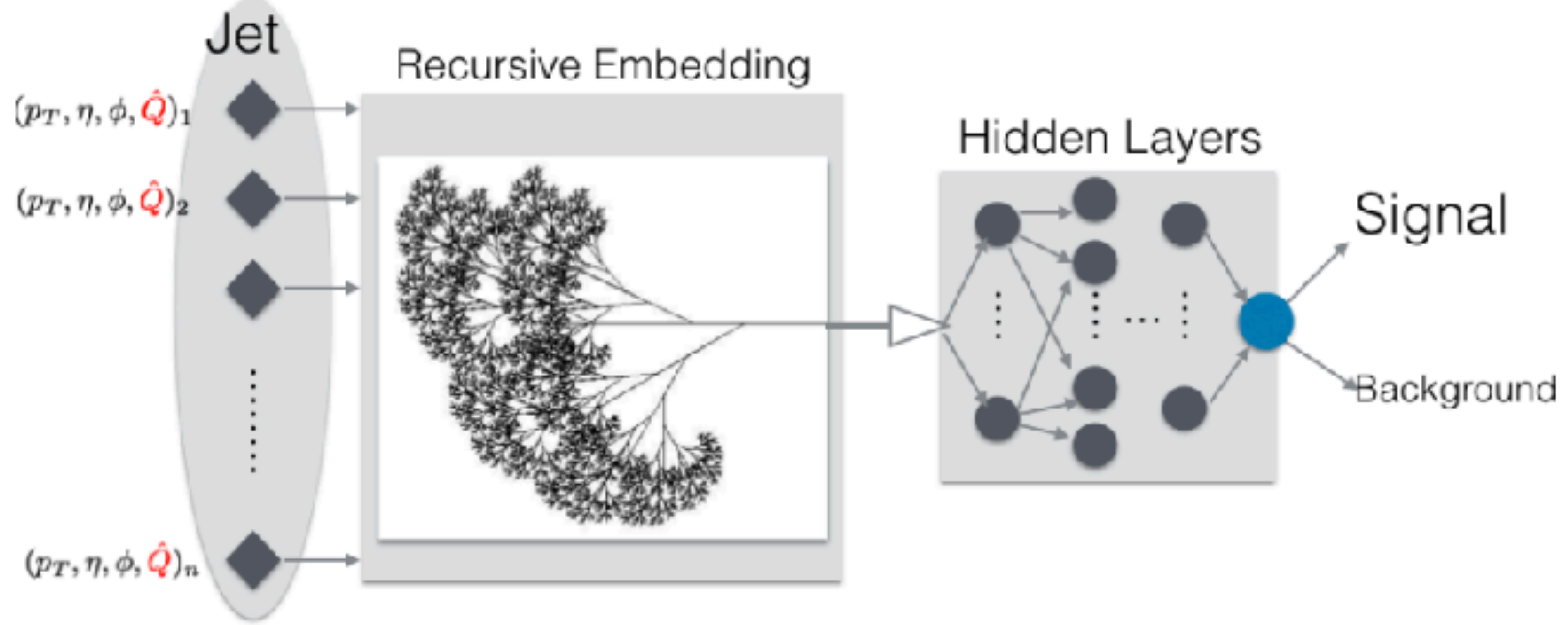
Impressive results with advanced techniques

Image recognition



Cogan, Kagan, Strauss, and Schwartzman [1407.5675]
Almeida, Backovic, Cliche, Lee, and Perelein [1501.15968]
de Oliveira, Kagan, Mackey, Nachman, and Schwartzman [1511.05190]
Baldi, Bauer, Eng, Sadowski, Whiteson [1603.09349]
Komiske, Metodiev, and Schwartz [1612.01551]

Language processing



Louppe, Cho, Becot, Cranmer [1702.00748]
Cheng [1711.02633]

Breaking open the black box

Advanced techniques often operate as a black box.

- No physical intuition for the parameters of the model
- If it works, do we care?

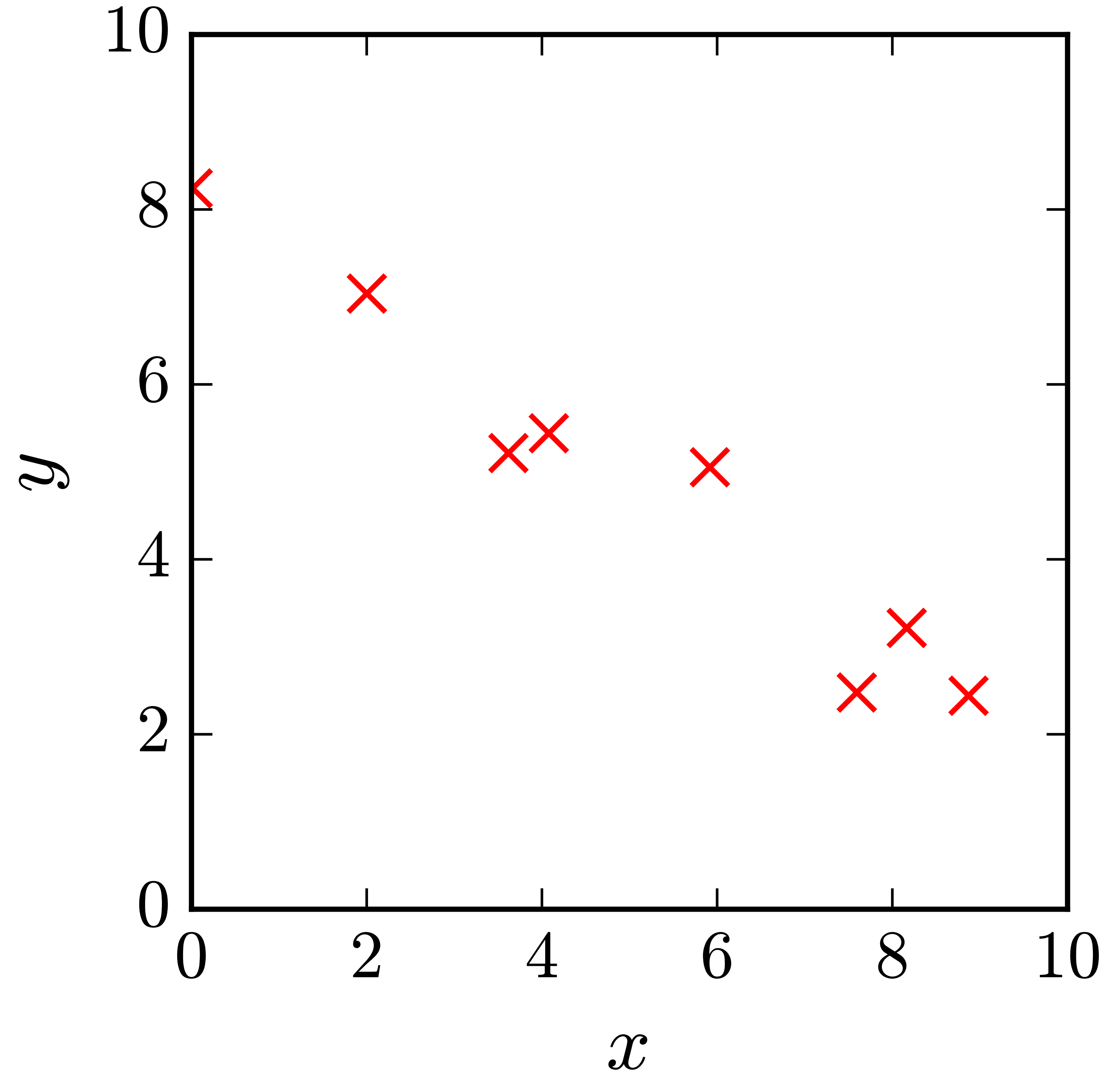
Outline for the talk

1. Review basics of fitting data
   a. Linear regression
   b. Logistic regression
2. Neural networks and deep learning
3. Controlling information through input variables
4. *Planing* to uncover what information the machine is learning from

Review: Linear Regression

<u>How to fit data</u>

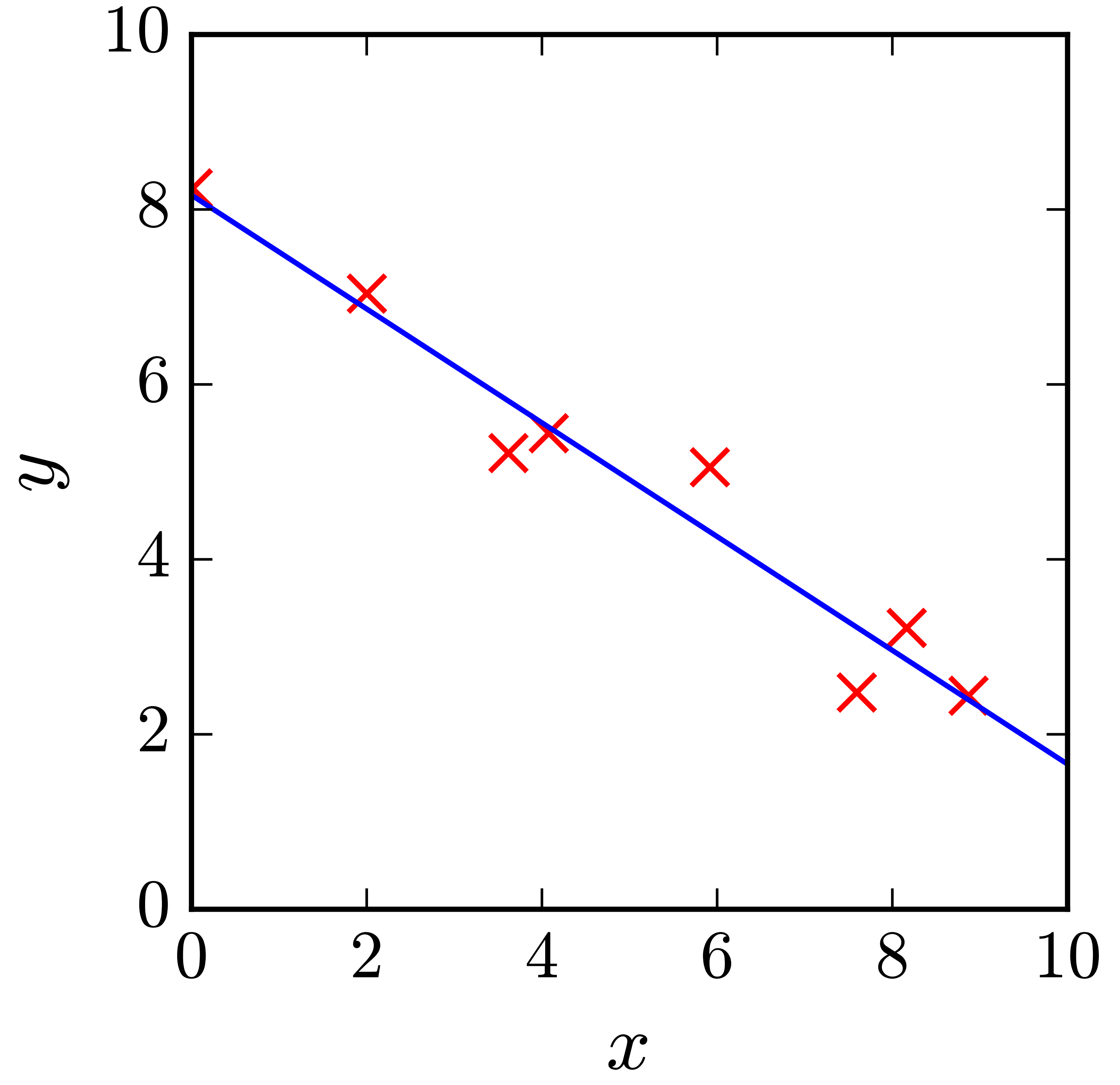
1. Plot the data
2. Define the function
   - $f(x, \vec{a}) = a_0 + a_1 x$
3. Choose how to know what fits best
   - a.k.a. *Loss Function*
   - MSE: $L(x, y, \vec{a}) = \frac{1}{N}\sum_{i=1}^{N}\left(f(x_i, \vec{a}) - y_i\right)^2$
5. Find the minimum error (loss) (cost)
   - $a_{\text{best}} = a \text{ when } \left(\frac{\partial L(x,y,\vec{a})}{\partial \vec{a}}\Big|_{x,y} = 0\right)$



Review: Linear Regression

<u>How to fit data</u>

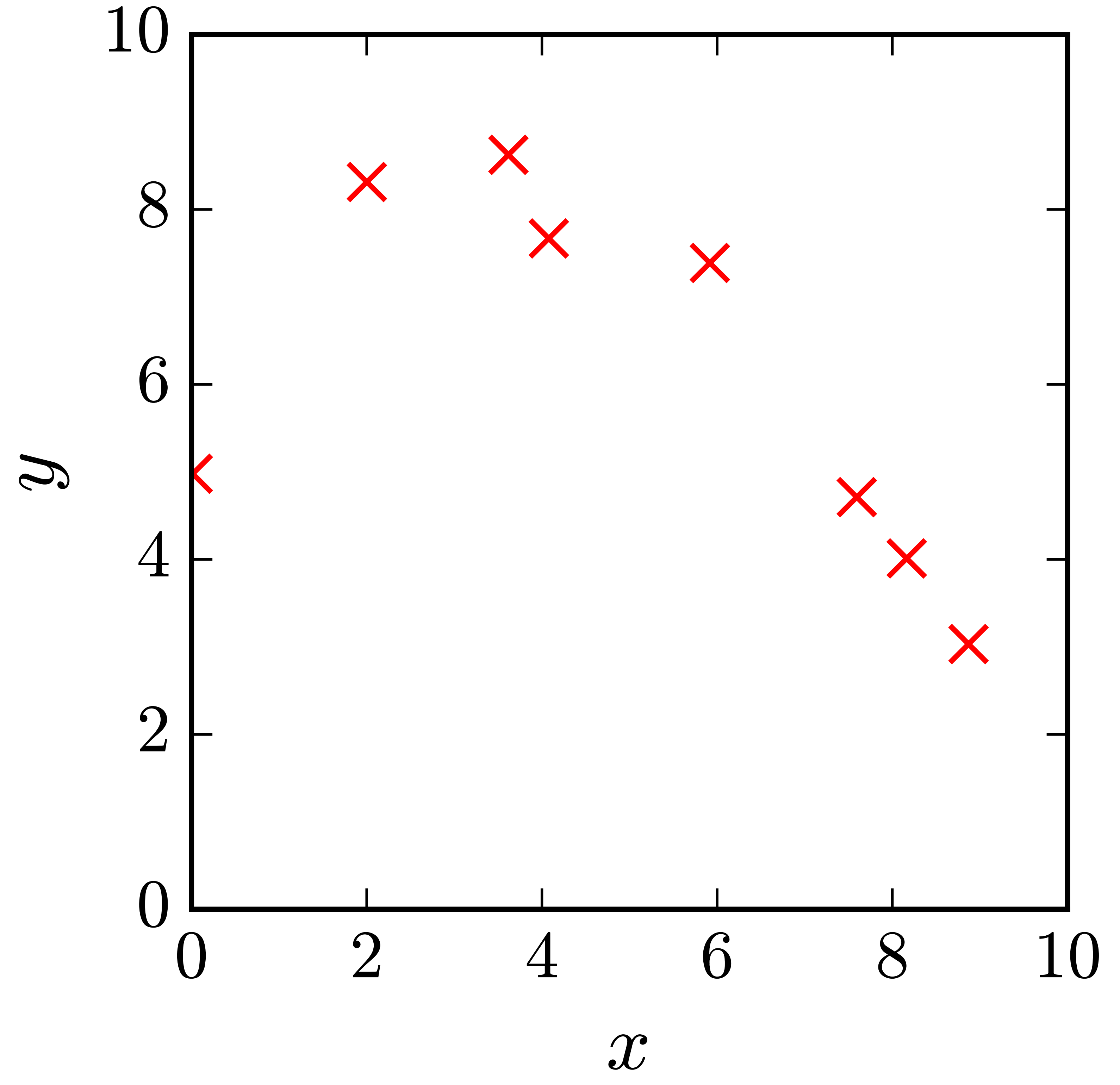
1. Plot the data
2. Define the function
   - $f(x, \vec{a}) = a_0 + a_1 x$
3. Choose how to know what fits best
   - a.k.a. *Loss Function*
   - MSE: $L(x, y, \vec{a}) = \frac{1}{N}\sum_{i=1}^{N}\left(f(x_i, \vec{a}) - y_i\right)^2$
5. Find the minimum error (loss) (cost)
   - $a_{\text{best}} = a \text{ when } \left(\left.\frac{\partial L(x, y, \vec{a})}{\partial \vec{a}}\right|_{x,y} = 0 \right)$



Review: Linear Regression

How to fit data

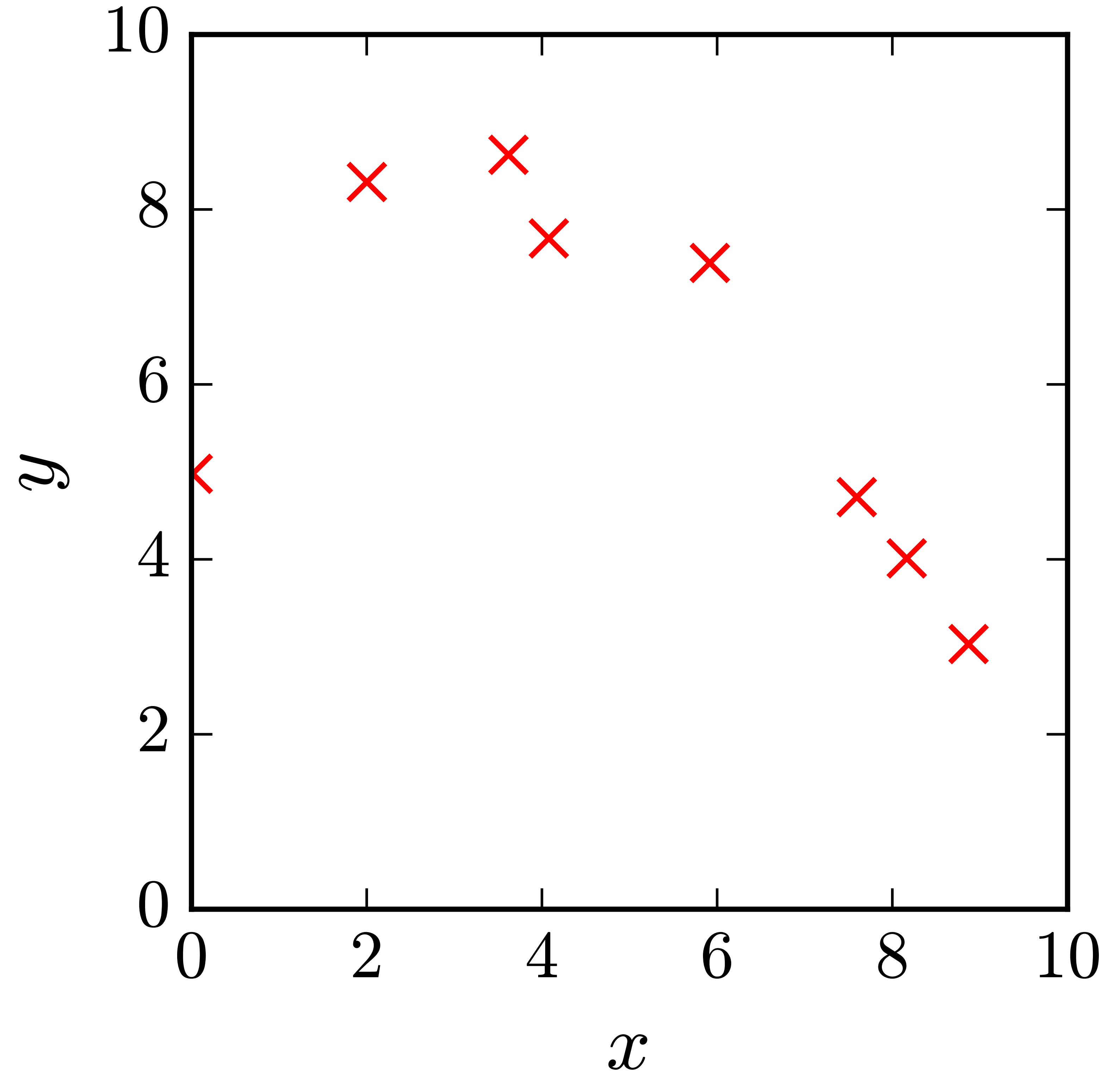
1. Plot the data
2. Define the function
   - $f(x, \vec{a}) = a_0 + a_1 x + a_2 x^2$
3. Choose how to know what fits best
   - a.k.a. *Loss Function*
   - MSE: $L(x, y, \vec{a}) = \dfrac{1}{N}\sum_{i=1}^{N}\left(f(x_i, \vec{a}) - y_i\right)^2$
5. Find the minimum error (loss) (cost)
   - $a_{\text{best}} = a \text{ when } \left(\left.\dfrac{\partial L(x, y, \vec{a})}{\partial \vec{a}}\right|_{x,y} = 0 \right)$



Review: ~~Linear~~ Quadratic? Regression

How to fit data

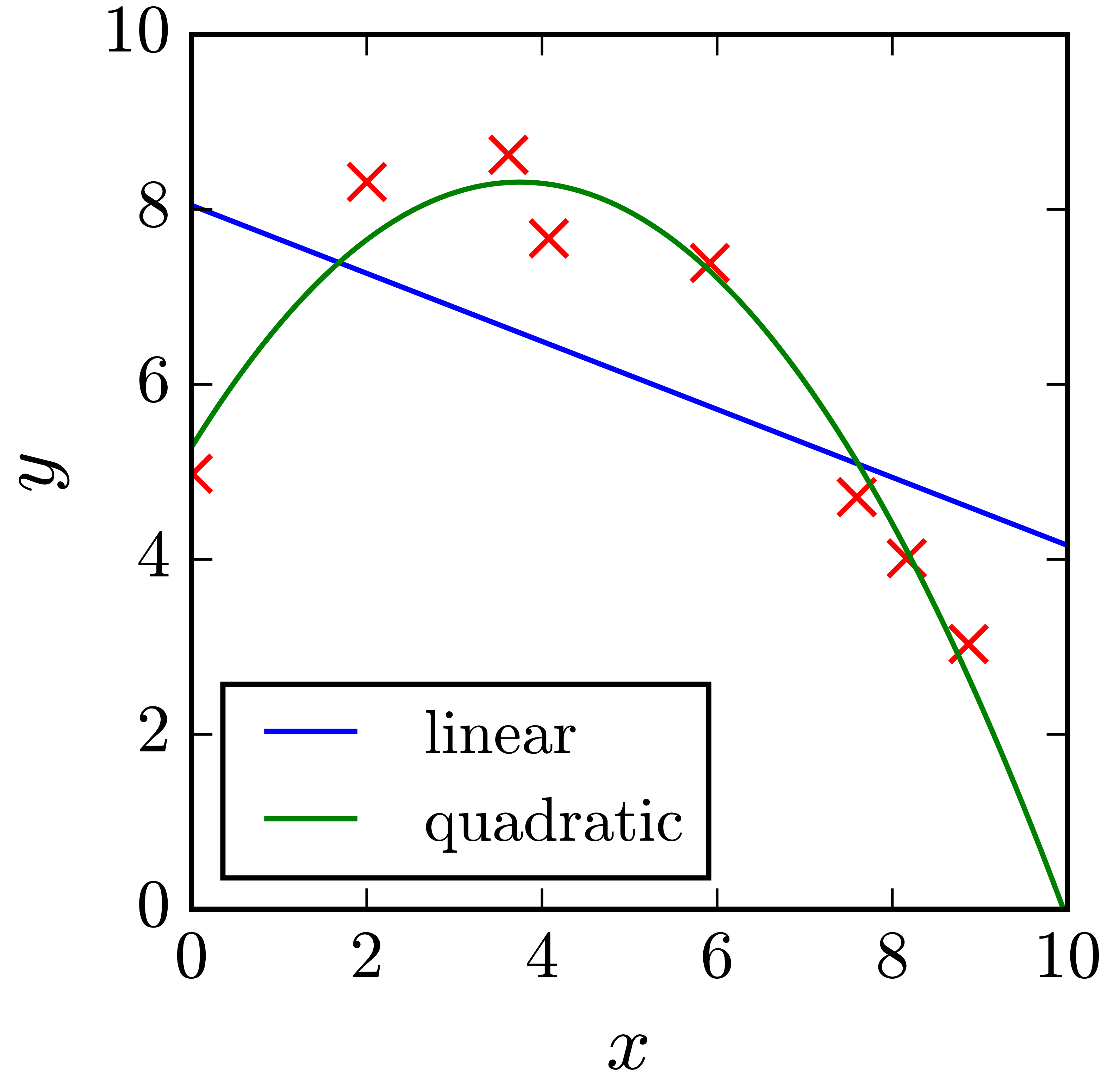
1. Plot the data
2. Define the function
   - $f(x,\vec{a}) = a_0 + a_1 x + a_2 x^2$
3. Choose how to know what fits best
   - a.k.a. *Loss Function*
   - MSE: $L(x,y,\vec{a}) = \dfrac{1}{N}\sum_{i=1}^{N}\left(f(x_i,\vec{a}) - y_i\right)^2$
5. Find the minimum error (loss) (cost)
   - $a_{\text{best}} = a \text{ when } \left(\left.\dfrac{\partial L(x,y,\vec{a})}{\partial \vec{a}}\right|_{x,y} = 0 \right)$



Review: ~~Linear~~ Quadratic? Regression

<u>How to fit data</u>

1. Plot the data
2. Define the function
   - $f(x, \vec{a}) = a_0 + a_1 x + a_2 x^2$
3. Choose how to know what fits best
   - a.k.a. *Loss Function*
   - MSE: $L(x, y, \vec{a}) = \dfrac{1}{N}\sum_{i=1}^{N}\left(f(x_i, \vec{a}) - y_i\right)^2$
5. Find the minimum error (loss) (cost)
   - $a_{\text{best}} = a \text{ when } \left(\left.\dfrac{\partial L(x, y, \vec{a})}{\partial \vec{a}}\right|_{x,y} = 0 \right)$

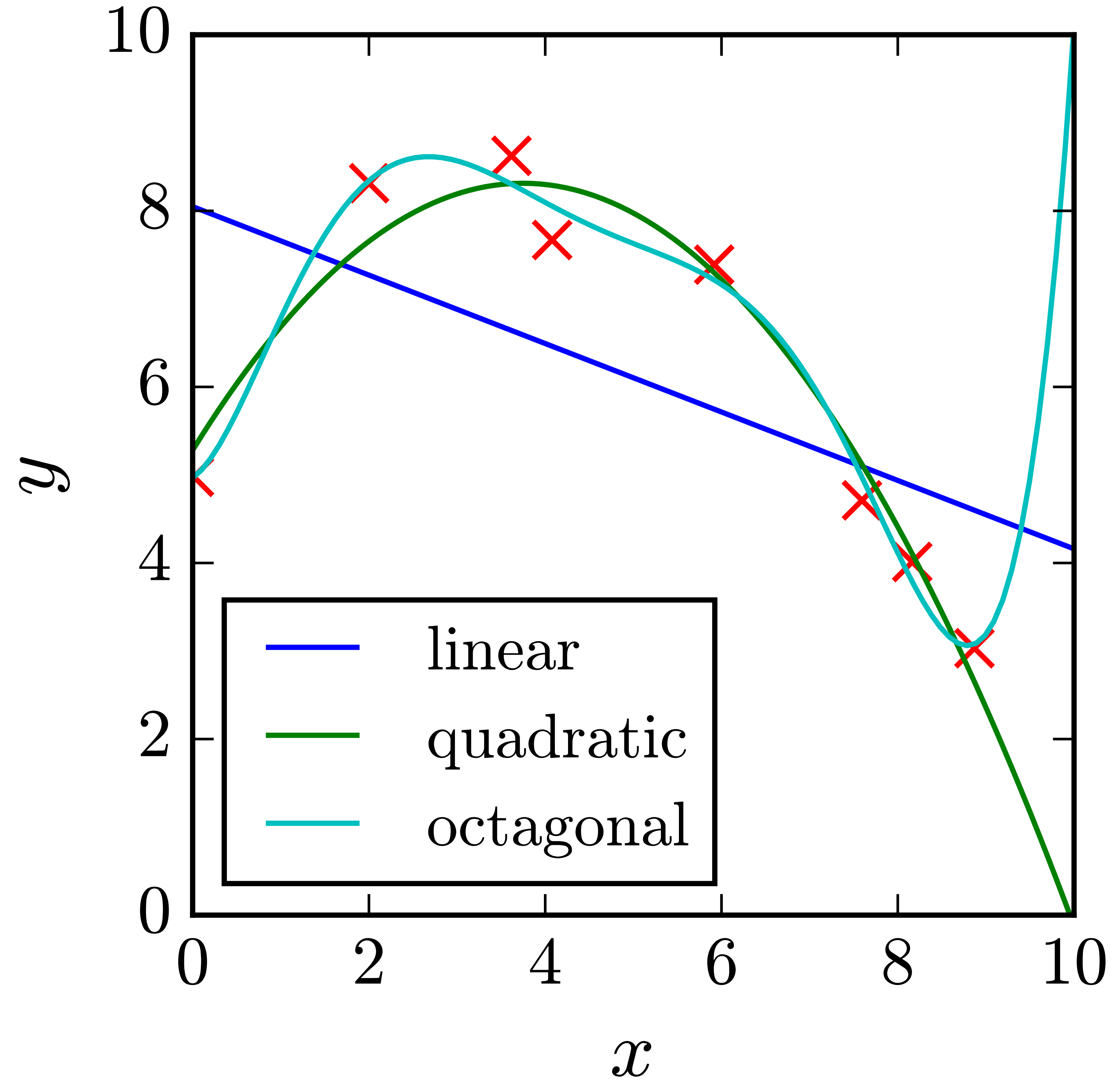


Is that good enough?
How many parameters can we add?

Review: ~~Linear~~ Quadratic? Regression

How to fit data

1. Plot the data
2. Define the function
   - $f(x, \vec{a}) = a_0 + a_1 x + a_2 x^2$
3. Choose how to know what fits best
   - a.k.a. *Loss Function*
   - MSE: $L(x, y, \vec{a}) = \frac{1}{N} \sum_{i=1}^{N} \left(f(x_i, \vec{a}) - y_i\right)^2$
5. Find the minimum error (loss) (cost)
   - $a_{\text{best}} = a \text{ when } \left(\frac{\partial L(x, y, \vec{a})}{\partial \vec{a}} \Big|_{x,y} = 0 \right)$

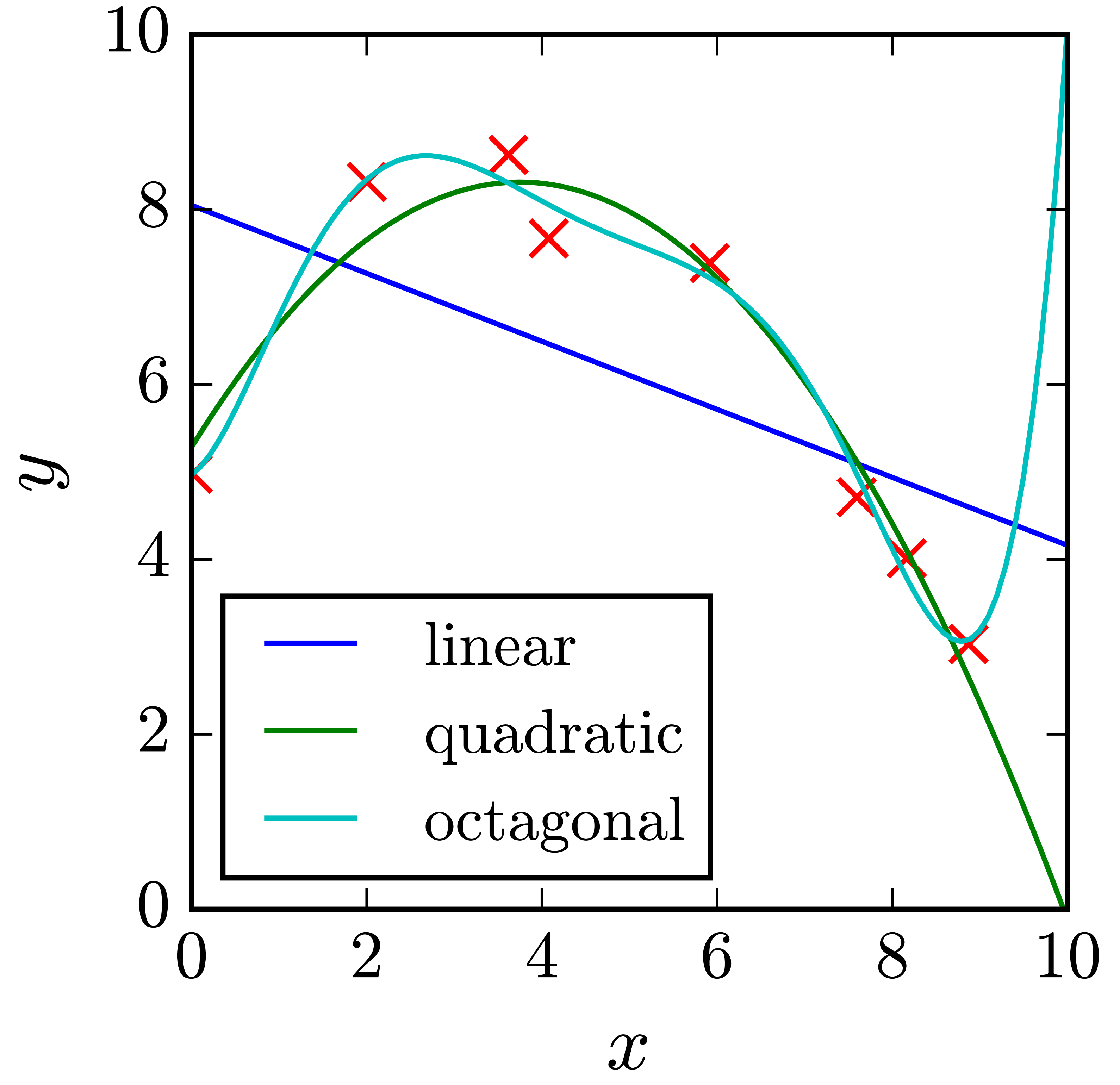


Is that good enough?
How many parameters can we add?

Review: ~~Linear~~ Quadratic? Regression

<u>How to fit data</u>

1. Plot the data
2. Define the function
   - $f(x, \vec{a}) = a_0 + a_1 x + a_2 x^2$
3. Choose how to know what fits best
   - a.k.a. *Loss Function*
   - MSE: $L(x, y, \vec{a}) = \frac{1}{N} \sum_{i=1}^{N} (f(x_i, \vec{a}) - y_i)^2$
5. Find the minimum error (loss) (cost)
   - $a_{\text{best}} = a \text{ when } \left(\frac{\partial L(x, y, \vec{a})}{\partial \vec{a}} \Big|_{x,y} = 0 \right)$



Avoid over fitting to be able to use for predictions

Logistic Regression

What if we are trying to predict a class, not a number?

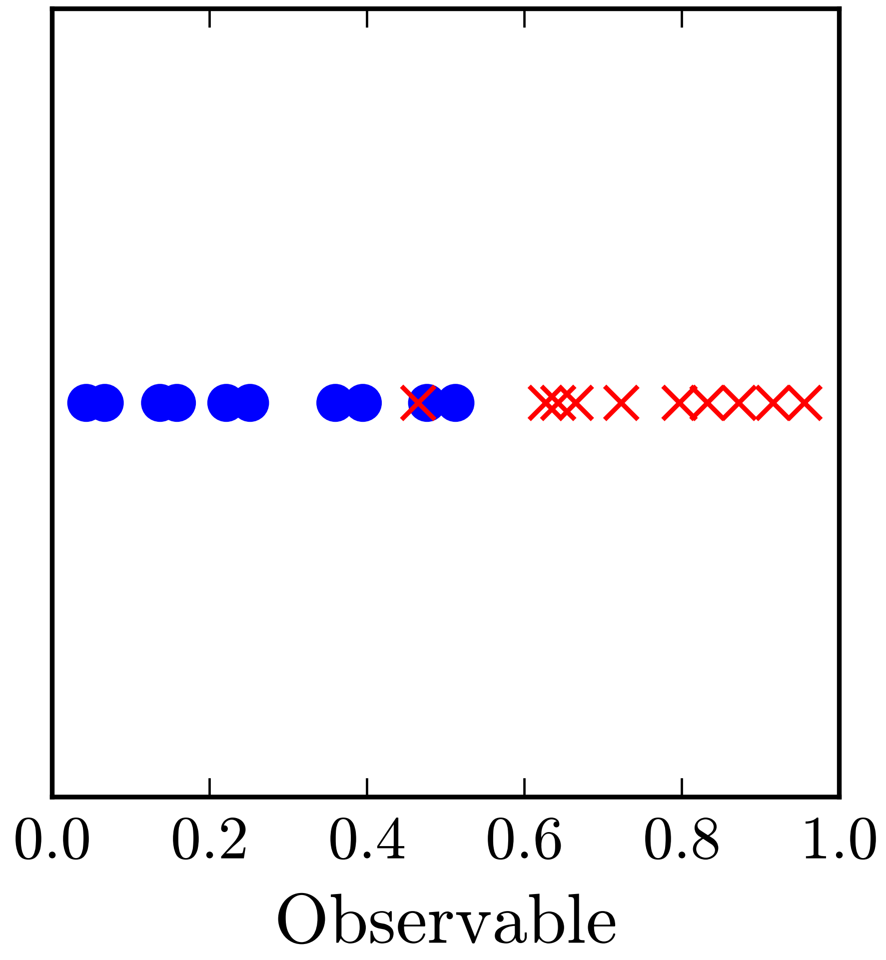
quark(s) gluon

Logistic Regression

What if we are trying to predict a class, not a number?

quark(s) gluon

What is the y-value we are trying to fit/predict?



Logistic Regression

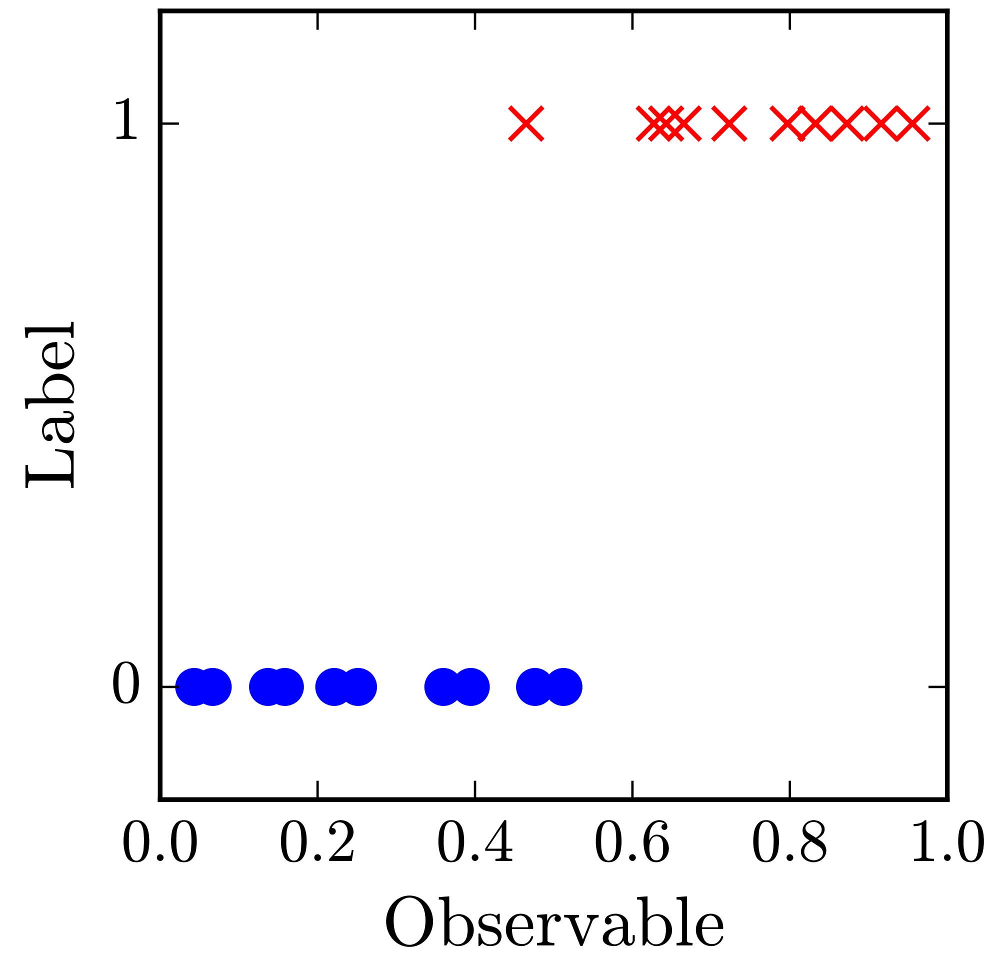
What if we are trying to predict a class, not a number?

quark(s) gluon

What is the y-value we are trying to fit/predict?

Define one class as 1 (Signal)

Other class as 0 (Background)

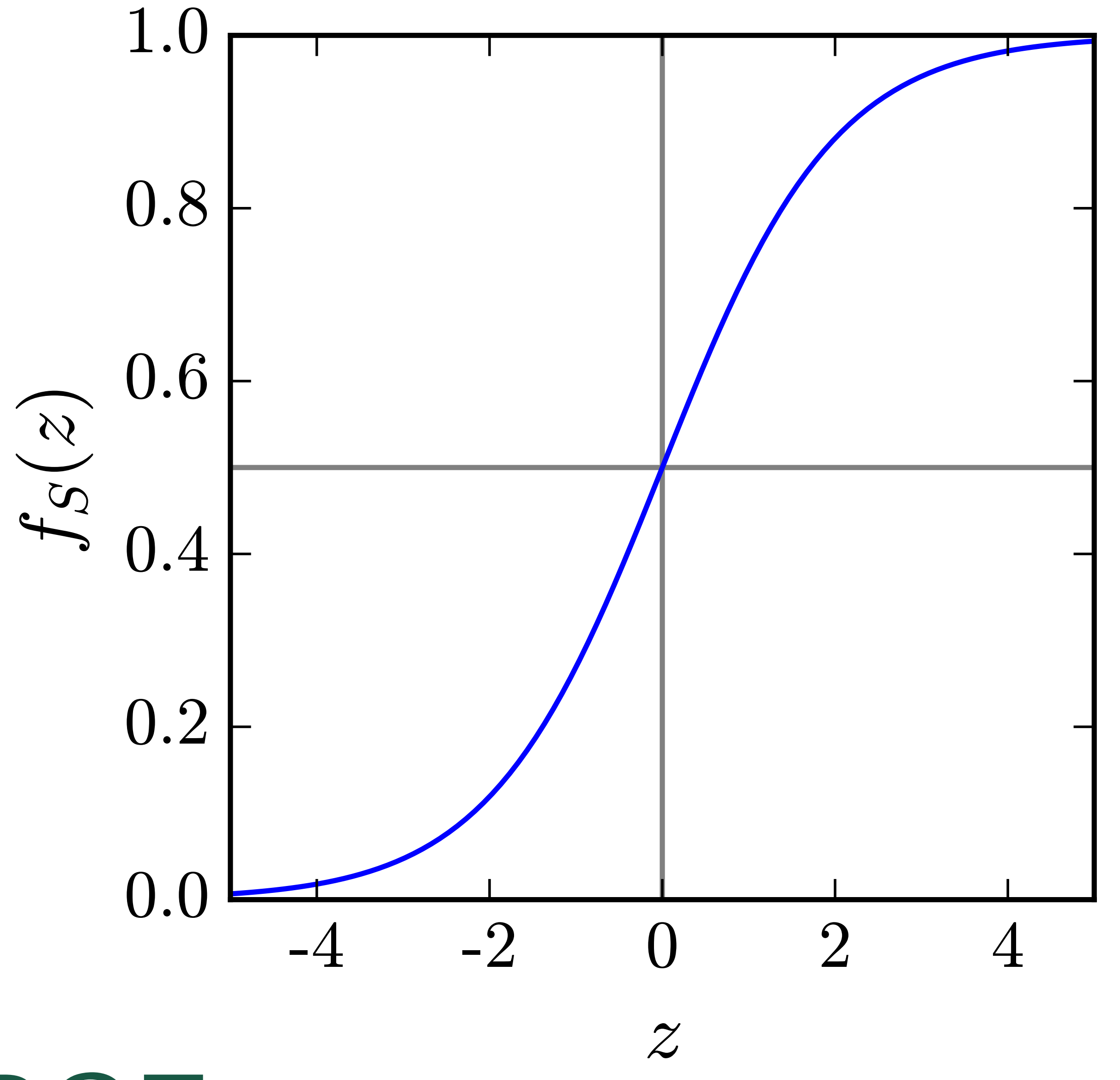


Logistic Regression

What if we are trying to predict a class, not a number?

- Change the shape of function: Logistic/Sigmoid function

$$f_S(z) = \frac{1}{1+\mathrm{e}^{-z}}$$



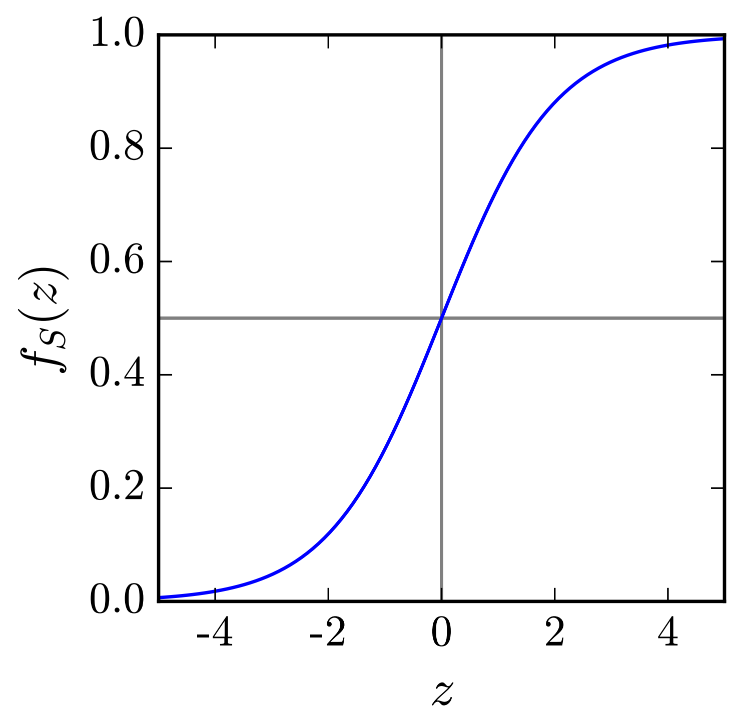
Does not add parameters

- Change the loss function: BCE

$$L(\vec{x},\vec{y},\vec{a}) = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log\left(f_S(p(x,a))\right) + (1-y_i)\log\left(1 - f_S(p(x,a))\right)\right)$$

Logistic Regression

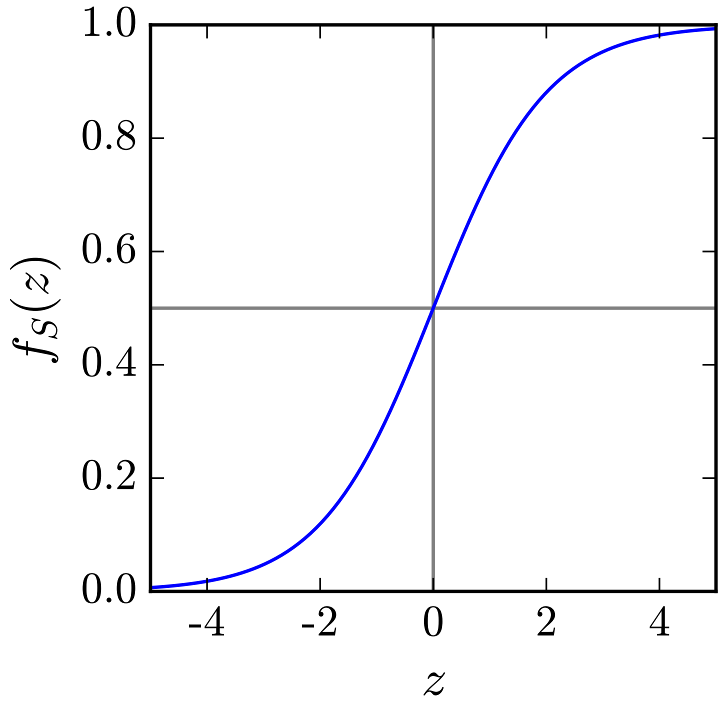
What if we are trying to predict a class, not a number?



$$L(\vec{x},\vec{y},\vec{a}) = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log\left(f_S(p(x,a))\right) + (1-y_i)\log\left(1-f_S(p(x,a))\right)\right)$$

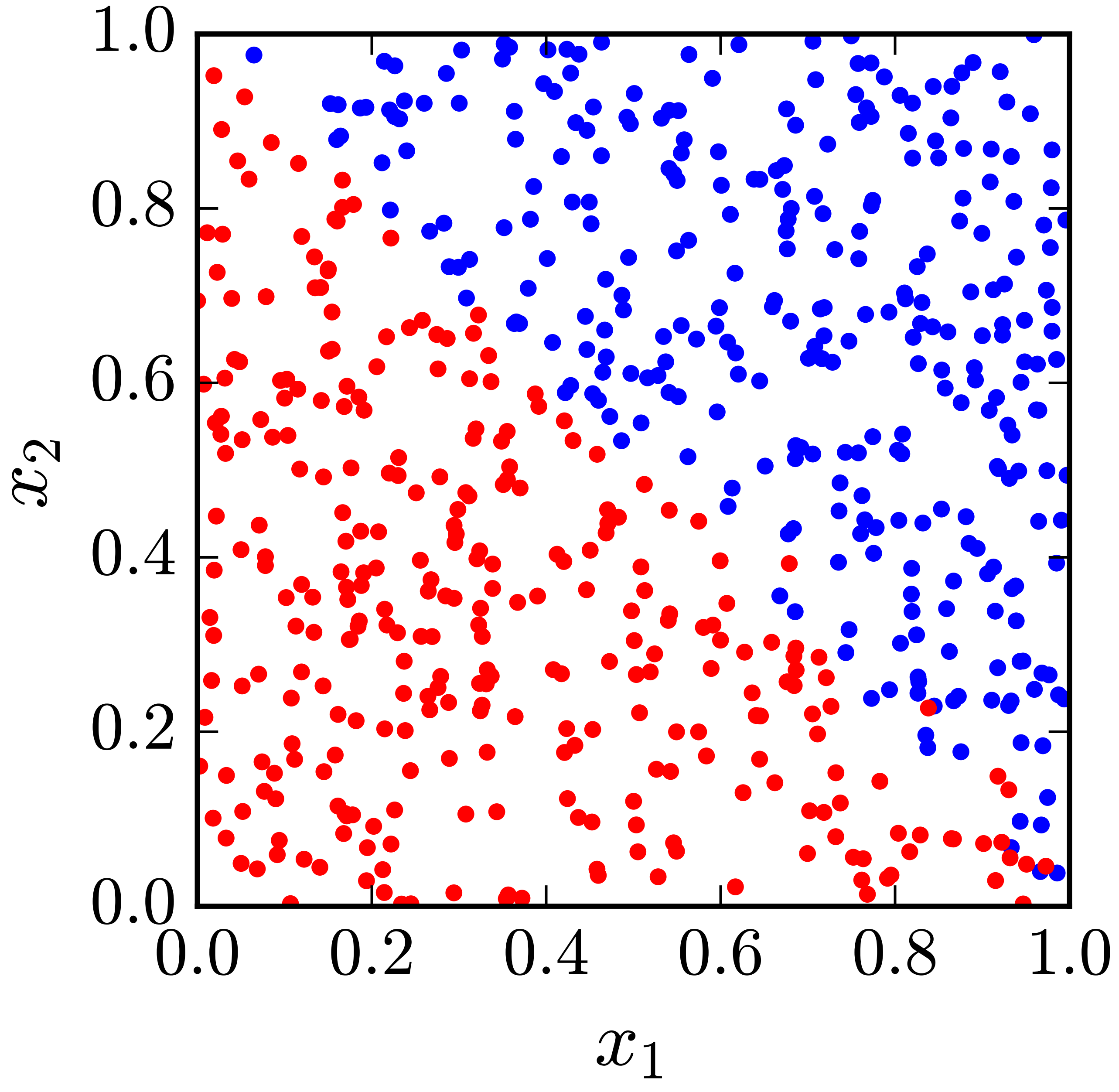
Logistic Regression

What if we are trying to predict a class, not a number?



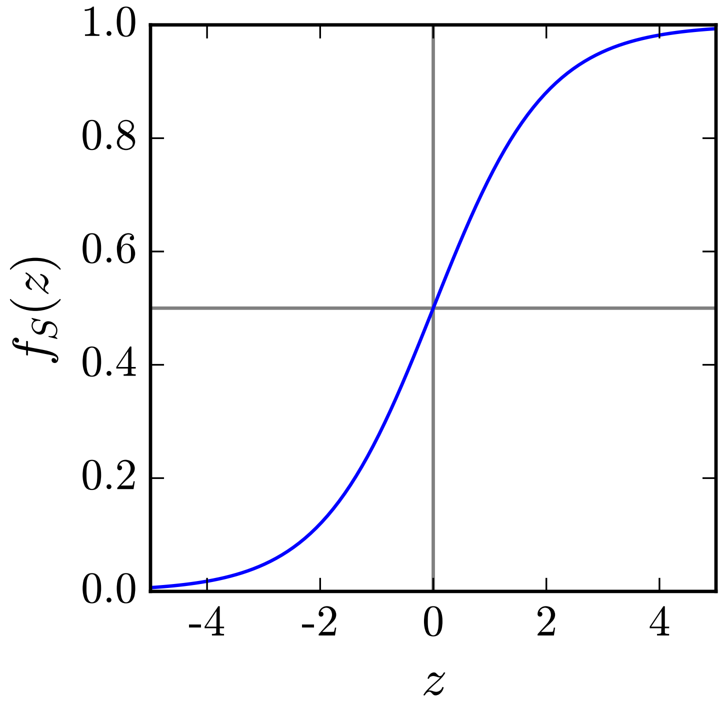
$$L(\vec{x},\vec{y},\vec{a}) = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log\Big(f_S(p(x,a))\Big) + (1-y_i)\log\Big(1 - f_S(p(x,a))\Big)\right)$$

What is $p(x,a)$?



Logistic Regression

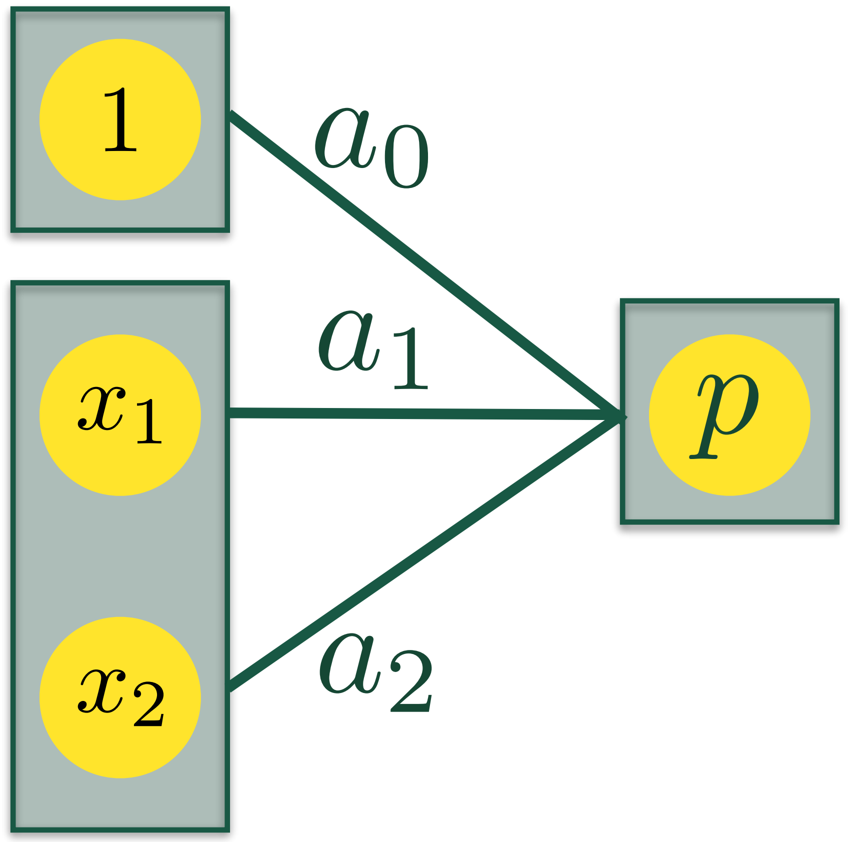
What if we are trying to predict a class, not a number?

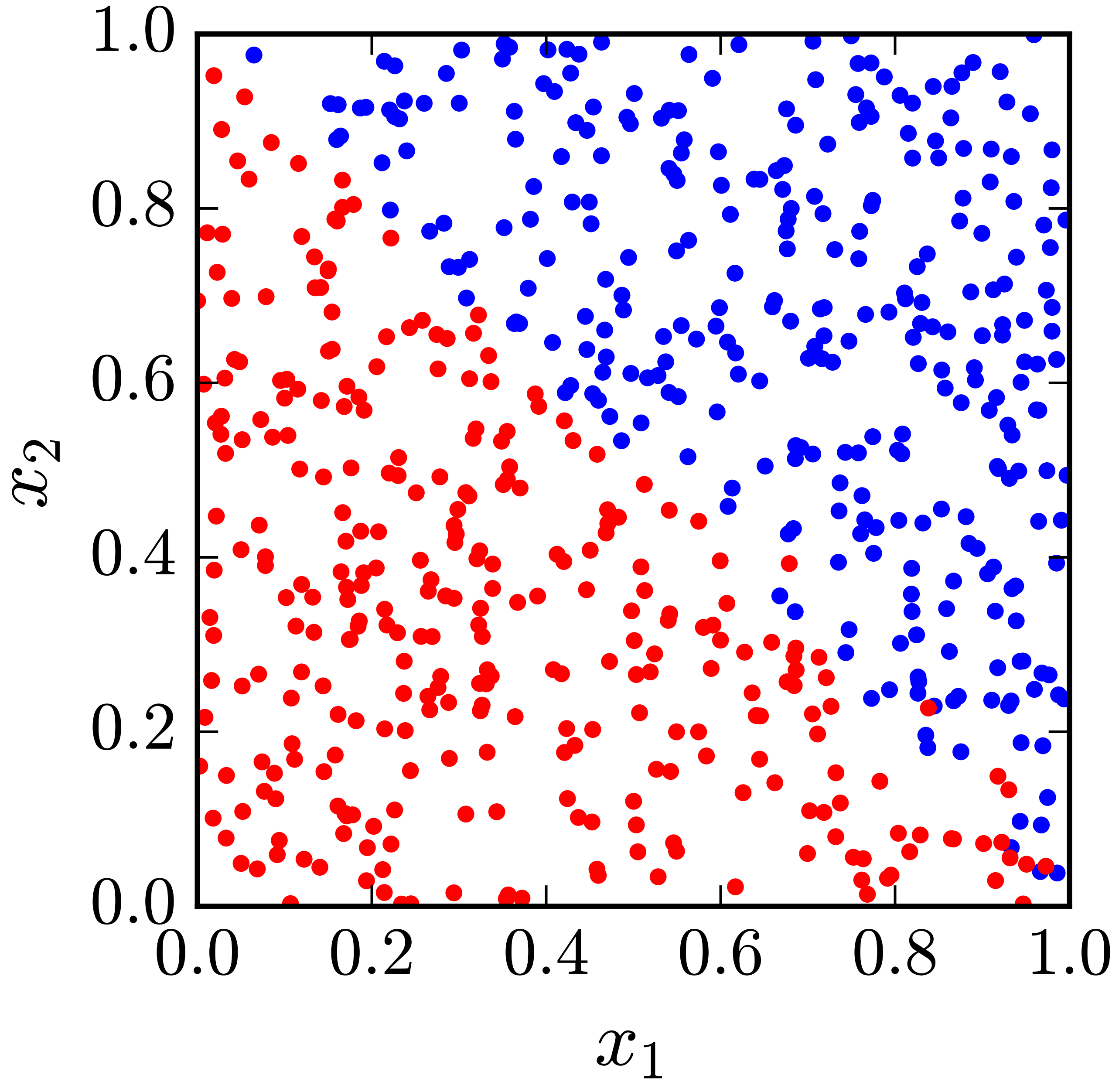


$$L(\vec{x},\vec{y},\vec{a}) = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log\left(f_S(p(x,a))\right) + (1-y_i)\log\left(1 - f_S(p(x,a))\right)\right)$$

What is $p(x,a)$?

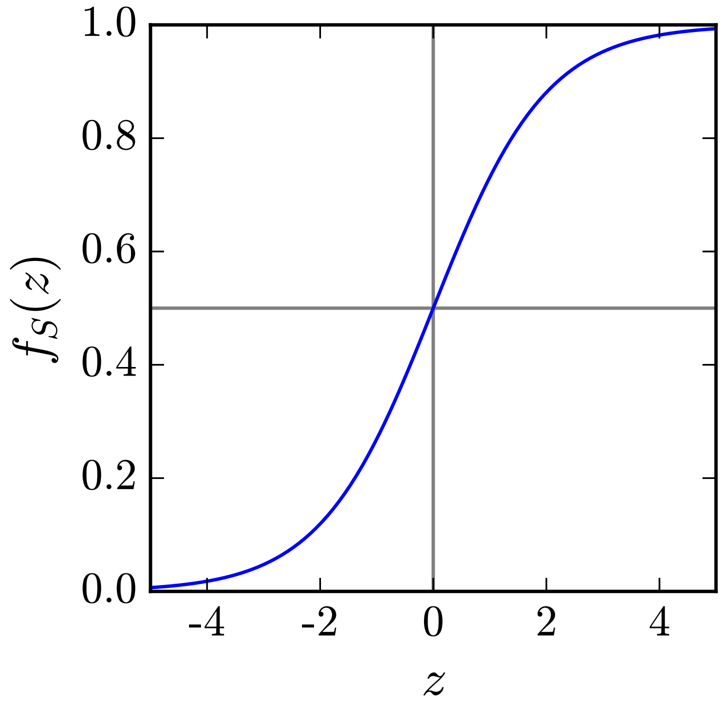
$$p(x,a) = a_0 + x_1 a_1 + x_2 a_2$$





Logistic Regression

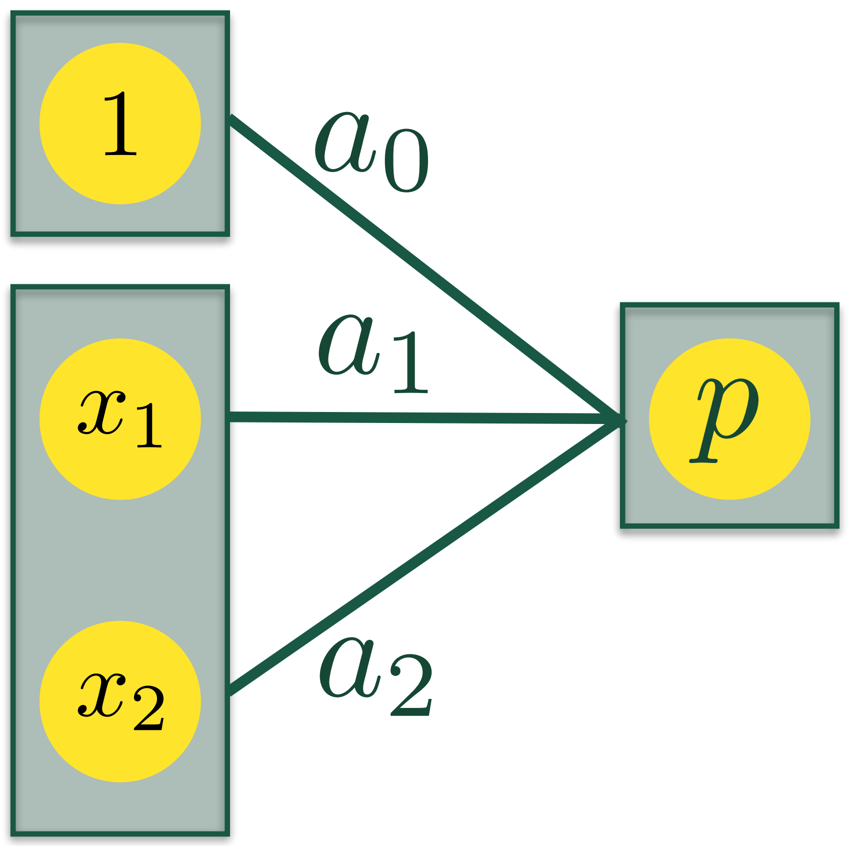
What if we are trying to predict a class, not a number?



$$L(\vec{x},\vec{y},\vec{a}) = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log\left(f_S(p(x,a))\right) + (1-y_i)\log\left(1-f_S(p(x,a))\right)\right)$$

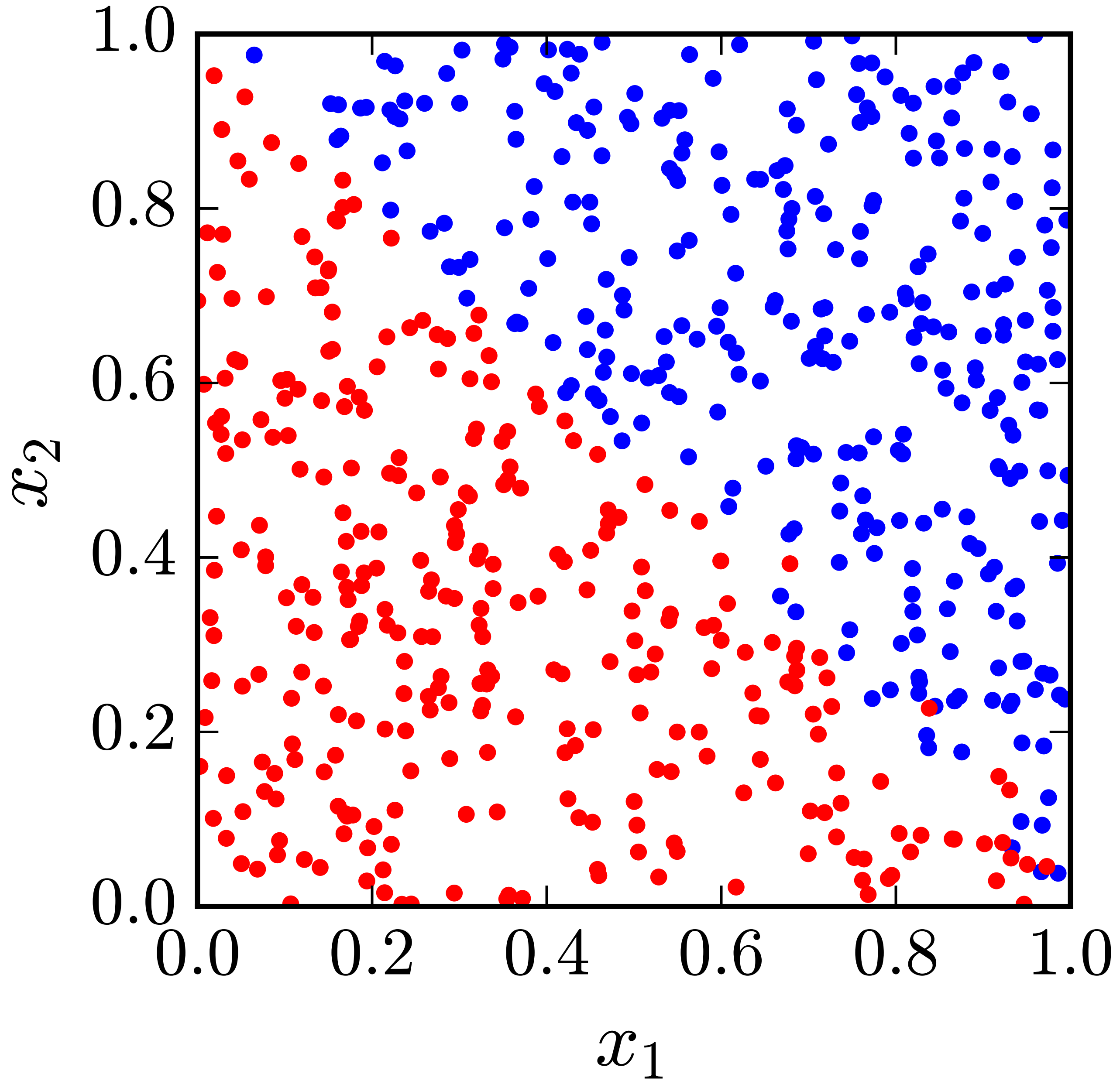
What is $p(x,a)$?

$$p(x,a) = a_0 + x_1 a_1 + x_2 a_2$$



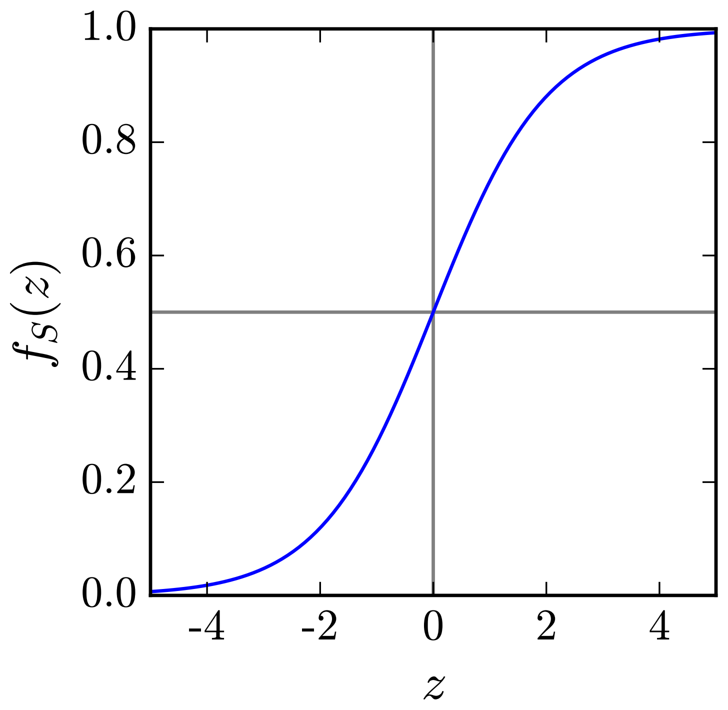
Minimize the loss with respect to $\vec{a}$

Boundary at $p(x,a) = 0$



Logistic Regression

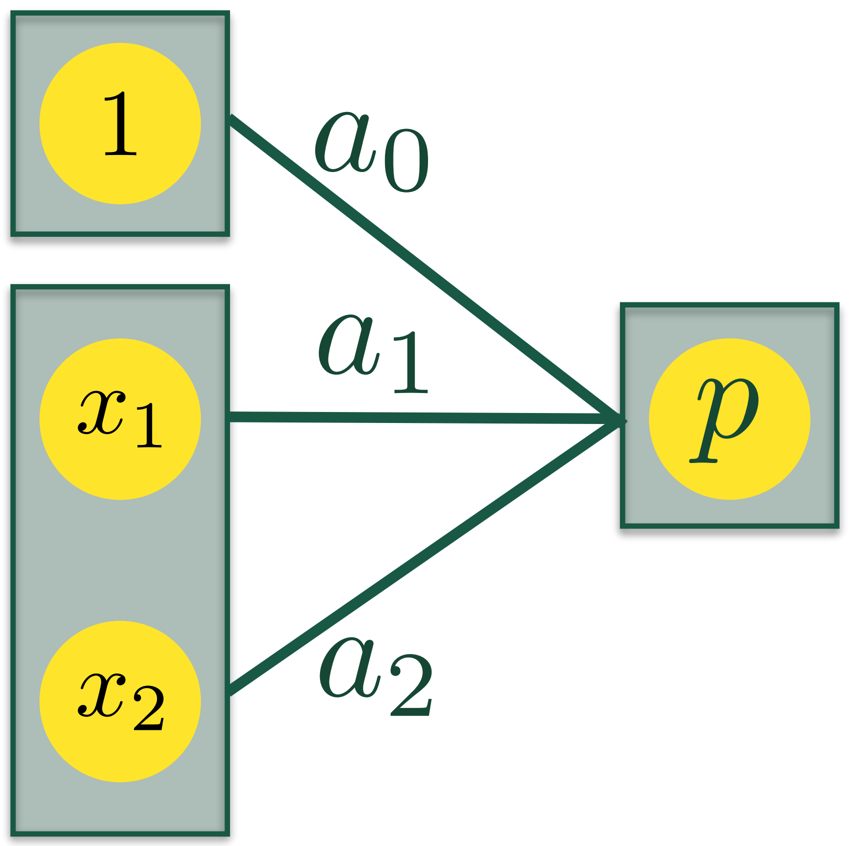
What if we are trying to predict a class, not a number?



$$L(\vec{x},\vec{y},\vec{a}) = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log\left(f_S(p(x,a))\right) + (1-y_i)\log\left(1 - f_S(p(x,a))\right)\right)$$

What is $p(x,a)$?

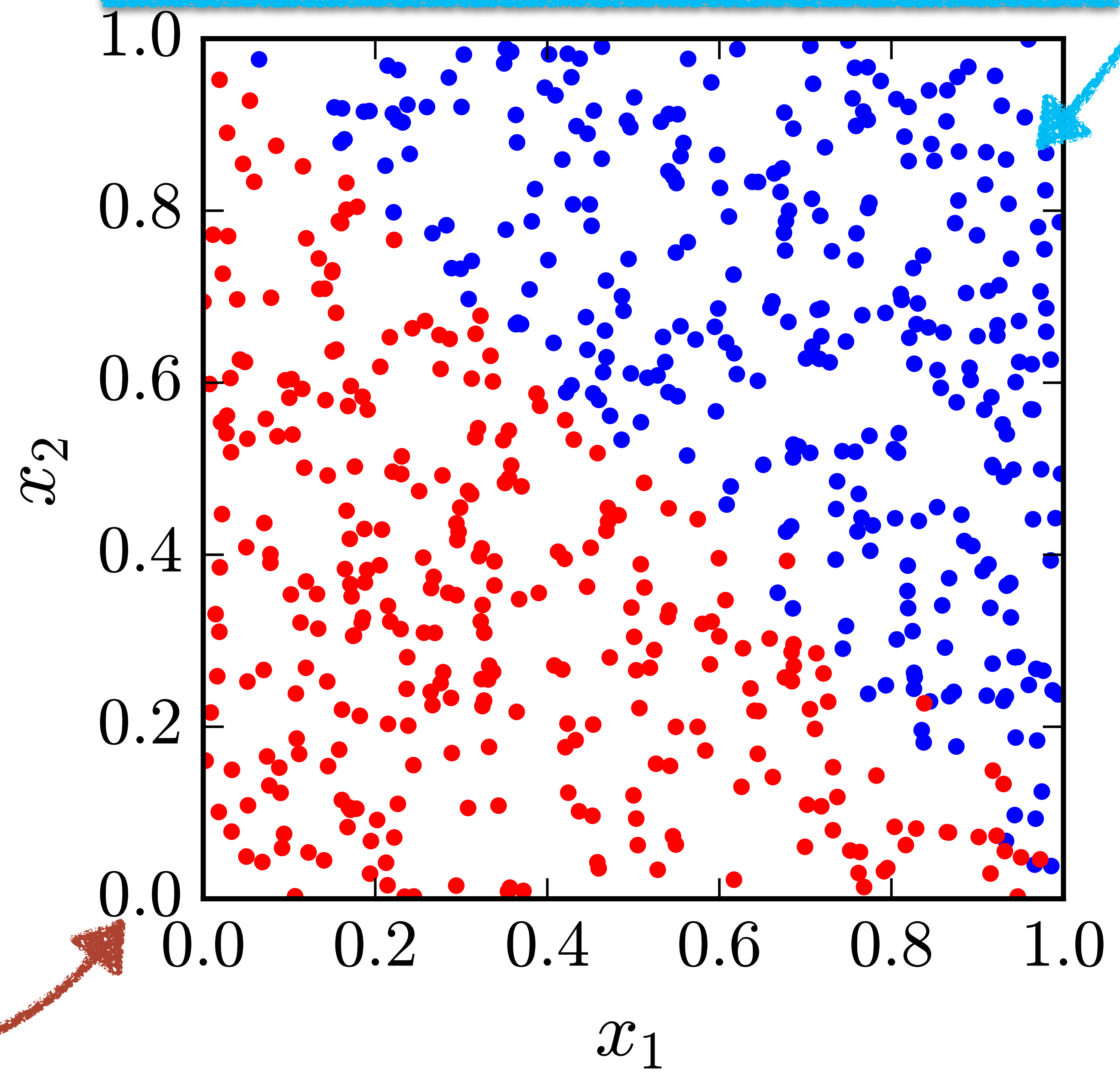
$$p(x,a) = a_0 + x_1 a_1 + x_2 a_2$$



Minimize the loss with respect to $\vec{a}$

Boundary at $p(x,a) = 0$

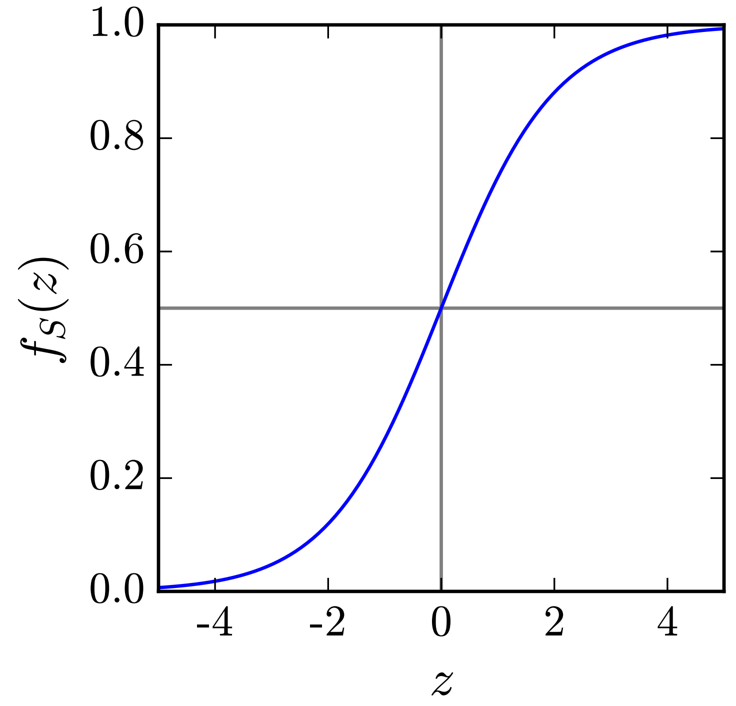
Large + values of p



Large - values of p

Logistic Regression

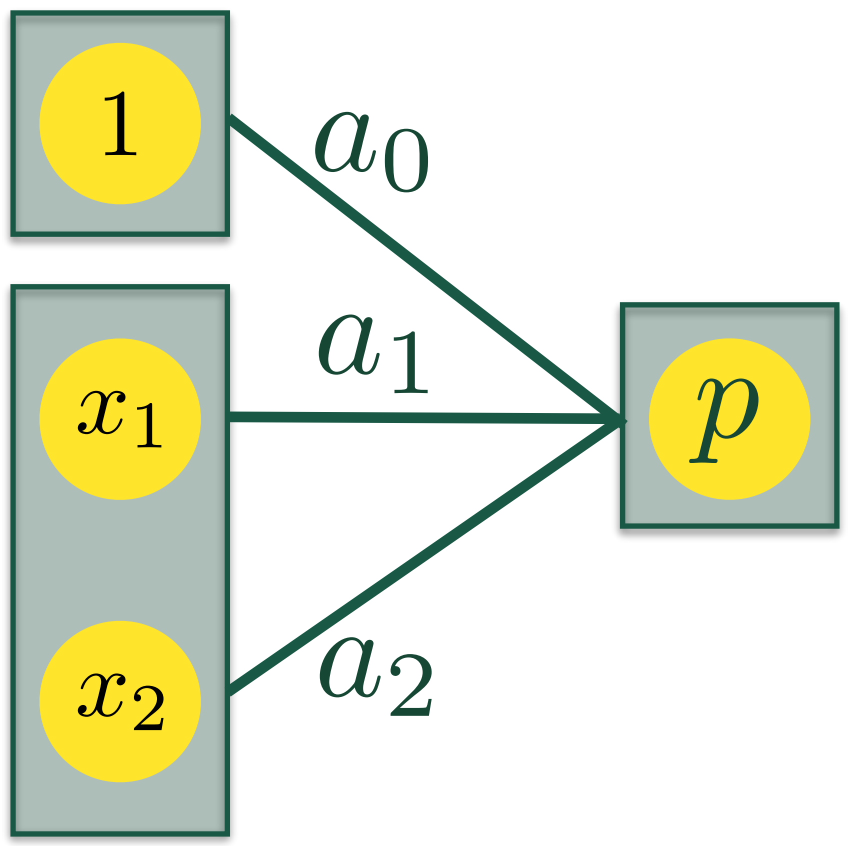
What if we are trying to predict a class, not a number?



$$L(\vec{x},\vec{y},\vec{a}) = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log\left(f_S(p(x,a))\right) + (1-y_i)\log\left(1 - f_S(p(x,a))\right)\right)$$

What is $p(x,a)$?

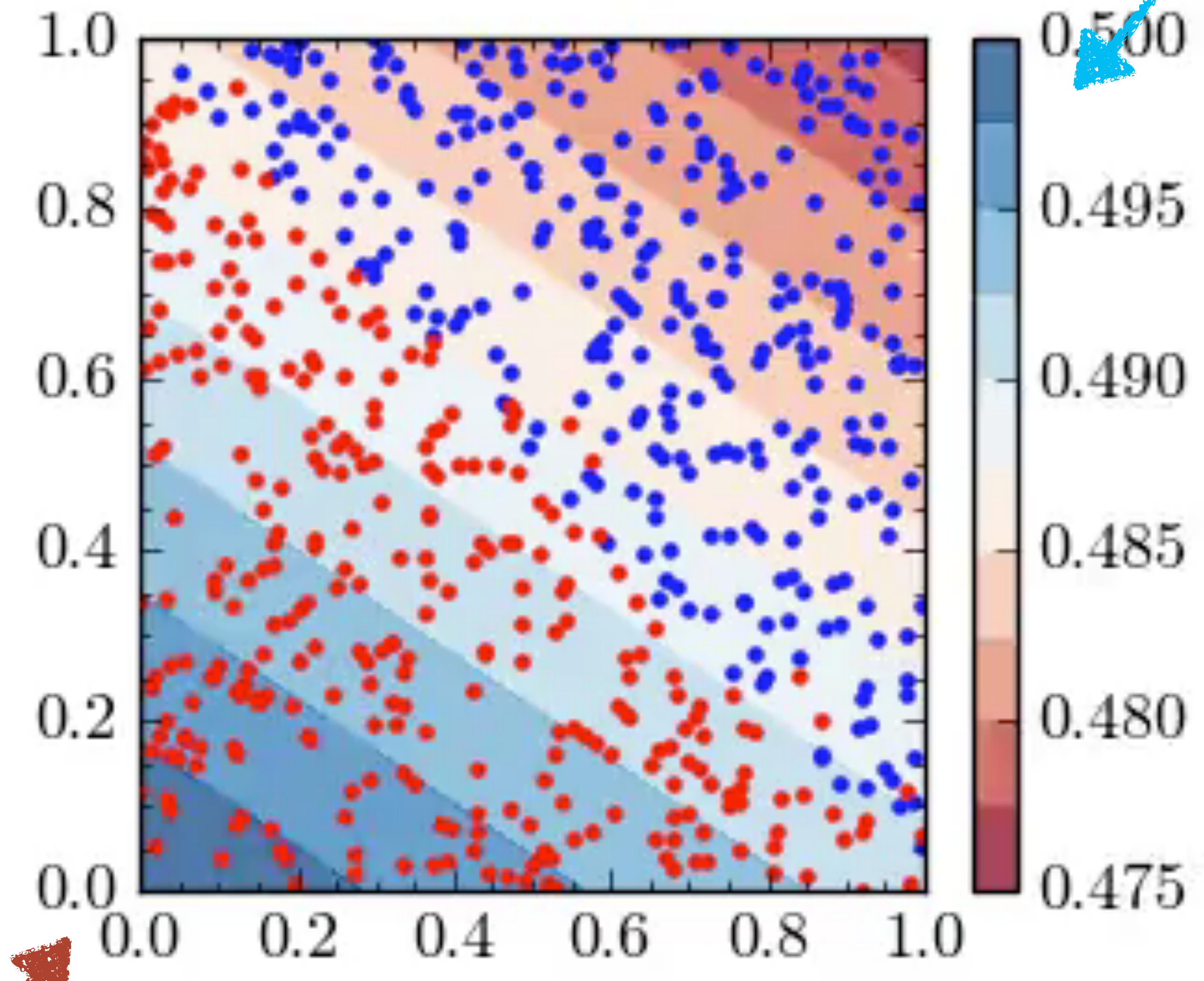
$$p(x,a) = a_0 + x_1 a_1 + x_2 a_2$$



Minimize the loss with respect to $\vec{a}$

Boundary at $p(x,a) = 0$

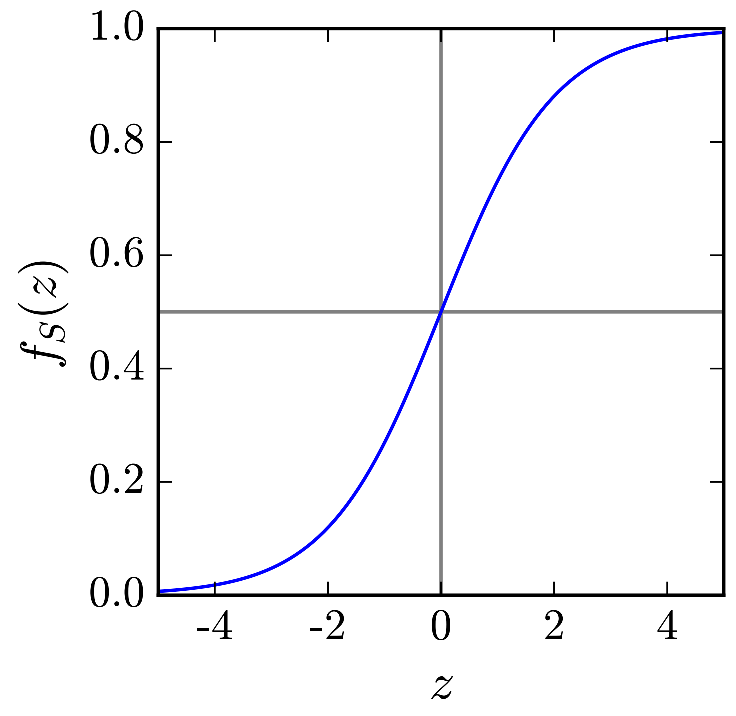
Large + values of p



Large - values of p

Logistic Regression

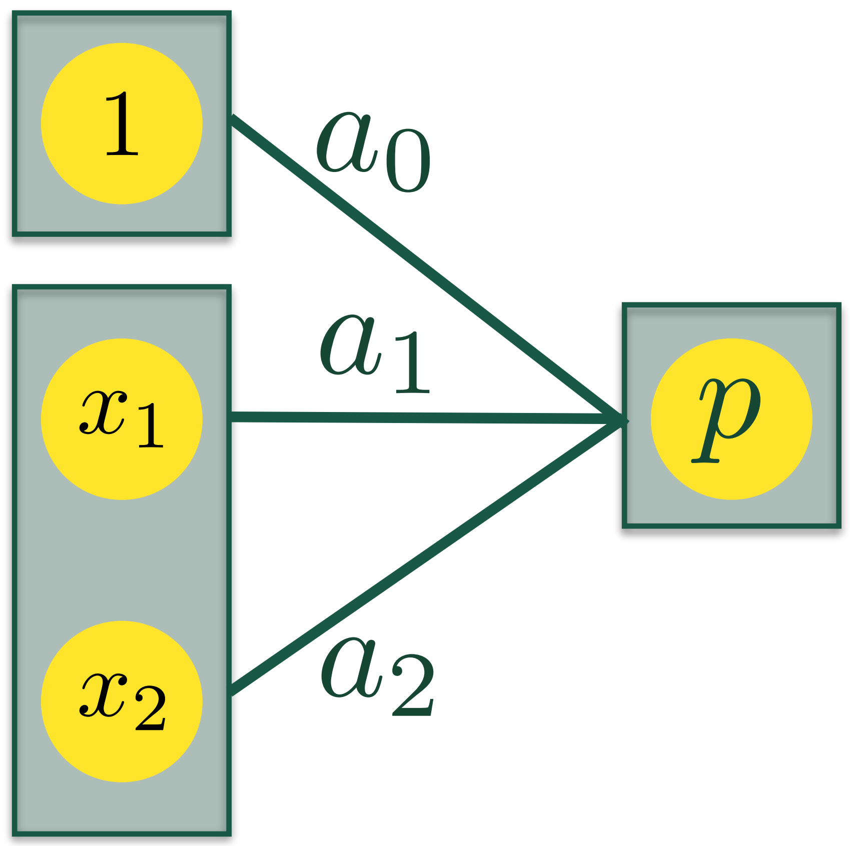
What if we are trying to predict a class, not a number?



$$L(\vec{x},\vec{y},\vec{a}) = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log\left(f_S(p(x,a))\right) + (1-y_i)\log\left(1 - f_S(p(x,a))\right)\right)$$

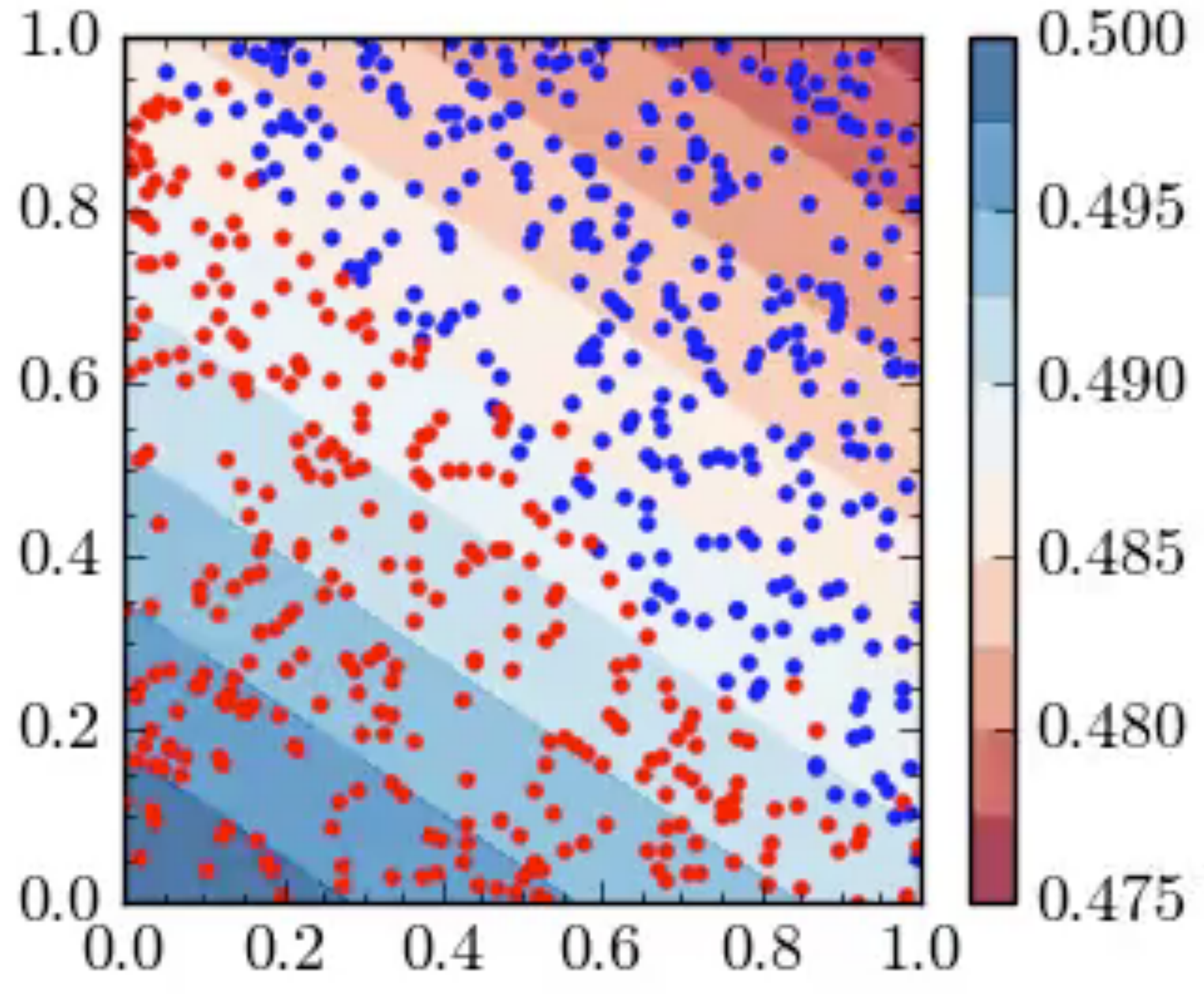
What is $p(x,a)$?

$$p(x,a) = a_0 + x_1 a_1 + x_2 a_2$$



Minimize the loss with respect to $\vec{a}$

Boundary at $p(x,a) = 0$

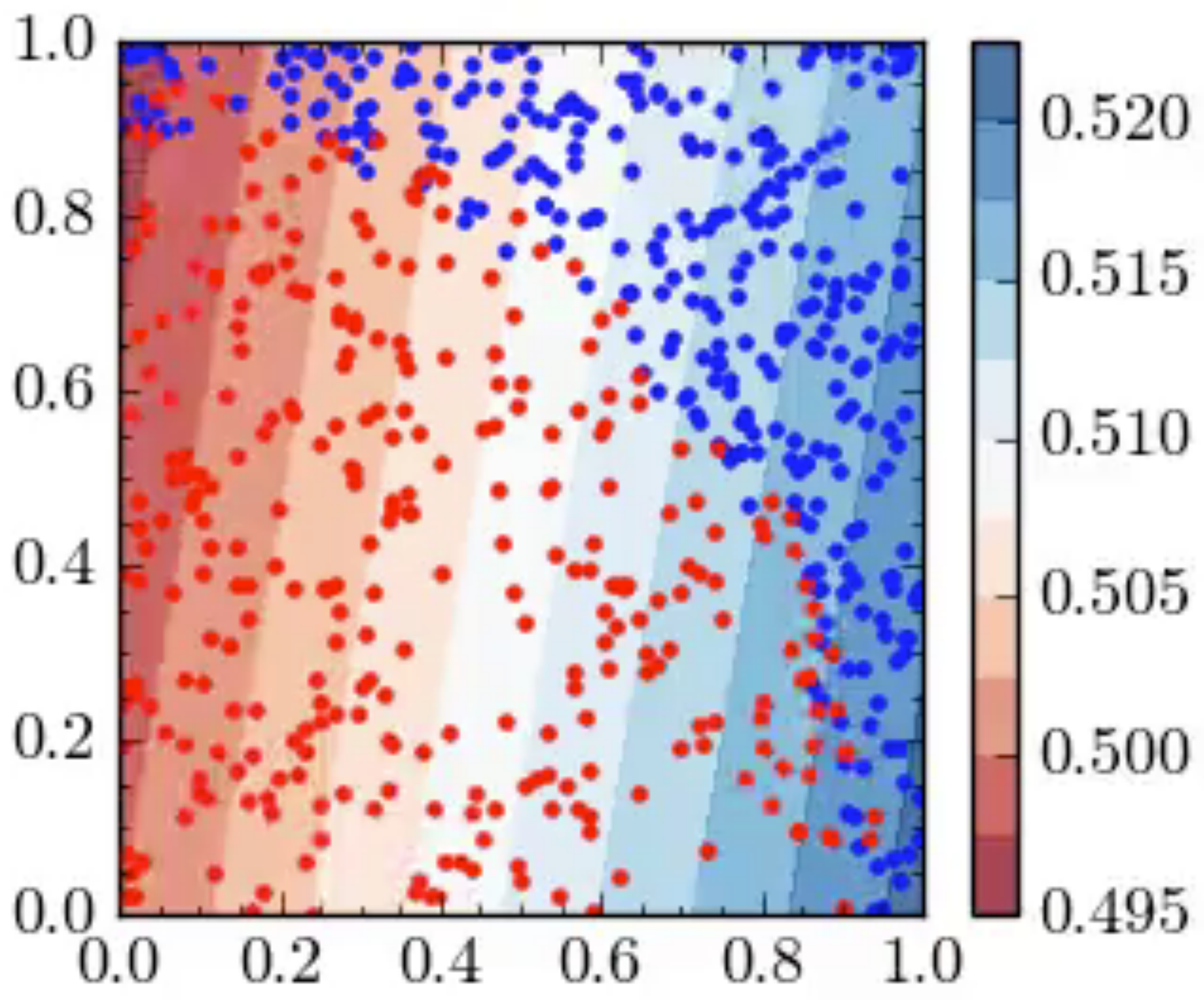


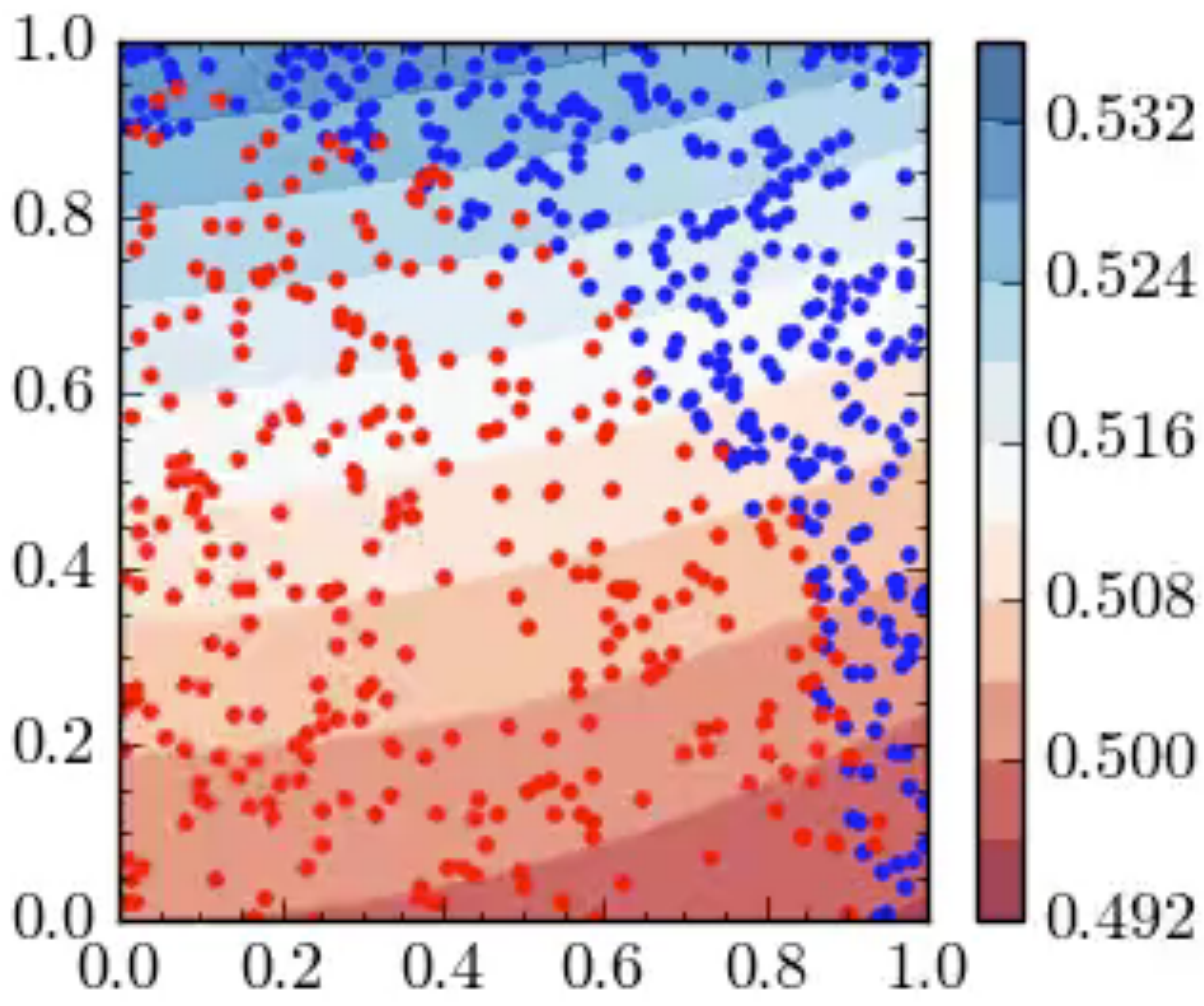
Logistic Regression

What if there is a shape in the data?

$$p(x,a) = a_0 + x_1 a_1 + x_2 a_2$$

$$\begin{aligned} p(x,a) = \ & a_0 + a_1 x_1 + a_2 x_2 \\ & + a_3 x_1^2 + a_4 x_2^2 + a_5 x_1 x_2 \end{aligned}$$



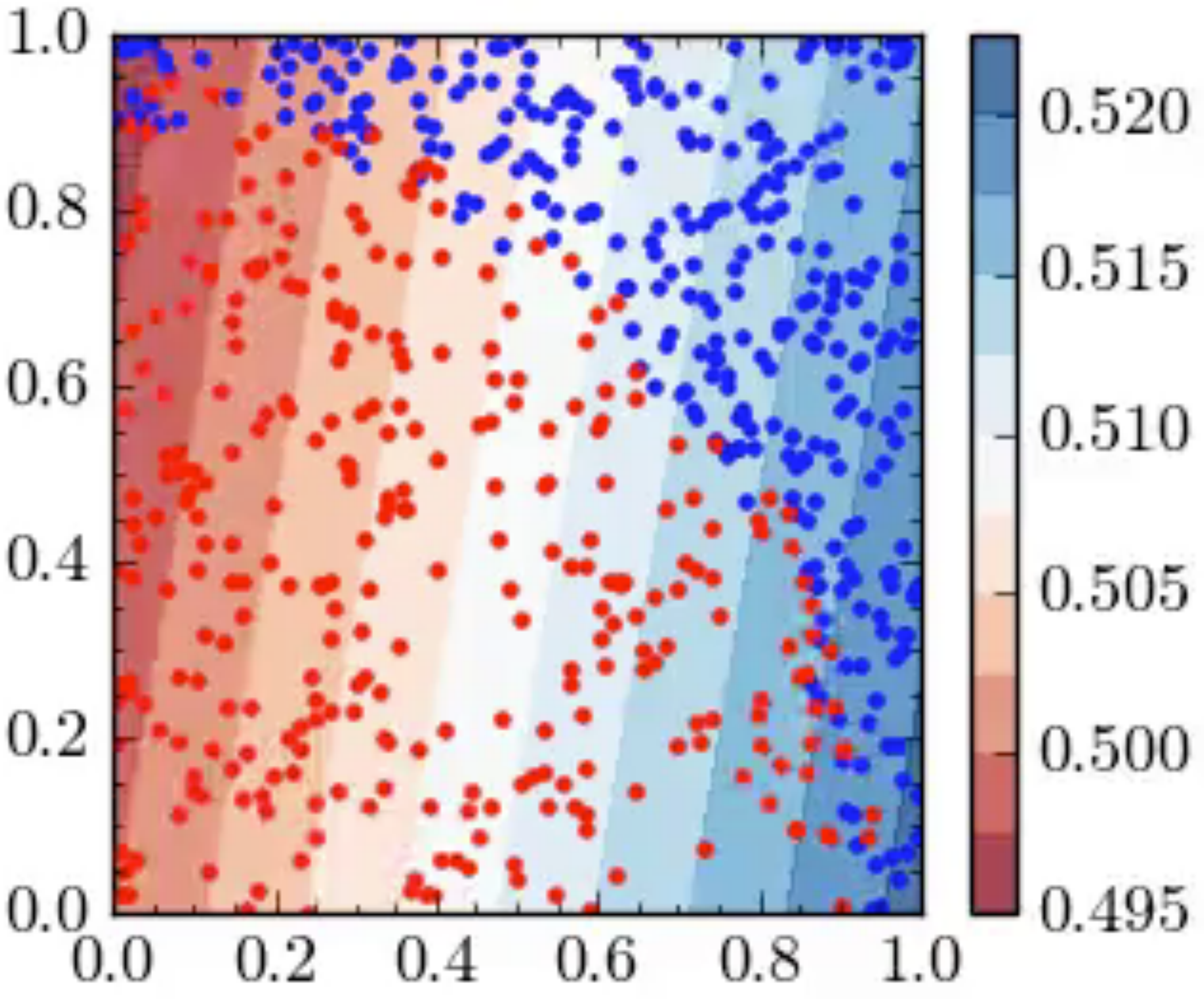


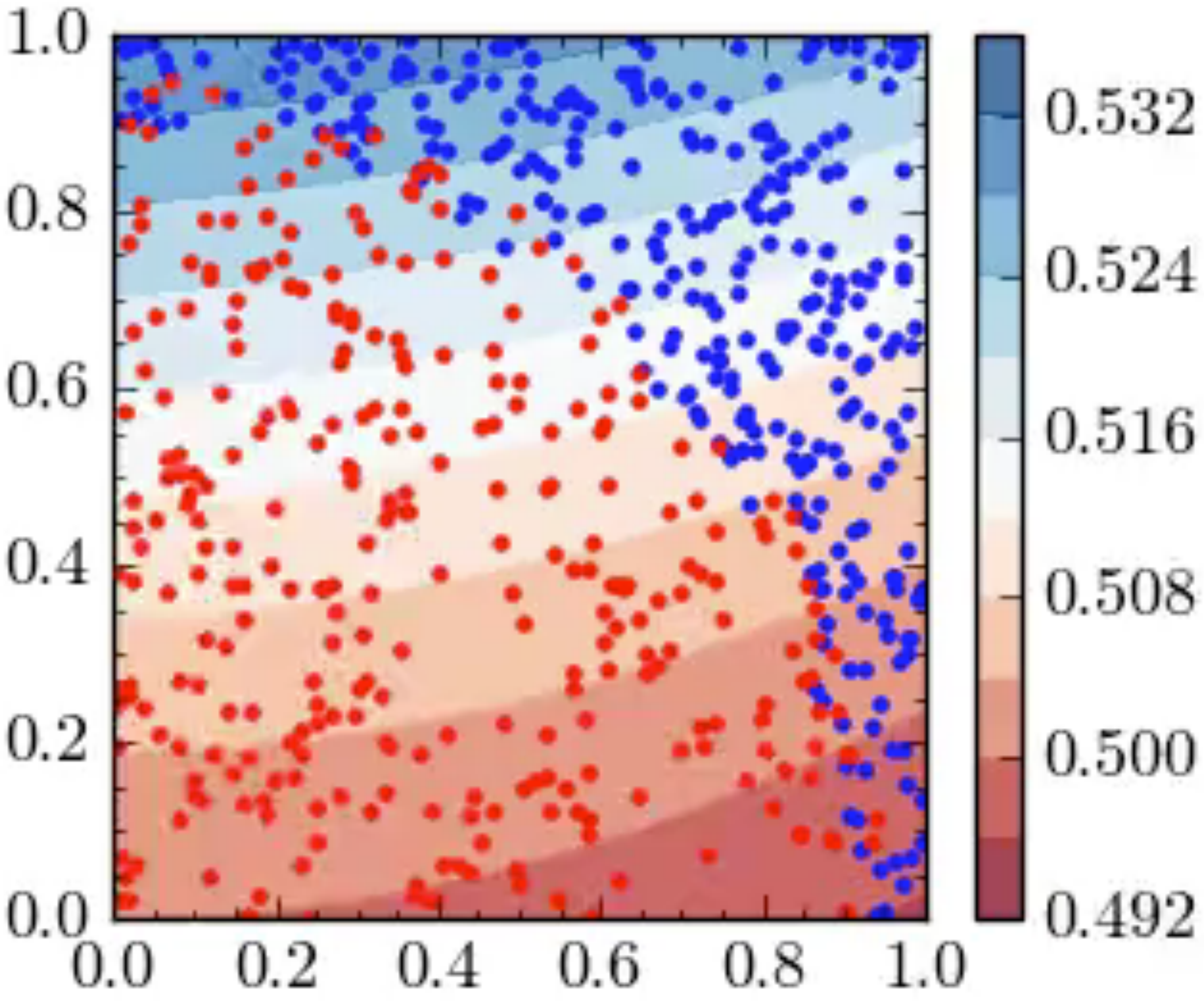
Logistic Regression

What if there is a shape in the data?

$$p(x,a) = a_0 + x_1 a_1 + x_2 a_2$$

$$p(x,a) = a_0 + a_1 x_1 + a_2 x_2 + a_3 x_1^2 + a_4 x_2^2 + a_5 x_1 x_2$$





Regression Review

Ideal

1. Choose physically motivated model
2. Learn best-fit parameters by minimizing loss function

Regression Review

Ideal

1. Choose physically motivated model
2. Learn best-fit parameters by minimizing loss function

How to deal with many inputs and hard to describe shapes in the data?

Regression Review

Ideal

1. Choose physically motivated model
2. Learn best-fit parameters by minimizing loss function

Create new variables/observables

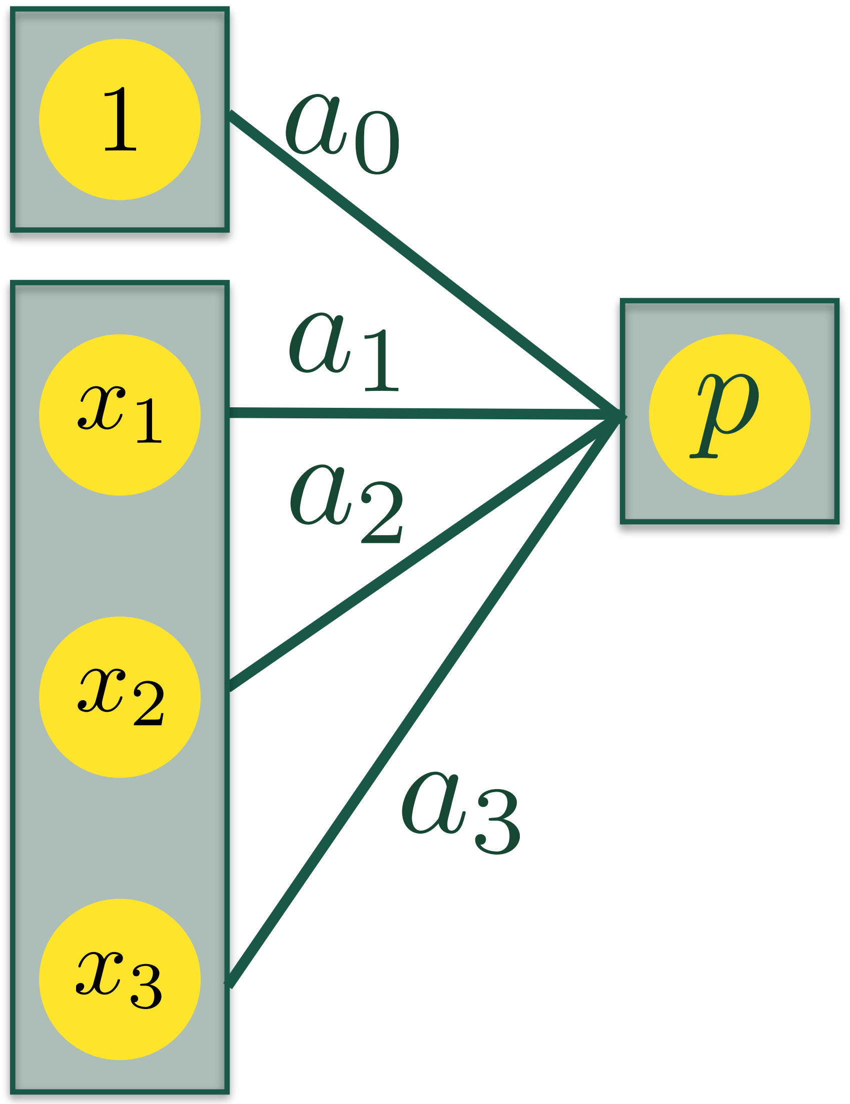
Let the machine choose the model

How to deal with many inputs and hard to describe shapes in the data?

Neural Networks

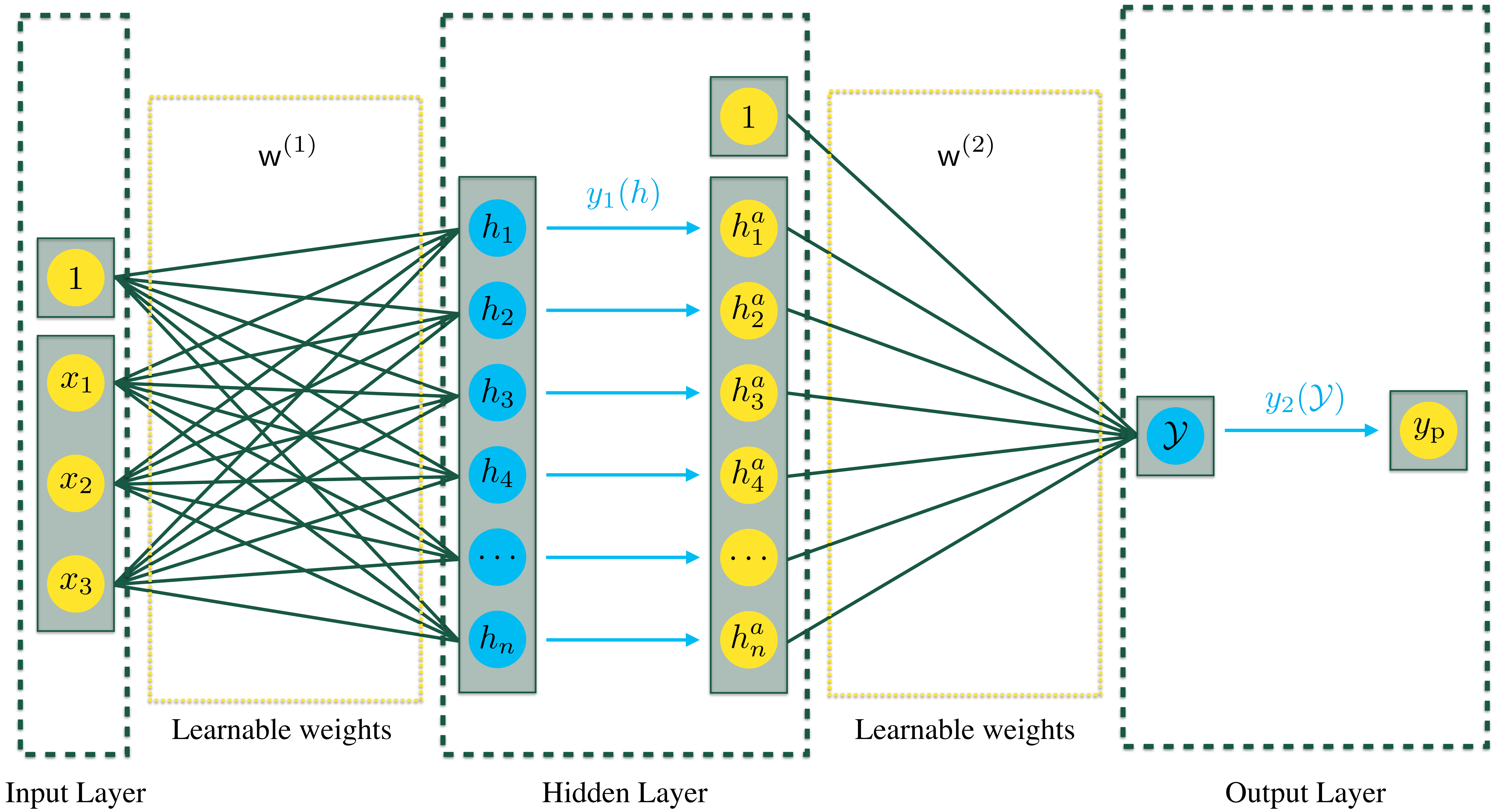
Neural Networks

- Can be used in classifications and numerical predictions
- Don't add more inputs, let machine find own shape
- Ability to learn 'any' function
- More nodes/hidden layers allows for more complex features



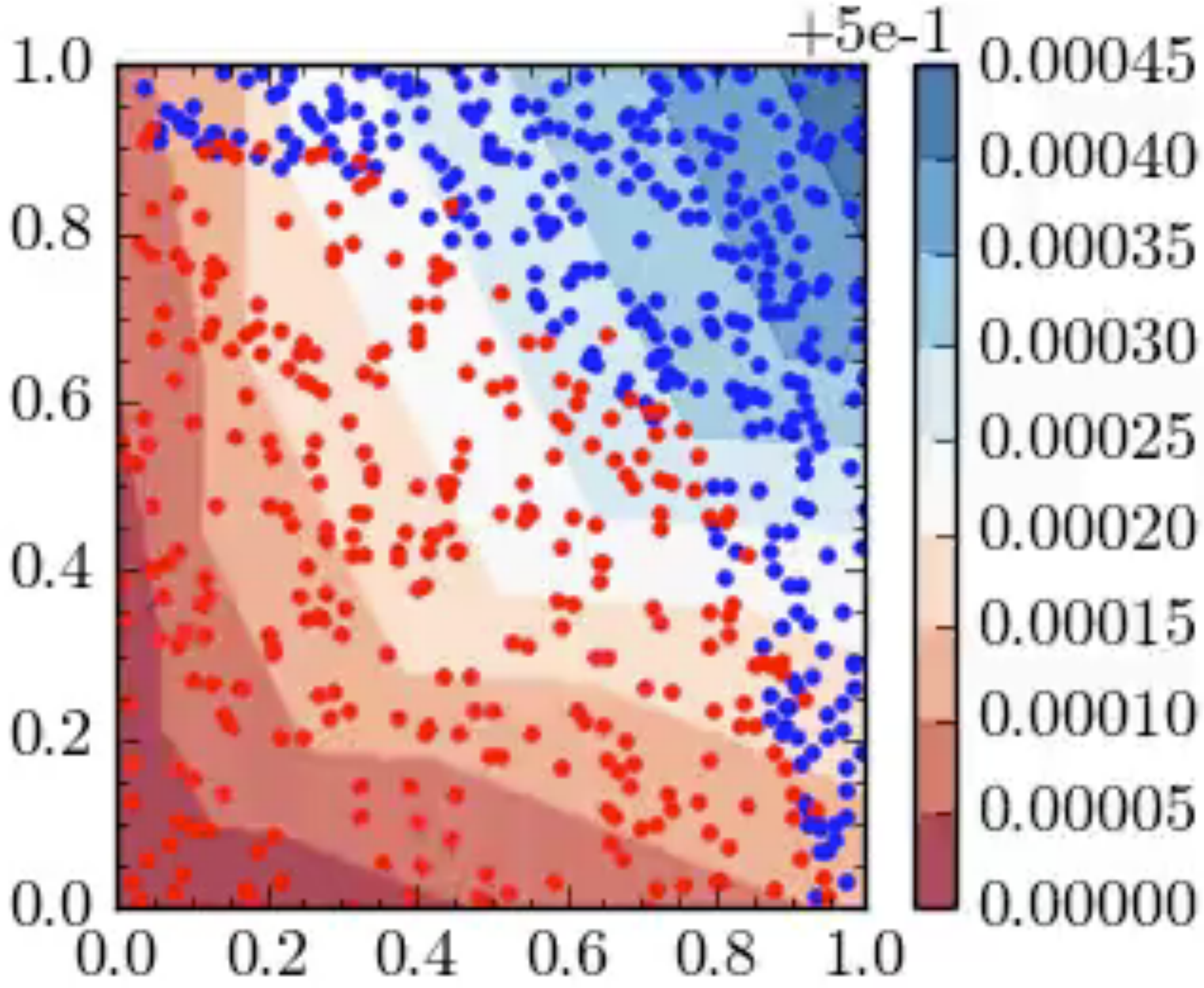
Neural Networks

- Can be used in classifications and numerical predictions
- Don't add more inputs, let machine find own shape
- Ability to learn 'any' function
- More nodes/hidden layers allows for more complex features



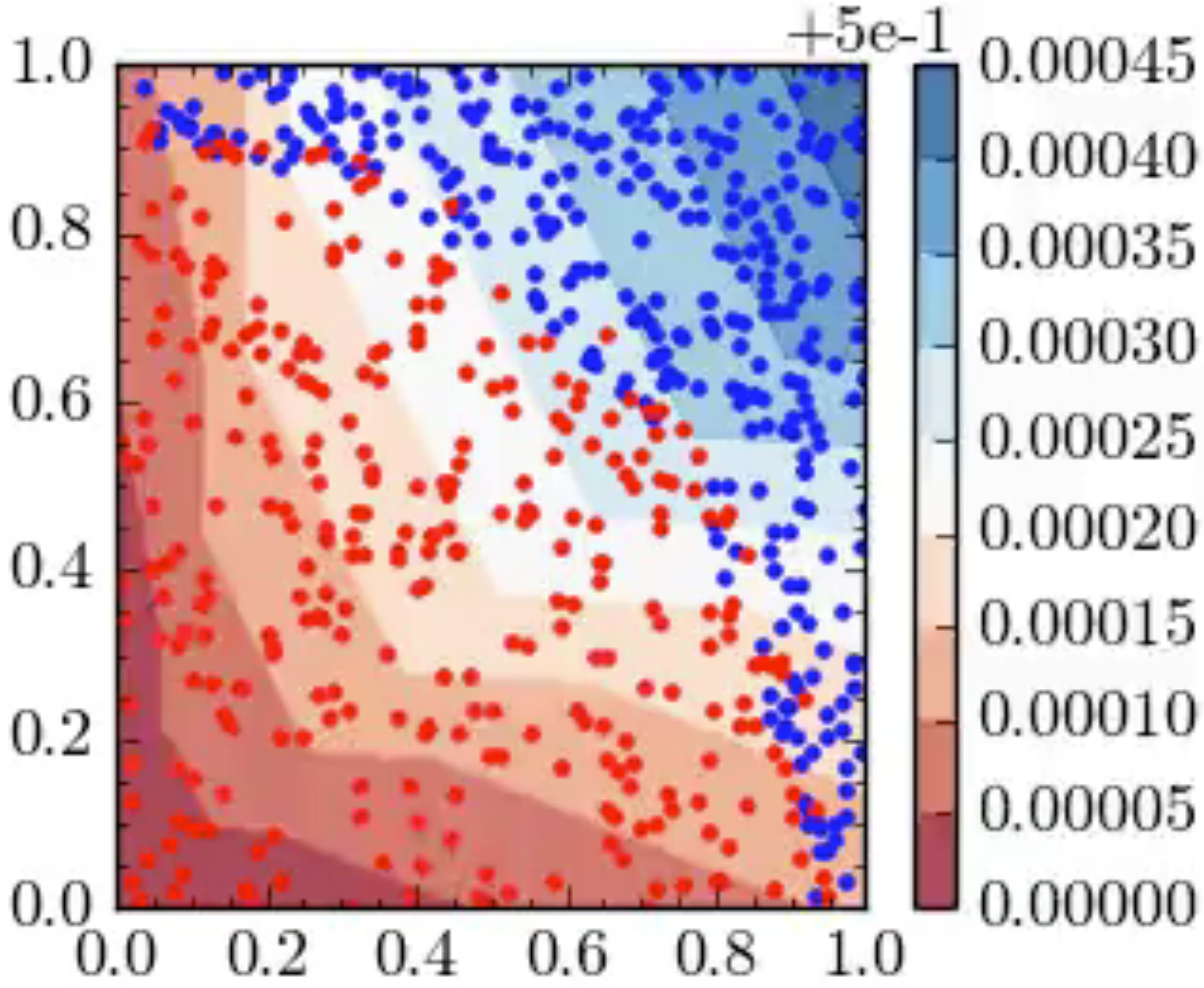
Neural Networks

- Can be used in classifications and numerical predictions
- Don't add more inputs, let machine find own shape
- Ability to learn 'any' function
- More nodes/hidden layers allows for more complex features



Neural Networks

- Can be used in classifications and numerical predictions
- Don't add more inputs, let machine find own shape
- Ability to learn 'any' function
- More nodes/hidden layers allows for more complex features



Neural Networks (Deep Learning)

ARTICLE

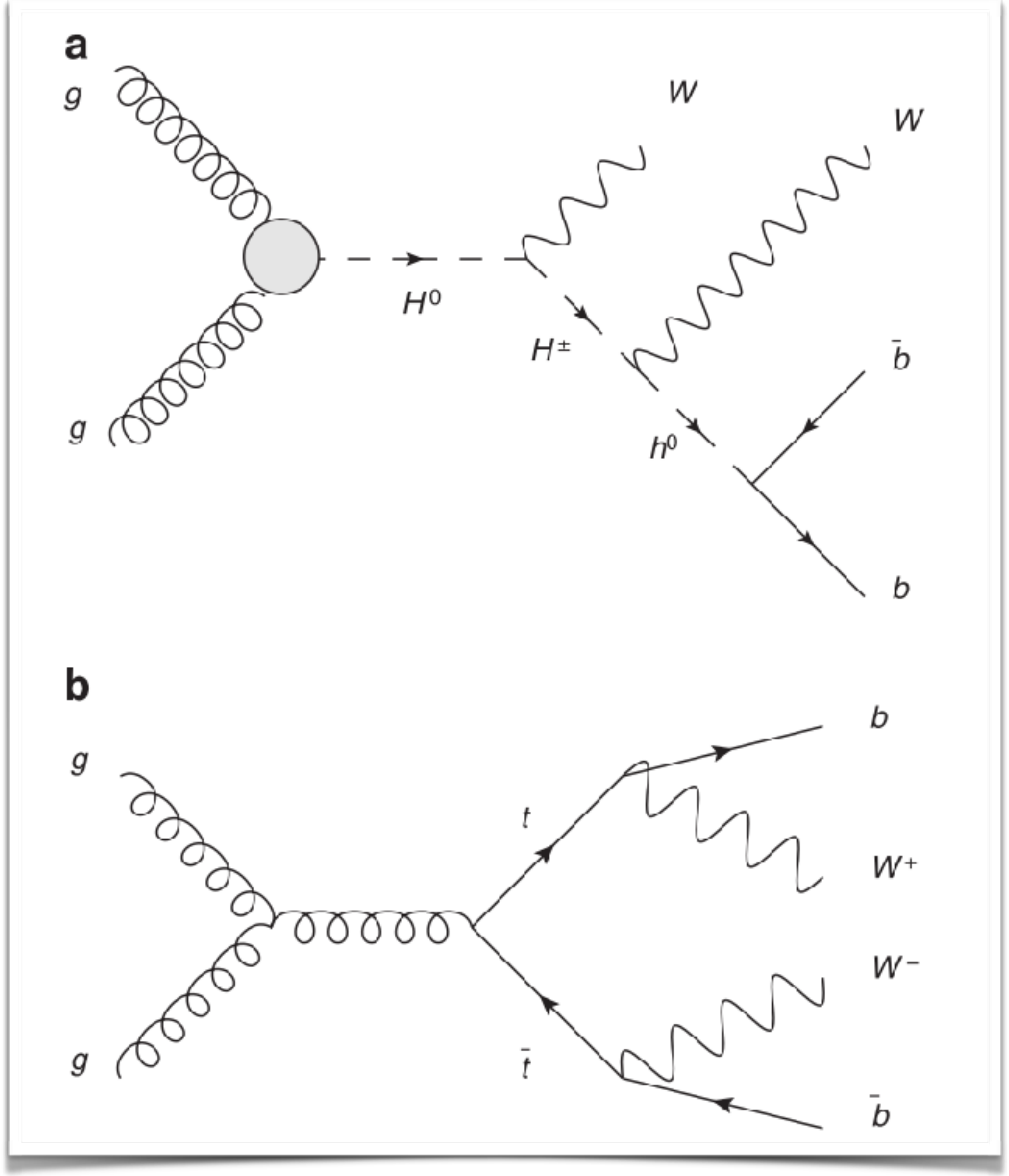
Received 19 Feb 2014 | Accepted 4 Jun 2014 | Published 2 Jul 2014 DOI: 10.1038/ncomms5308

Searching for exotic particles in high-energy physics with deep learning

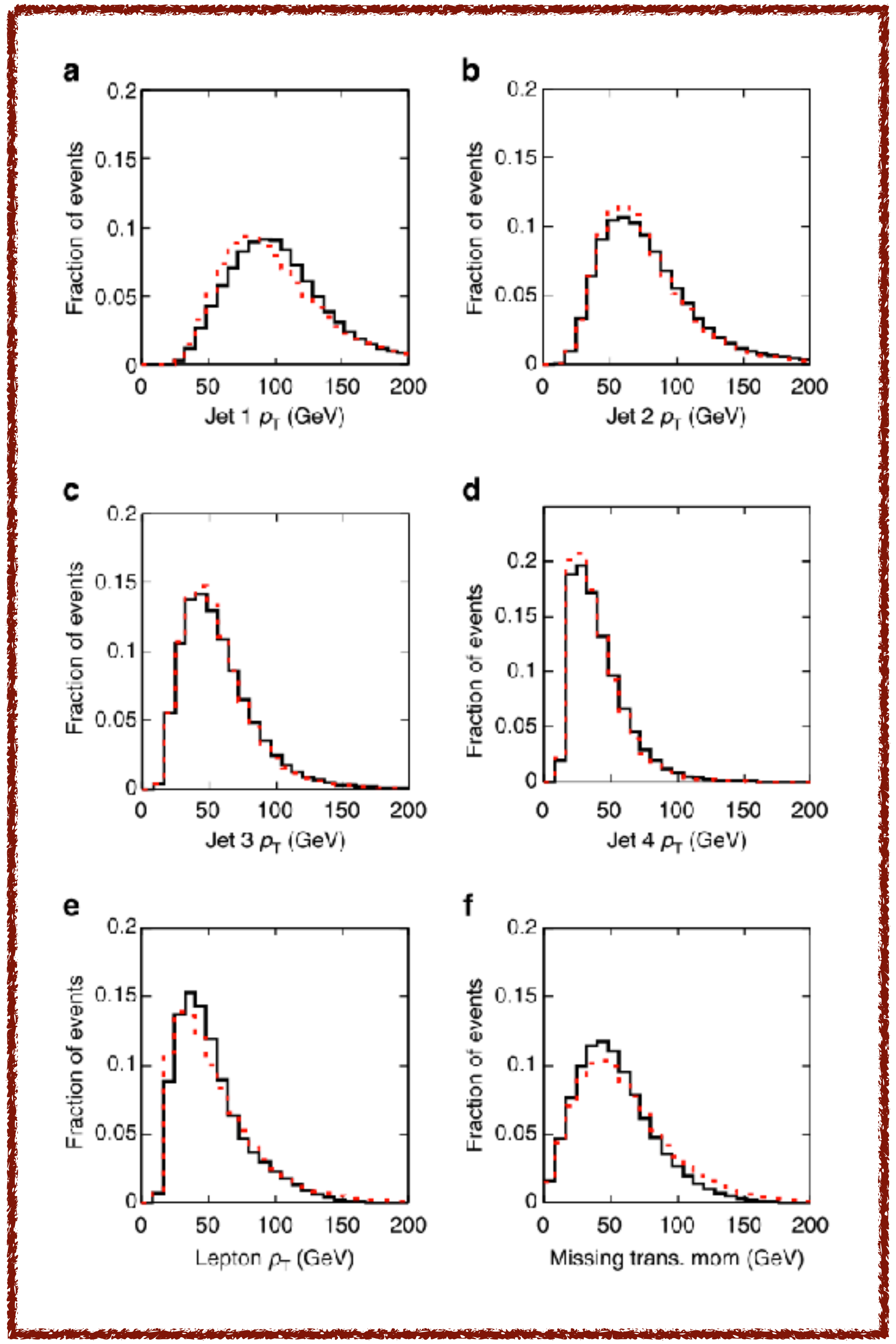
P. Baldi[1], P. Sadowski[1] & D. Whiteson[2]

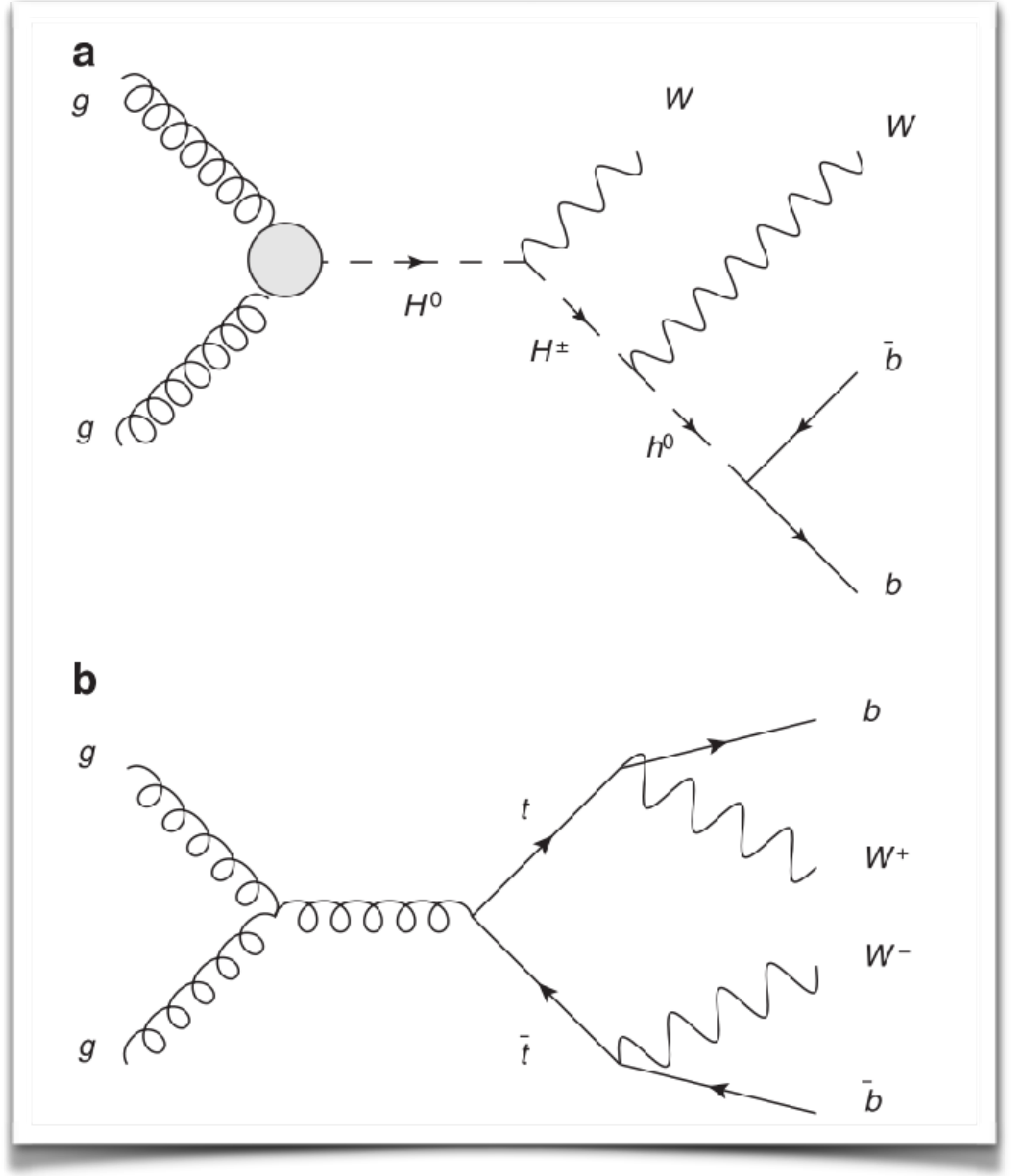
[1402.4735]

- One of first papers to show ***deep learning*** <u>outperforming</u> standard techniques in HEP
- Compares shallow and deep networks on raw and high-level features



Neural Networks (Deep Learning)

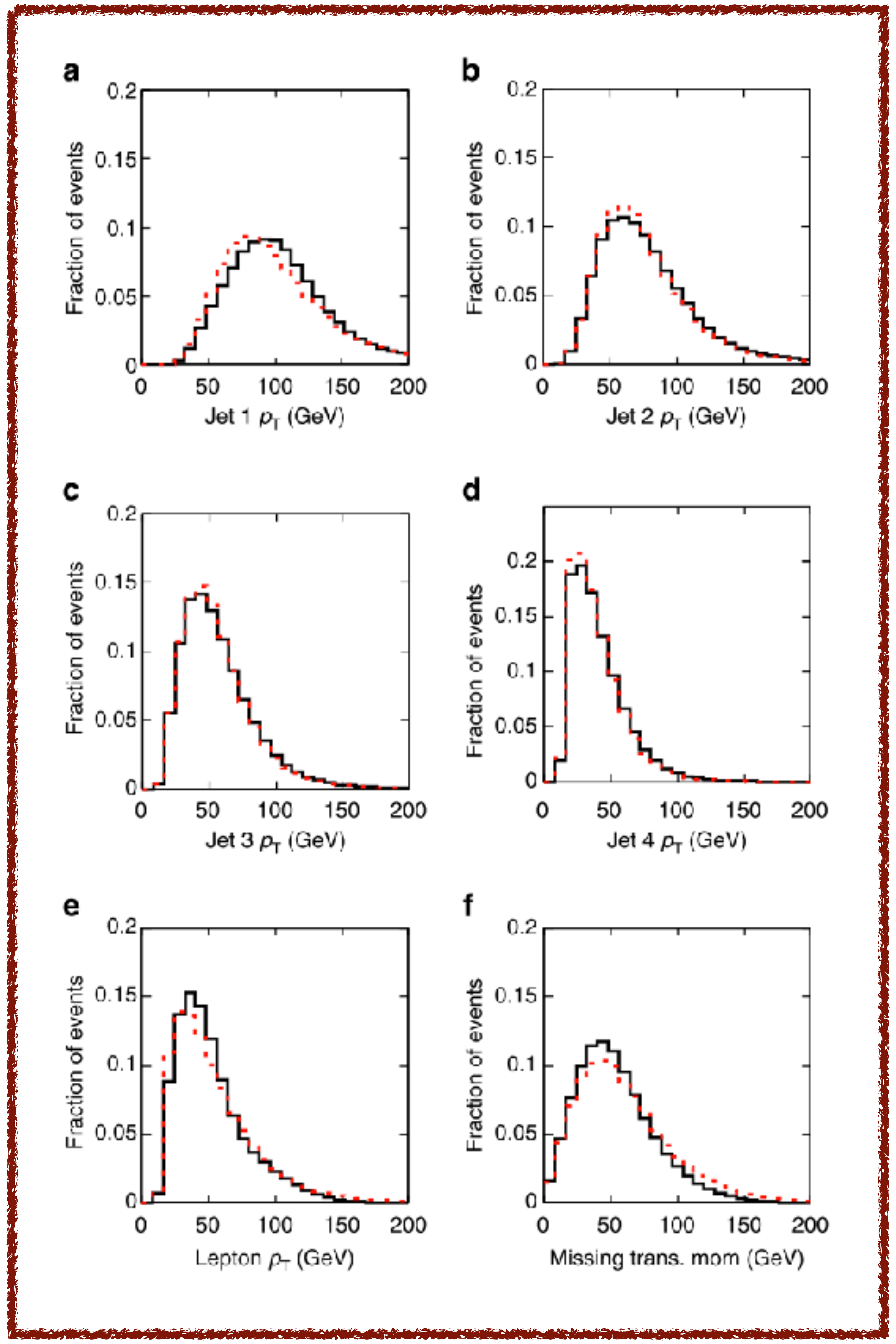


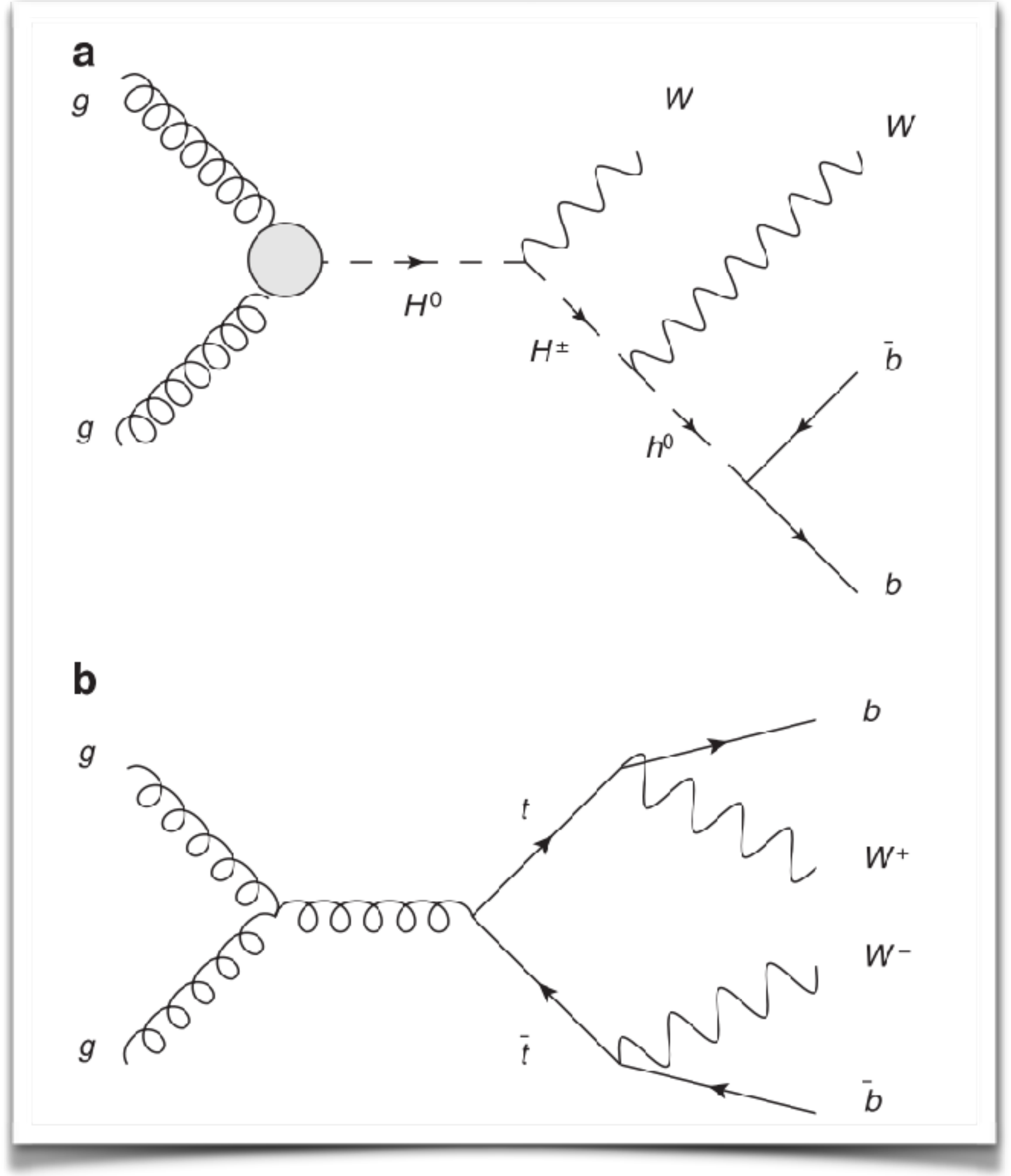


21 raw features for semi-leptonic channel

Not much separation in individual features

Neural Networks (Deep Learning)

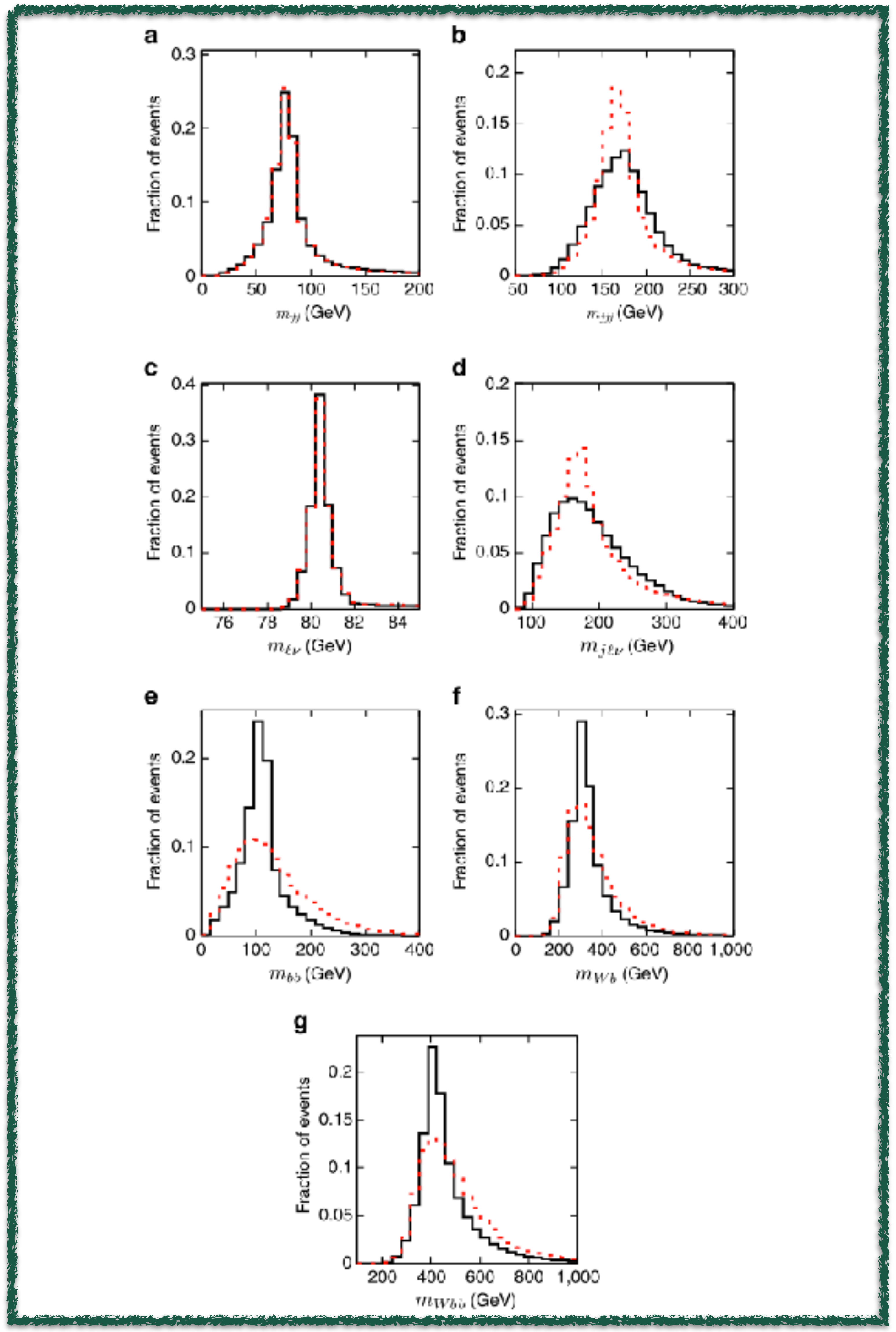




Invariant masses of intermediate sates

$$m_{jj} \quad m_{jjj} \quad m_{\ell\nu} \quad m_{j\ell\nu}$$

$$m_{bb} \quad m_{Wb} \quad m_{Wbb}$$



Neural Networks (Deep Learning)

11 million training examples
1 hidden layer shallow network
5 layer deep network

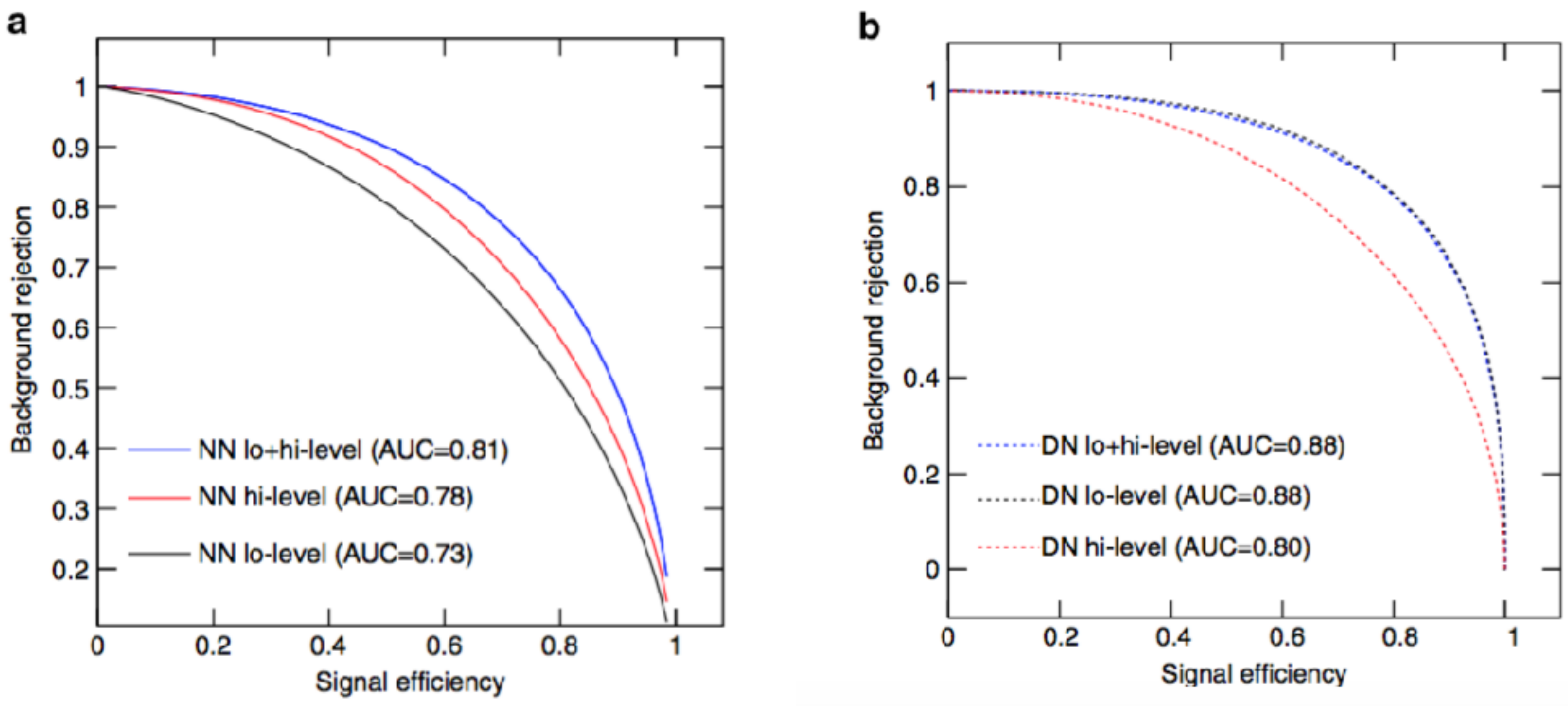


Table 1 | Performance for Higgs benchmark.

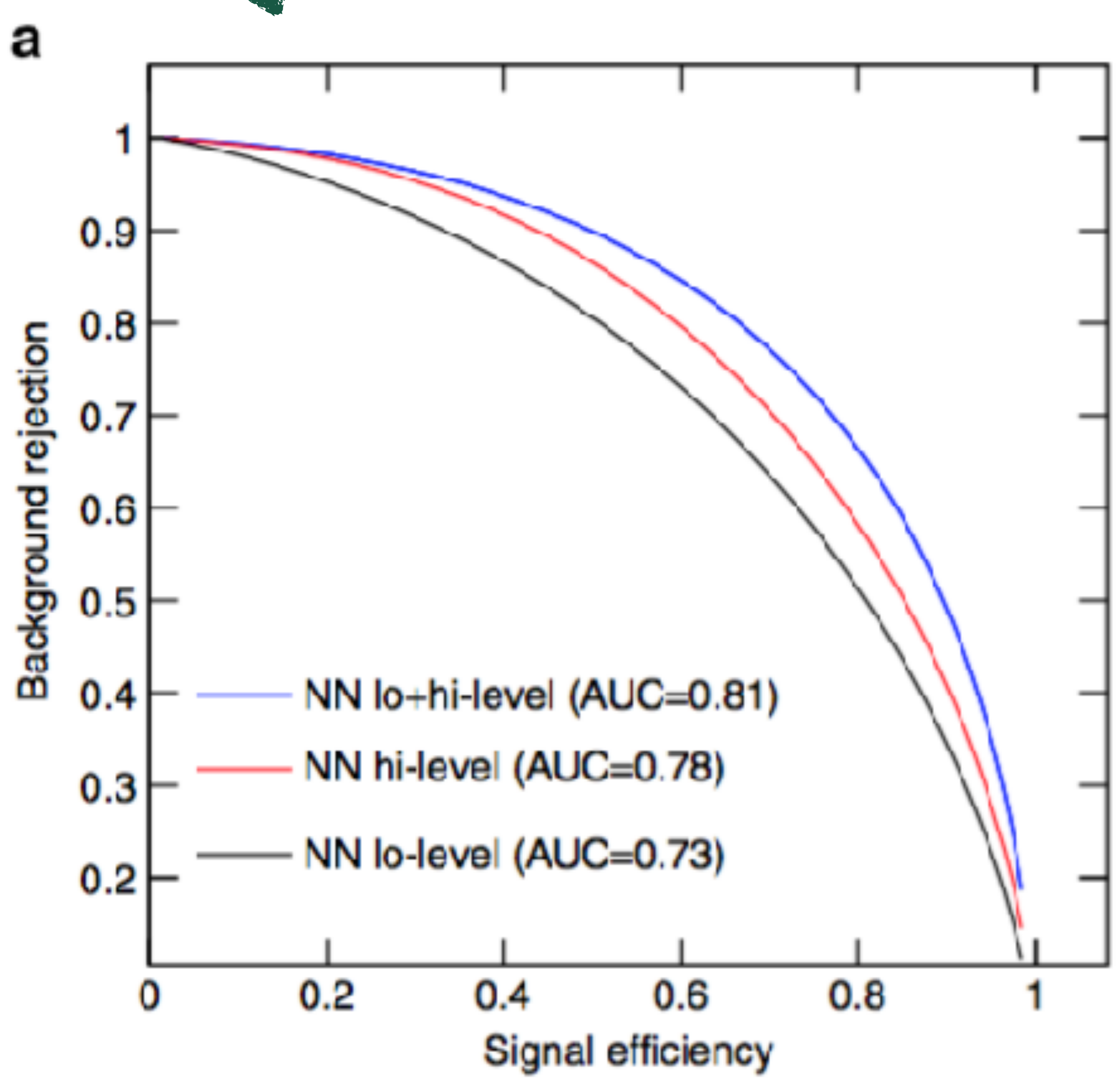
Technique	Low-level	High-level	Complete
AUC			
BDT	0.73 (0.01)	0.78 (0.01)	0.81 (0.01)
NN	0.733 (0.007)	0.777 (0.001)	0.816 (0.004)
DN	0.880 (0.001)	0.800 (<0.001)	0.885 (0.002)
Discovery significance			
NN	2.5σ	3.1σ	3.7σ
DN	4.9σ	3.6σ	5.0σ

Comparison of the performance of several learning techniques: boosted decision trees (BDT), shallow neural networks (NN), and deep neural networks (DN) for three sets of input features: low-level features, high-level features and the complete set of features. Each neural network was trained five times with different random initializations. The table displays the mean area under the curve (AUC) of the signal-rejection curve in Fig. 7, with s.d. in parentheses as well as the expected significance of a discovery (in units of Gaussian σ) for 100 signal events and $1,000 \pm 50$ background events.

Methods trained with only the high-level features, however, have a weaker performance than those trained with the full suite of features, which suggests that despite the insight represented by the high-level features, they do not capture all the information contained in the low-level features. The deep-learning techniques show nearly equivalent performance using the low-level features and the complete features, suggesting that they are automatically discovering the insight contained in the high-level features.

Neural Networks (Deep Learning)

11 million training examples
1 hidden layer shallow network
5 layer deep network



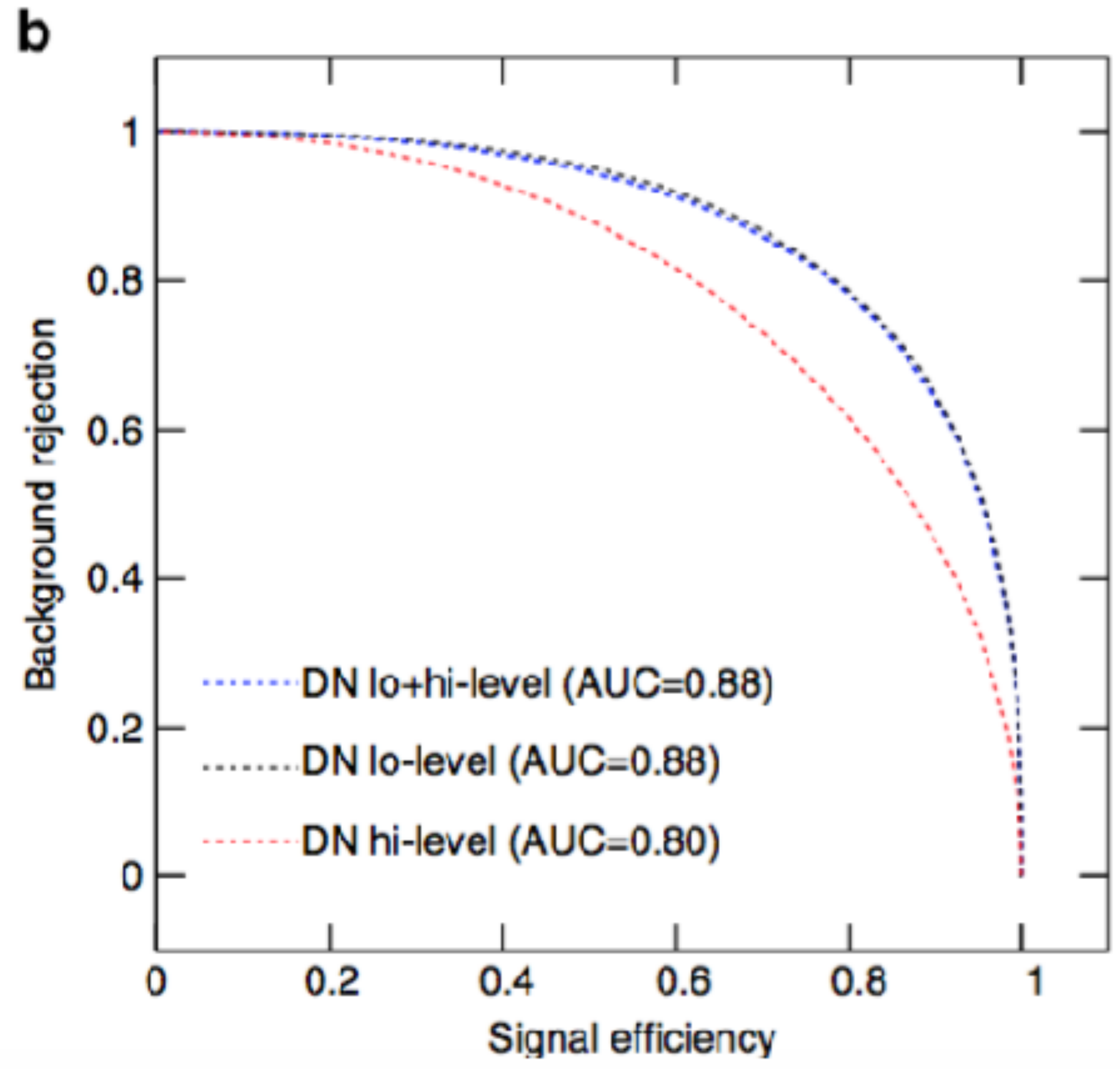


Table 1 | Performance for Higgs benchmark.

Technique	Low-level	High-level	Complete
AUC			
BDT	0.73 (0.01)	0.78 (0.01)	0.81 (0.01)
NN	0.733 (0.007)	0.777 (0.001)	0.816 (0.004)
DN	0.880 (0.001)	0.800 (<0.001)	0.885 (0.002)
Discovery significance			
NN	2.5σ	3.1σ	3.7σ
DN	4.9σ	3.6σ	5.0σ

Comparison of the performance of several learning techniques: boosted decision trees (BDT), shallow neural networks (NN), and deep neural networks (DN) for three sets of input features: low-level features, high-level features and the complete set of features. Each neural network was trained five times with different random initializations. The table displays the mean area under the curve (AUC) of the signal-rejection curve in Fig. 7, with s.d. in parentheses as well as the expected significance of a discovery (in units of Gaussian σ) for 100 signal events and $1{,}000 \pm 50$ background events.

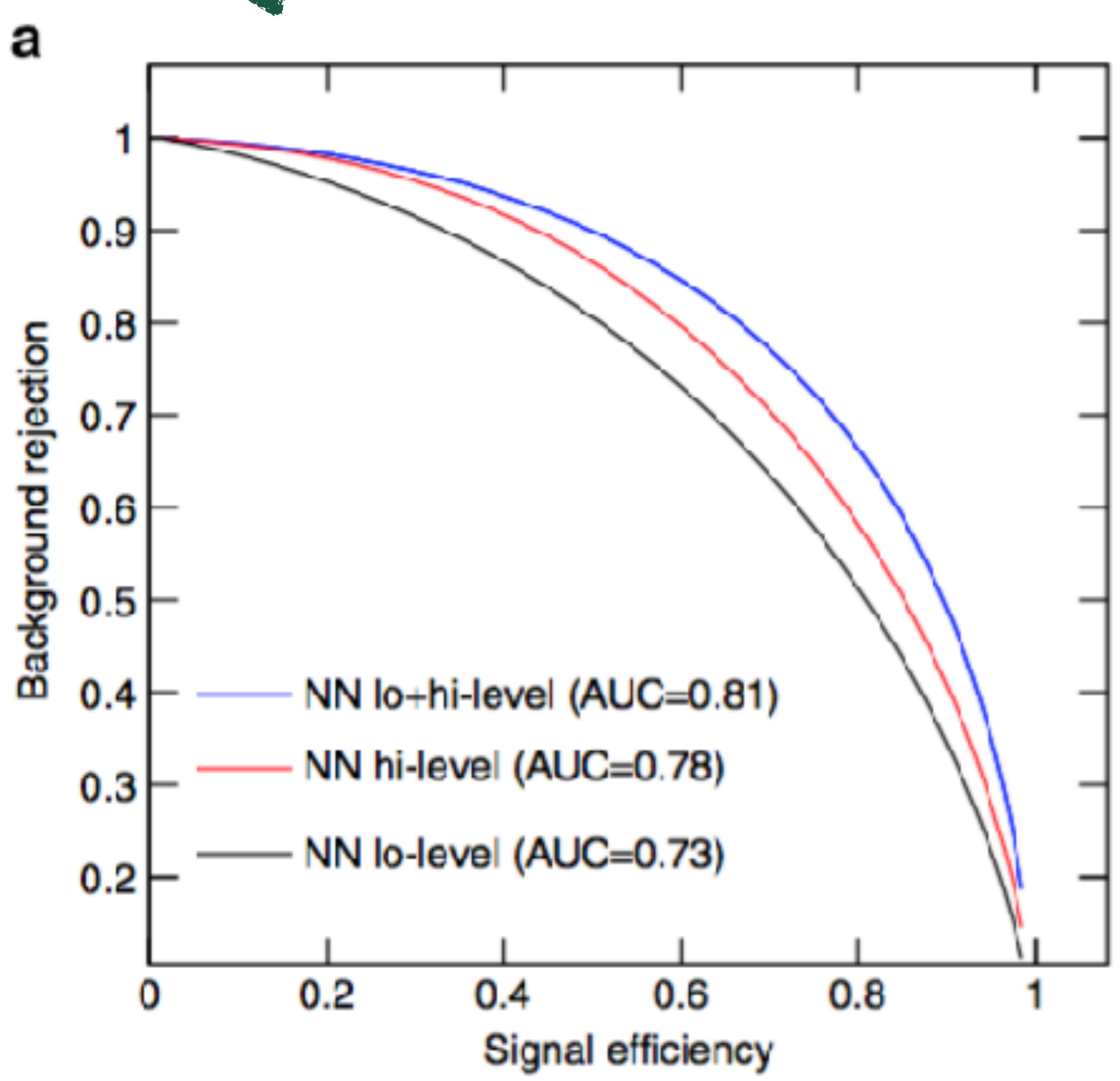
Methods trained with only the high-level features, however, have a weaker performance than those trained with the full suite of features, which suggests that despite the insight represented by the high-level features, they do not capture all the information contained in the low-level features. The deep-learning techniques show nearly equivalent performance using the low-level features and the complete features, suggesting that they are automatically discovering the insight contained in the high-level features.

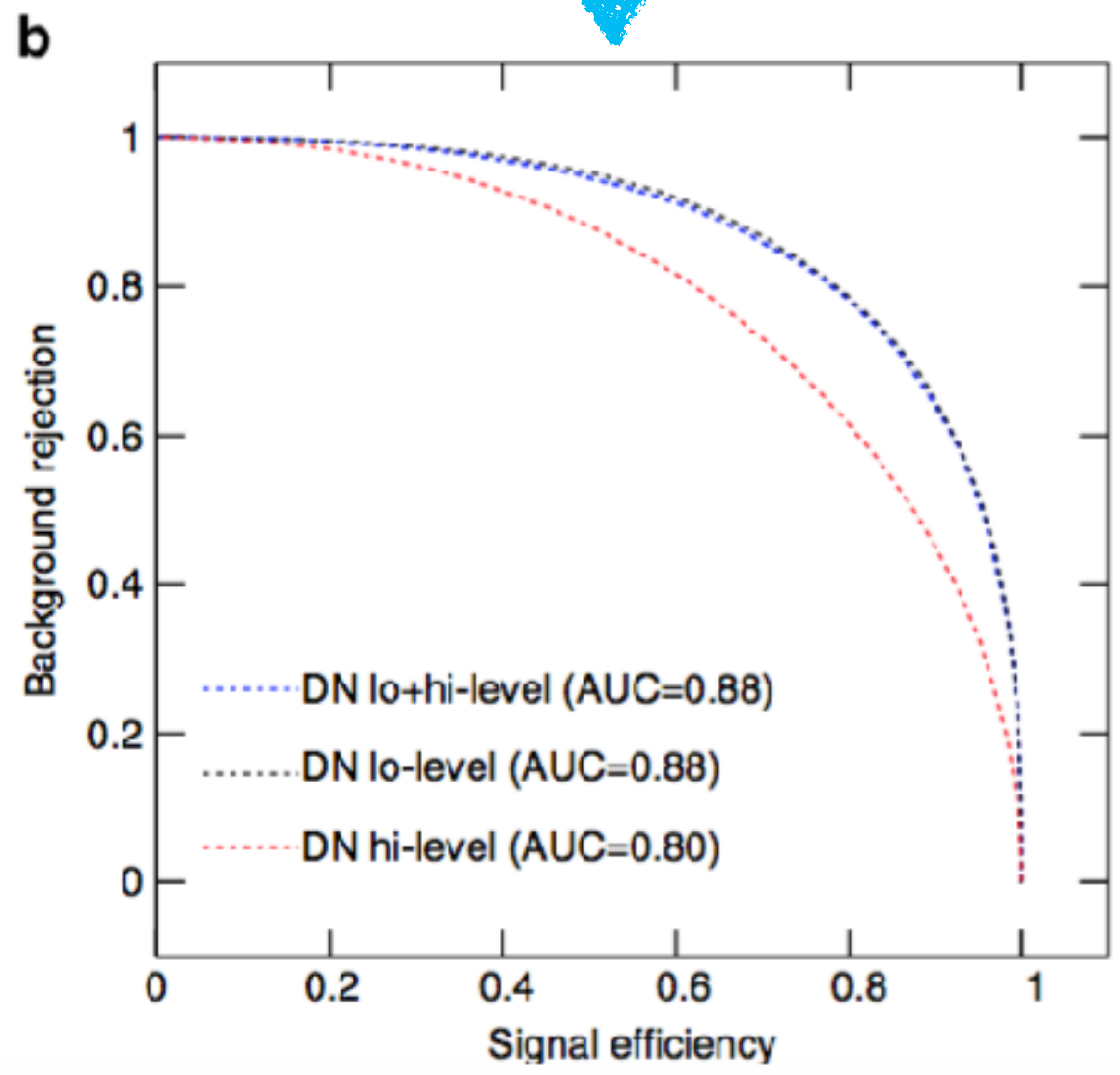
Neural Networks (Deep Learning)

11 million training examples
1 hidden layer shallow network
5 layer deep network

Table 1 | Performance for Higgs benchmark.			
Technique	**Low-level**	**High-level**	**Complete**
AUC			
BDT	0.73 (0.01)	0.78 (0.01)	0.81 (0.01)
NN	0.733 (0.007)	0.777 (0.001)	0.816 (0.004)
DN	0.880 (0.001)	0.800 (<0.001)	0.885 (0.002)
Discovery significance			
NN	2.5σ	3.1σ	3.7σ
DN	4.9σ	3.6σ	5.0σ

Comparison of the performance of several learning techniques: boosted decision trees (BDT), shallow neural networks (NN), and deep neural networks (DN) for three sets of input features: low-level features, high-level features and the complete set of features. Each neural network was trained five times with different random initializations. The table displays the mean area under the curve (AUC) of the signal-rejection curve in Fig. 7, with s.d. in parentheses as well as the expected significance of a discovery (in units of Gaussian σ) for 100 signal events and $1{,}000 \pm 50$ background events.





Methods trained with only the high-level features, however, have a weaker performance than those trained with the full suite of features, which suggests that despite the insight represented by the high-level features, they do not capture all the information contained in the low-level features. The deep-learning techniques show nearly equivalent performance using the low-level features and the complete features, suggesting that they are automatically discovering the insight contained in the high-level features.

Neural Networks (Deep Learning)

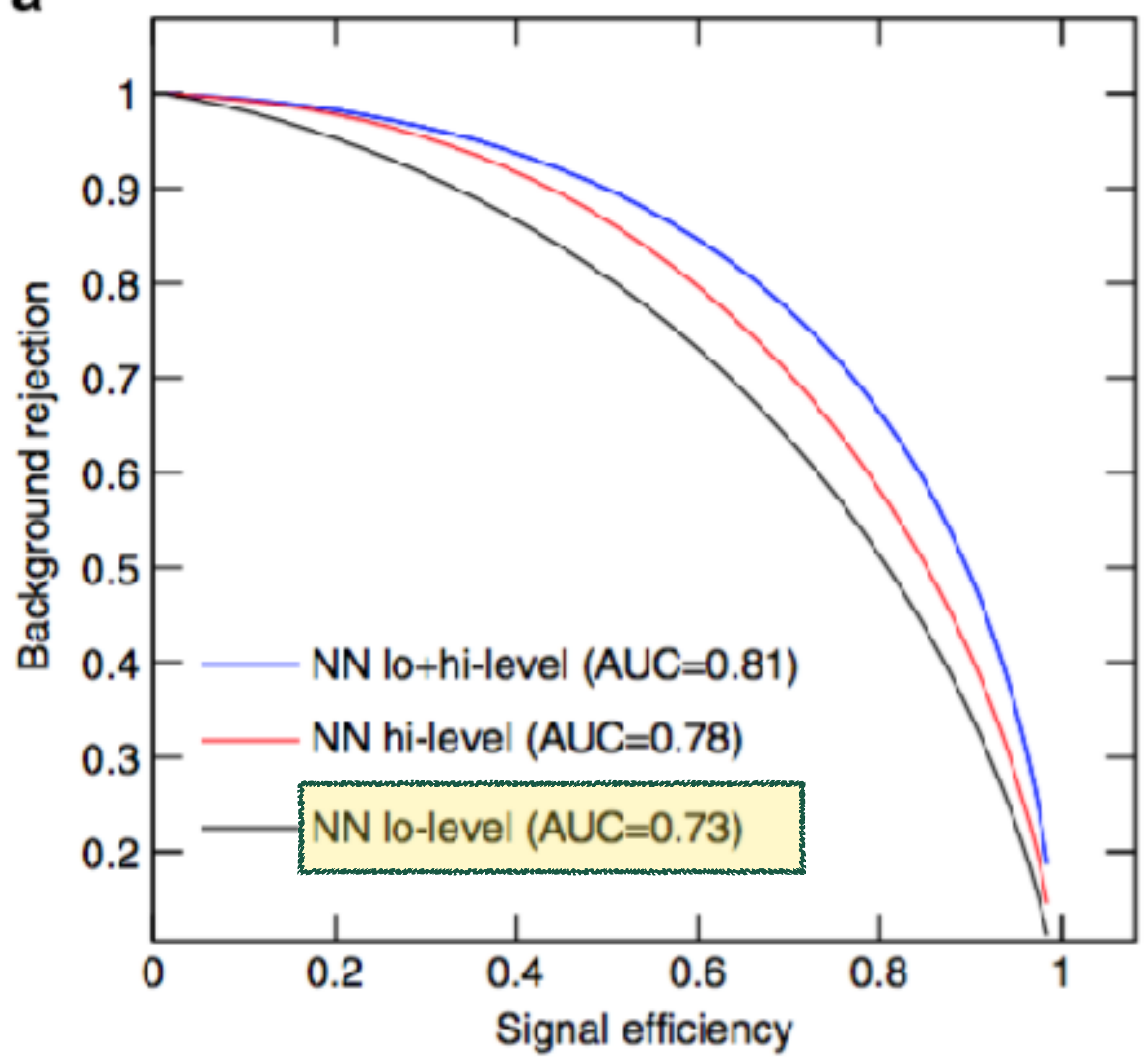
11 million training examples
1 hidden layer shallow network
5 layer deep network

Table 1 | Performance for Higgs benchmark.

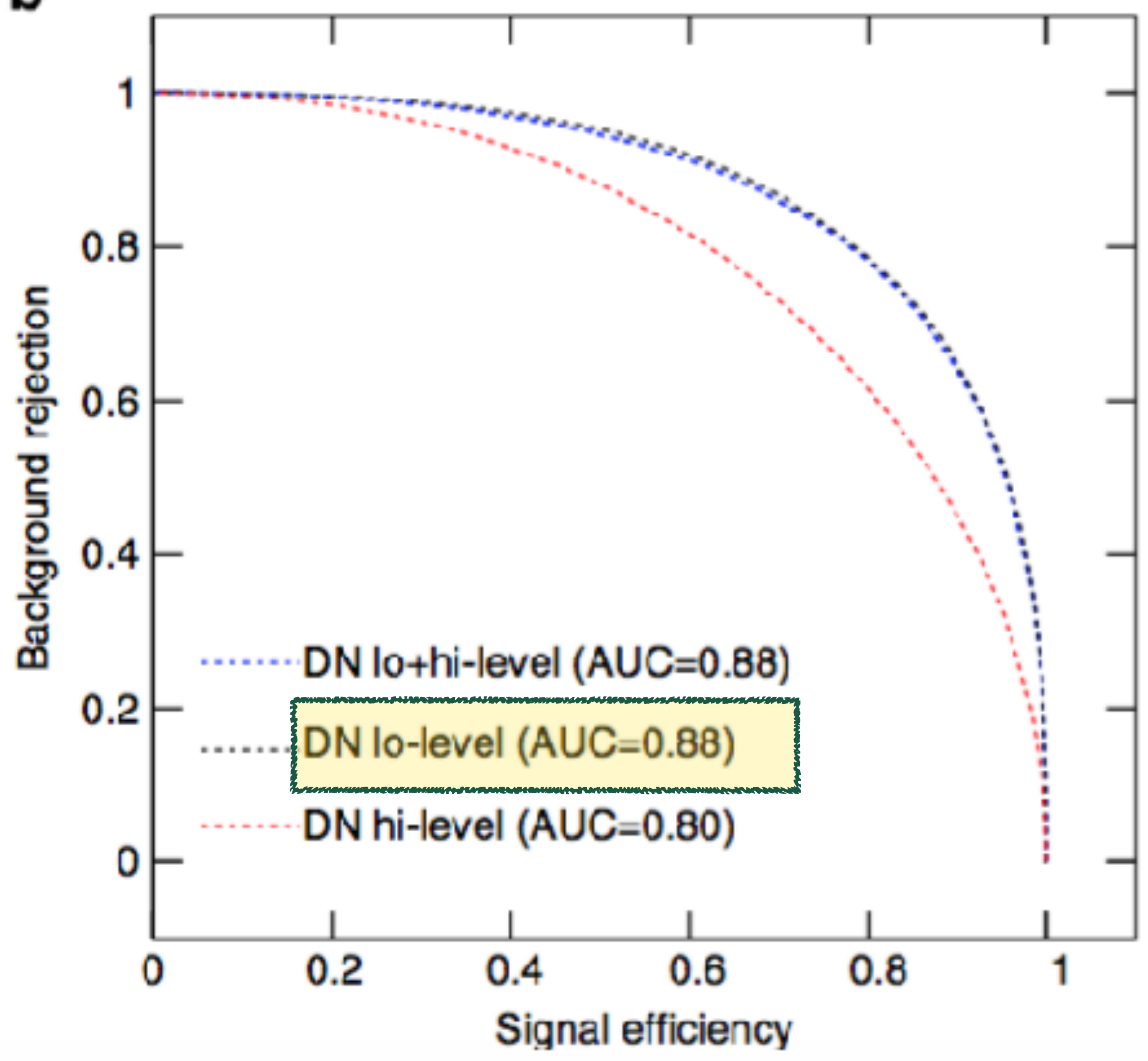
Technique	Low-level	High-level	Complete
AUC			
BDT	0.73 (0.01)	0.78 (0.01)	0.81 (0.01)
NN	0.733 (0.007)	0.777 (0.001)	0.816 (0.004)
DN	0.880 (0.001)	0.800 (<0.001)	0.885 (0.002)
Discovery significance			
NN	2.5σ	3.1σ	3.7σ
DN	4.9σ	3.6σ	5.0σ

Comparison of the performance of several learning techniques: boosted decision trees (BDT), shallow neural networks (NN), and deep neural networks (DN) for three sets of input features: low-level features, high-level features and the complete set of features. Each neural network was trained five times with different random initializations. The table displays the mean area under the curve (AUC) of the signal-rejection curve in Fig. 7, with s.d. in parentheses as well as the expected significance of a discovery (in units of Gaussian σ) for 100 signal events and $1{,}000 \pm 50$ background events.

a



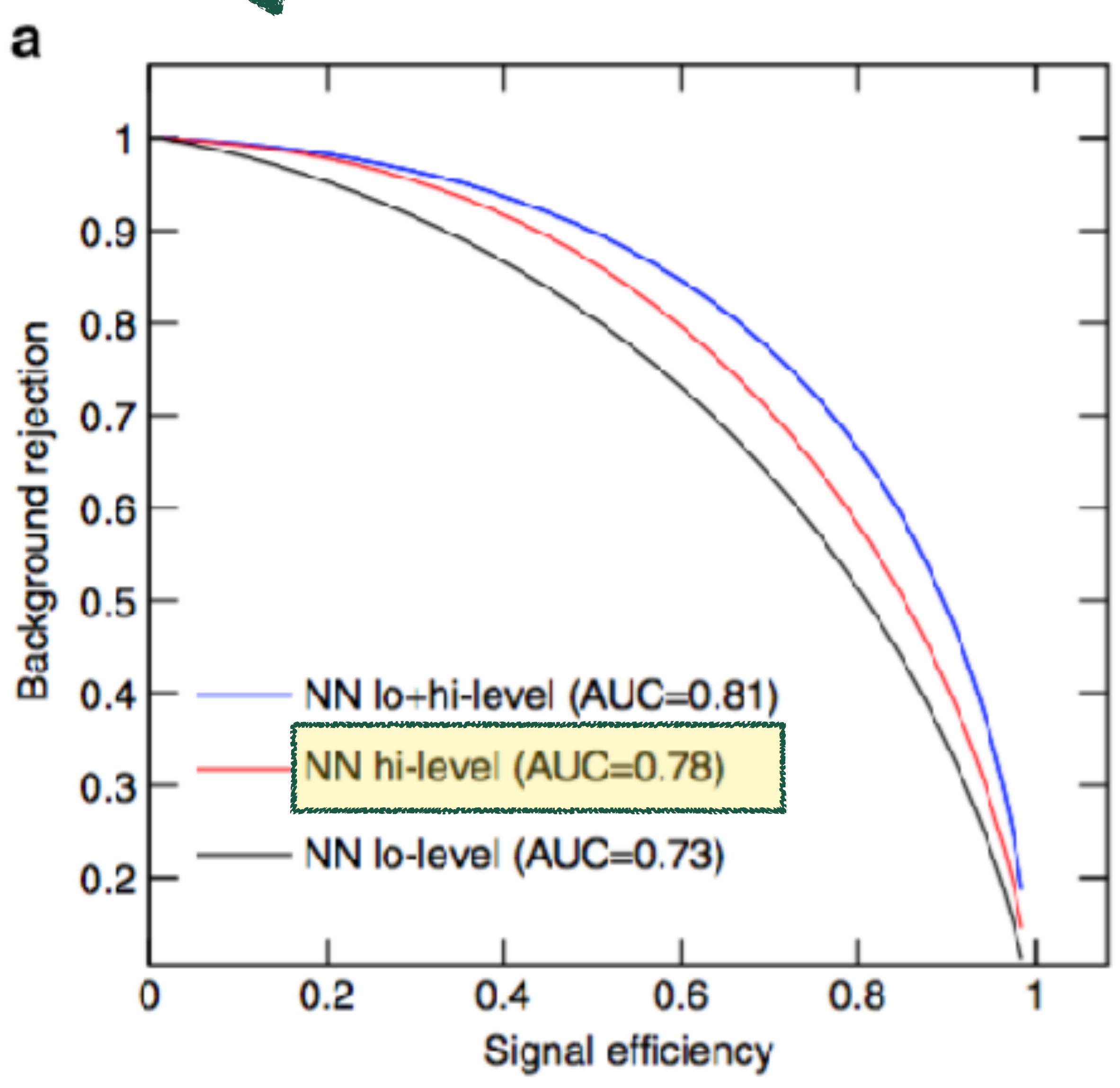
b



Methods trained with only the high-level features, however, have a weaker performance than those trained with the full suite of features, which suggests that despite the insight represented by the high-level features, they do not capture all the information contained in the low-level features. The deep-learning techniques show nearly equivalent performance using the low-level features and the complete features, suggesting that they are automatically discovering the insight contained in the high-level features.

Neural Networks (Deep Learning)

11 million training examples
1 hidden layer shallow network
5 layer deep network



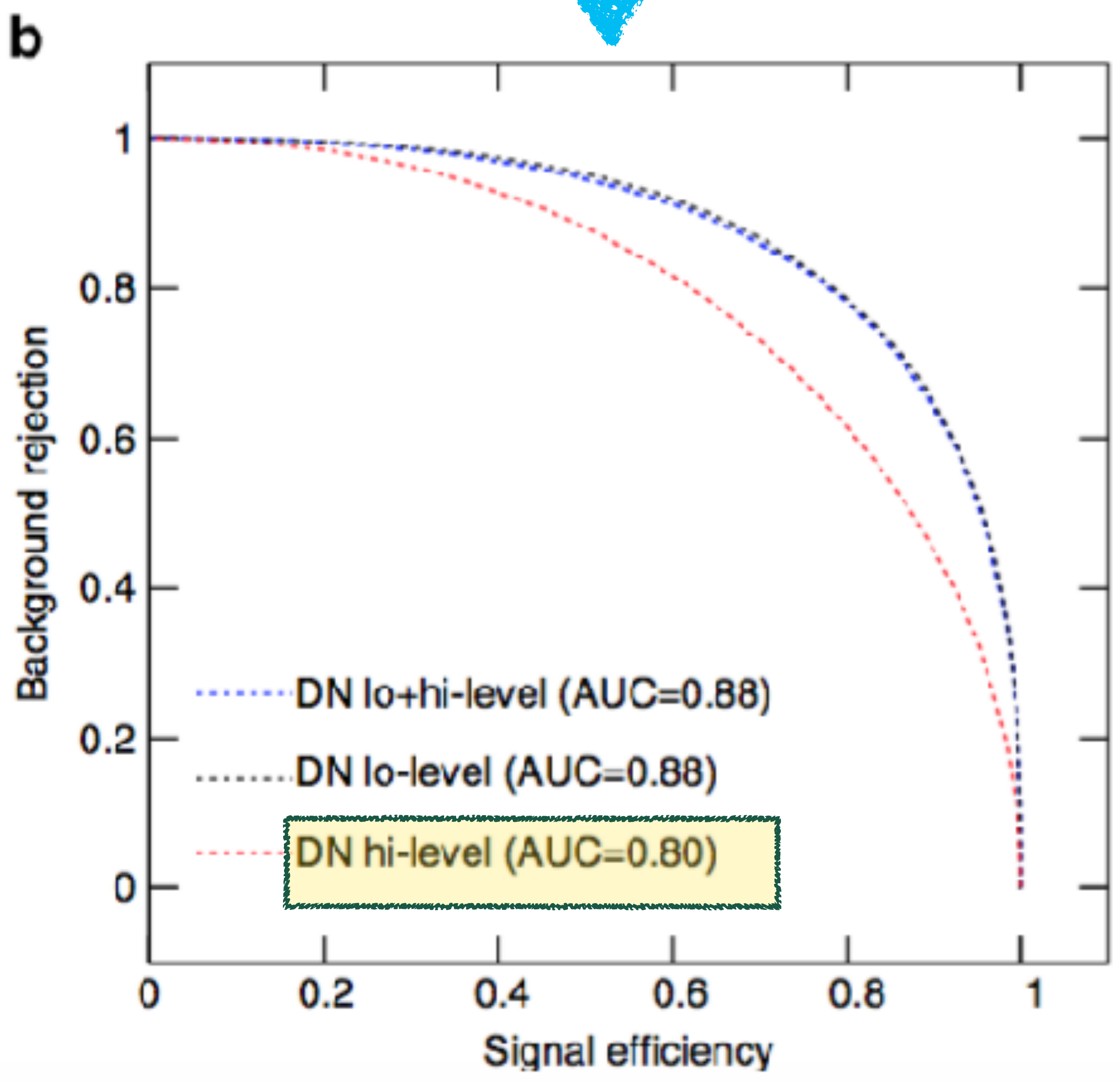


Table 1 | Performance for Higgs benchmark.

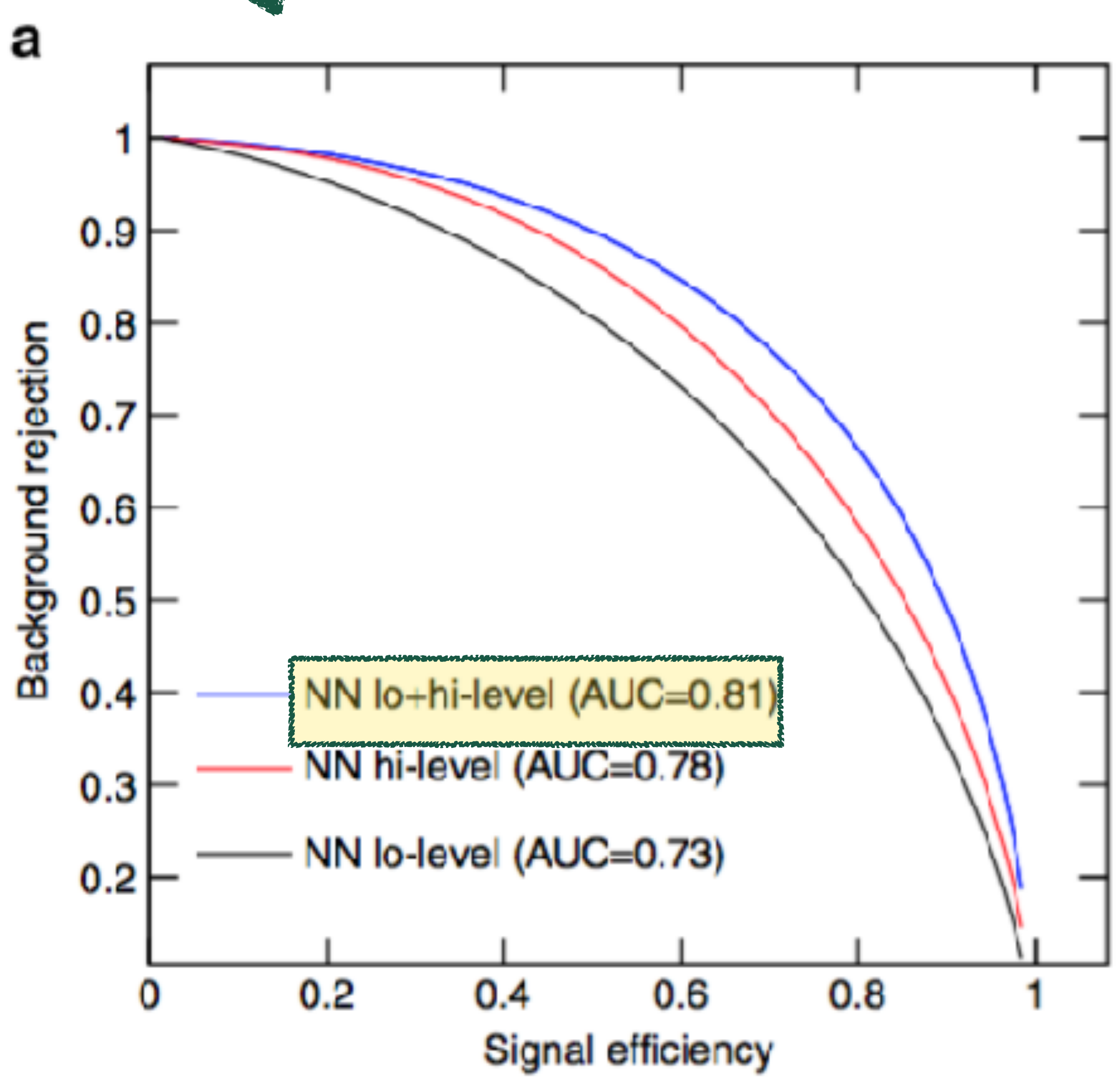
Technique	Low-level	High-level	Complete
AUC			
BDT	0.73 (0.01)	0.78 (0.01)	0.81 (0.01)
NN	0.733 (0.007)	0.777 (0.001)	0.816 (0.004)
DN	0.880 (0.001)	0.800 (< 0.001)	0.885 (0.002)
Discovery significance			
NN	2.5σ	3.1σ	3.7σ
DN	4.9σ	3.6σ	5.0σ

Comparison of the performance of several learning techniques: boosted decision trees (BDT), shallow neural networks (NN), and deep neural networks (DN) for three sets of input features: low-level features, high-level features and the complete set of features. Each neural network was trained five times with different random initializations. The table displays the mean area under the curve (AUC) of the signal-rejection curve in Fig. 7, with s.d. in parentheses as well as the expected significance of a discovery (in units of Gaussian σ) for 100 signal events and $1{,}000 \pm 50$ background events.

Methods trained with only the high-level features, however, have a weaker performance than those trained with the full suite of features, which suggests that despite the insight represented by the high-level features, they do not capture all the information contained in the low-level features. The deep-learning techniques show nearly equivalent performance using the low-level features and the complete features, suggesting that they are automatically discovering the insight contained in the high-level features.

Neural Networks (Deep Learning)

11 million training examples
1 hidden layer shallow network
5 layer deep network



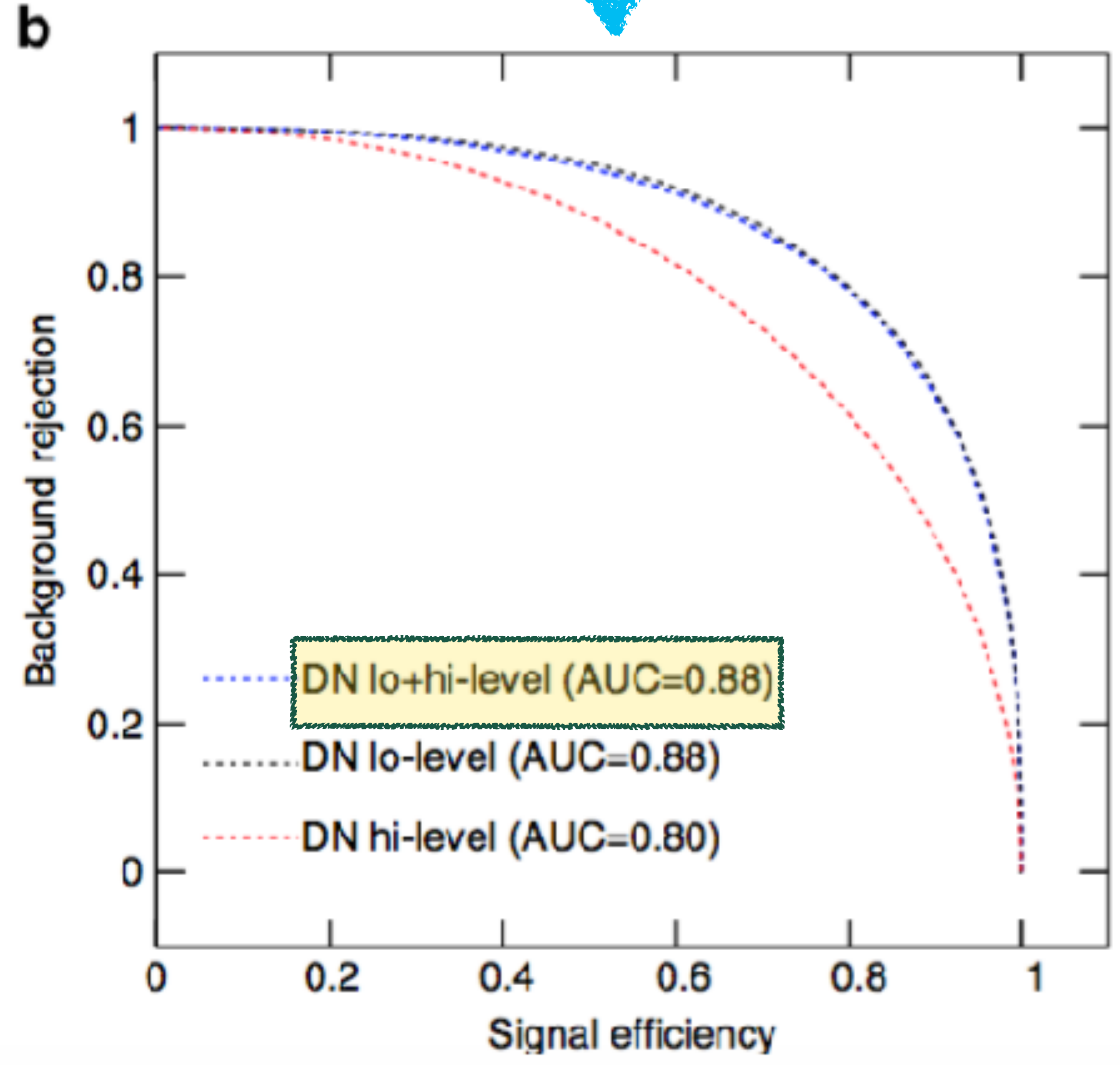


Table 1 | Performance for Higgs benchmark.

Technique	Low-level	High-level	Complete
AUC			
BDT	0.73 (0.01)	0.78 (0.01)	0.81 (0.01)
NN	0.733 (0.007)	0.777 (0.001)	0.816 (0.004)
DN	0.880 (0.001)	0.800 (<0.001)	0.885 (0.002)
Discovery significance			
NN	2.5σ	3.1σ	3.7σ
DN	4.9σ	3.6σ	5.0σ

Comparison of the performance of several learning techniques: boosted decision trees (BDT), shallow neural networks (NN), and deep neural networks (DN) for three sets of input features: low-level features, high-level features and the complete set of features. Each neural network was trained five times with different random initializations. The table displays the mean area under the curve (AUC) of the signal-rejection curve in Fig. 7, with s.d. in parentheses as well as the expected significance of a discovery (in units of Gaussian σ) for 100 signal events and $1{,}000 \pm 50$ background events.

Methods trained with only the high-level features, however, have a weaker performance than those trained with the full suite of features, which suggests that despite the insight represented by the high-level features, they do not capture all the information contained in the low-level features. The deep-learning techniques show nearly equivalent performance using the low-level features and the complete features, suggesting that they are automatically discovering the insight contained in the high-level features.

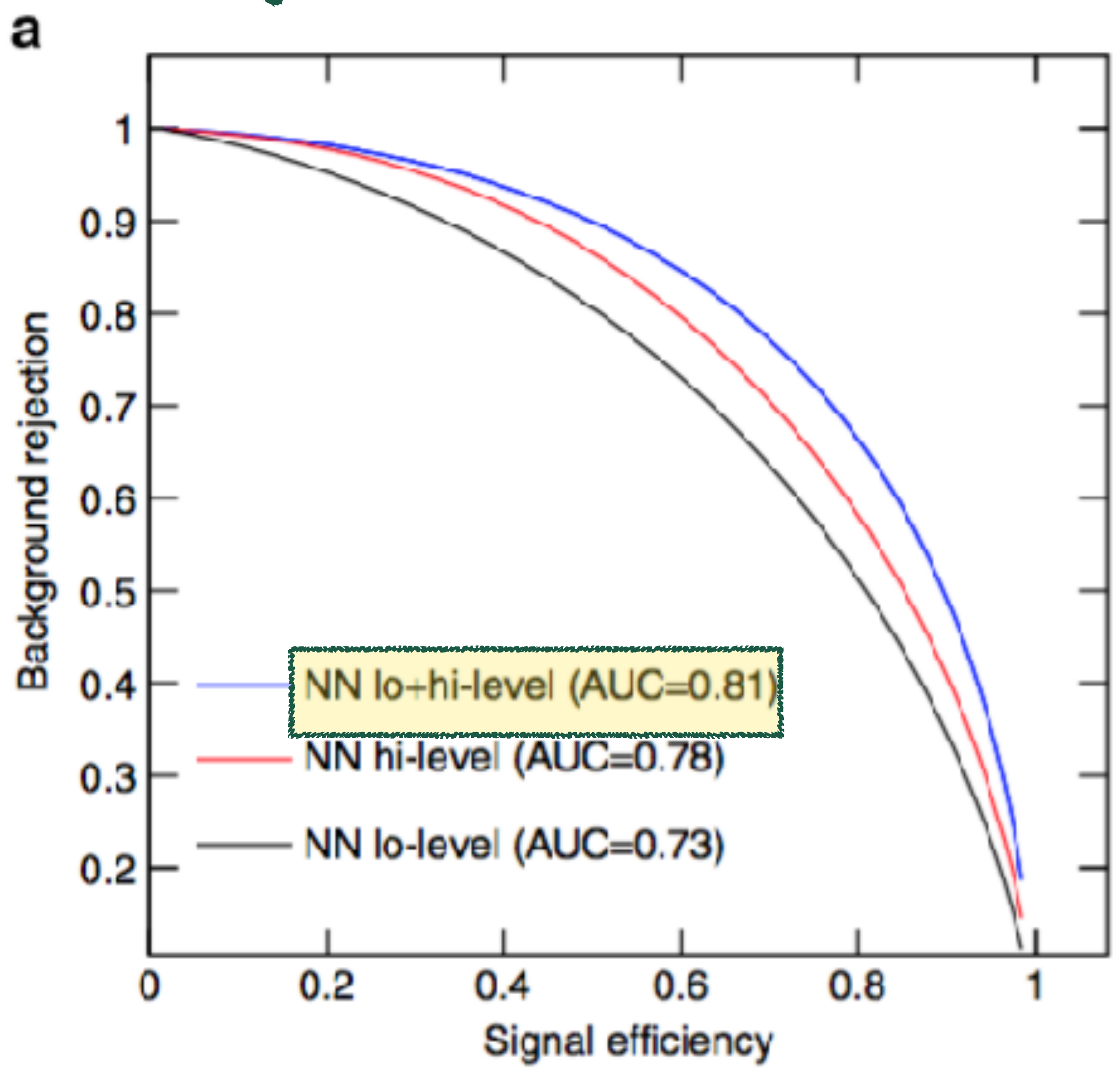
Neural Networks (Deep Learning)

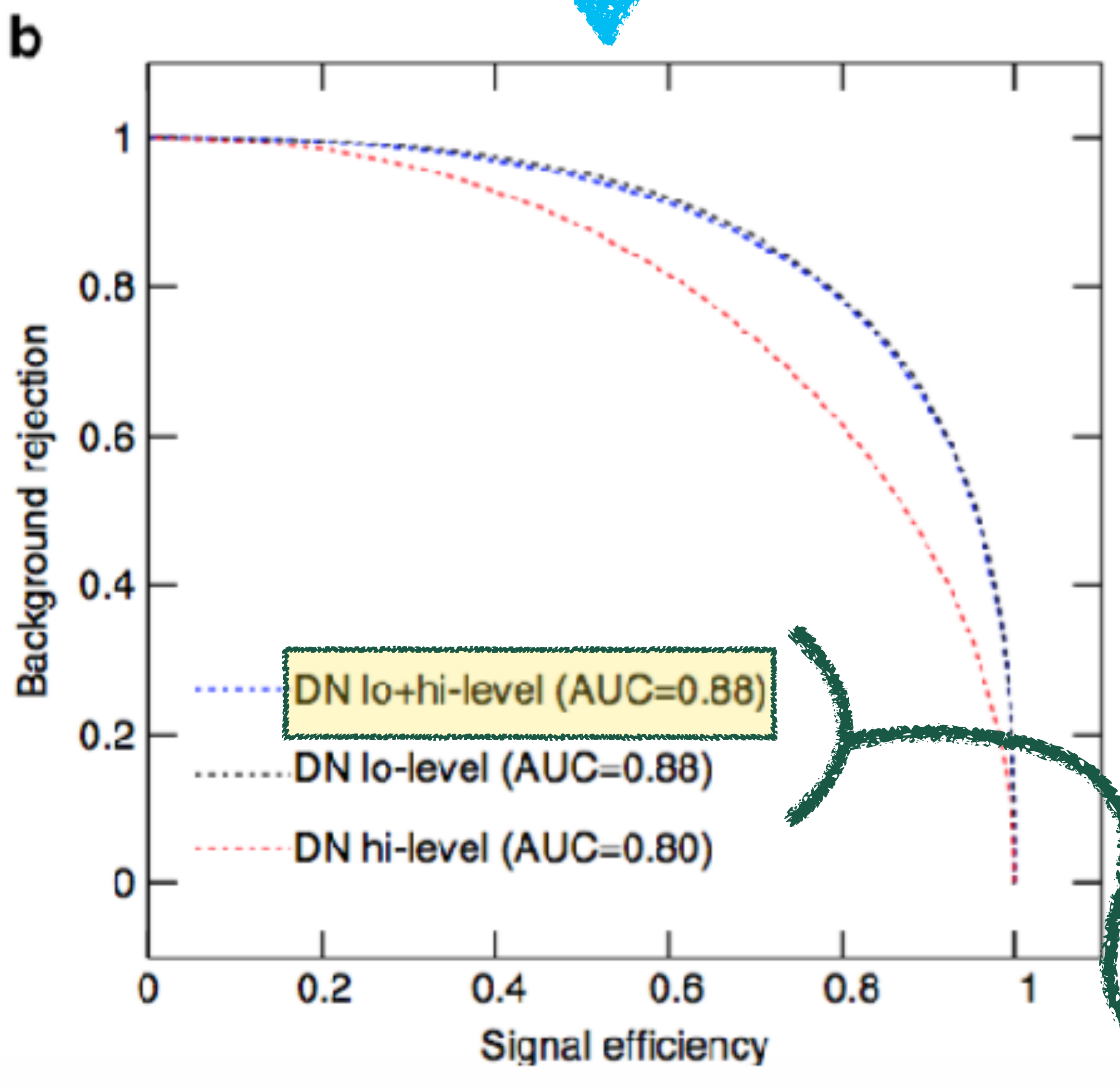
11 million training examples
1 hidden layer shallow network
5 layer deep network

Table 1 | Performance for Higgs benchmark.

Technique	Low-level	High-level	Complete
AUC			
BDT	0.73 (0.01)	0.78 (0.01)	0.81 (0.01)
NN	0.733 (0.007)	0.777 (0.001)	0.816 (0.004)
DN	0.880 (0.001)	0.800 (<0.001)	0.885 (0.002)
Discovery significance			
NN	2.5σ	3.1σ	3.7σ
DN	4.9σ	3.6σ	5.0σ

Comparison of the performance of several learning techniques: boosted decision trees (BDT), shallow neural networks (NN), and deep neural networks (DN) for three sets of input features: low-level features, high-level features and the complete set of features. Each neural network was trained five times with different random initializations. The table displays the mean area under the curve (AUC) of the signal-rejection curve in Fig. 7, with s.d. in parentheses as well as the expected significance of a discovery (in units of Gaussian σ) for 100 signal events and 1,000 $\pm$ 50 background events.





Methods trained with only the high-level features, however, have a weaker performance than those trained with the full suite of features, which suggests that despite the insight represented by the high-level features, they do not capture all the information contained in the low-level features. The deep-learning techniques show nearly equivalent performance using the low-level features and the complete features, suggesting that they are automatically discovering the insight contained in the high-level features.

Deep learning finds more information than our high-level variables

- What are our variables missing?
- How do we know if all information has been used?
- What is the machine learning?
- What information does the machine learn?
- What information is important for the machine?
- …?

Deep learning finds more information than our high-level variables

- What are our variables missing?
- How do we know if all information has been used?
- What is the machine learning?
- What information does the machine learn?
- What information is important for the machine?
- …?

Datta and Larkoski [1704.08249]

- Sets of observables that completely and minimally span N-body phase space.

Komiske, Metodiev, and Thaler [1712.07124]

- Energy flow polynomials, a complete linear basis for jet substructure

Deep learning finds more information than our high-level variables

- What are our variables missing?
- How do we know if all information has been used?
- What is the machine learning?
- What information does the machine learn?
- What information is important for the machine?
- …?

Datta and Larkoski [1704.08249]

- Sets of observables that completely and minimally span N-body phase space.

Komiske, Metodiev, and Thaler [1712.07124]

- Energy flow polynomials, a complete linear basis for jet substructure

Chang, Cohen, and **BO** [1709.10106]

- Remove information to reveal what machine learned from

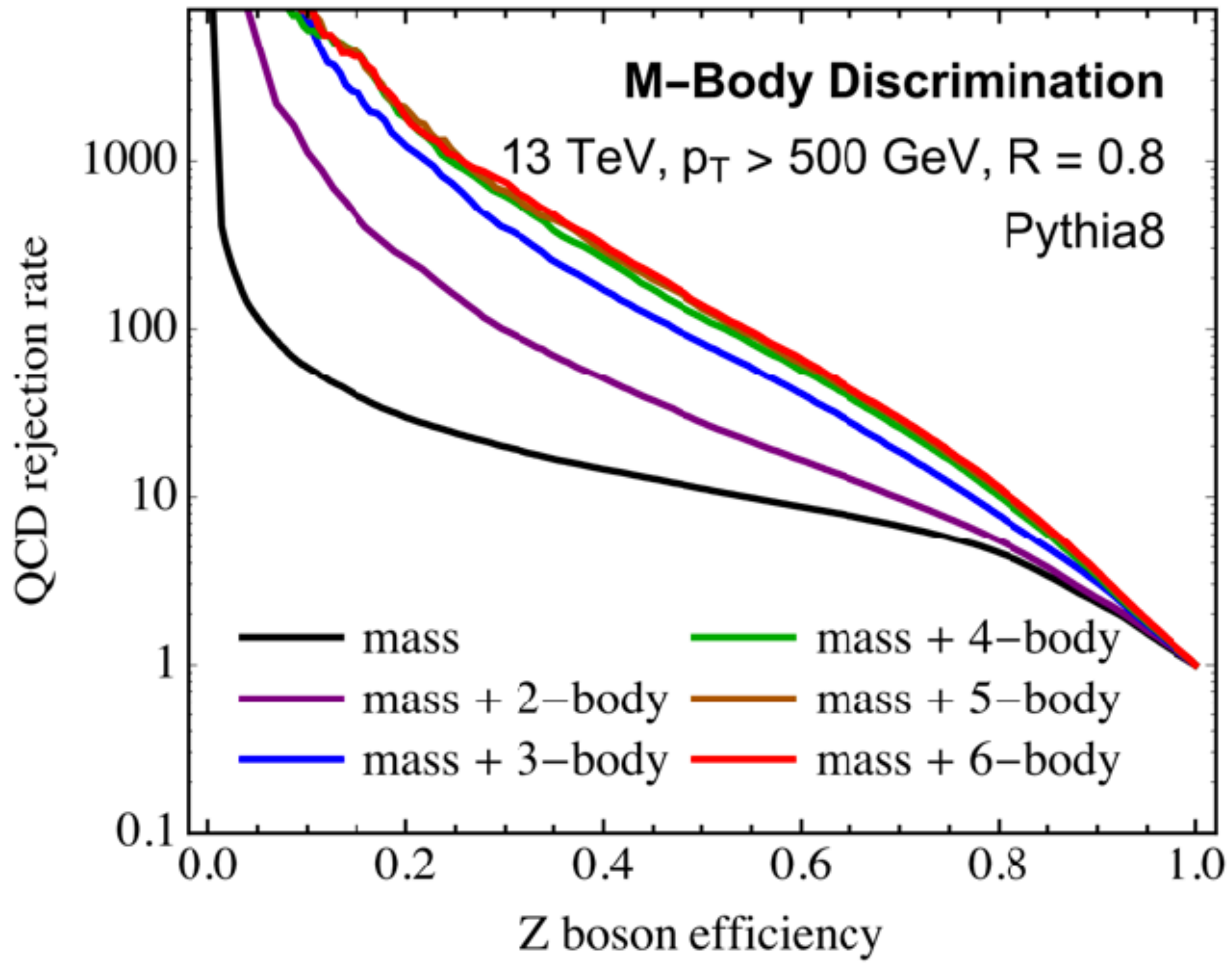
How much information is in a jet?

Distinguish hadronic, boosted *Z* from QCD background

(3*M* - 4) unique variables in *M*-body phase space

$$\tau_N^{(\beta)} = \frac{1}{p_{TJ}} \sum_{i \in \mathrm{Jet}} p_{Ti} \min\left\{\mathrm{R}_{1i}^{\beta}, \mathrm{R}_{2i}^{\beta}, \cdots, \mathrm{R}_{\mathrm{Ni}}^{\beta}, \right\}$$

$$\begin{aligned}\{&\tau_1^{(0.5)}, \tau_1^{(1)}, \tau_1^{(2)}, \\ &\tau_2^{(0.5)}, \tau_2^{(1)}, \tau_2^{(2)}, \\ &\cdots, \\ &\tau_{M-2}^{(0.5)}, \tau_{M-2}^{(1)}, \tau_{M-2}^{(2)}, \\ &\tau_{M-1}^{(1)}, \tau_{M-1}^{(2)}\}\end{aligned}$$



Information in a jet that is useful for discriminating QCD jets from Z bosons is saturated by only considering observables that are sensitive to 4-body phase space.

Datta and Larkoski [1704.08249]

How much information is in a jet?

Distinguish hadronic, boosted Z from QCD background

Still uses a deep neural network, don't know what combination of the information the machine uses. Is all of it necessary?

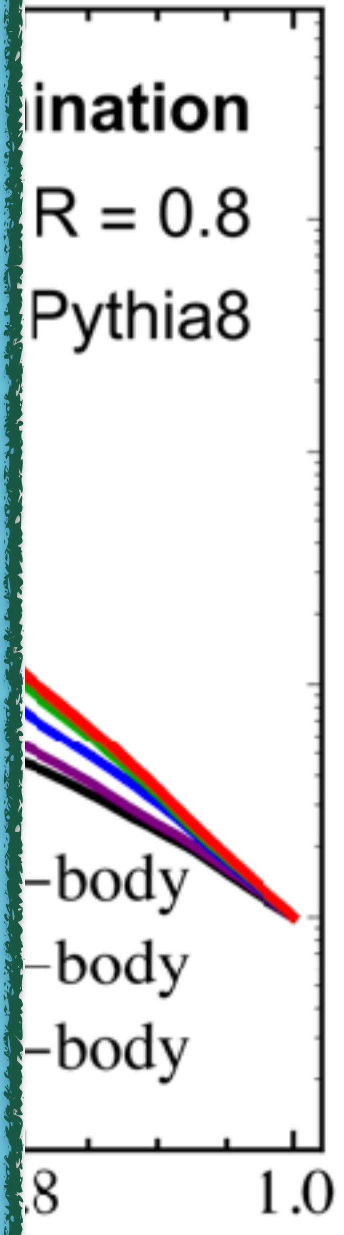
Use Energy Flow Polynomials to get a liner basis? (Komiske, Metodiev, and Thaler [1712.07124])

Find out what the machine is learning from?

(3M - 4) un

$$\tau_N^{(\beta)} = \frac{\ldots}{p\ldots}$$

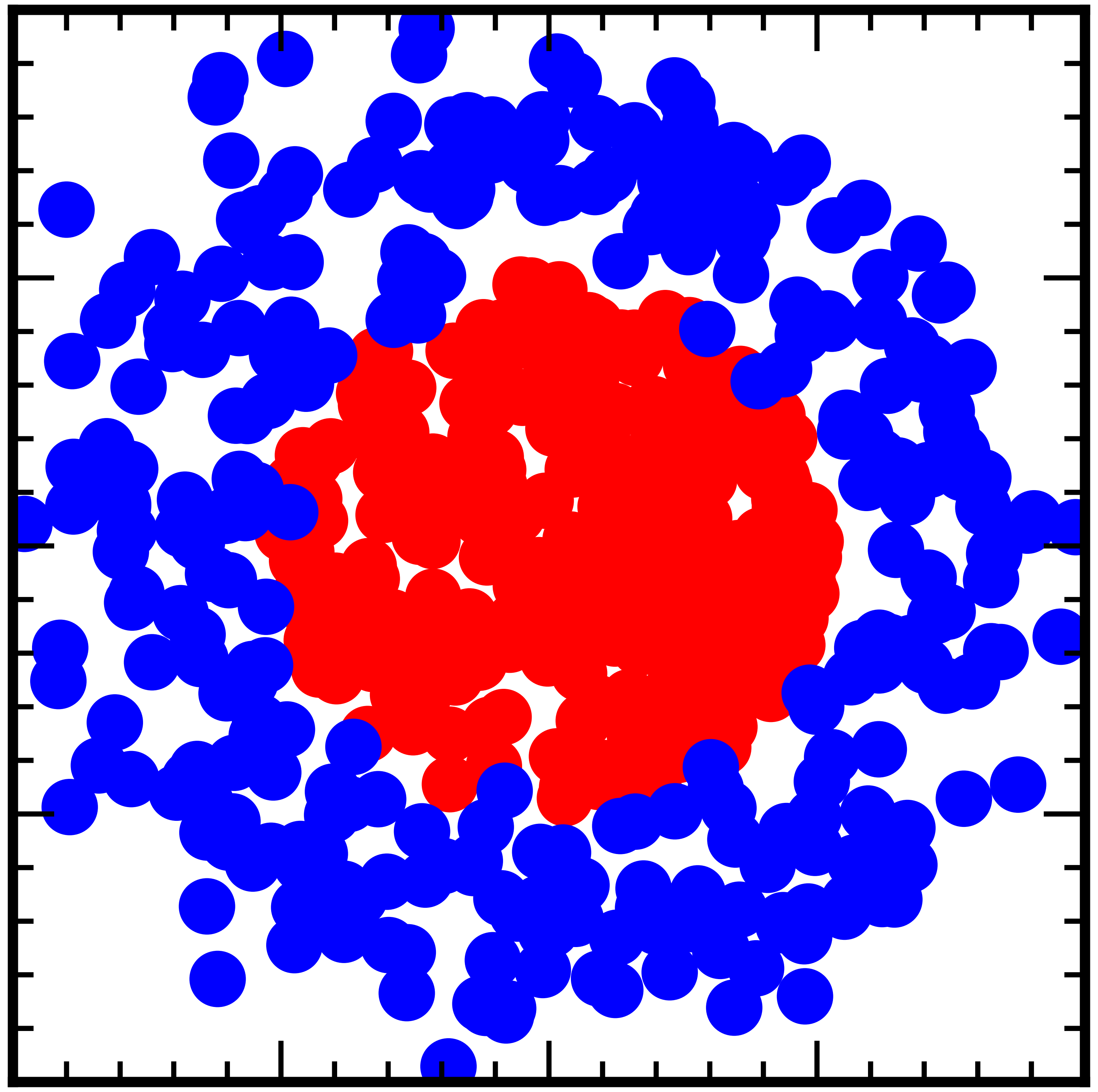
$$\{\tau_1^{(0.5)}, \tau_1^{(\ldots}, \ldots$$
$$\tau_2^{(0.5)}, \tau_2^{(\ldots}, \ldots$$
$$\cdots,$$
$$\tau_{M-2}^{(0.5)}, \tau_{M-2}^{(1)}, \tau_{M-2}^{(2)},$$
$$\tau_{M-1}^{(1)}, \tau_{M-1}^{(2)}\}$$



Information in a jet that is useful for discriminating QCD jets from Z bosons is saturated by only considering observables that are sensitive to 4-body phase space.

Datta and Larkoski [1704.08249]

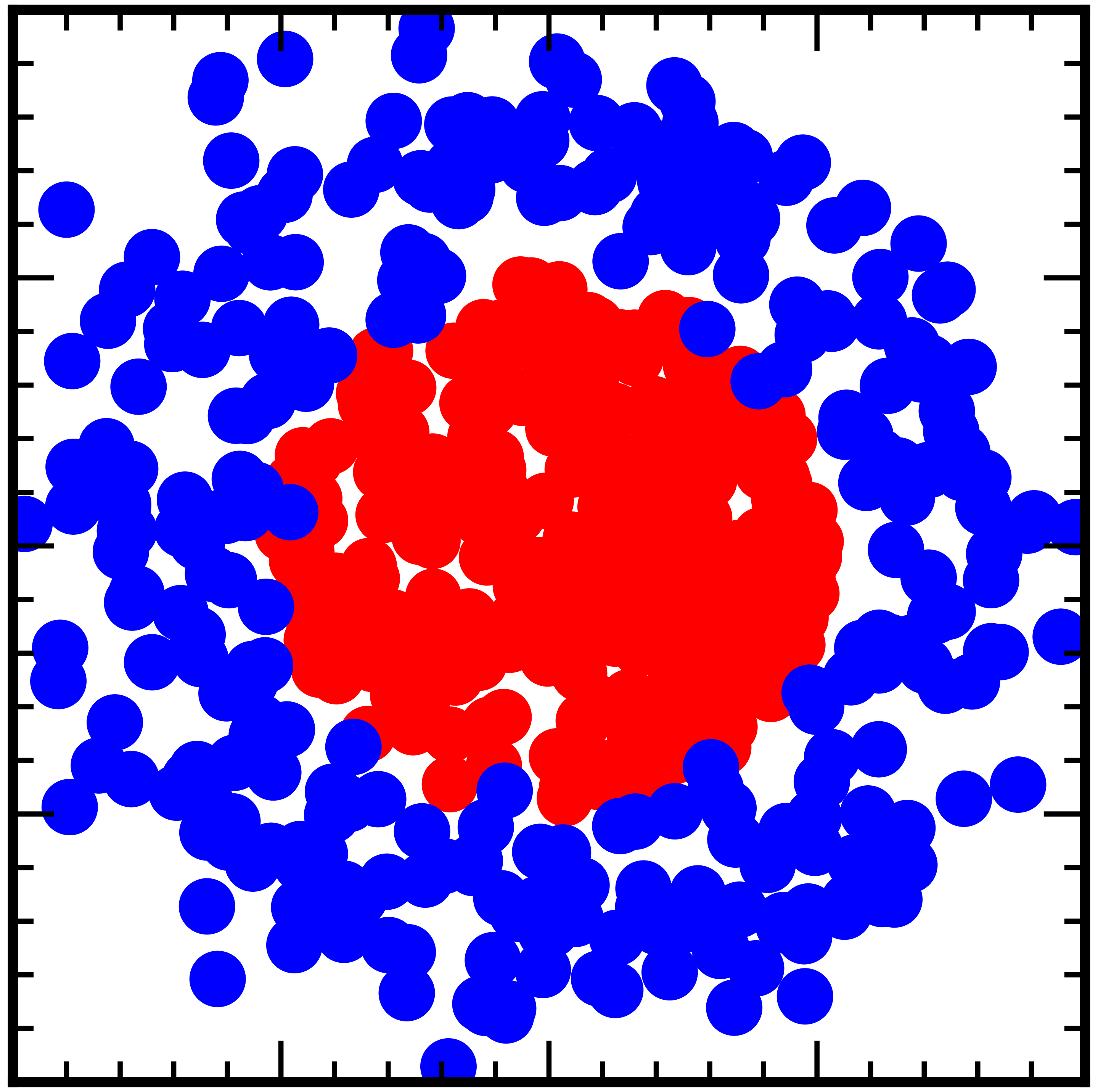
What is the machine learning?

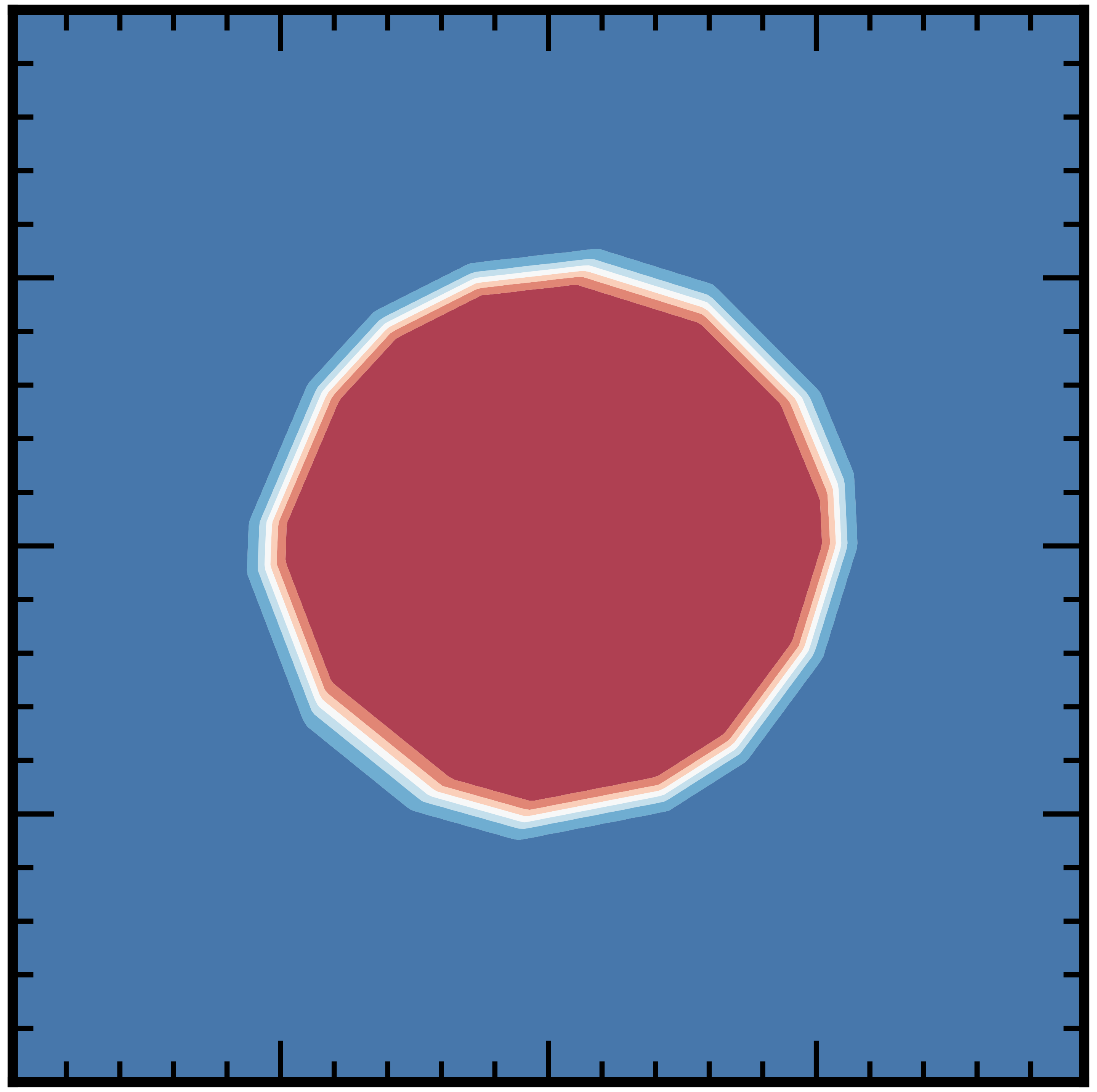


Does the machine learn a circle?

Chang, Cohen, and BO [arXiv:1709.10106]

What is the machine learning?

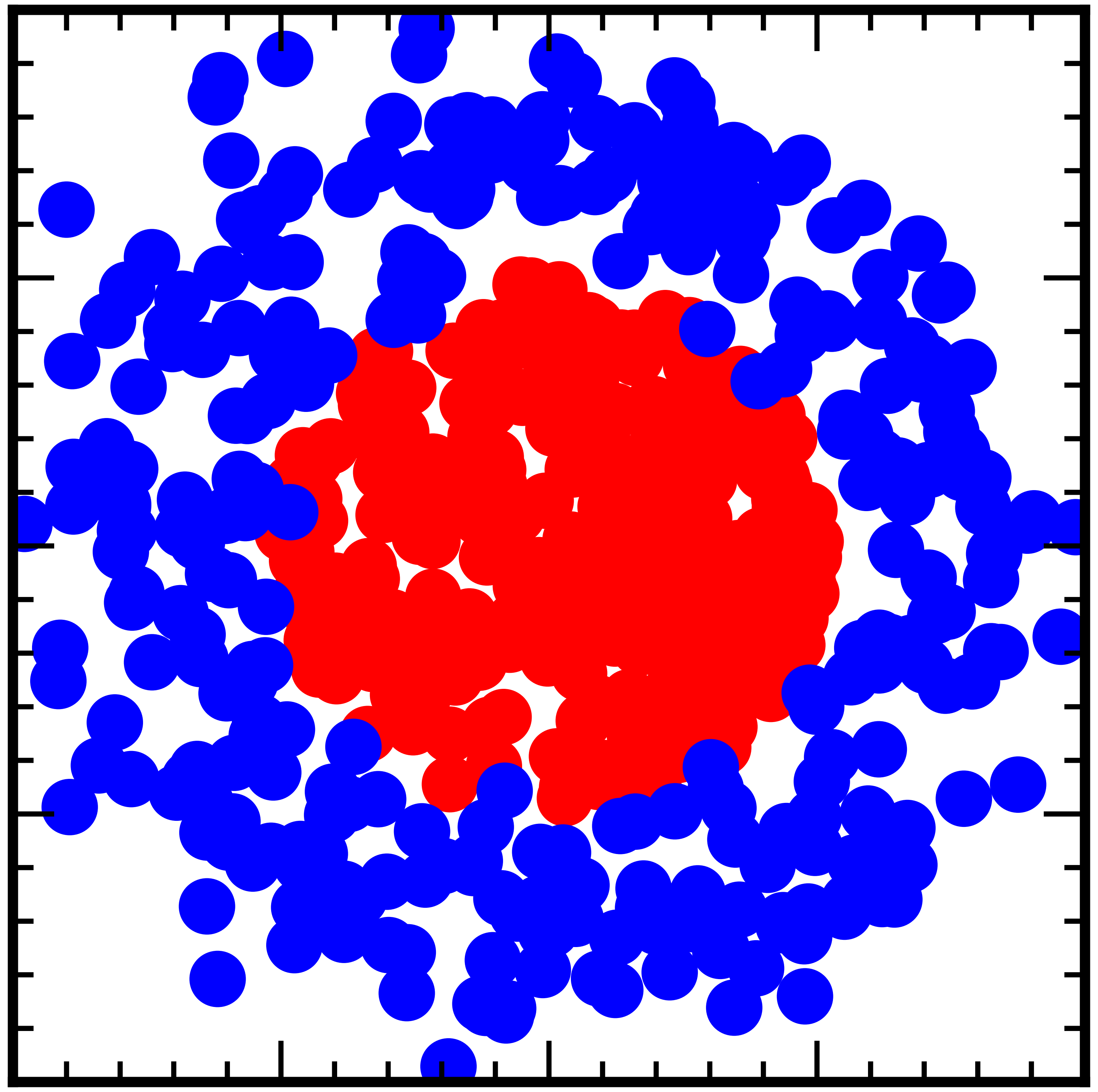


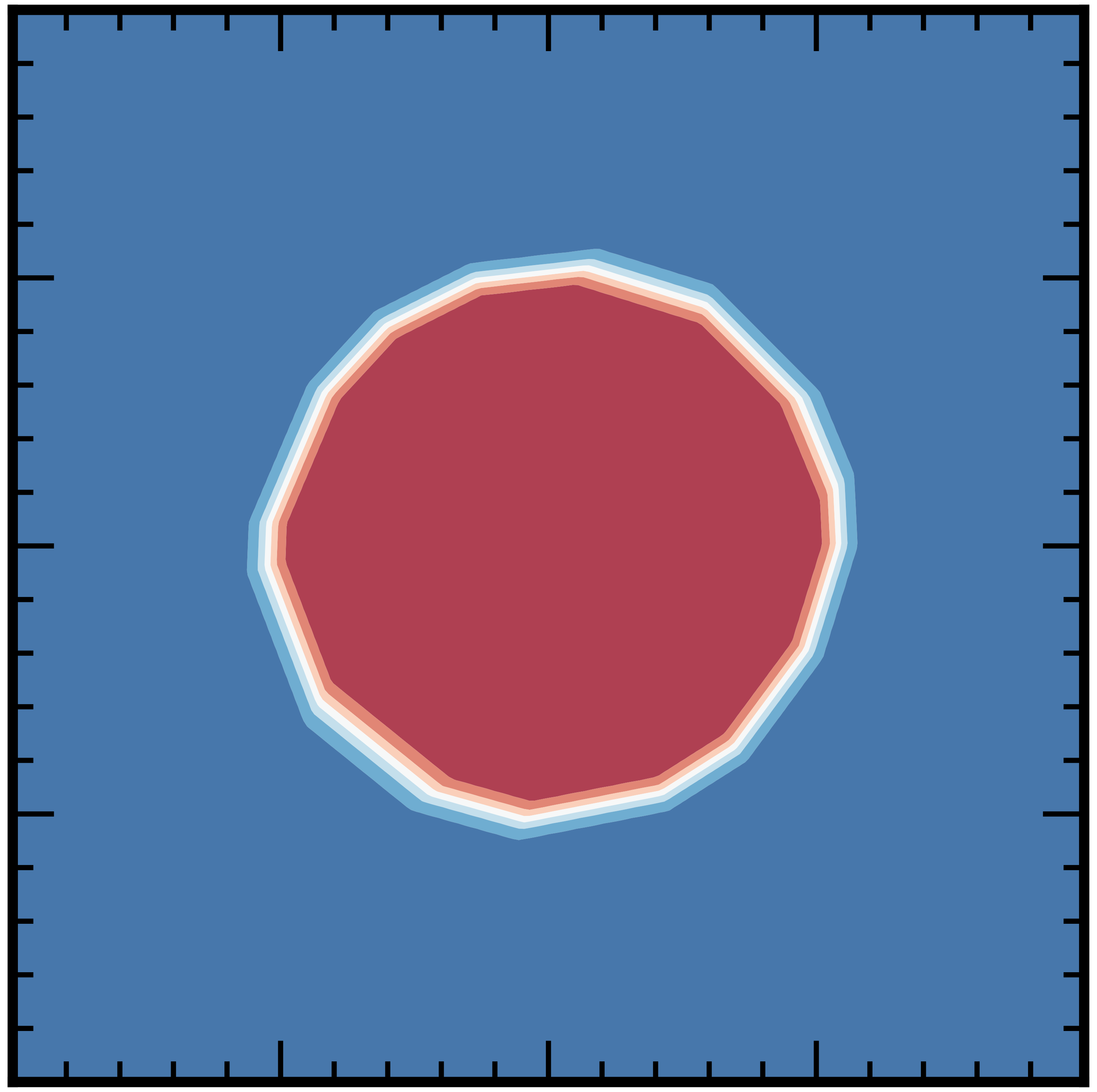


Does the machine learn a circle?

Chang, Cohen, and BO [arXiv:1709.10106]

What is the machine learning?

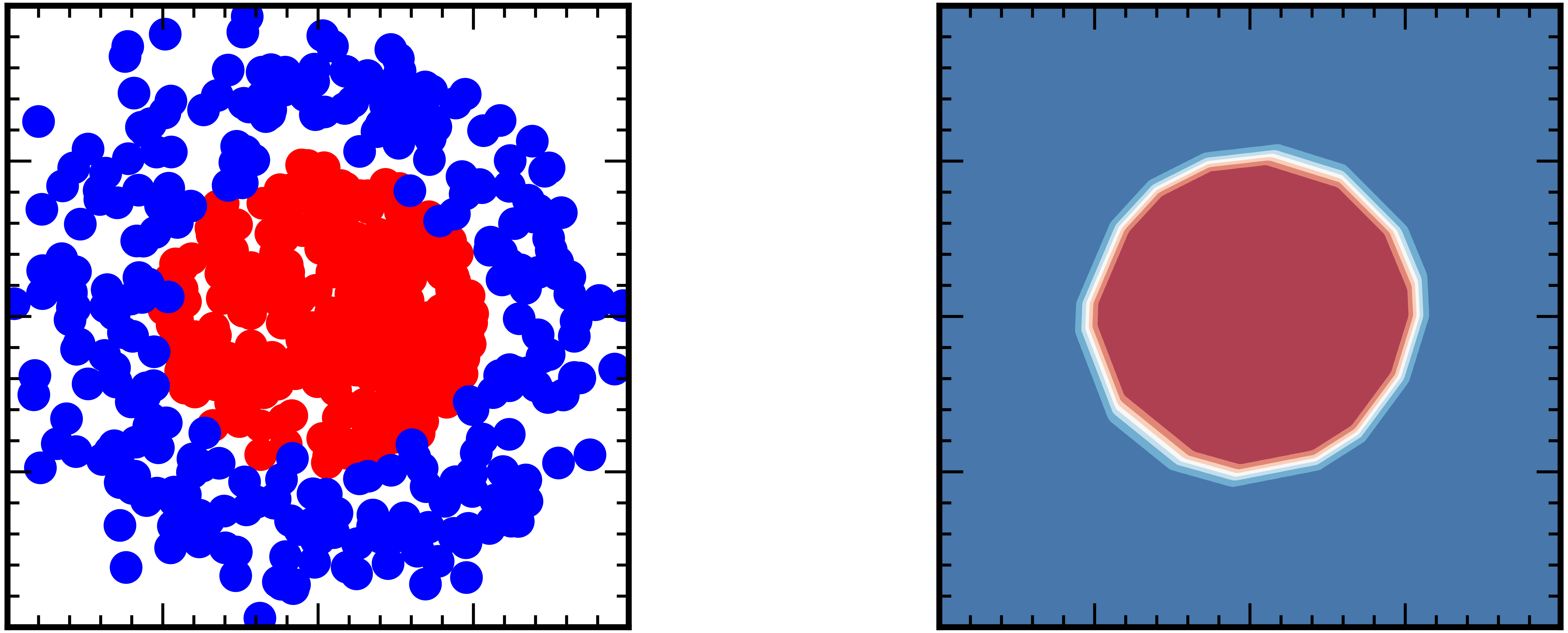




Does the machine learn a circle? Does it learn $x^2 + y^2 \leq r^2$?

Chang, Cohen, and BO [arXiv:1709.10106]

What is the machine learning?



Does the machine learn a circle? Does it learn $x^2 + y^2 \leq r^2$?

It has learned generically where events are in “any” parameter space. Wrong question to ask.

Chang, Cohen, and BO [arXiv:1709.10106]

What is the machine learning?

Can we find what information the machine is learning?

What is the machine learning?

Can we find what information the machine is learning?

Planing

See also de Oliveria, Kagan, Nachman, Schwartzman [arXiv:1511.05190]

(a) Train machine on low level data
(b) Compute low level AUC
(c) Choose a variable: compute (planing) weights
(d) Train machine on weighted (planed) data
(e) Compute planed AUC
(f) Compare: looking for significant performance drop

Removing information from training samples

Similar to what experiments do with different p_T samples

What is the machine learning?

Can we find what information the machine is learning?

Planing

See also de Oliveria, Kagan, Nachman, Schwartzman [arXiv:1511.05190]

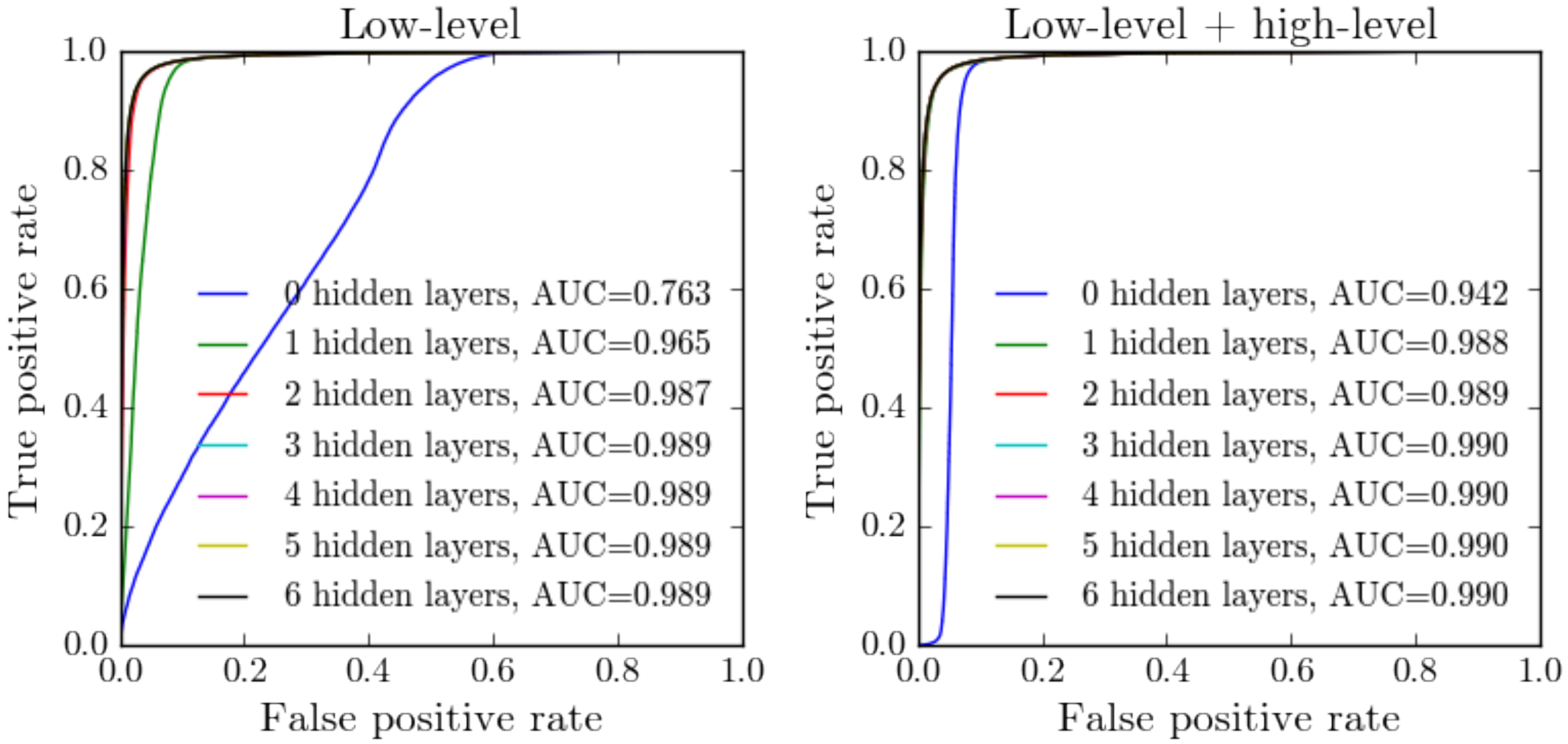
(a) Train machine on low level data
(b) Compute low level AUC
(c) Choose a variable: compute (planing) weights
(d) Train machine on weighted (planed) data
(e) Compute planed AUC
(f) Compare: looking for significant performance drop

Saturation

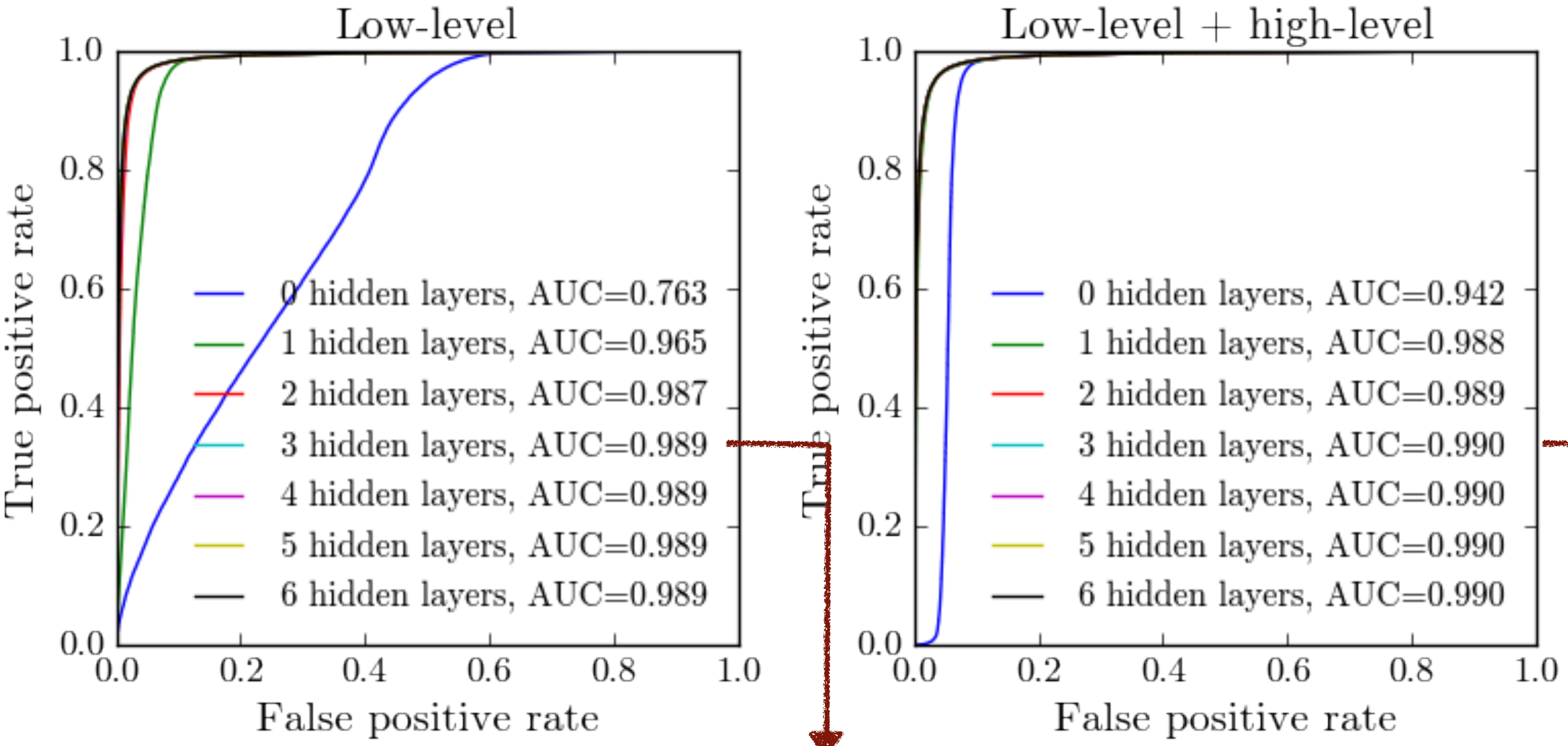
Used in Baldi, Sadowski, Whiteson [arXiv:1402.4735 and 1410.3469]; Baldi, Bauer, Eng, Sadowski, Whiteson [arXiv:1603.09349]; Guest, Collado, Baldi, Hsu, Urban, Whiteson [arXiv:1607.08633]; Datta, Larkoski [arXiv:1704.08249]; Aguilar-Saavedra, Collins, Mishra [arXiv:1709.01087]

(a) Train network on low level data
(b) Compute low level AUC
(c) Add high level variable
(d) Train new machine using low + high level info
(e) Compute AUC
(f) No performance gain implies information has been learned

Technical aside: using saturation to pick the network

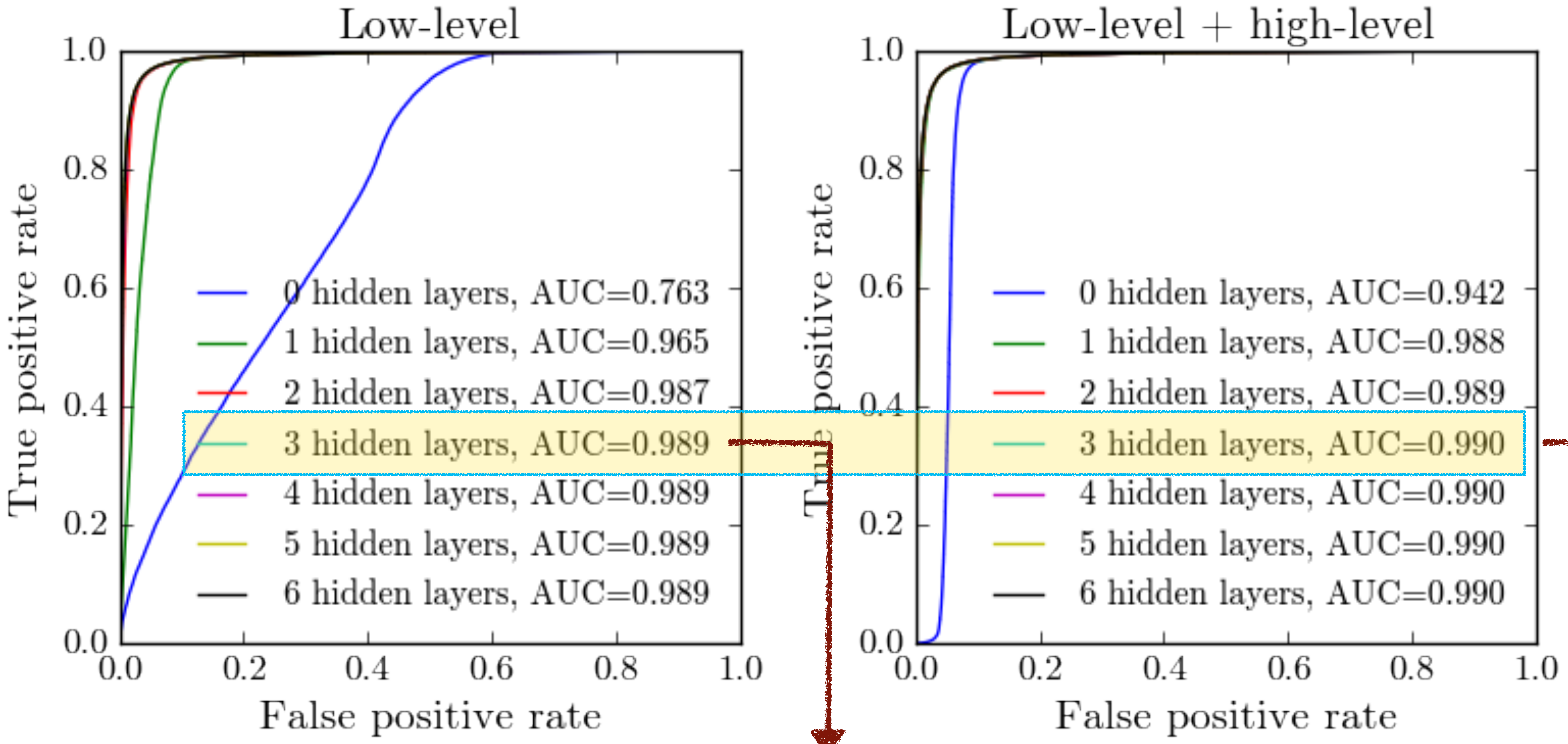


Technical aside: using saturation to pick the network



Minimal improvement

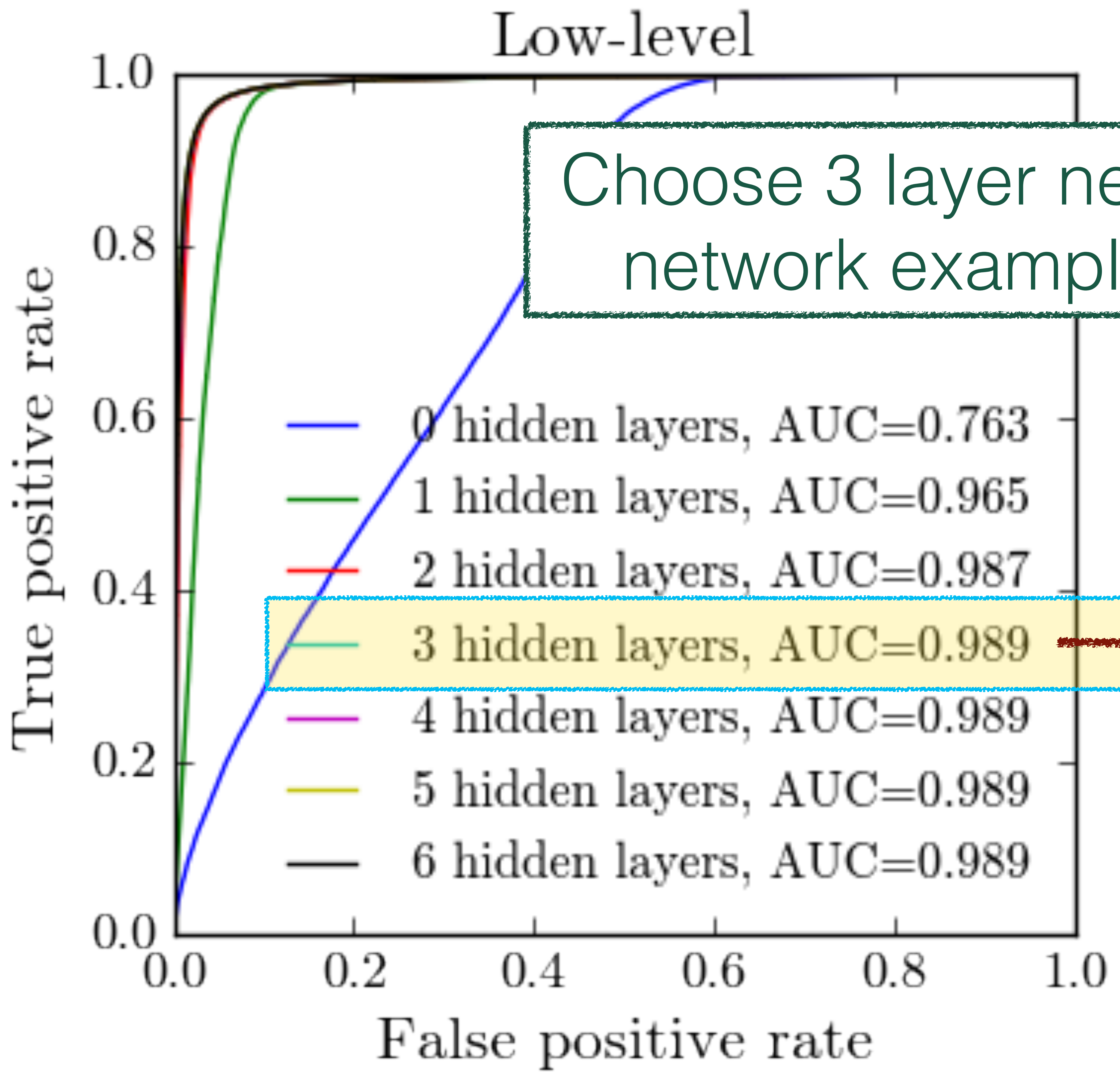
Technical aside: using saturation to pick the network

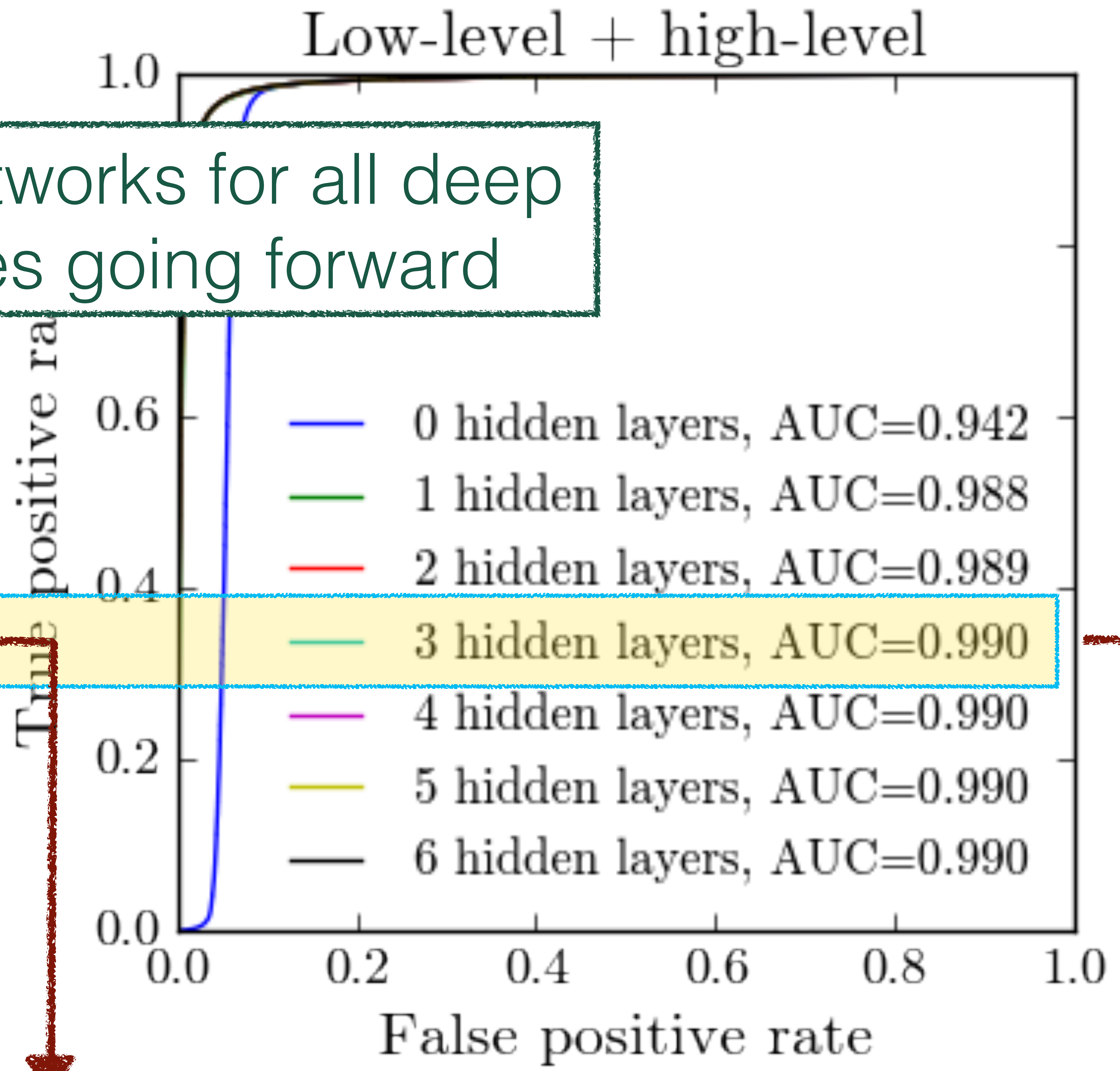


high-level not adding much

Minimal improvement

Technical aside: using saturation to pick the network



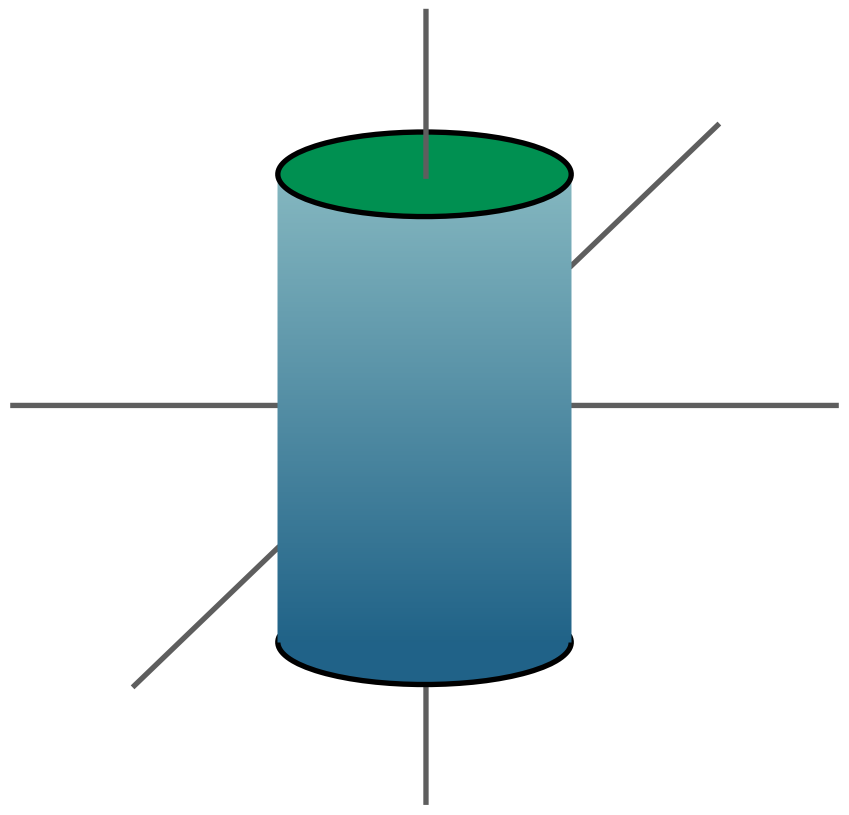


Choose 3 layer networks for all deep network examples going forward

high-level not adding much

Minimal improvement

Example with a toy model

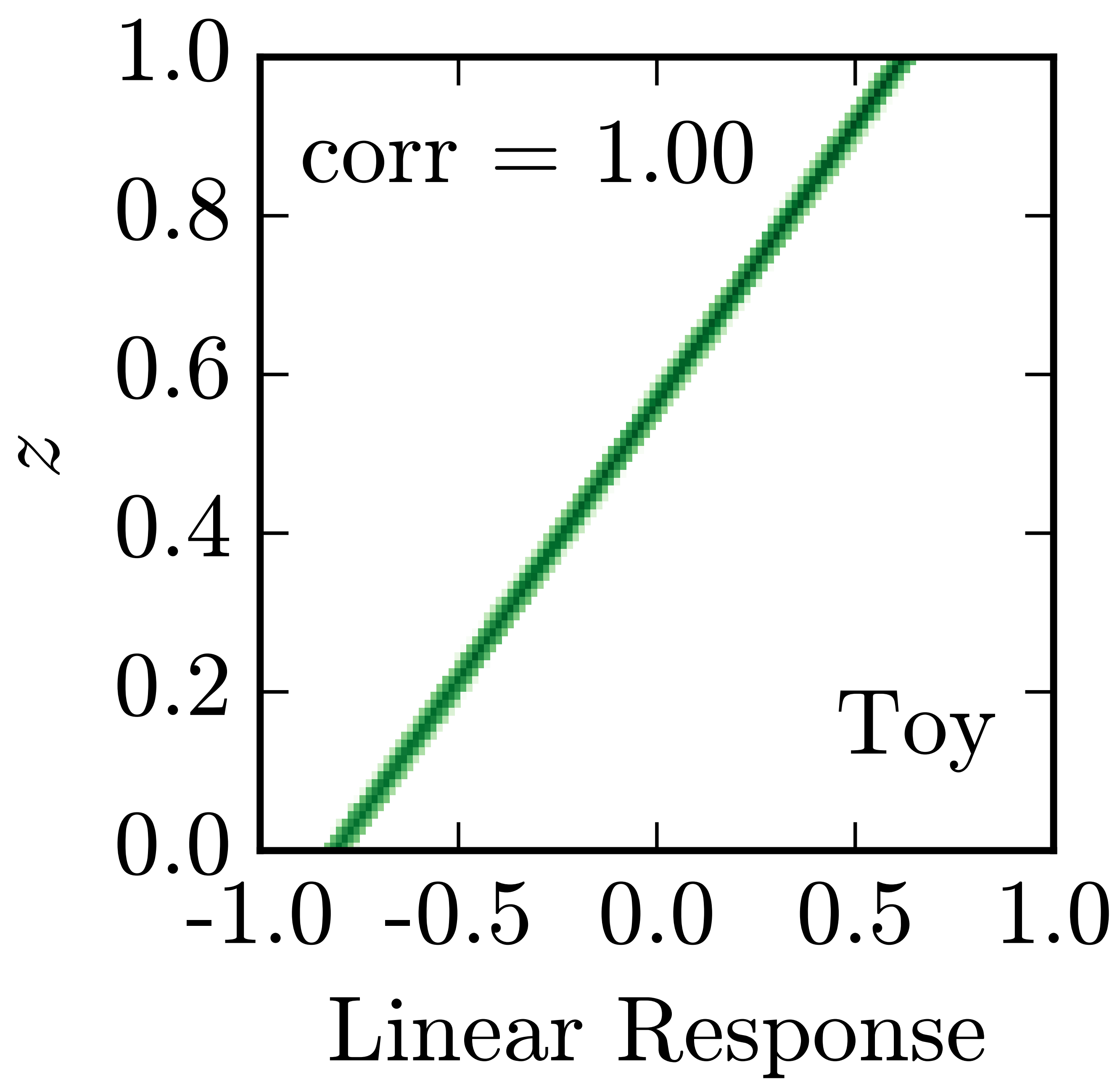


Signal Distribution: $f(\vec{x}) = \left[\Theta\left(r_0 - r\right) + C_r\right] \cdot \left[z \cdot B_z + C_z\right]$

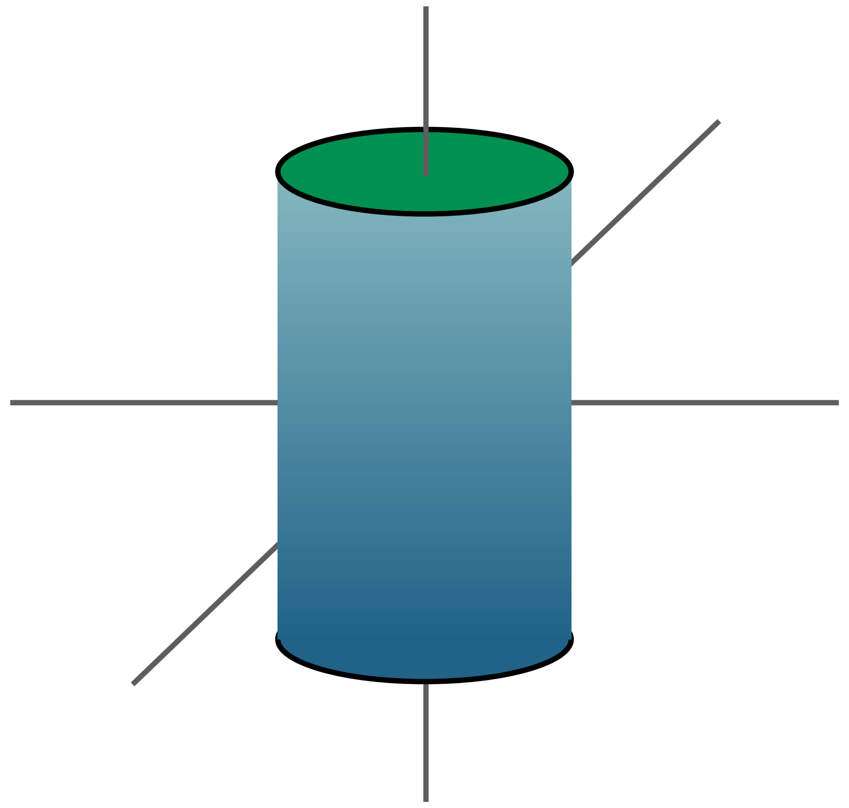
Background Distribution: Uniform

Results

(x, y, z)	r	Planed	Linear AUC	Deep AUC
✓	✘	✘	0.61275(01)	0.81243(45)
✓	✓	✘	0.79672(01)	0.81388(23)
✓	✘	r	0.61030(01)	0.61026(02)
✓	✘	(r, z)	0.5081(16)	0.49998(03)



Example with a toy model



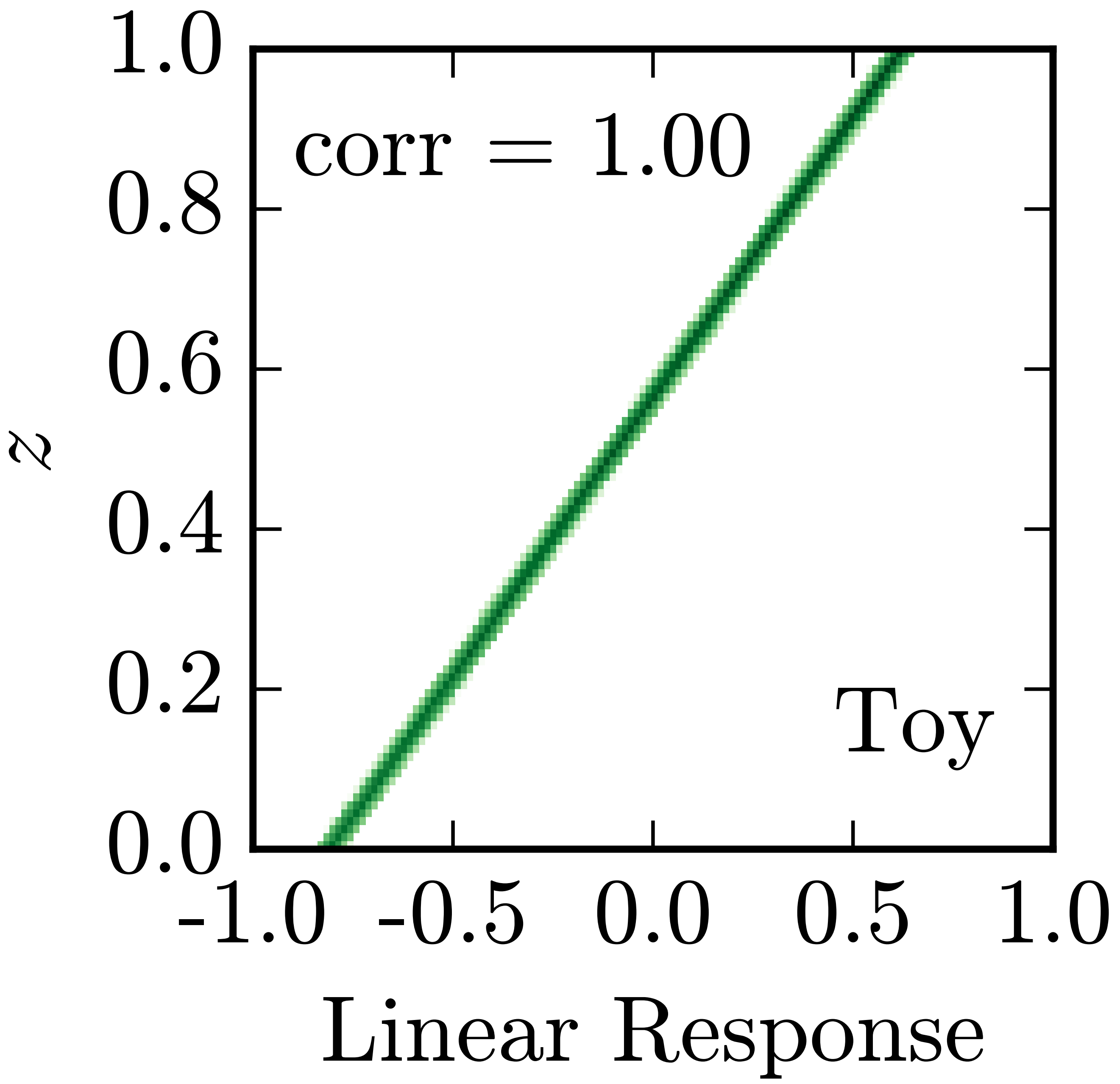
Signal Distribution: $f(\vec{x}) = [\Theta(r_0 - r) + C_r] \cdot [z \cdot B_z + C_z]$

Background Distribution: Uniform

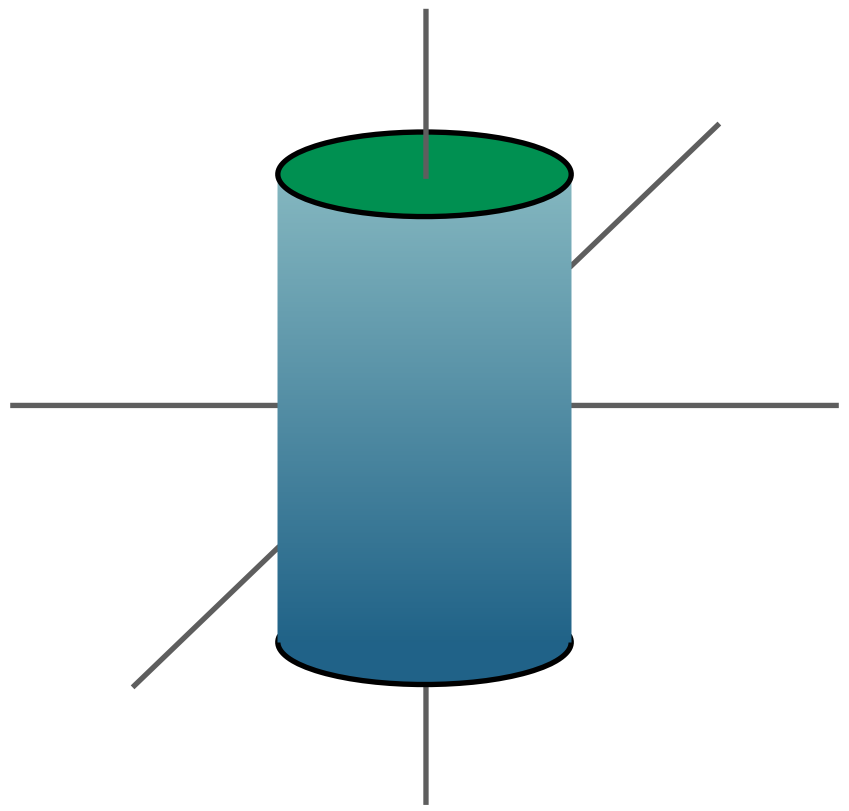
Check Saturation

Results

(x, y, z)	r	PLANED	LINEAR AUC	DEEP AUC
✓	✘	✘	0.61275(01)	0.81243(45)
✓	✓	✘	0.79672(01)	0.81388(23)
✓	✘	r	0.61030(01)	0.61026(02)
✓	✘	(r, z)	0.5081(16)	0.49998(03)



Example with a toy model



Signal Distribution: $f(\vec{x}) = [\Theta(r_0 - r) + C_r] \cdot [z \cdot B_z + C_z]$

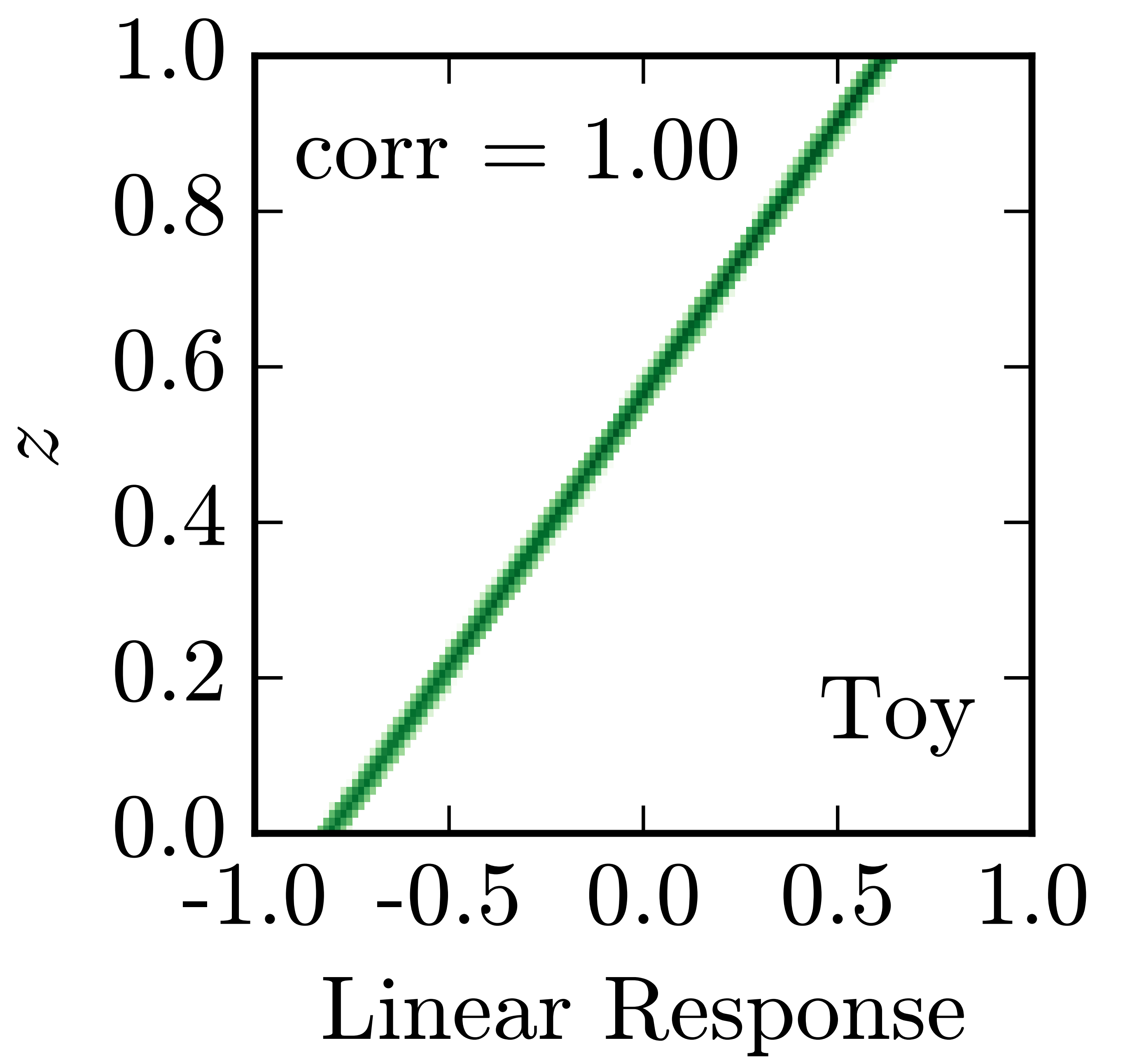
Background Distribution: Uniform

Check Saturation

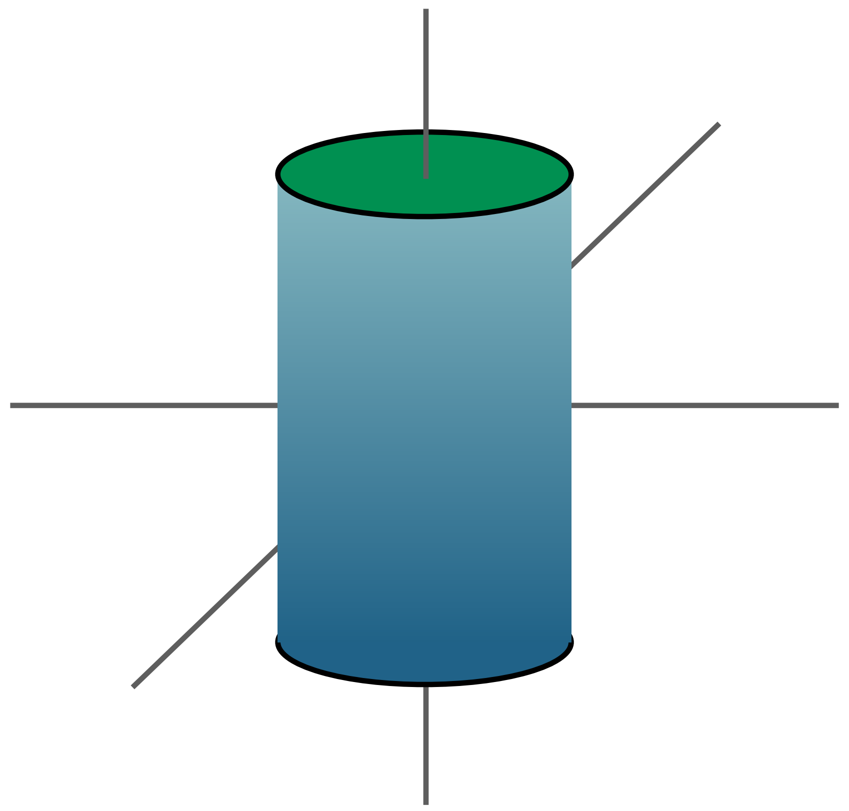
Results

(x, y, z)	r	PLANED	LINEAR AUC	DEEP AUC
✓	✘	✘	0.61275(01)	0.81243(45)
✓	✓	✘	0.79672(01)	0.81388(23)
✓	✘	r	0.61030(01)	0.61026(02)
✓	✘	(r, z)	0.5081(16)	0.49998(03)

No discrimination left



Example with a toy model



Signal Distribution: $f(\vec{x}) = [\Theta(r_0 - r) + C_r] \cdot [z \cdot B_z + C_z]$

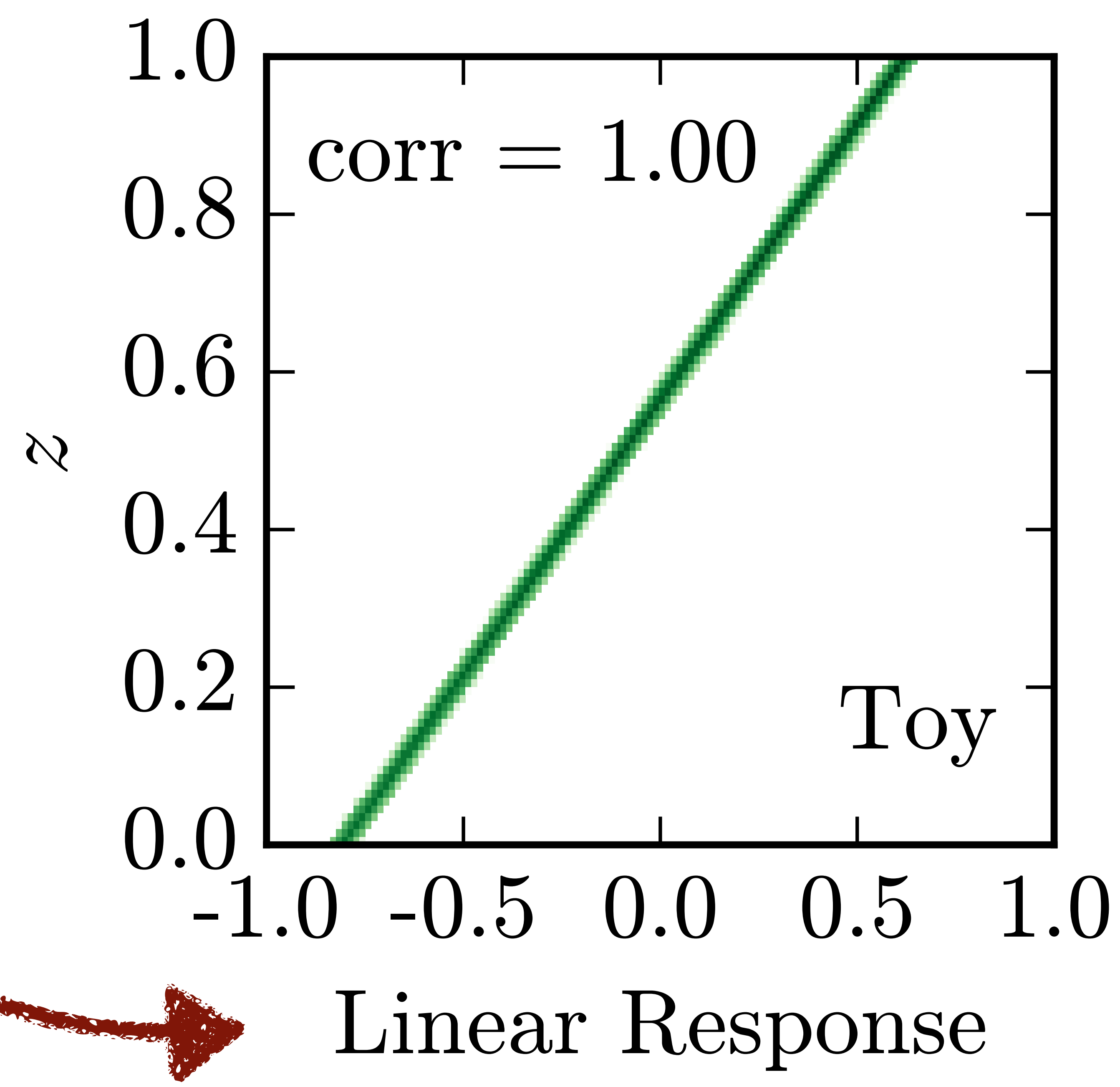
Background Distribution: Uniform

Check Saturation

Results

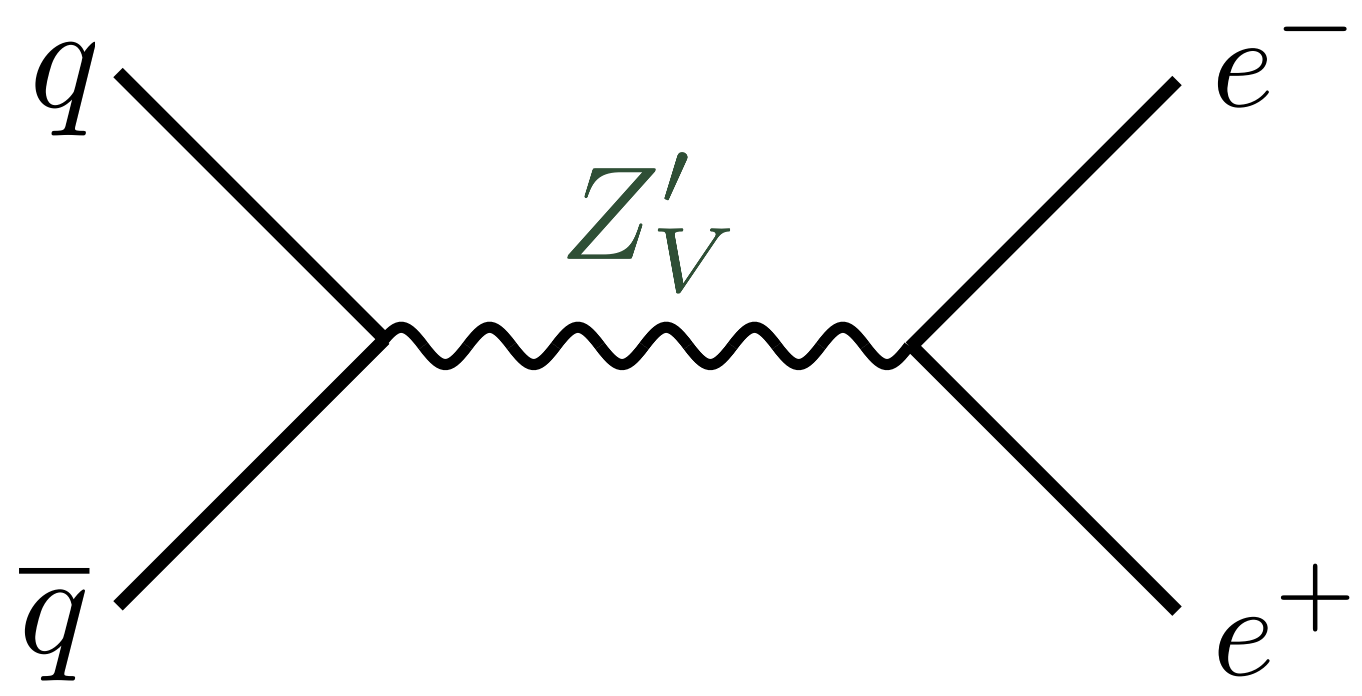
(x, y, z)	r	PLANED	LINEAR AUC	DEEP AUC
✓	✘	✘	0.61275(01)	0.81243(45)
✓	✓	✘	0.79672(01)	0.81388(23)
✓	✘	r	0.61030(01)	0.61026(02)
✓	✘	(r, z)	0.5081(16)	0.49998(03)

No discrimination left



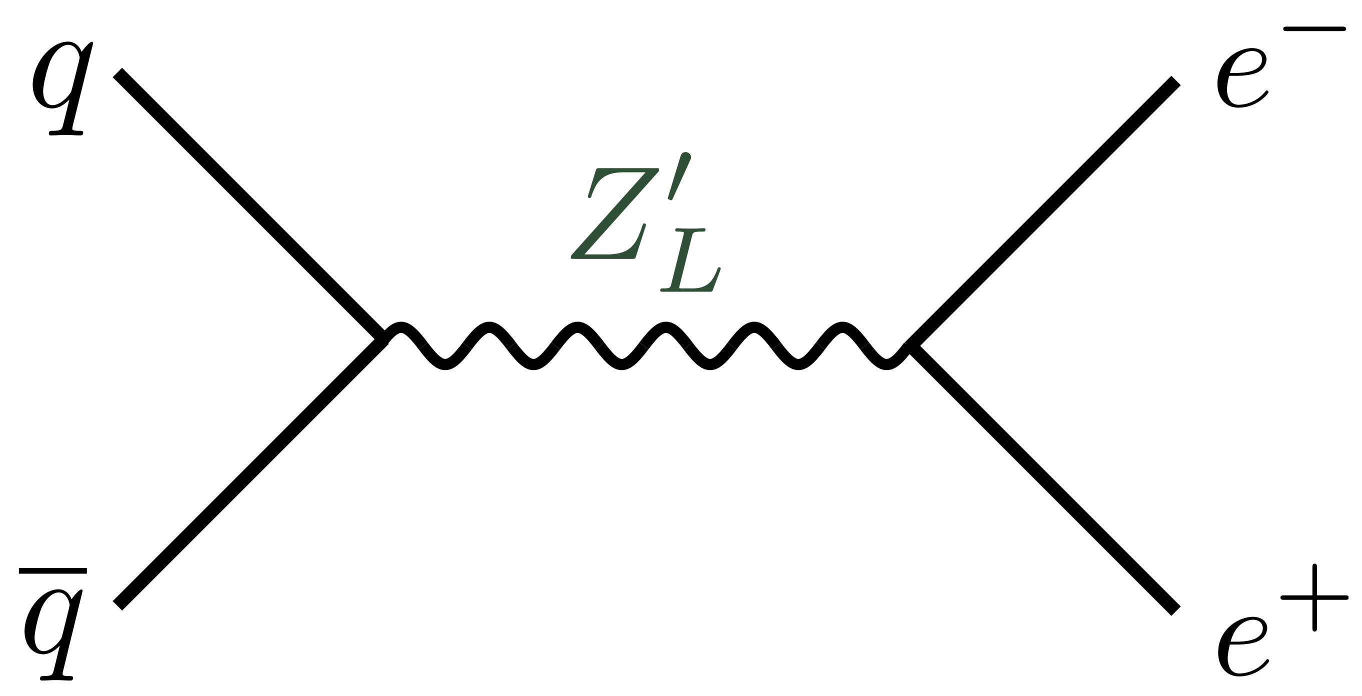
BSM Models

$$\mathcal{L} \supset Z'_\mu \sum_f Q_f \left(g_L \bar{f} \gamma^\mu P_L f + g_R \bar{f} \gamma^\mu P_R f \right)$$



Vector couplings

$$g_L = g_R$$



Left-handed couplings

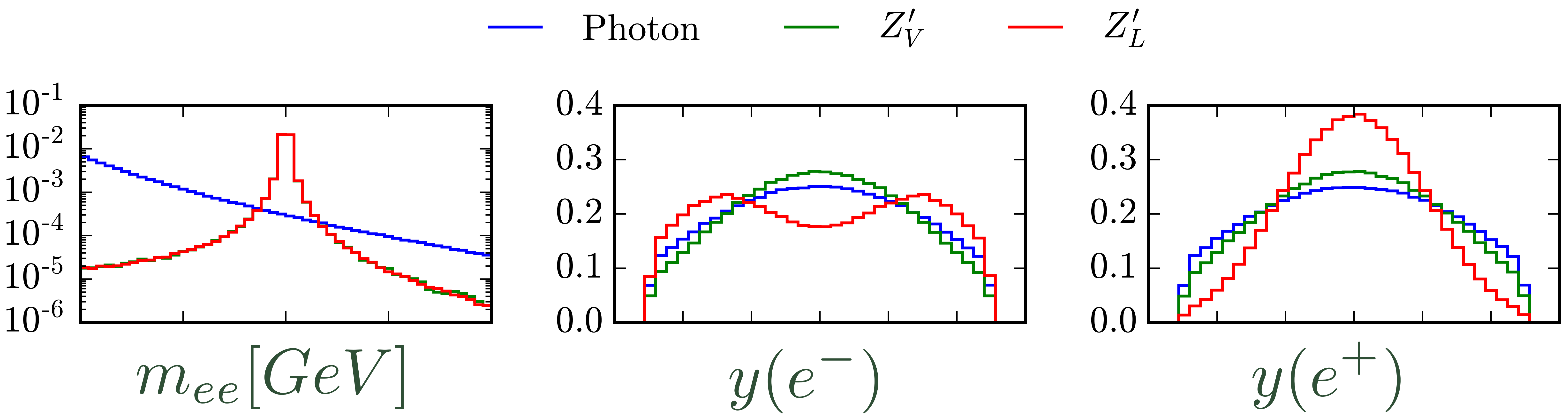
$$g_R = 0$$

8 low-level features (4-momentum of electron and positron)

What is the machine learning?

How much information is there to learn in a given distribution?

Original distributions



Planed distributions

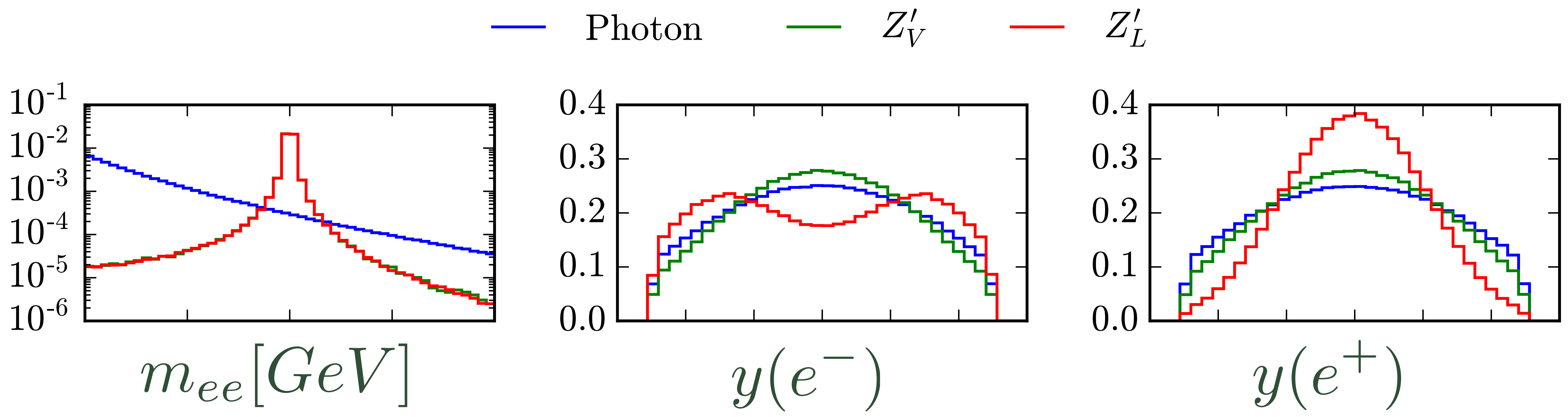
Vector Couplings

$(E,\vec{p})$	m	Planed	Linear AUC	Deep AUC
✓	✗	✗	0.746221(01)	0.988510(98)
✓	✓	✗	0.938967(01)	0.989007(03)
✓	✗	m	0.50550(29)	0.4942(48)

What is the machine learning?

How much information is there to learn in a given distribution?

Original distributions



Planed distributions

Vector Couplings

$(E, \vec{p})$	m	Planed	Linear AUC	Deep AUC
✓	✗	✗	0.746221(01)	0.988510(98)
✓	✓	✗	0.938967(01)	0.989007(03)
✓	✗	m	0.50550(29)	0.4942(48)

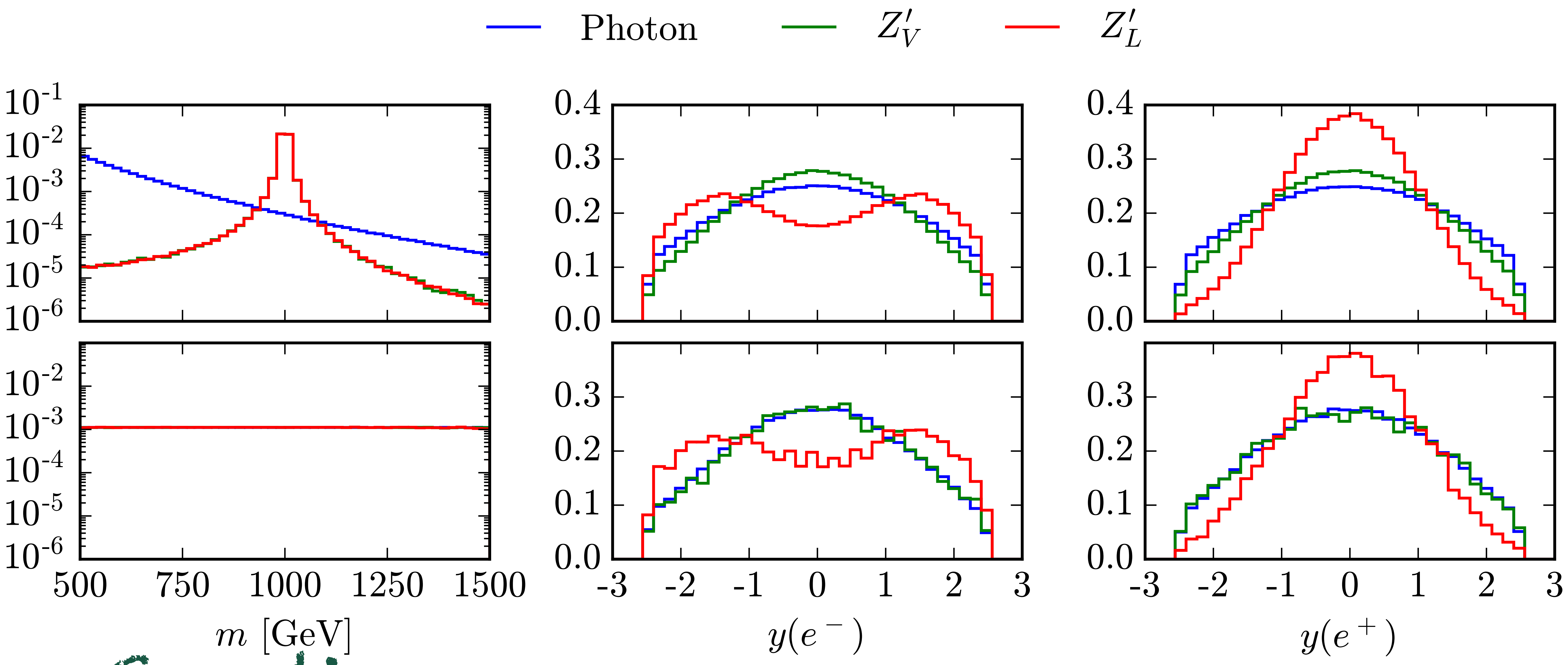
Something more than m_{ee}?

What is the machine learning?

How much information is there to learn in a given distribution?

Original distributions

Planed distributions



Vector Couplings

$(E,\vec{p})$	m	PLANED	LINEAR AUC	DEEP AUC
✓	✘	✘	0.746221(01)	0.988510(98)
✓	✓	✘	0.938967(01)	0.989007(03)
✓	✘	m	0.50550(29)	0.4942(48)

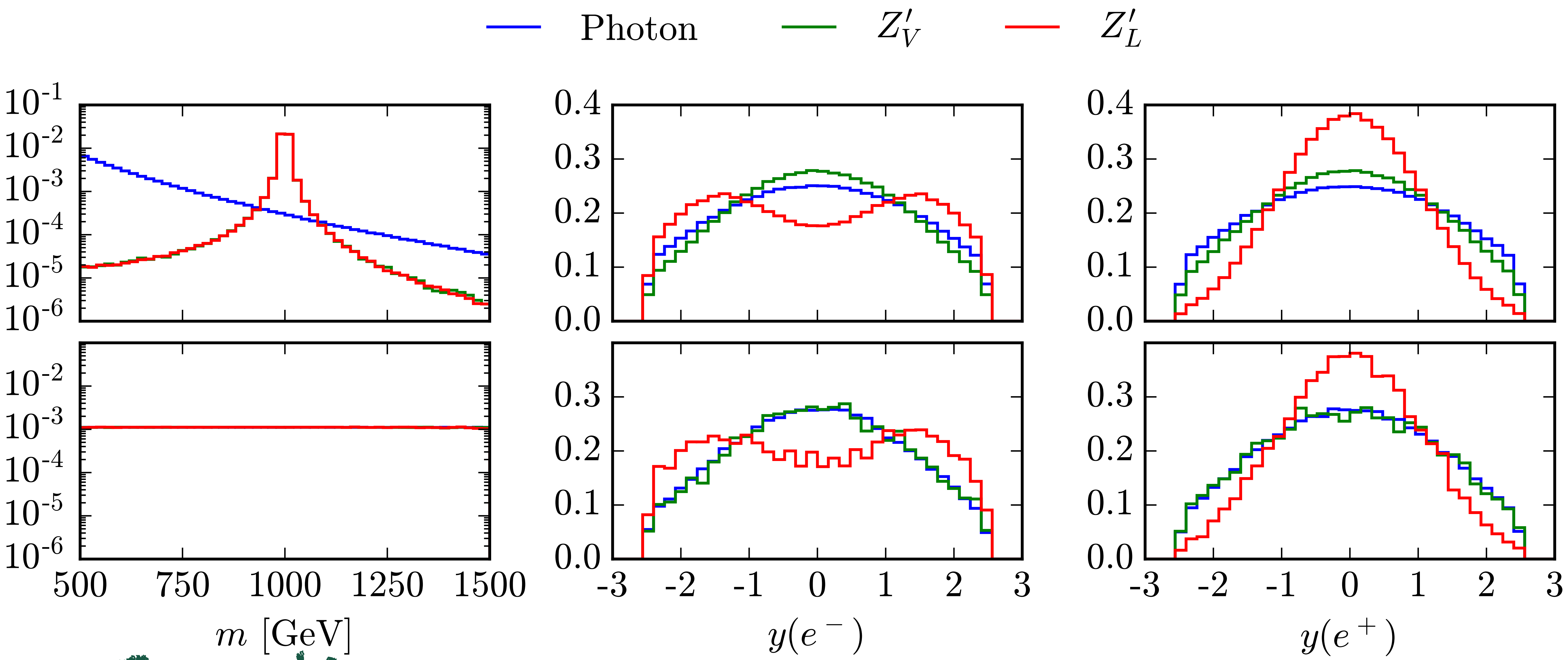
Something more than m_{ee}?

What is the machine learning?

How much information is there to learn in a given distribution?

Original distributions

Planed distributions



Vector Couplings

$(E,\vec{p})$	m	PLANED	LINEAR AUC	DEEP AUC
✓	✗	✗	0.746221(01)	0.988510(98)
✓	✓	✗	0.938967(01)	0.989007(03)
✓	✗	m	0.50550(29)	0.4942(48)

Something more than m_{ee}?

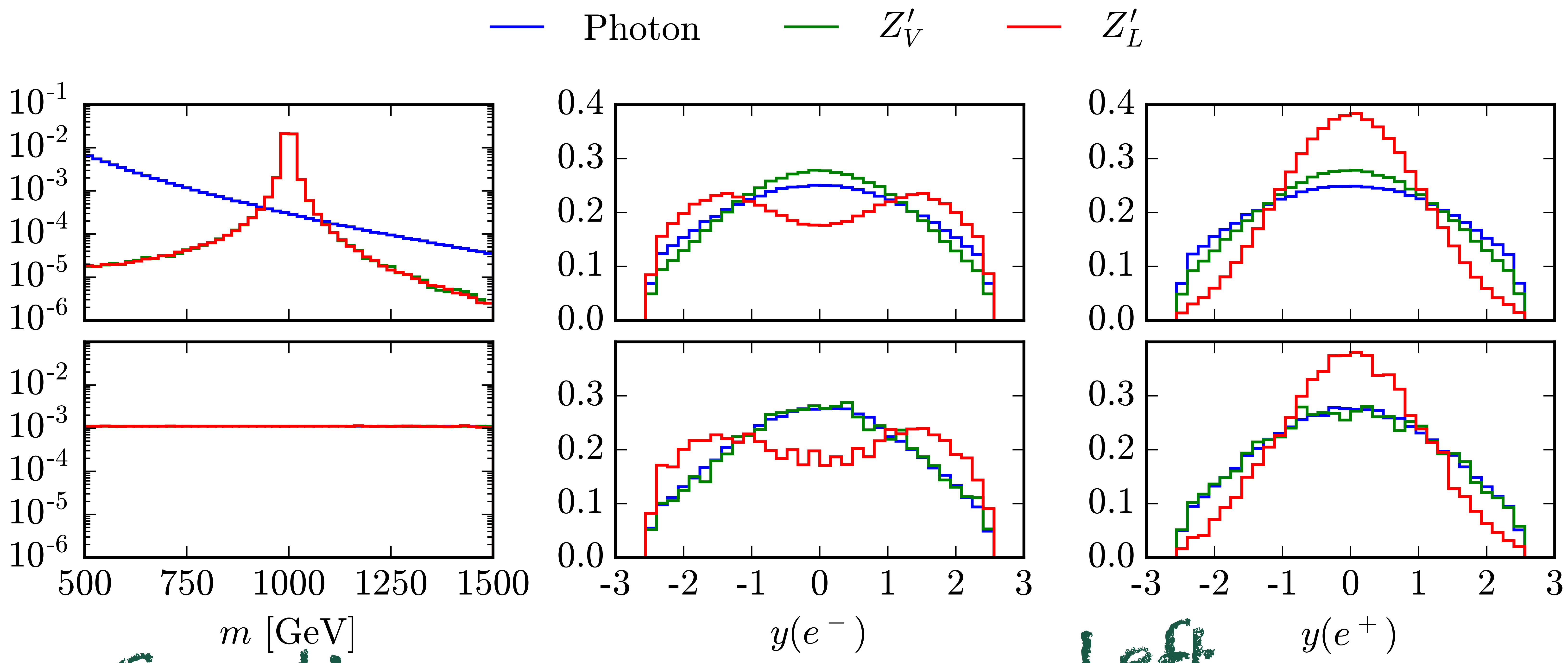
NO!

What is the machine learning?

How much information is there to learn in a given distribution?

Original distributions

Planed distributions



Vector Couplings

$(E,\vec{p})$	m	Planed	Linear AUC	Deep AUC
✓	✗	✗	0.746221(01)	0.988510(98)
✓	✓	✗	0.938967(01)	0.989007(03)
✓	✗	m	0.50550(29)	0.4942(48)

Left

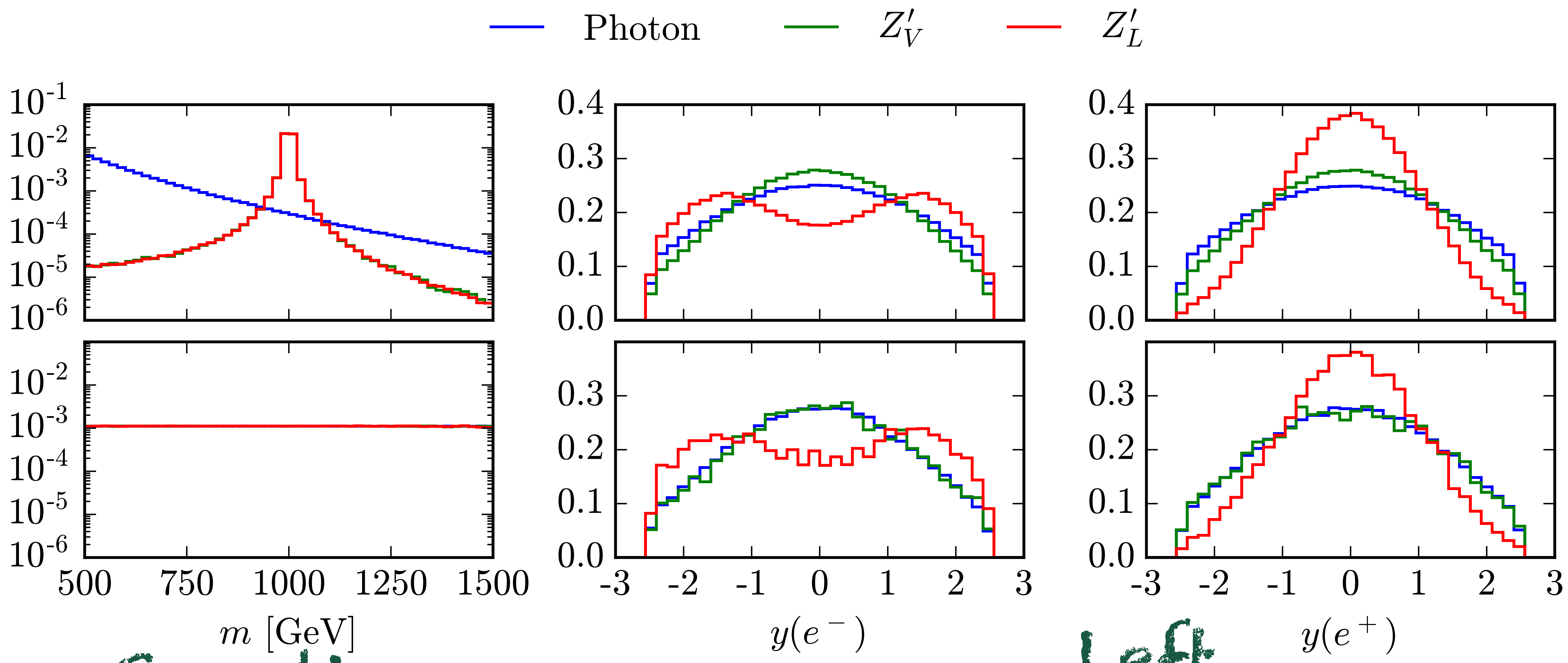
$(E,\vec{p})$	m	Planed	Linear AUC	Deep AUC
✓	✗	✗	0.763280(05)	0.989353(59)
✓	✓	✗	0.942004(02)	0.989826(10)
✓	✗	m	0.626648(28)	0.6258(24)
✓	✗	$(m, \Delta\|y\|)$	0.52421(15)	0.5320(25)

What is the machine learning?

How much information is there to learn in a given distribution?

Original distributions

Planed distributions



Vector Couplings

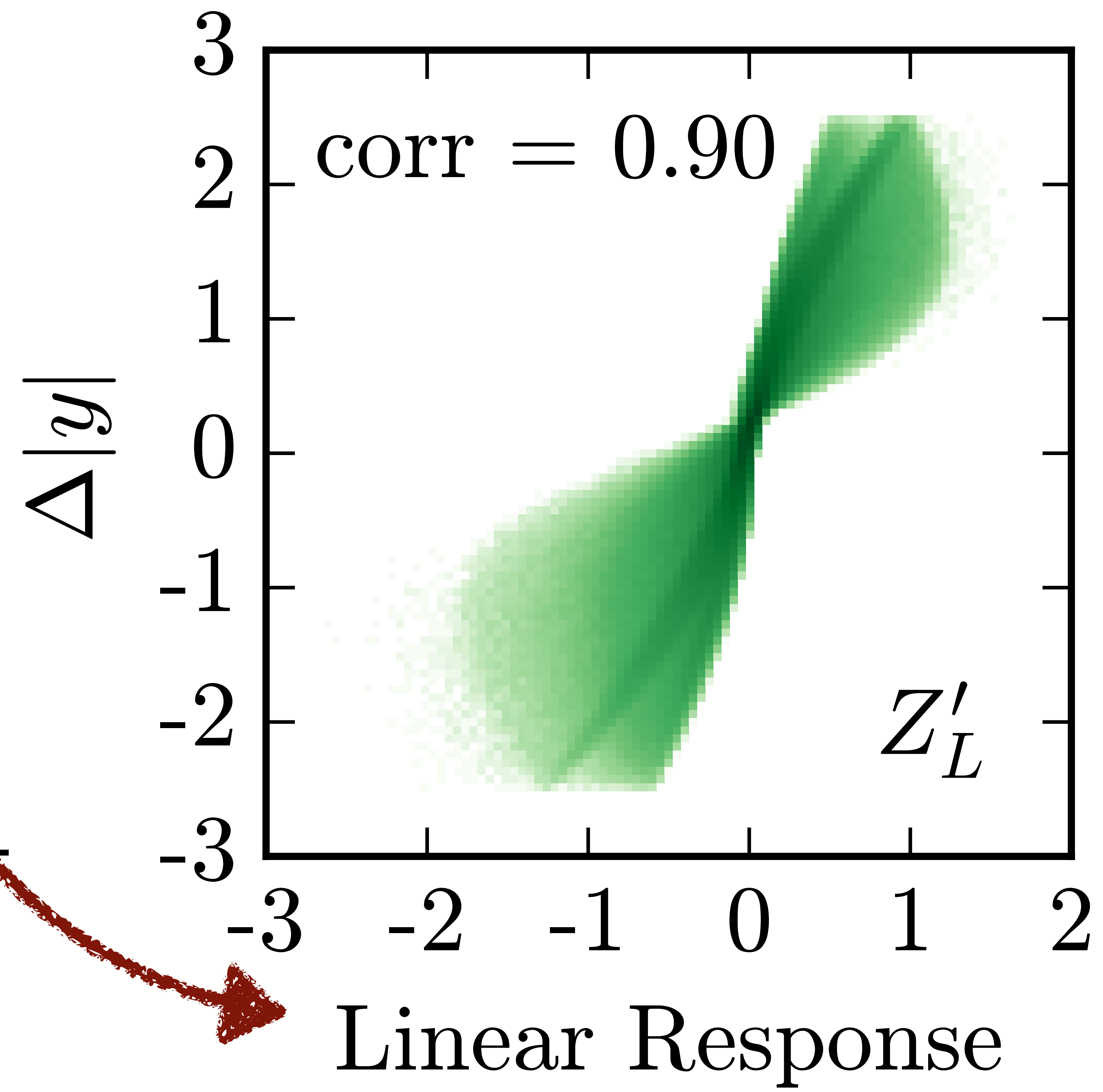
$(E,\vec{p})$	m	PLANED	LINEAR AUC	DEEP AUC
✓	✗	✗	0.746221(01)	0.988510(98)
✓	✓	✗	0.938967(01)	0.989007(03)
✓	✗	m	0.50550(29)	0.4942(48)

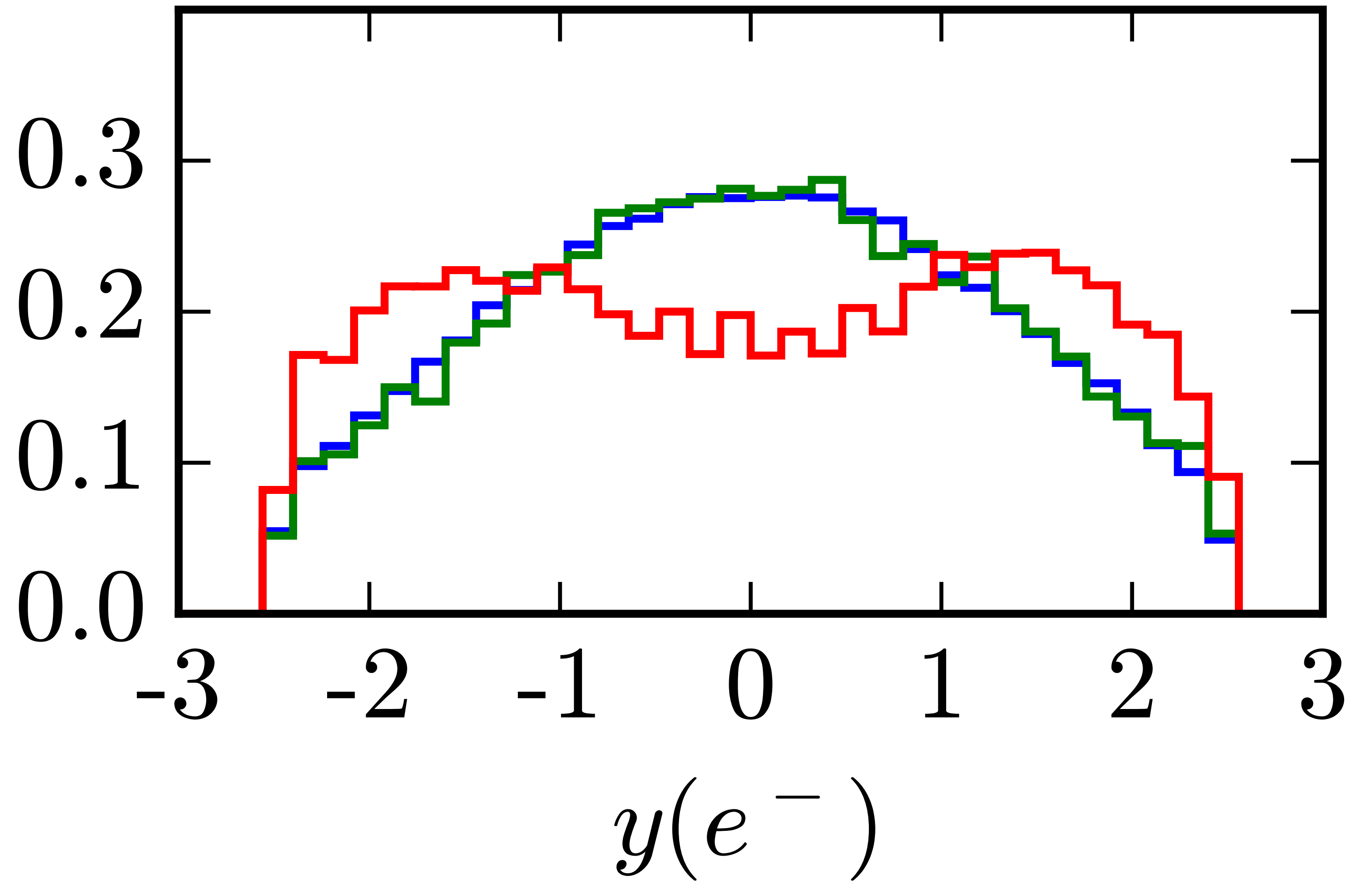
Left

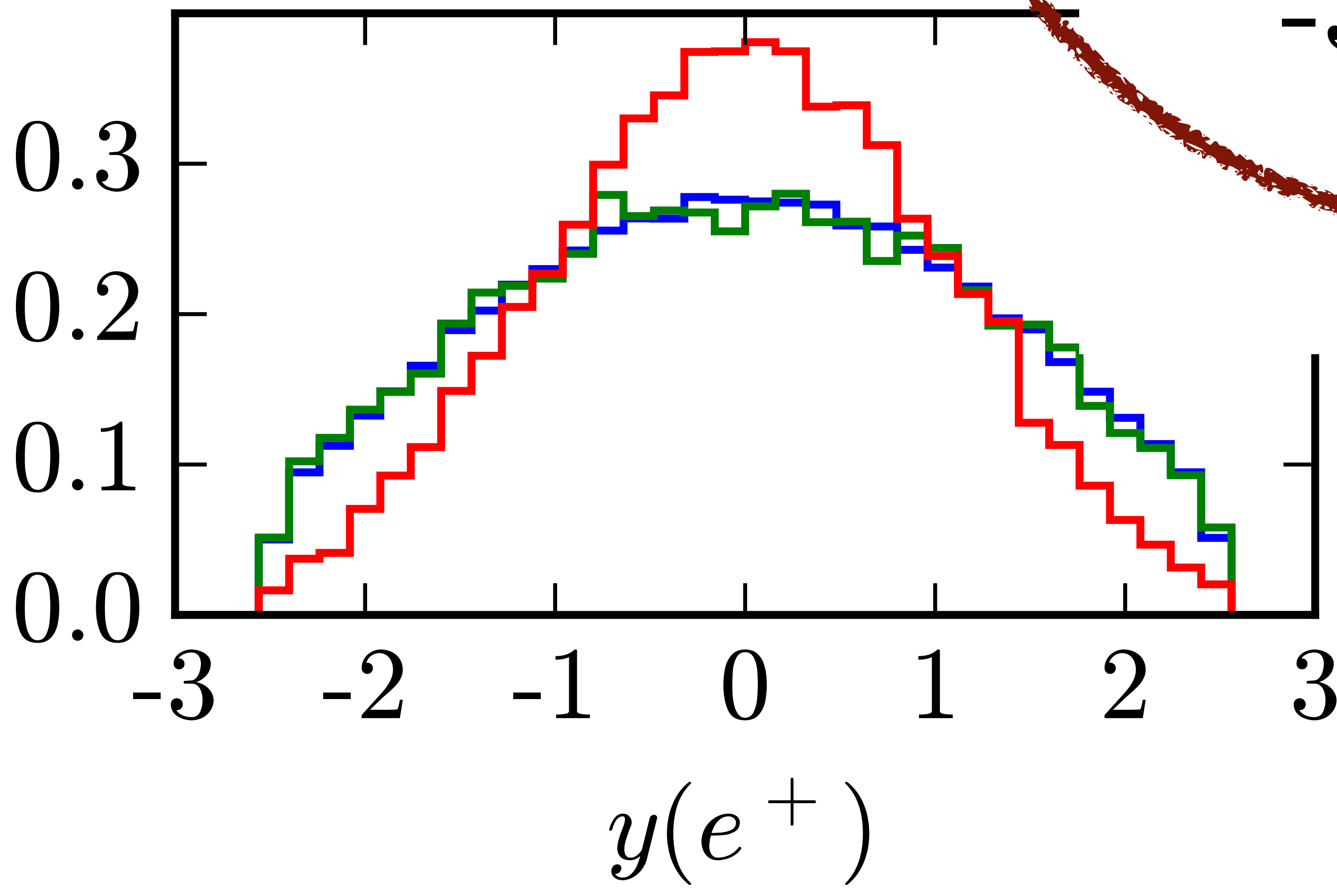
$(E,\vec{p})$	m	PLANED	LINEAR AUC	DEEP AUC
✓	✗	✗	0.763280(05)	0.989353(59)
✓	✓	✗	0.942004(02)	0.989826(10)
✓	✗	m	0.626648(28)	0.6258(24)
✓	✗	$(m, \Delta\lvert y\rvert)$	0.52421(15)	0.5320(25)

What is the machine learning?

$(E,\vec{p})$	m	PLANED	LINEAR AUC	DEEP AUC
✓	✗	✗	0.763280(05)	0.989353(59)
✓	✓	✗	0.942004(02)	0.989826(10)
✓	✗	m	0.626648(28)	0.6258(24)
✓	✗	$(m, \Delta\|y\|)$	0.52421(15)	0.5320(25)

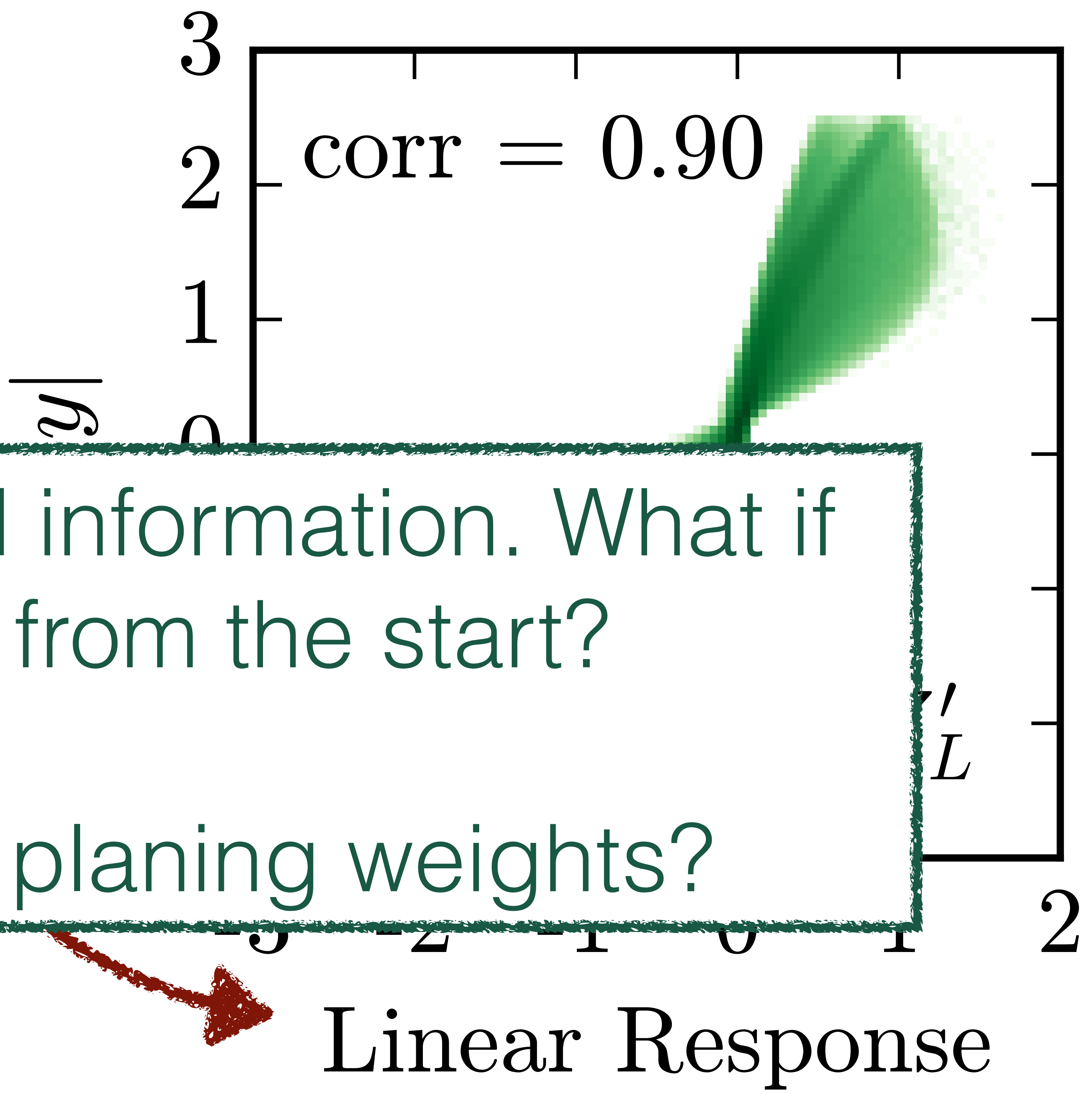






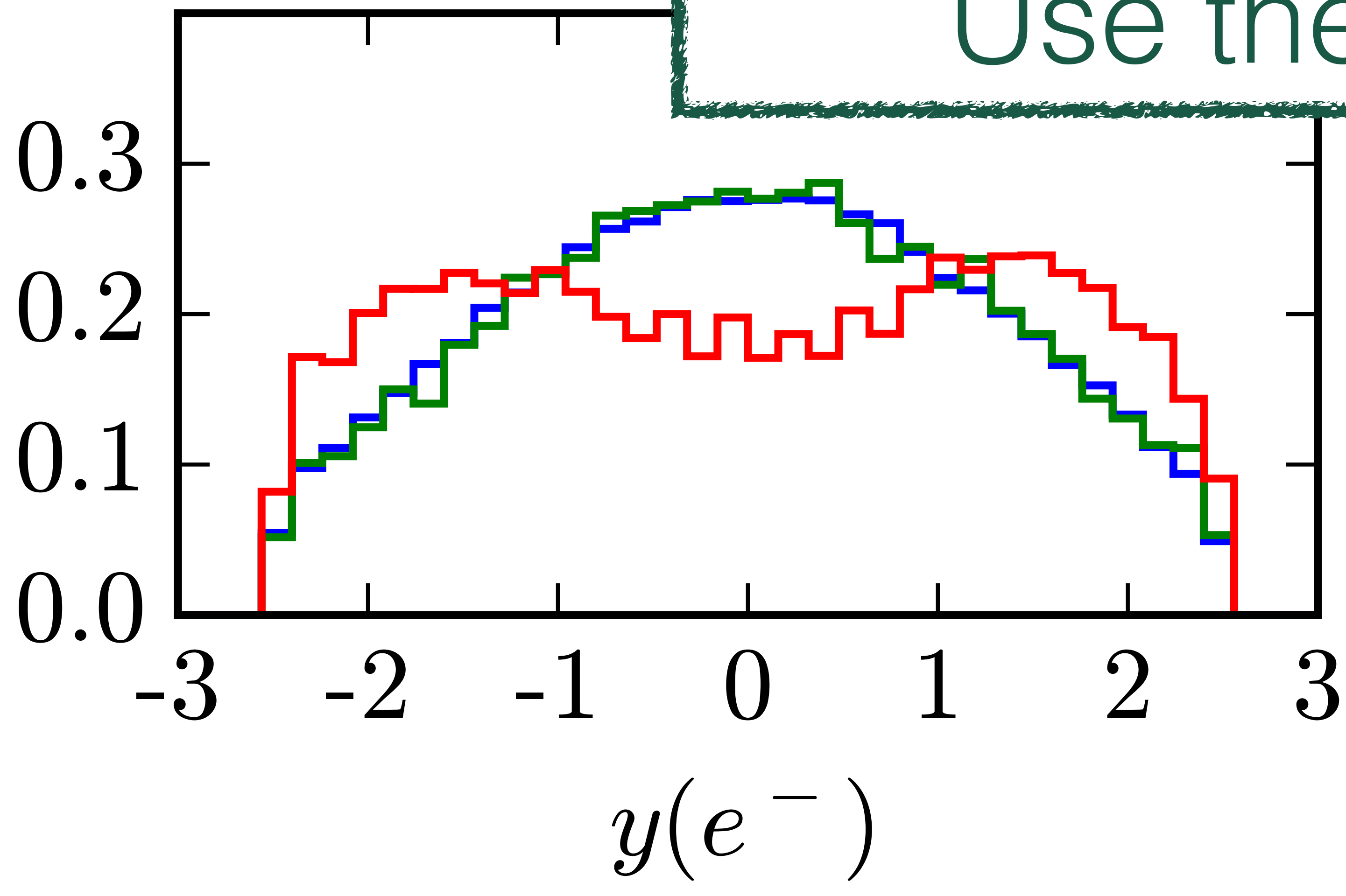
What is the machine learning?

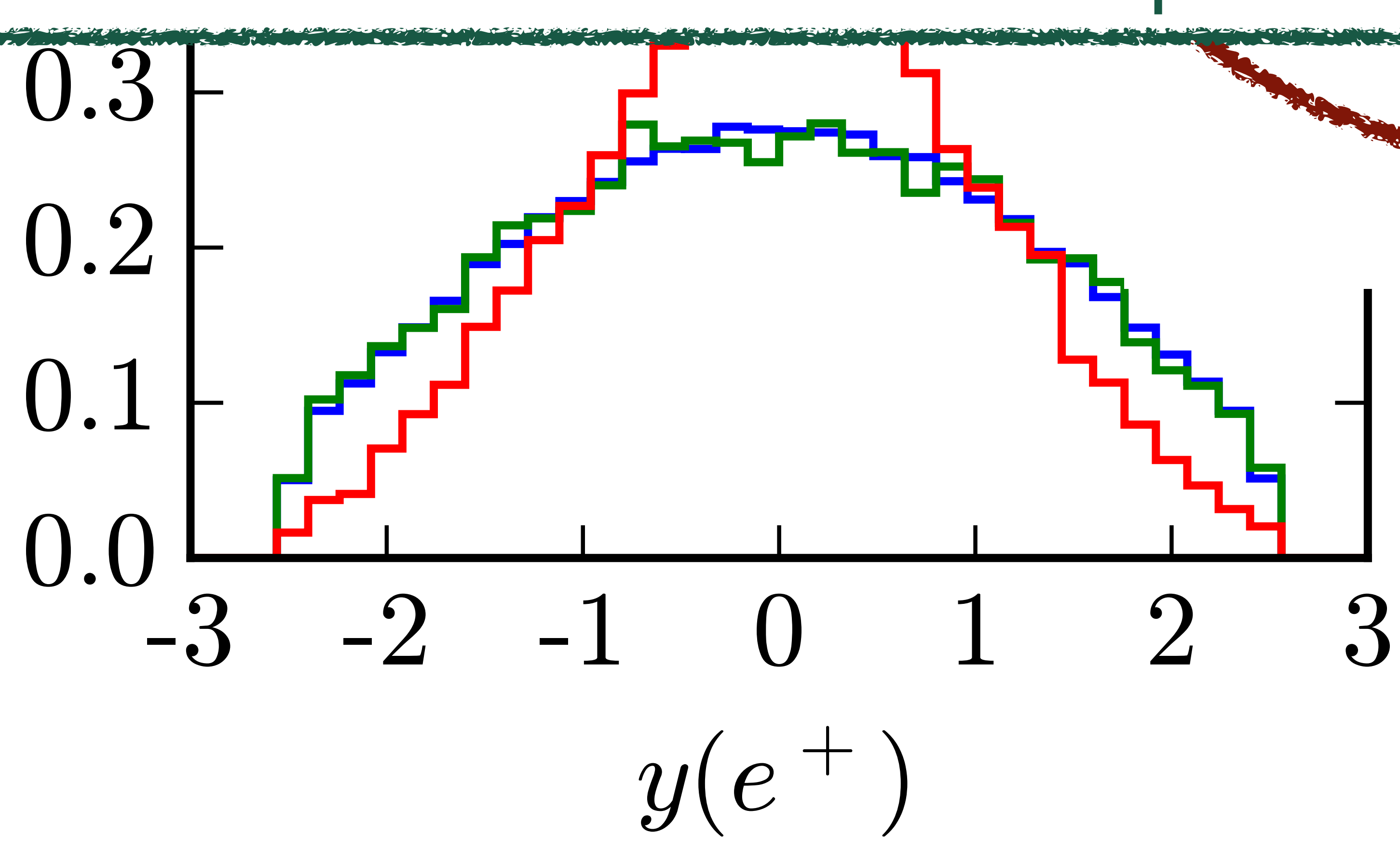
$(E, \vec{p})$	m	PLANED	LINEAR AUC	DEEP AUC
✓	✗	✗	0.763280(05)	0.989353(59)
✓	✓	✗	0.942004(02)	0.989826(10)
✓	✗	m	0.626648(28)	0.6258(24)
✓	✗	$(m, \Delta\|y\|)$	0.52421(15)	0.5320(25)



Iterative procedure to remove all information. What if we don't know the physics from the start?

Use the machine to learn the planing weights?





Conclusion

- Machine learning trades intuitive, physical parameters for improved results
  - Image pixels
  - Grammar of QCD and jet clustering
  - Etc.

- Jet substructure has observables which span the space, can show when adding more information doesn't help

- Through planing, possible to find remove information, while maintaining the same network structure. Allows one to determine what information the network needs to learn.

Backup

ROC Curves

Basic Neural Network Example

What is machine learning for?

Particle Physics Interlude

Machine learning particle physics

Can identify and measure photons, electrons, muons, and things made of quarks

Neutrinos (and some BSM particles) escape detection

Beams travel in ±z direction, no momentum in (x, y) plane

Particle Physics Interlude

Machine learning particle physics

Can identify and measure photons, electrons, muons, and things made of quarks

Energy and momentum vector

Neutrinos (and some BSM particles) escape detection

Beams travel in ±z direction, no momentum in (x, y) plane

Particle Physics Interlude

Machine learning particle physics

Can identify and measure photons, electrons, muons, and things made of quarks

Energy and momentum vector

jets (b-jets)

Neutrinos (and some BSM particles) escape detection

Beams travel in ±z direction, no momentum in (x, y) plane

Particle Physics Interlude

Machine learning particle physics

Can identify and measure photons, electrons, muons, and things made of quarks

Energy and momentum vector

jets (b-jets)

Neutrinos (and some BSM particles) escape detection

Beams travel in ±z direction, no momentum in (x, y) plane

Missing momentum in (x, y) plane

Particle Physics Interlude

Machine learning particle physics

Can identify and measure photons, electrons, muons, and things made of quarks

Energy and momentum vector

jets (b-jets)

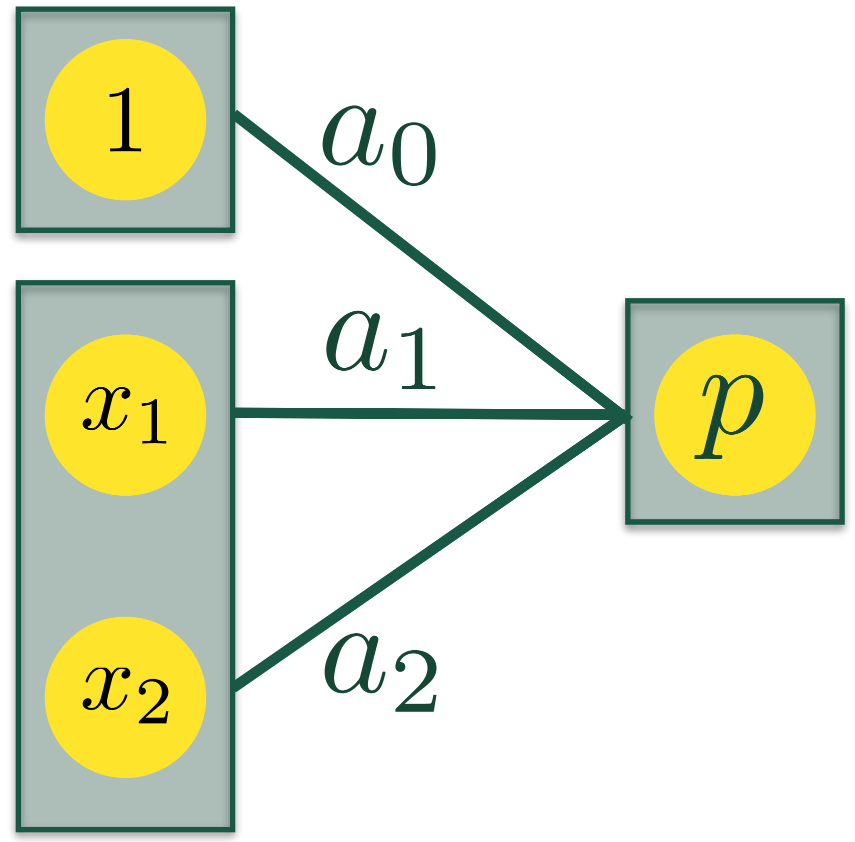
Which heavy particle decayed to the final state particles?

Neutrinos (and some BSM particles) escape detection

Beams travel in ±z direction, no momentum in (x, y) plane

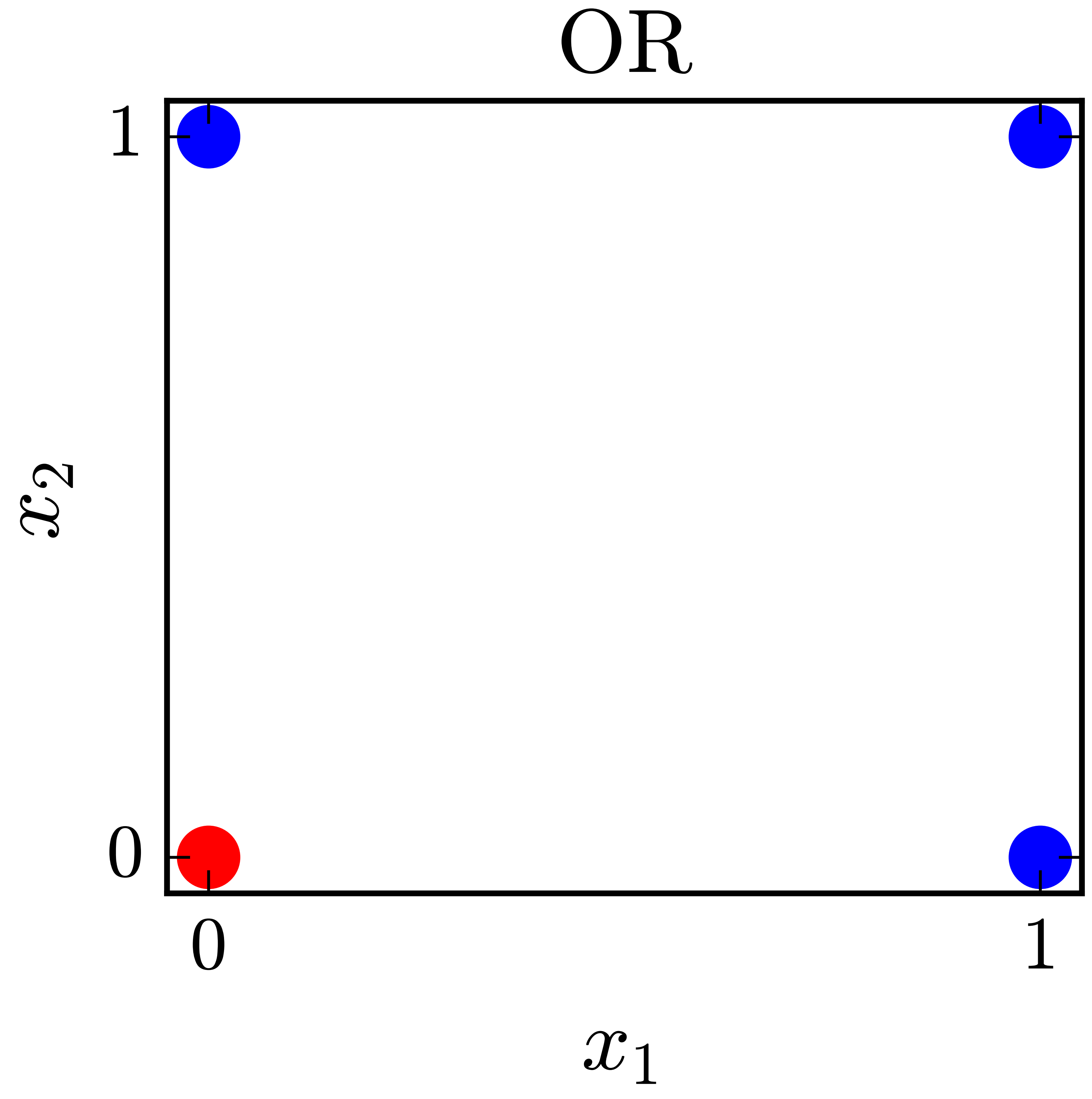
Missing momentum in (x, y) plane

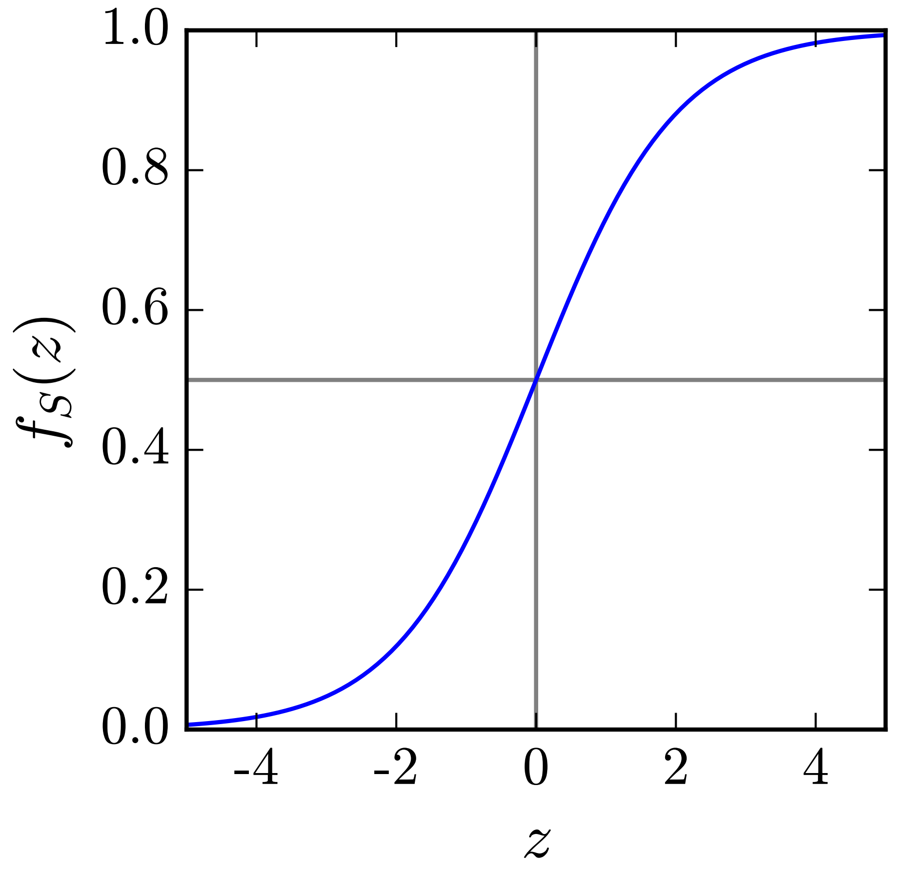
Neural Networks



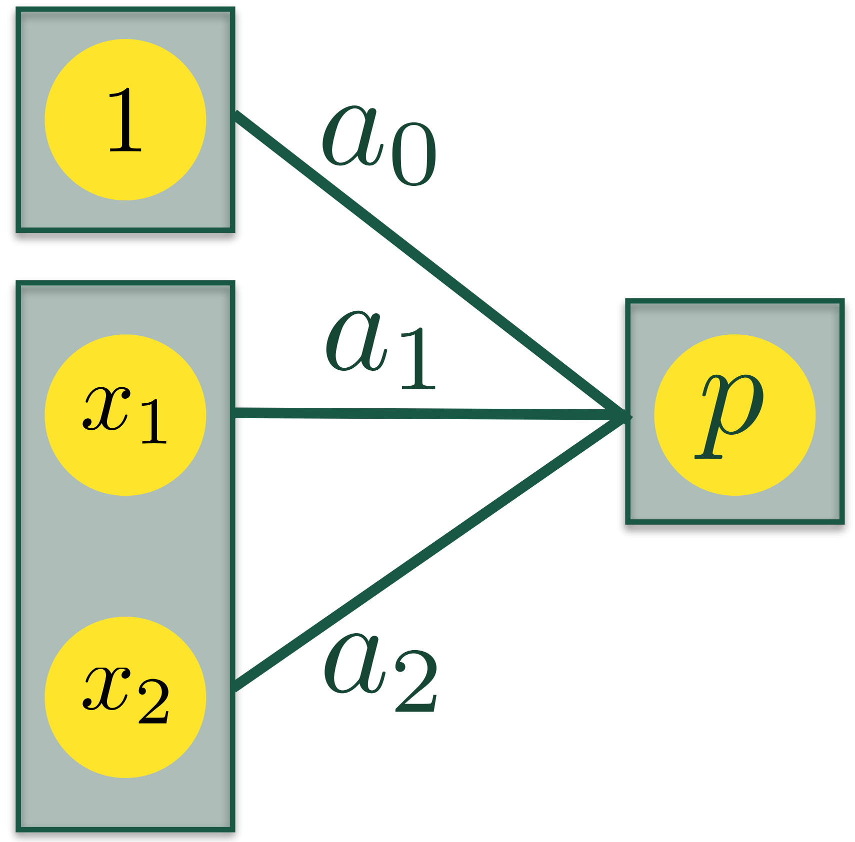
OR

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

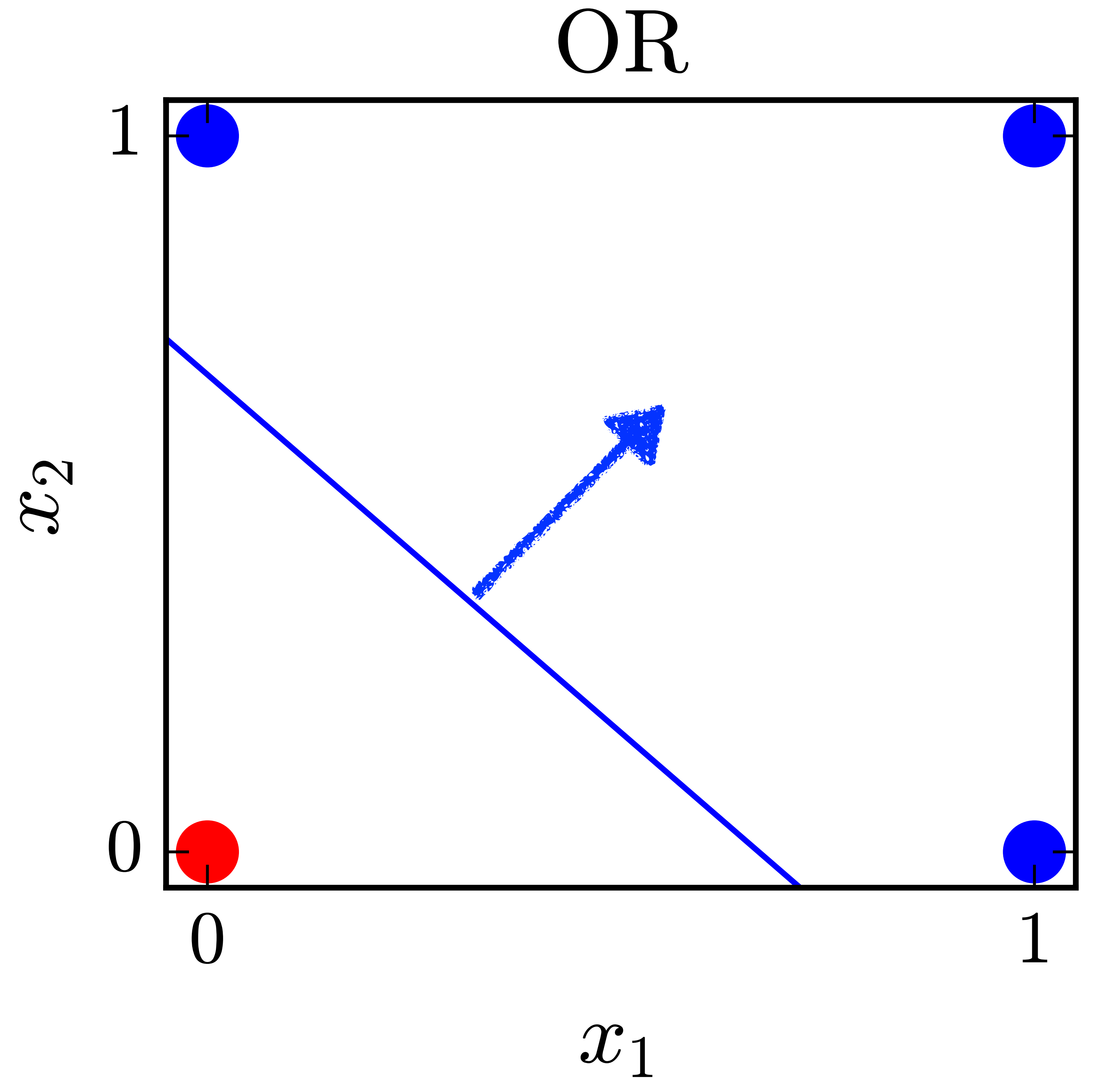


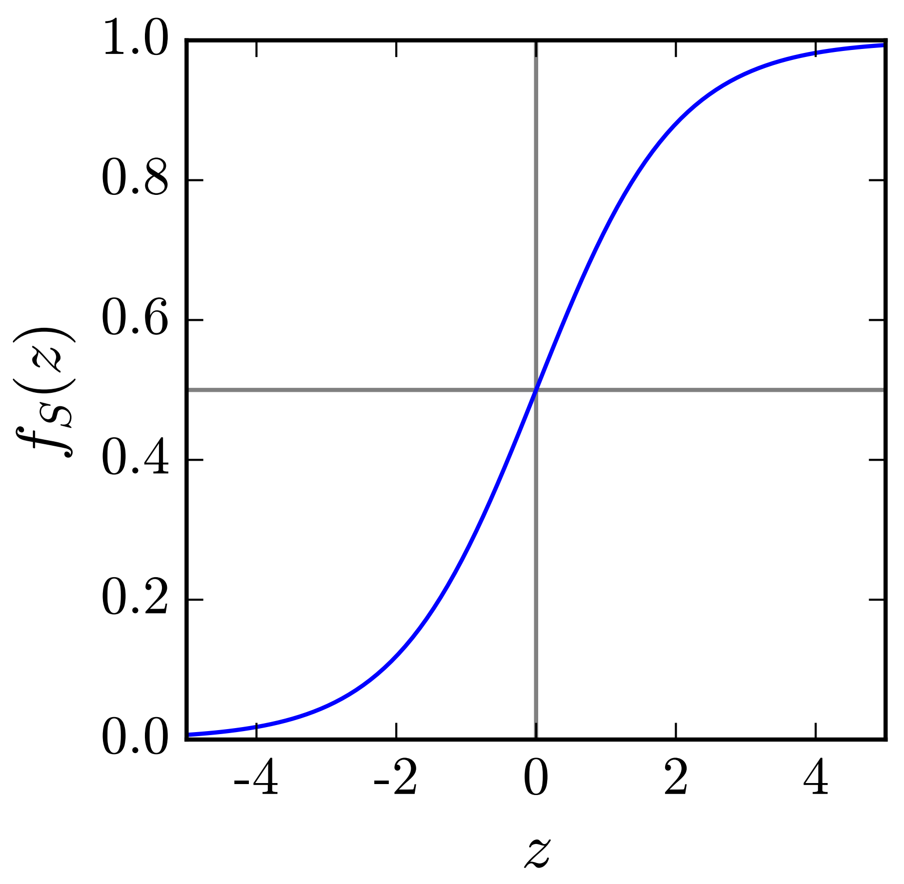


Neural Networks



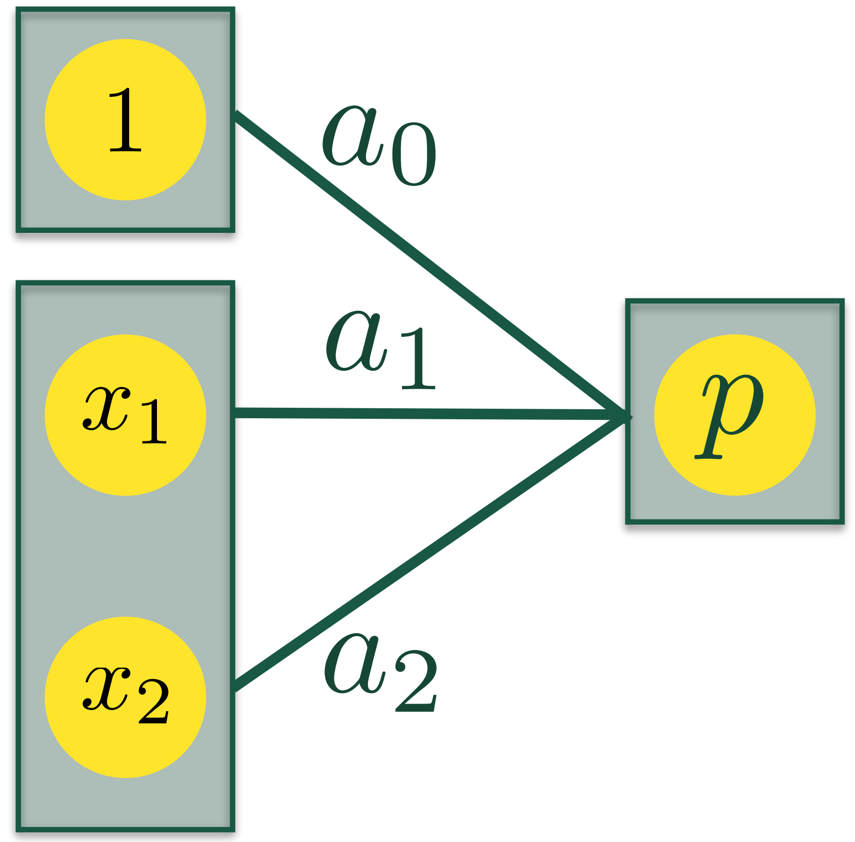
OR		
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

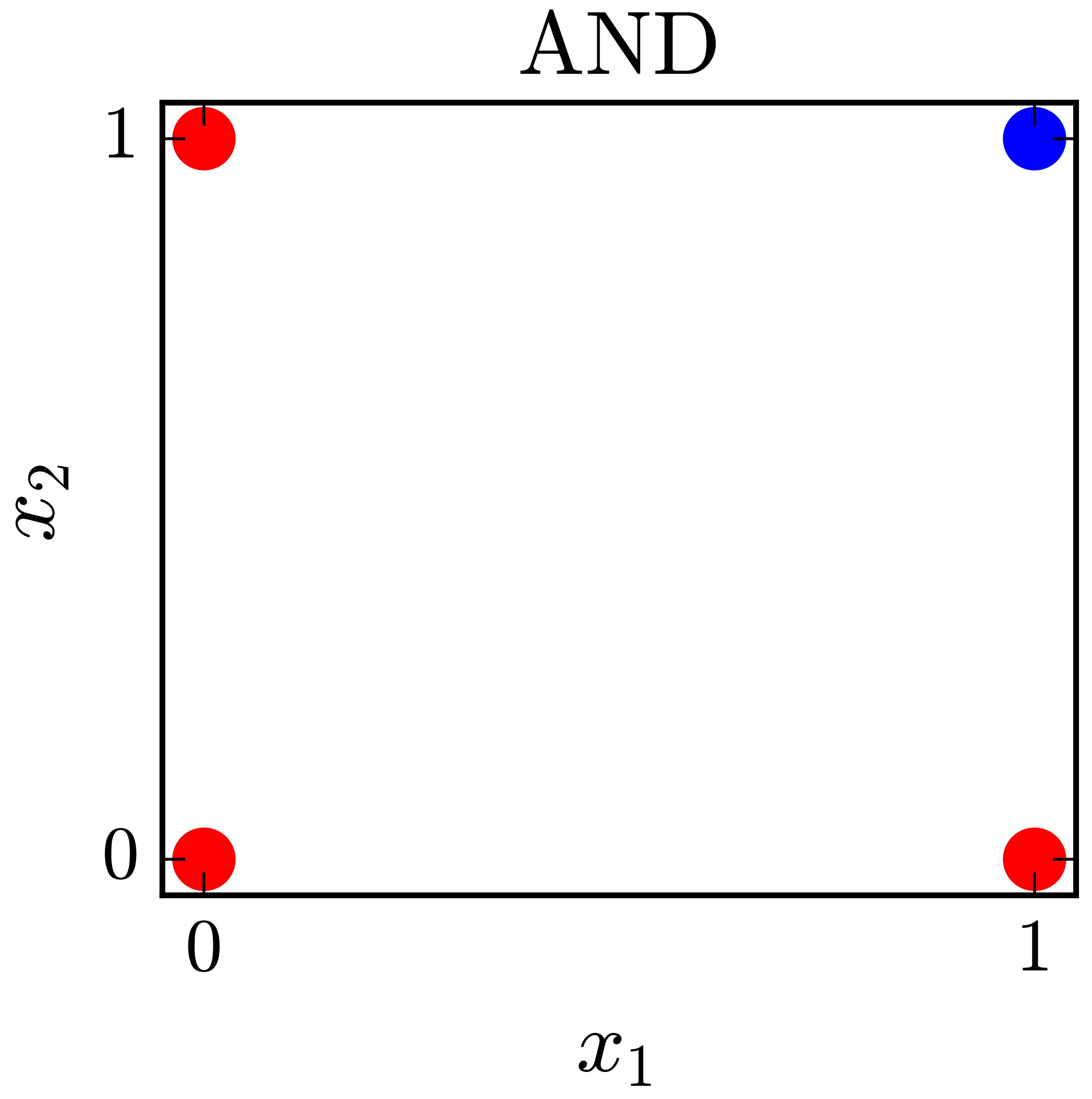




$$a_0 = -10, \quad a_1 = 15, \quad a_2 = 15$$

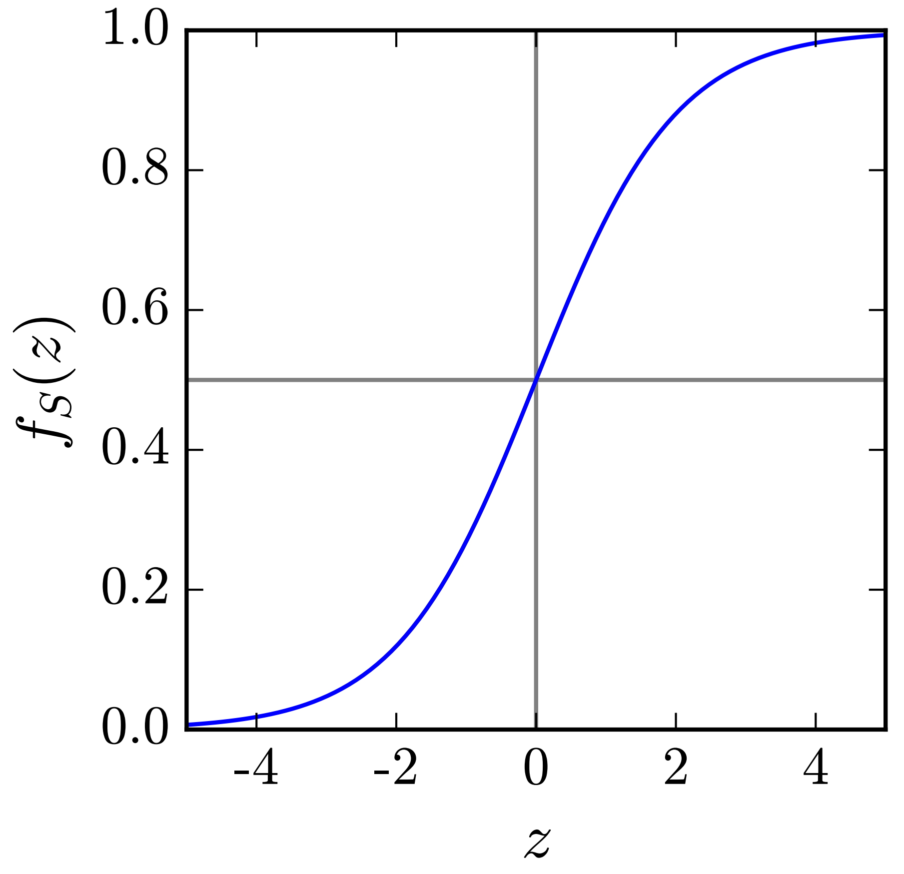
Neural Networks



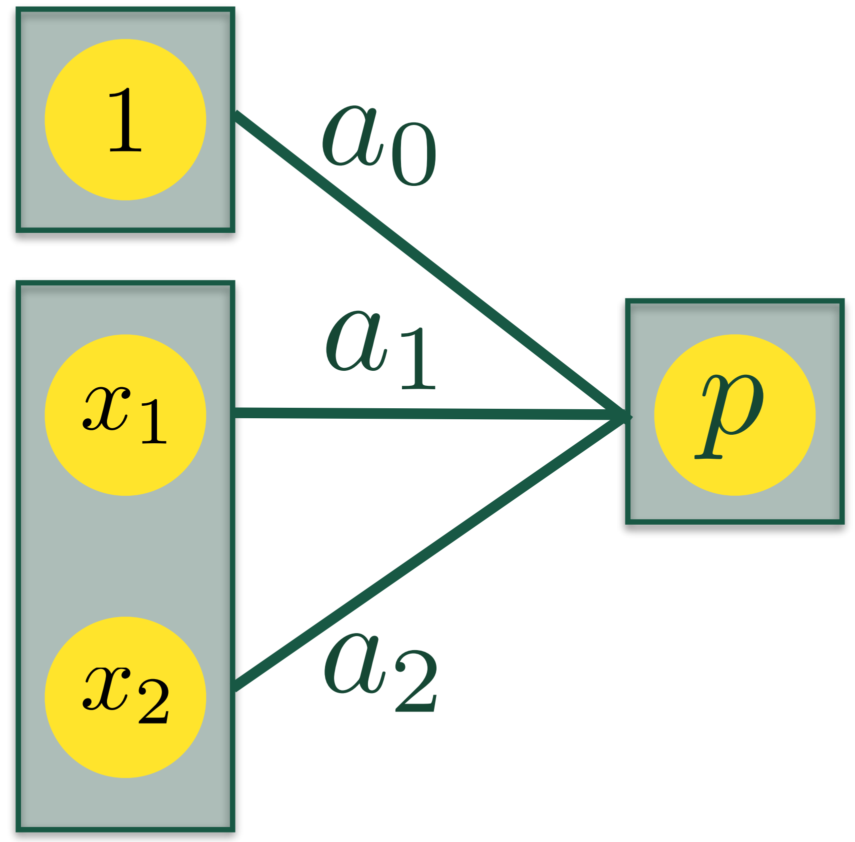


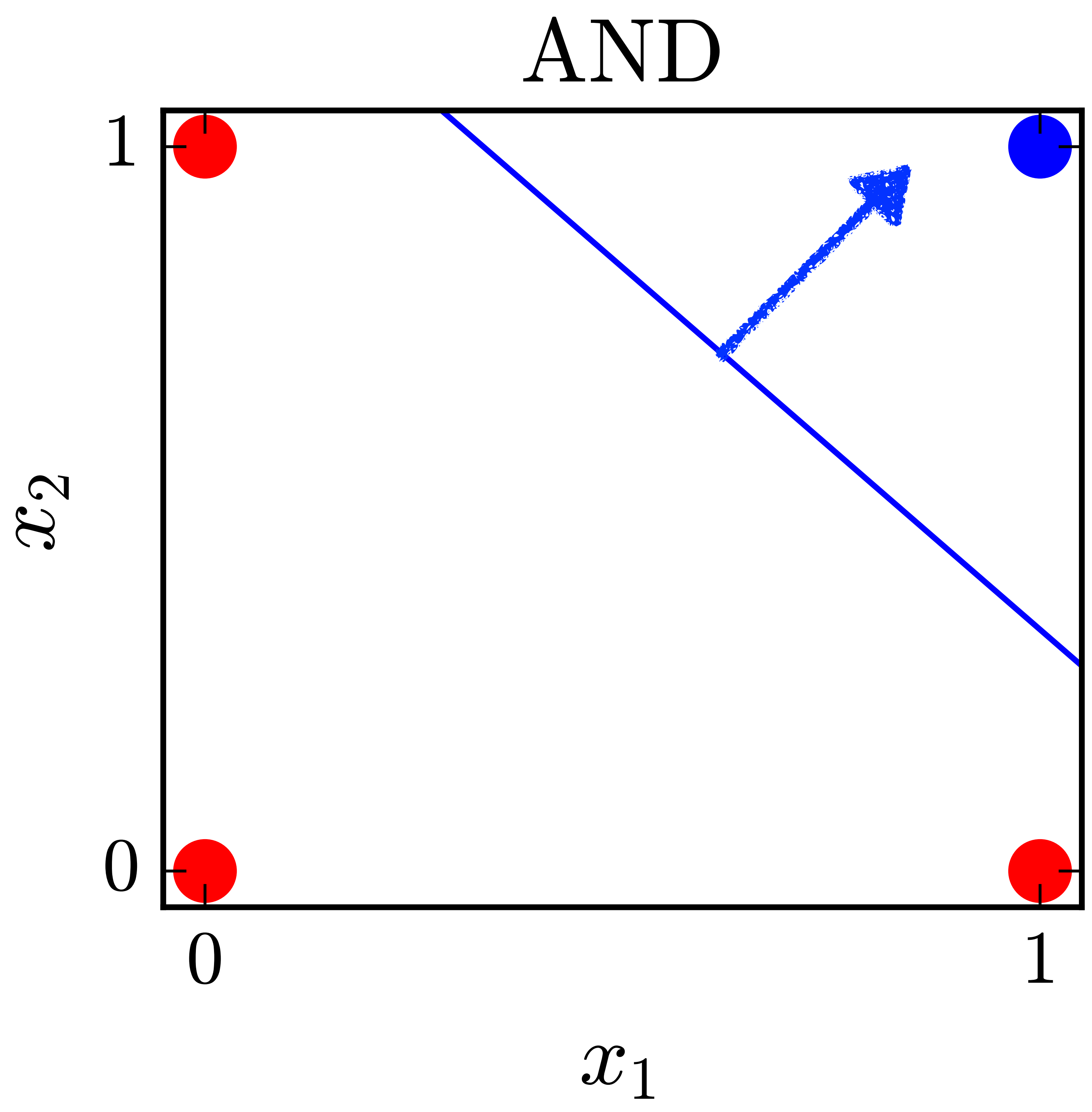
AND

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1



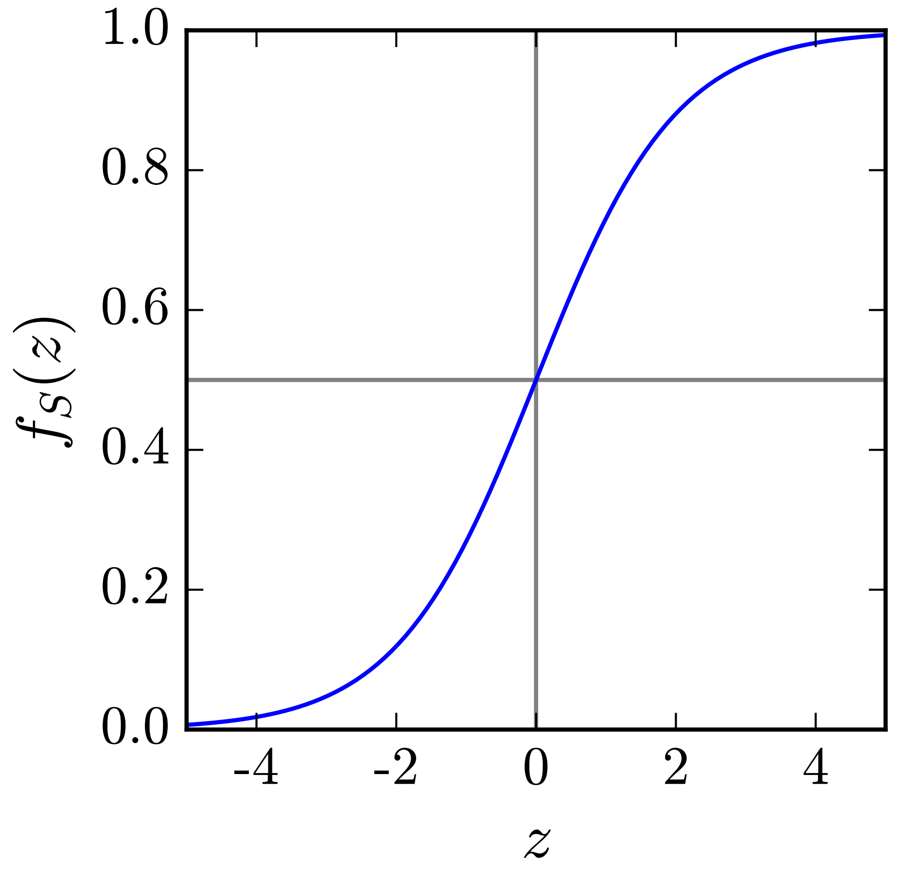
Neural Networks





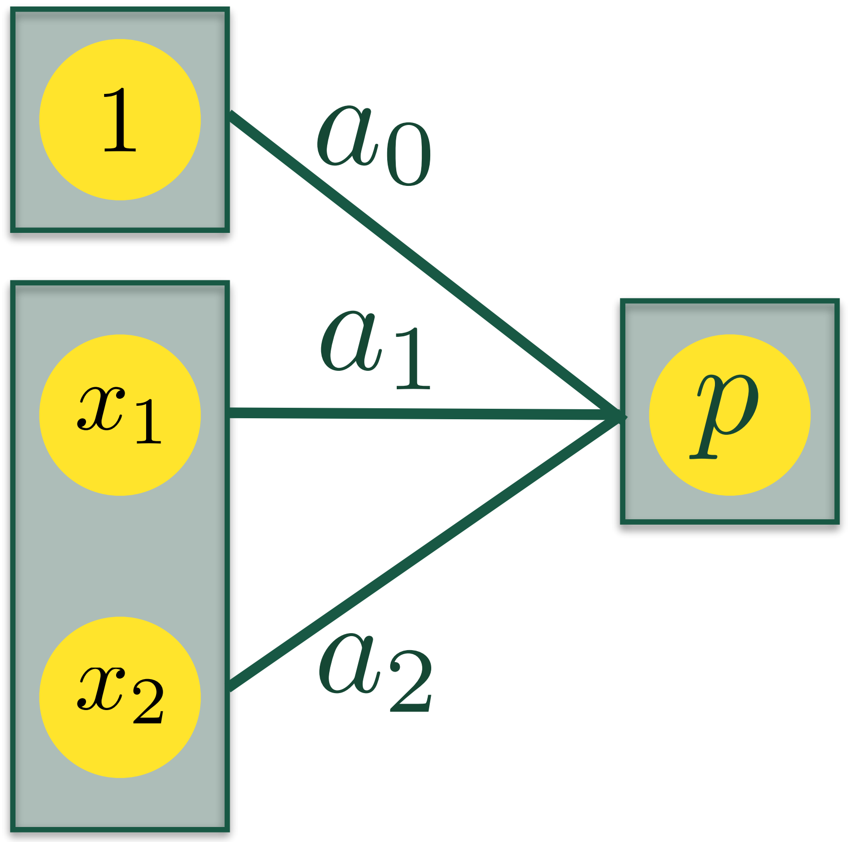
AND

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1



$$a_0 = -20, \quad a_1 = 15, \quad a_2 = 15$$

Neural Networks



OR

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

$$a_0 = -10, \quad a_1 = 15, \quad a_2 = 15$$

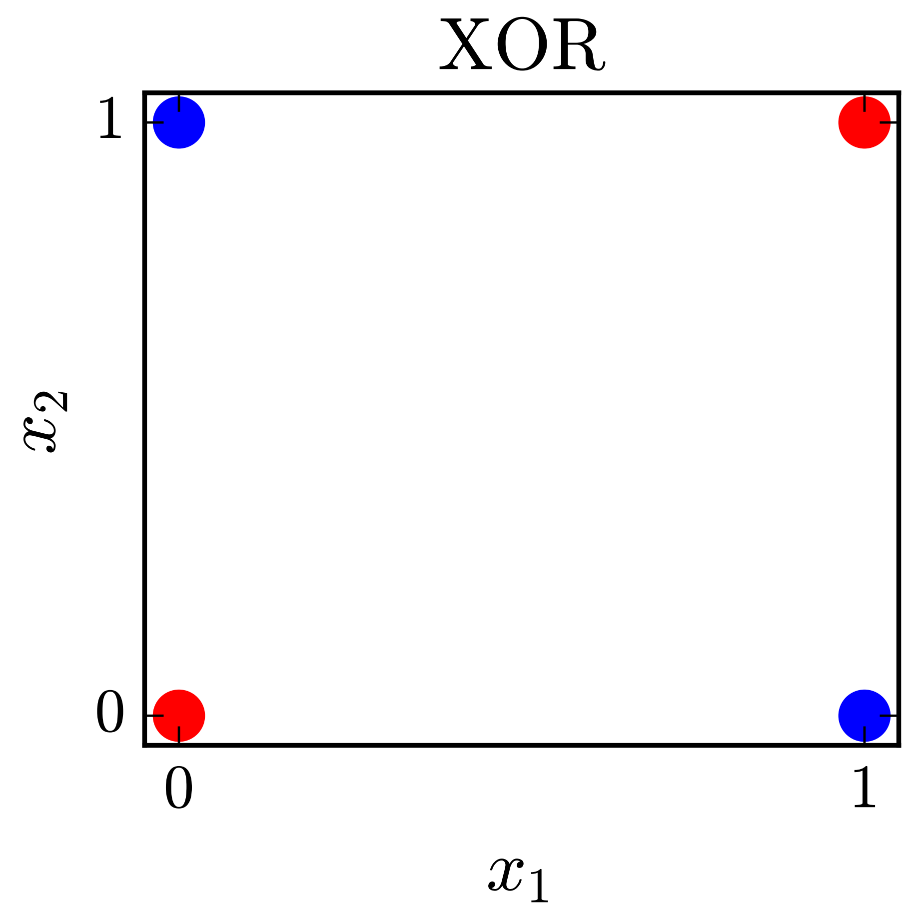
AND

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

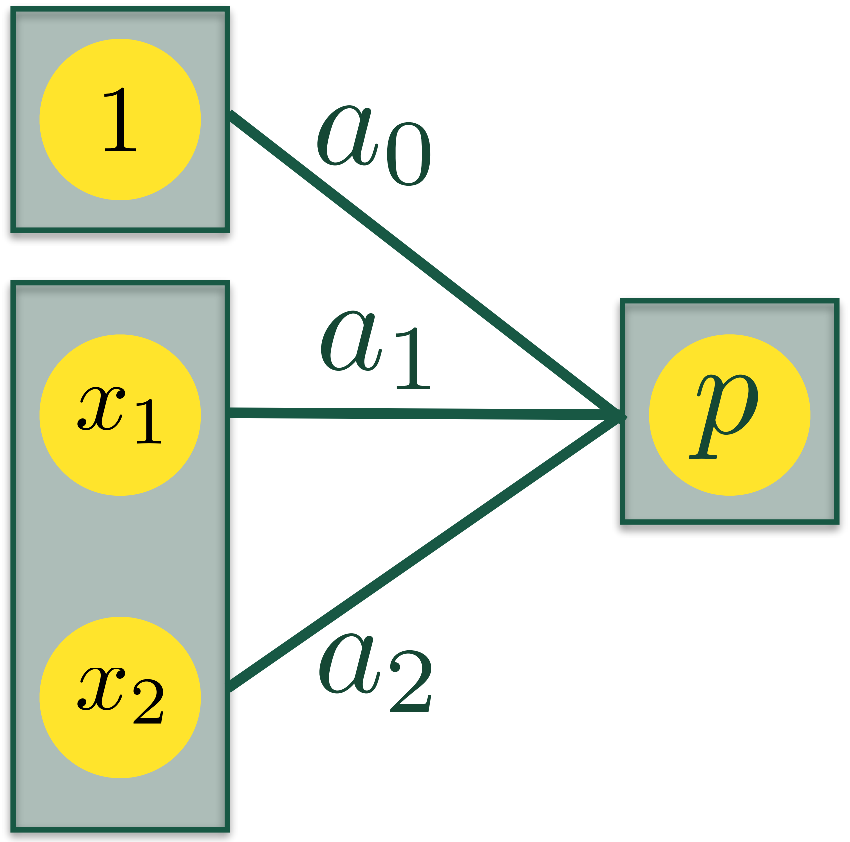
$$a_0 = -20, \quad a_1 = 15, \quad a_2 = 15$$

XOR

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0



Neural Networks



OR

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

$$a_0 = -10, \quad a_1 = 15, \quad a_2 = 15$$

AND

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

$$a_0 = -20, \quad a_1 = 15, \quad a_2 = 15$$

XOR

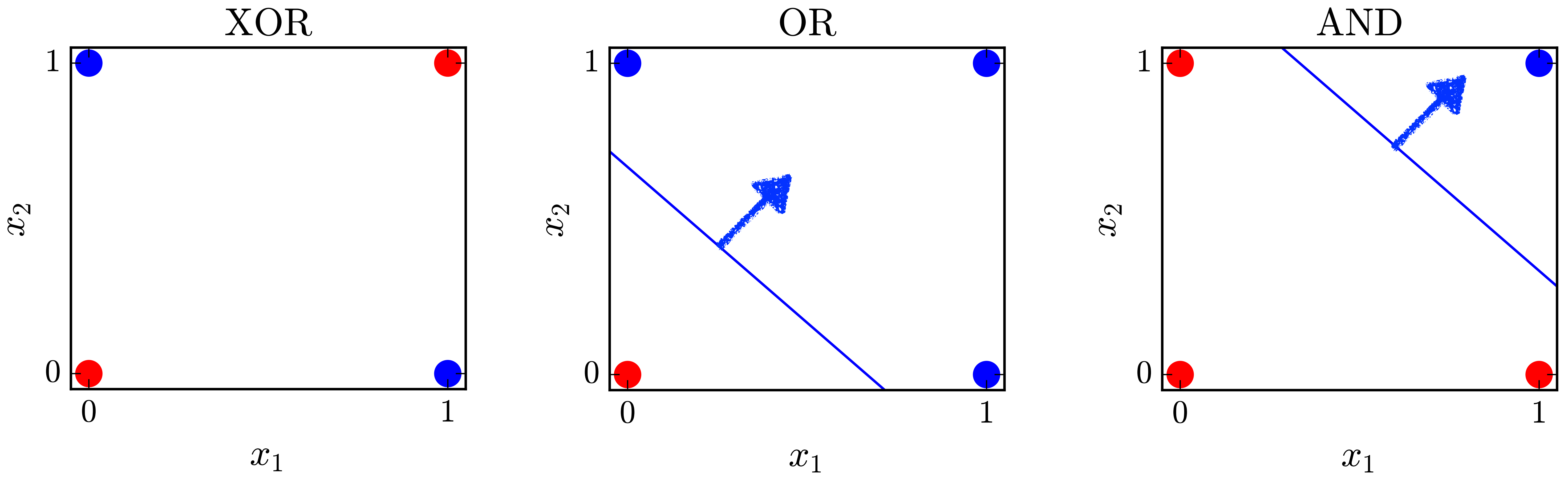
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

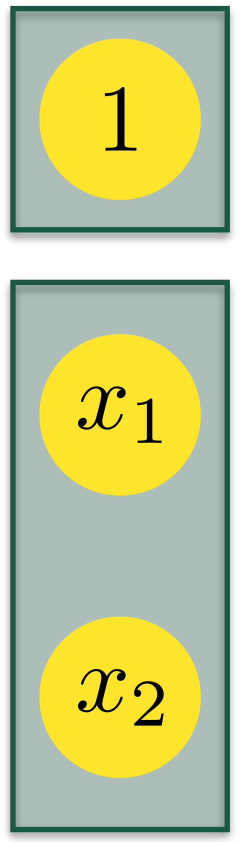
This system cannot produce XOR

(cannot make a two sided cut)

Neural Networks

XOR		
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

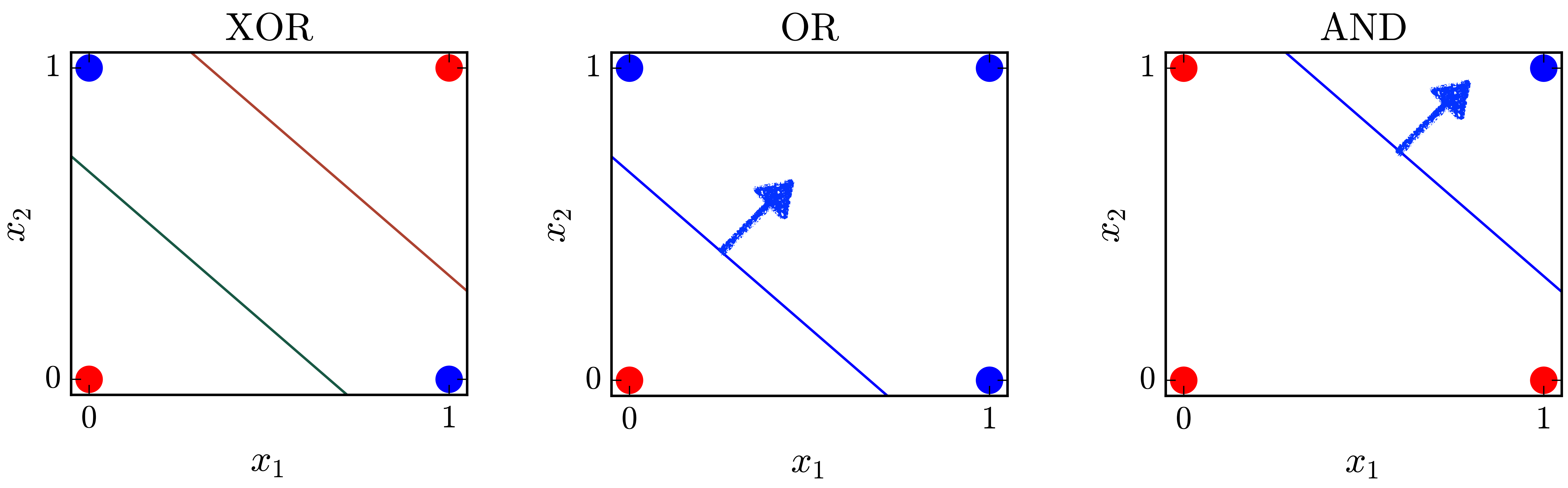


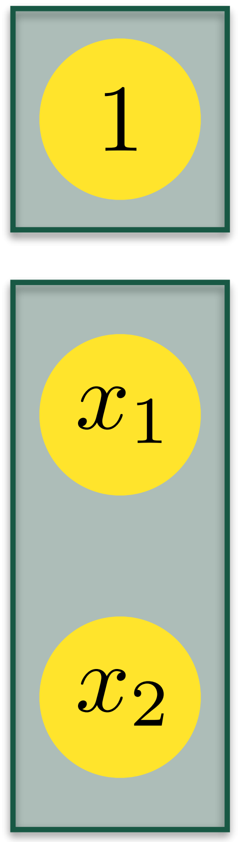


Neural Networks

XOR

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

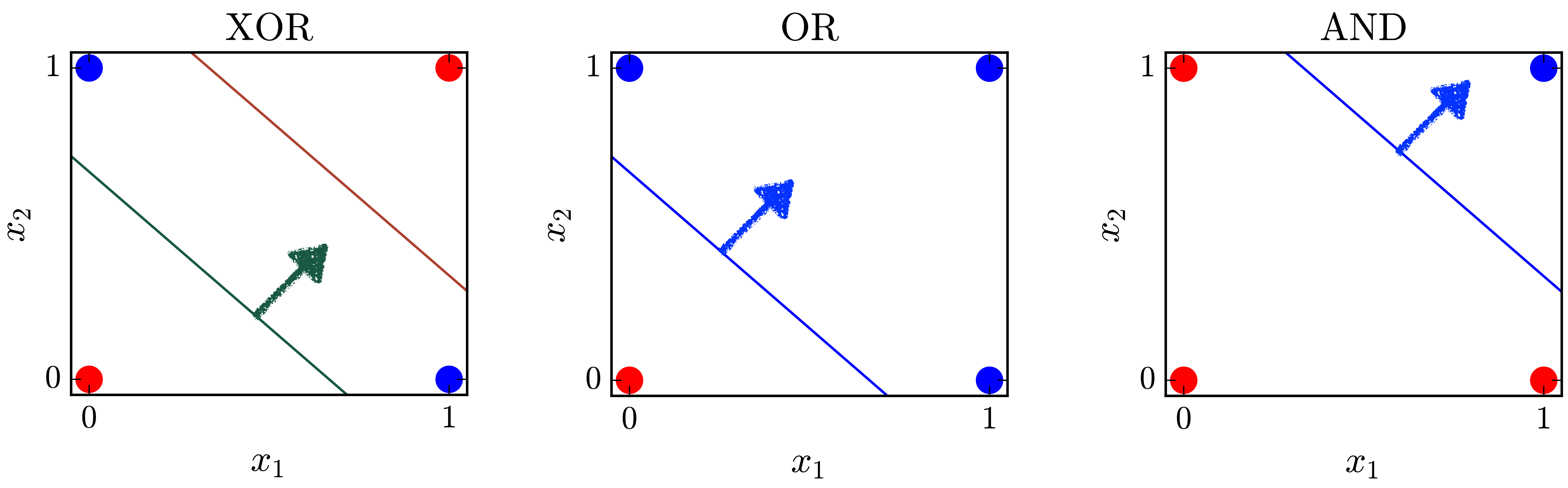


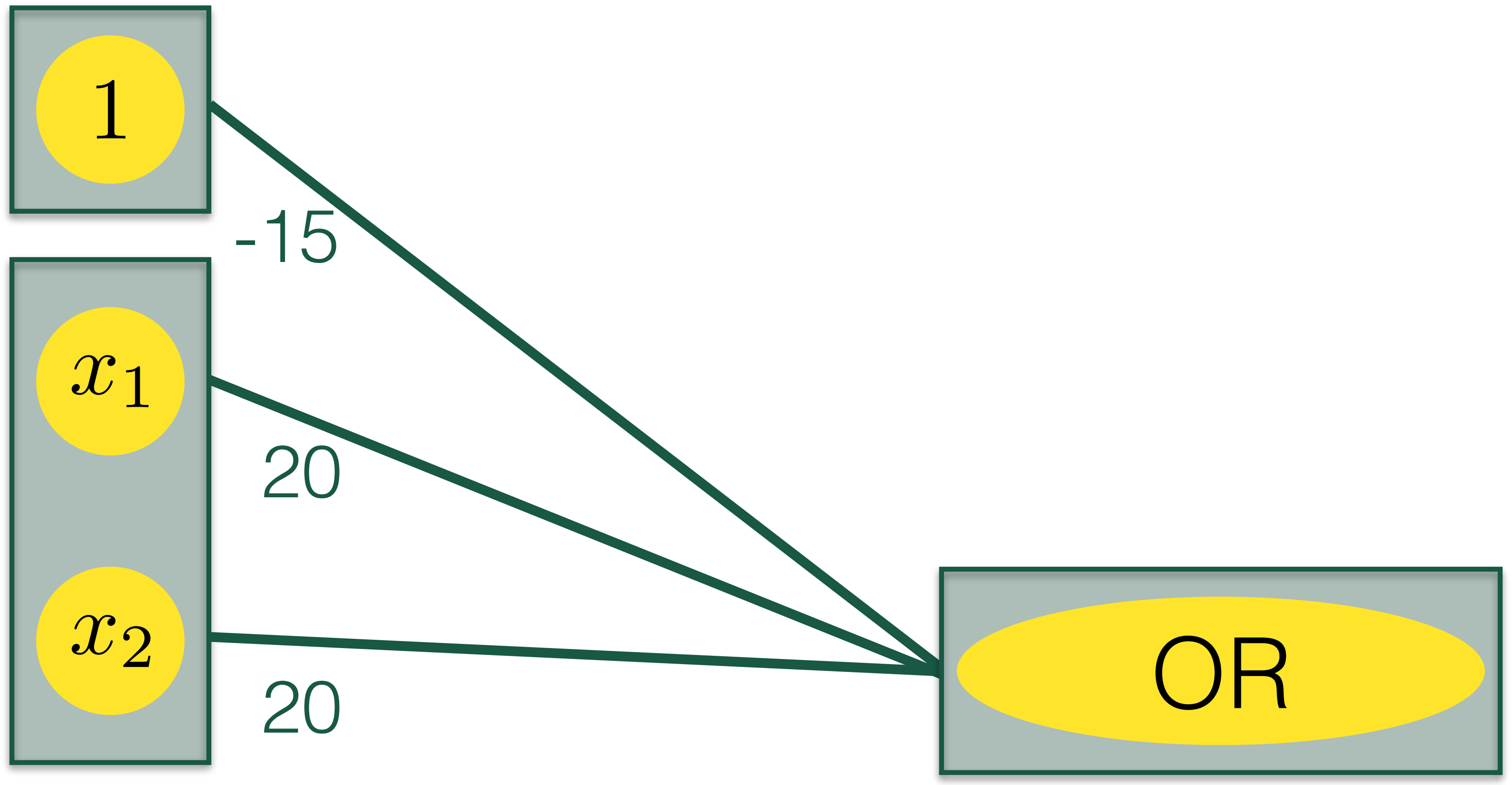


Neural Networks

XOR

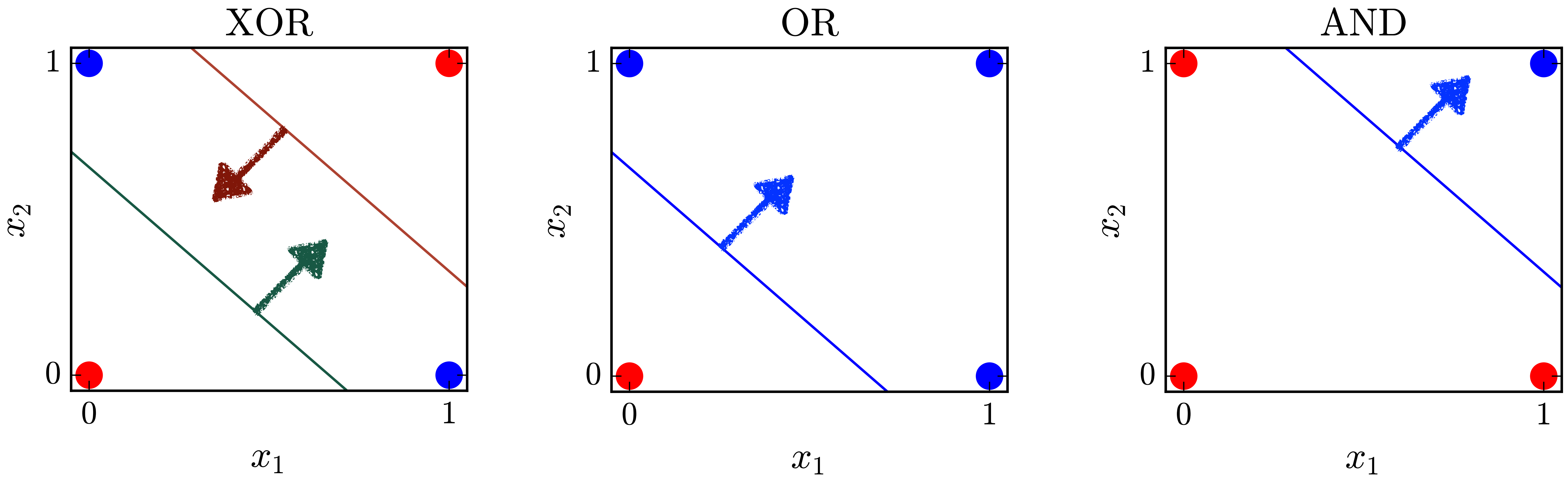
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

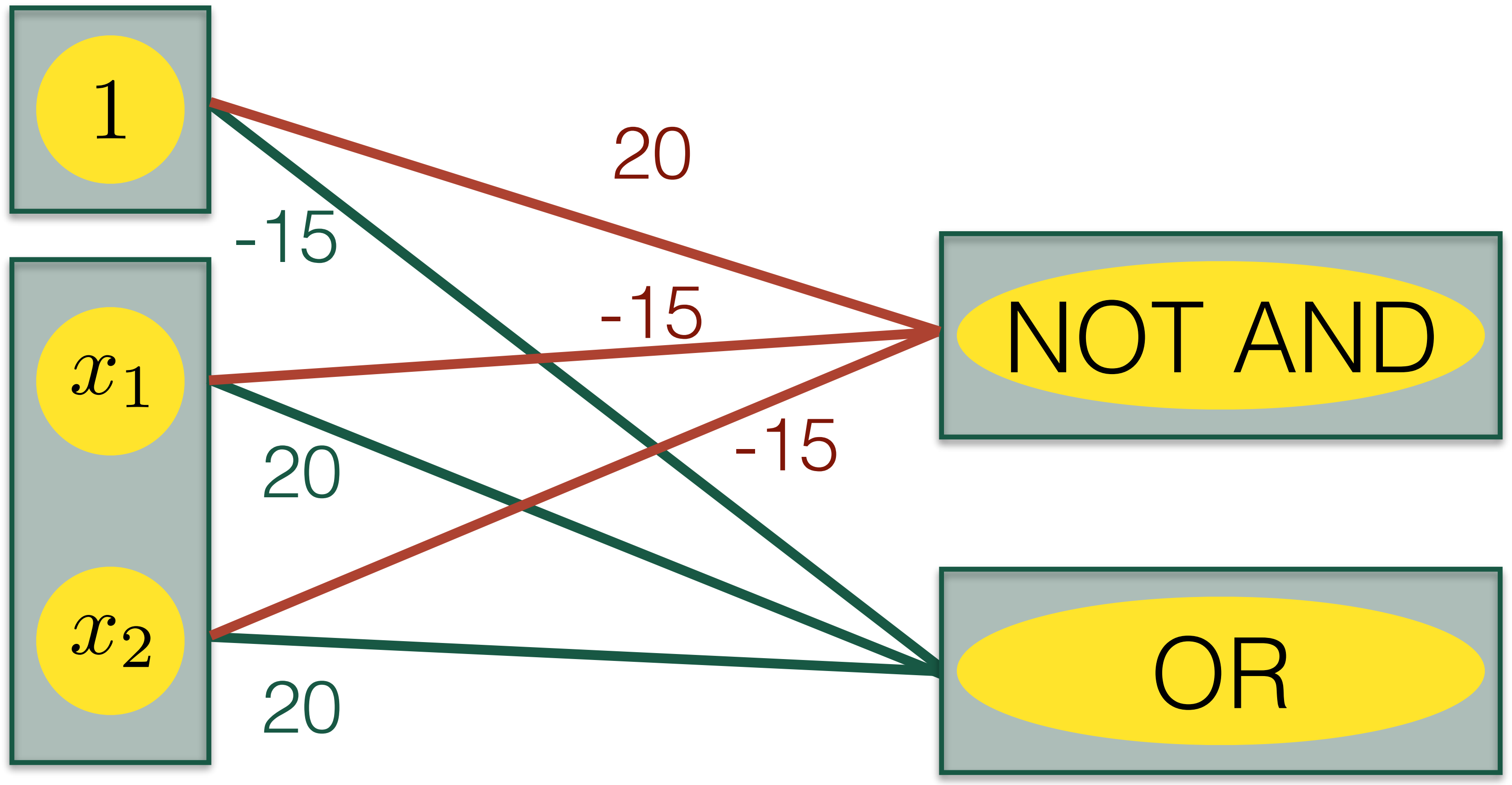




Neural Networks

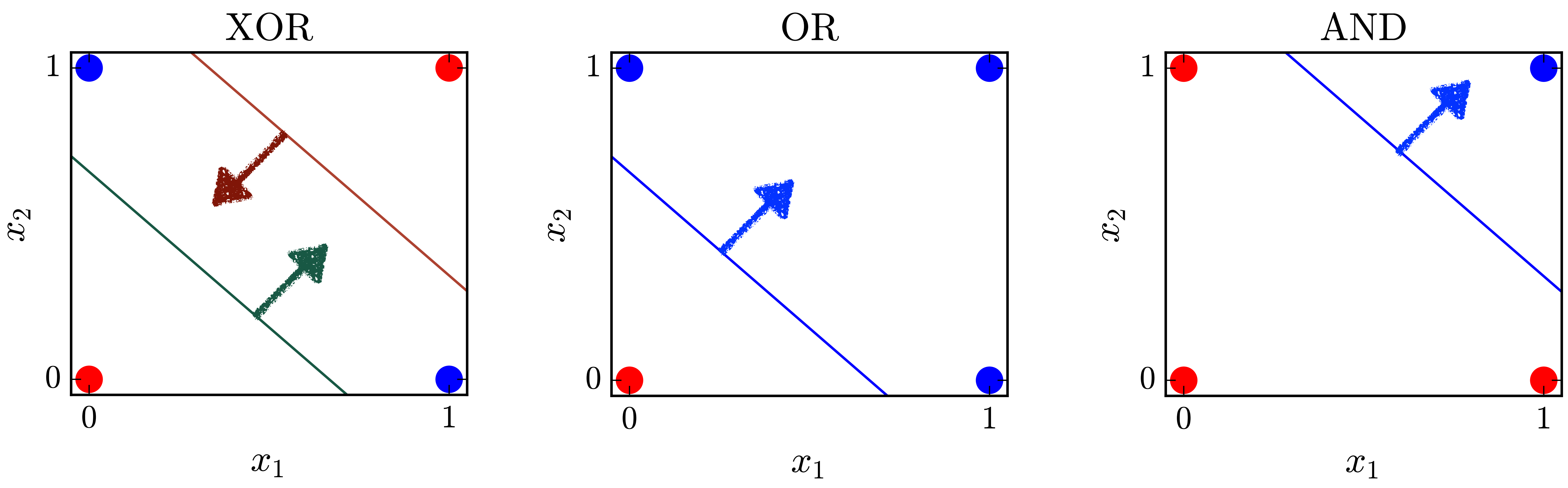
XOR		
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

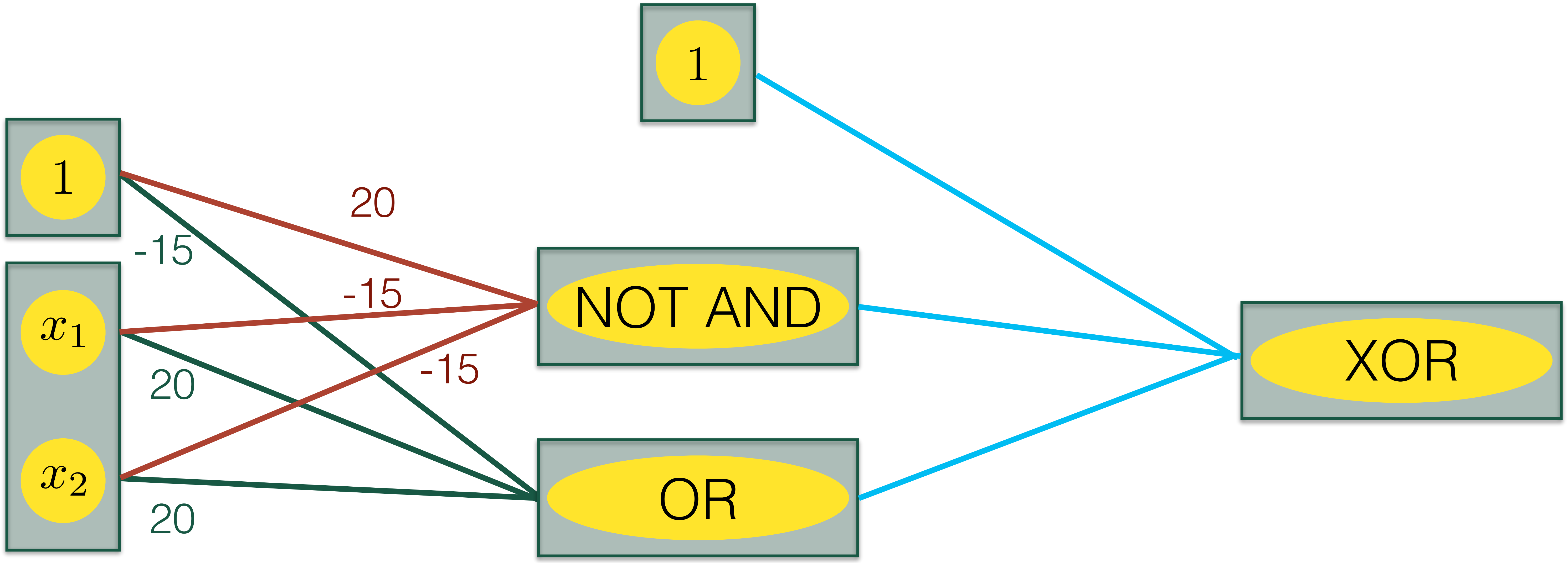




Neural Networks

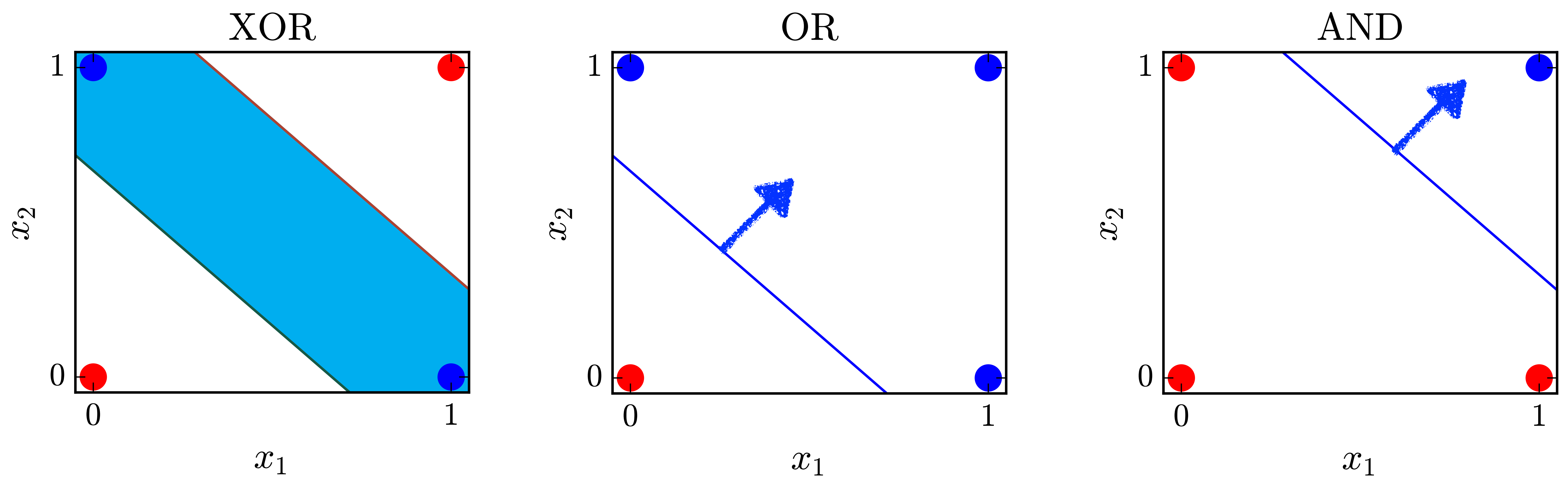
XOR		
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

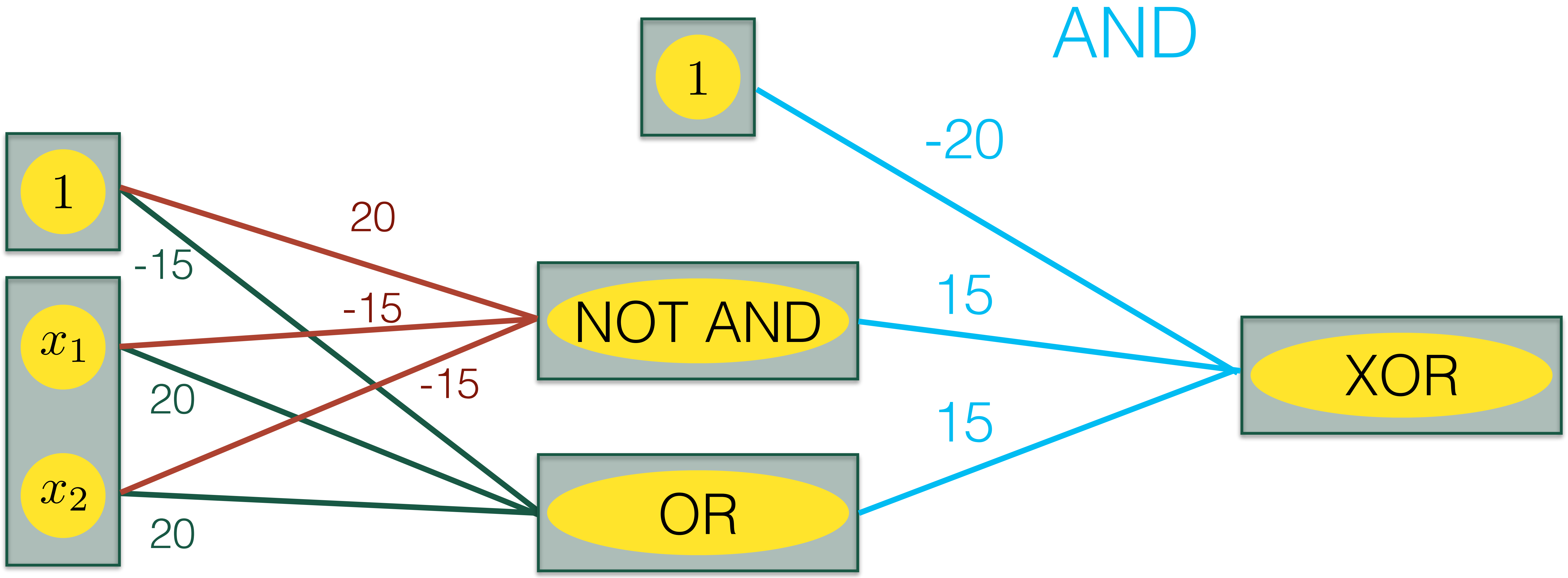




Neural Networks

XOR		
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

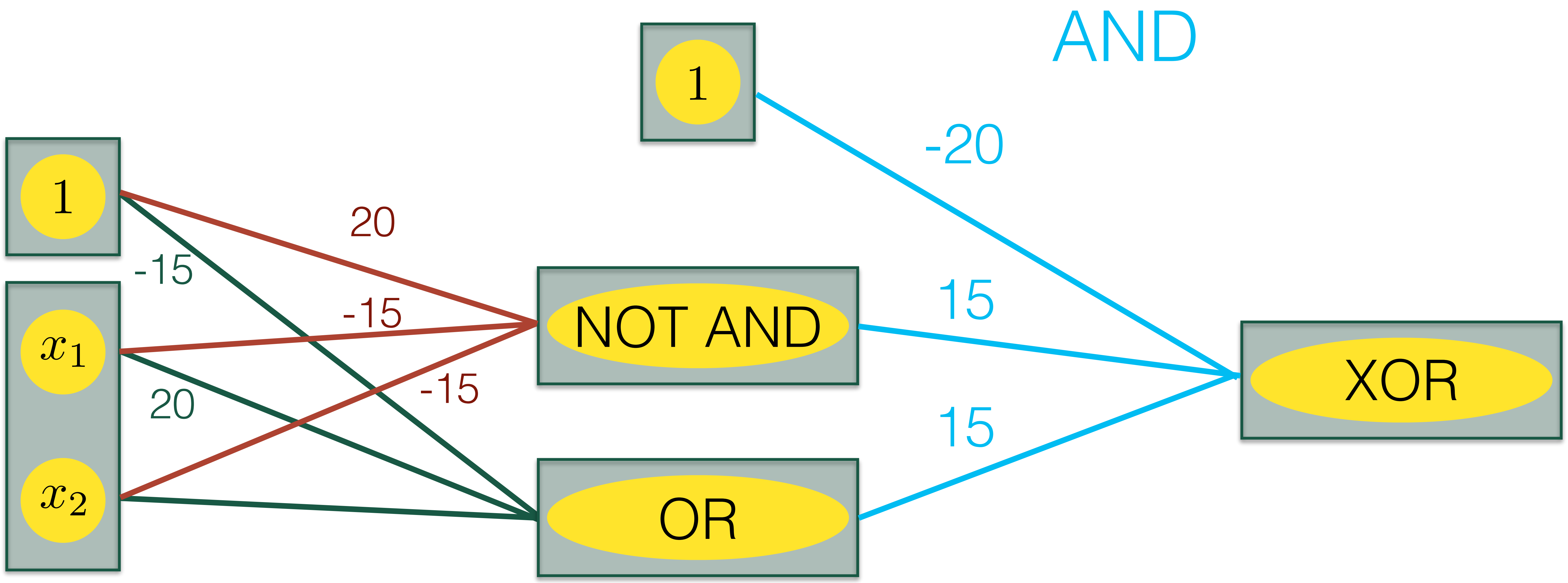




Neural Networks

XOR

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

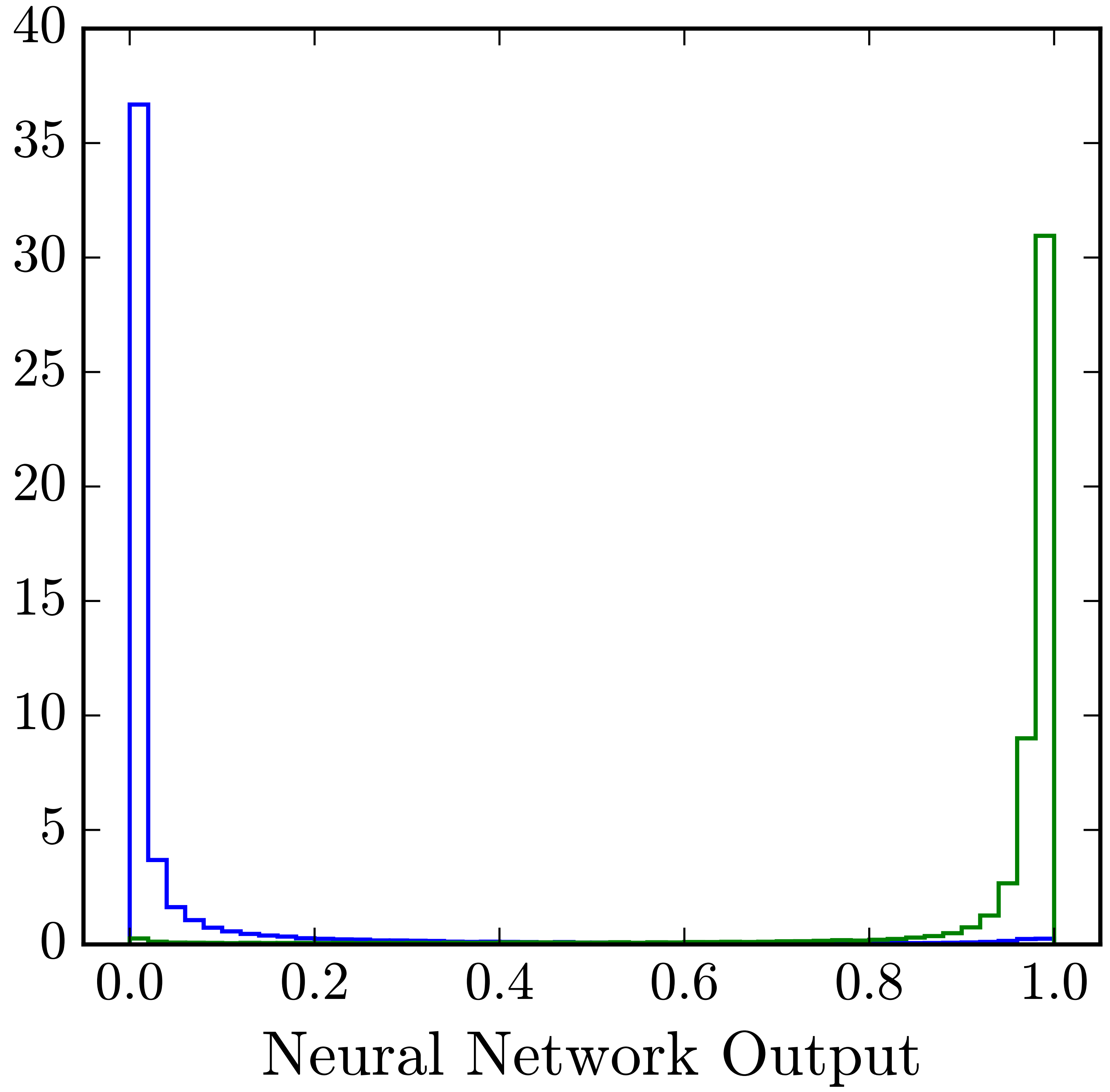


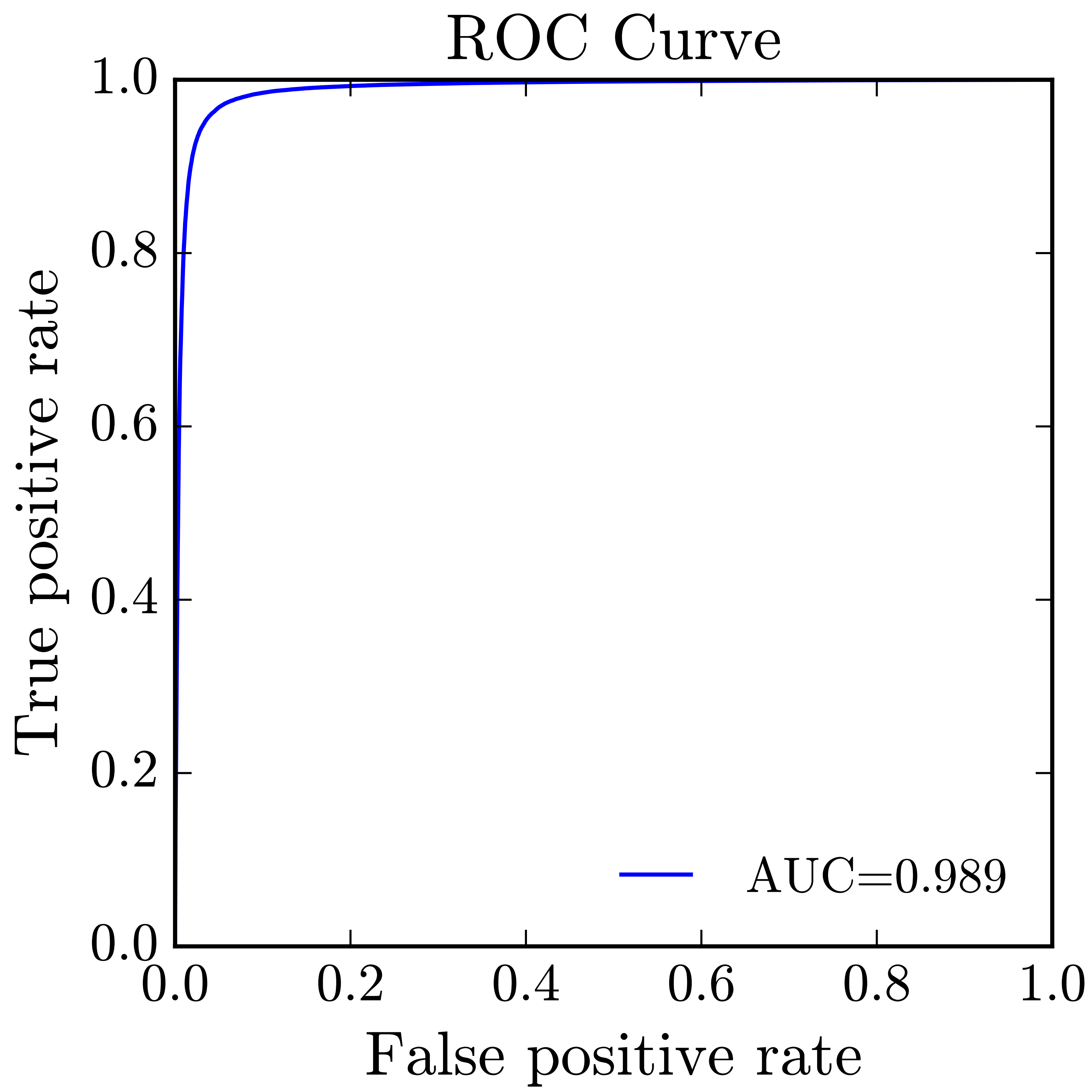
Simple example showing that neural network can access ‘high-level’ functions

To learn weights, need LARGE training set and CPU time

Measuring Classifiers

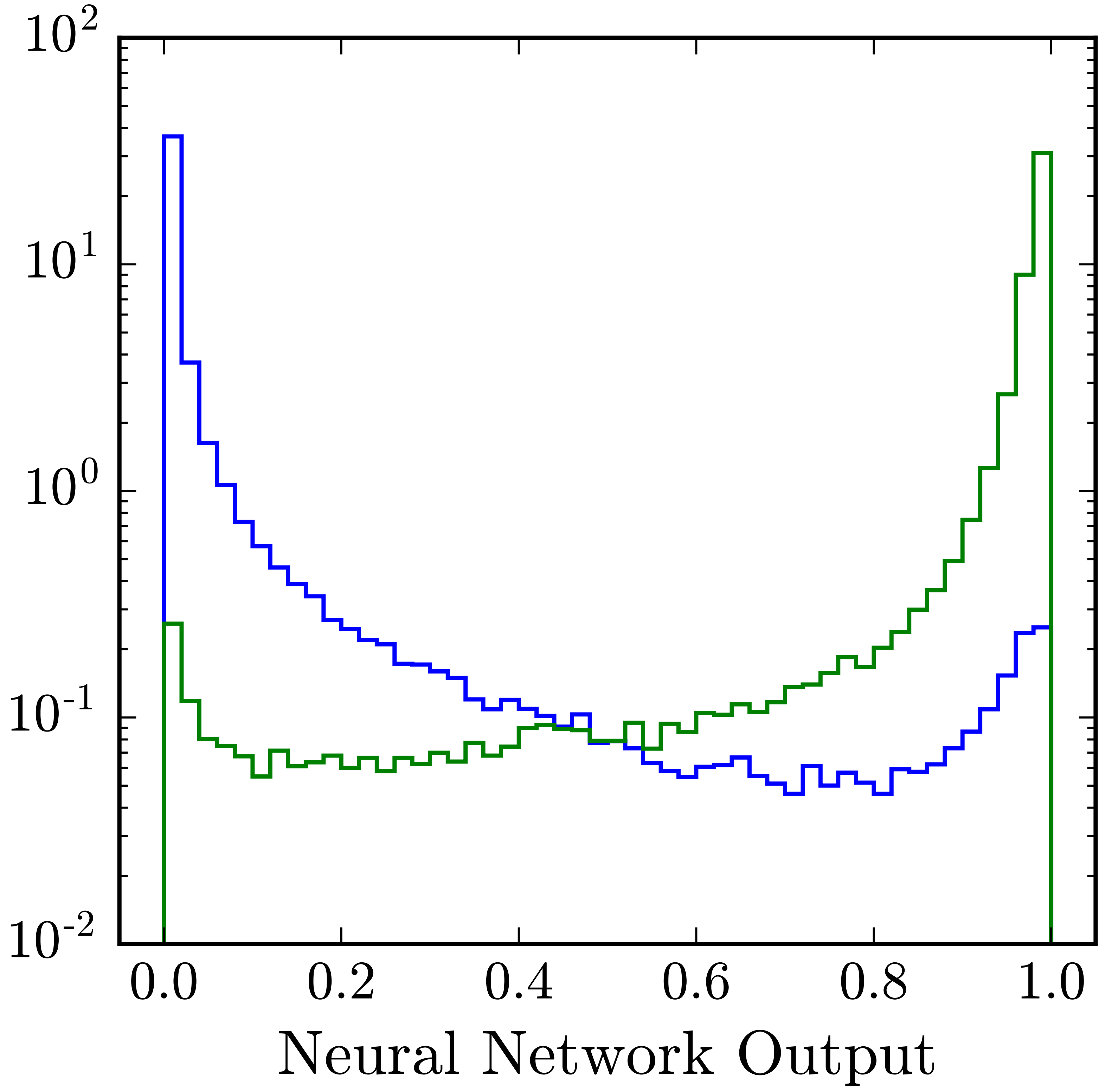
How good is the classifier?

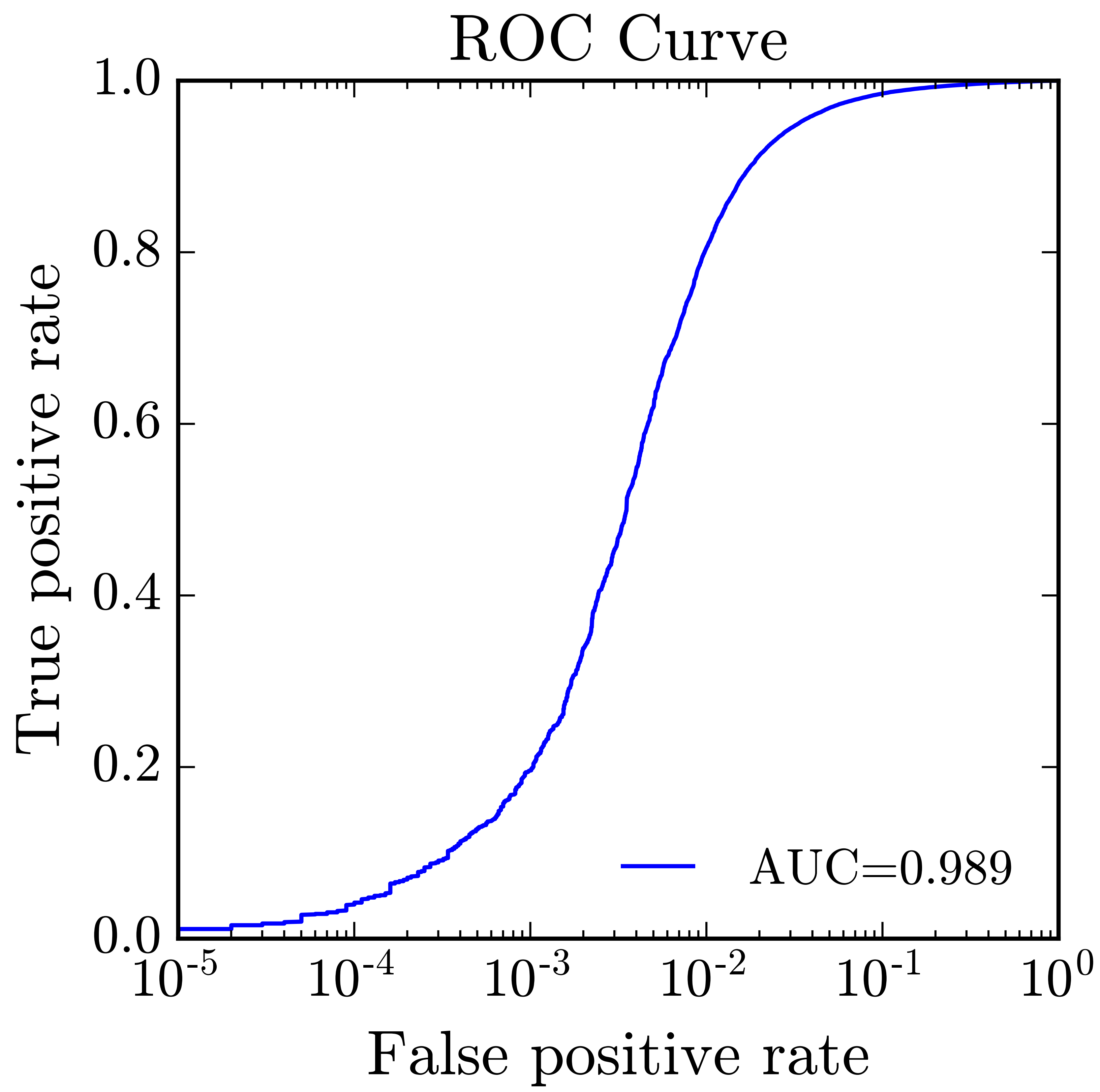




Measuring Classifiers

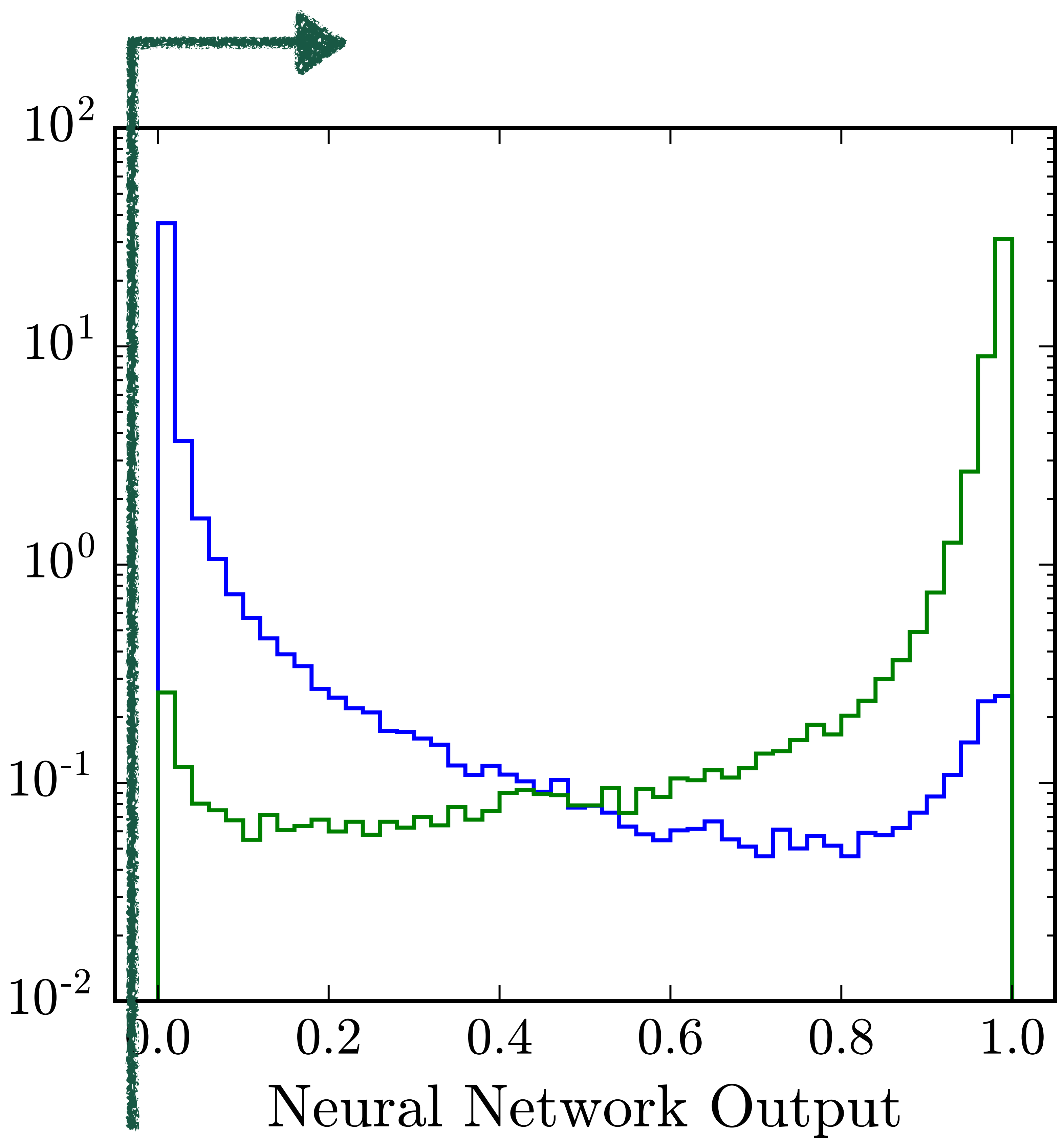
How good is the classifier?

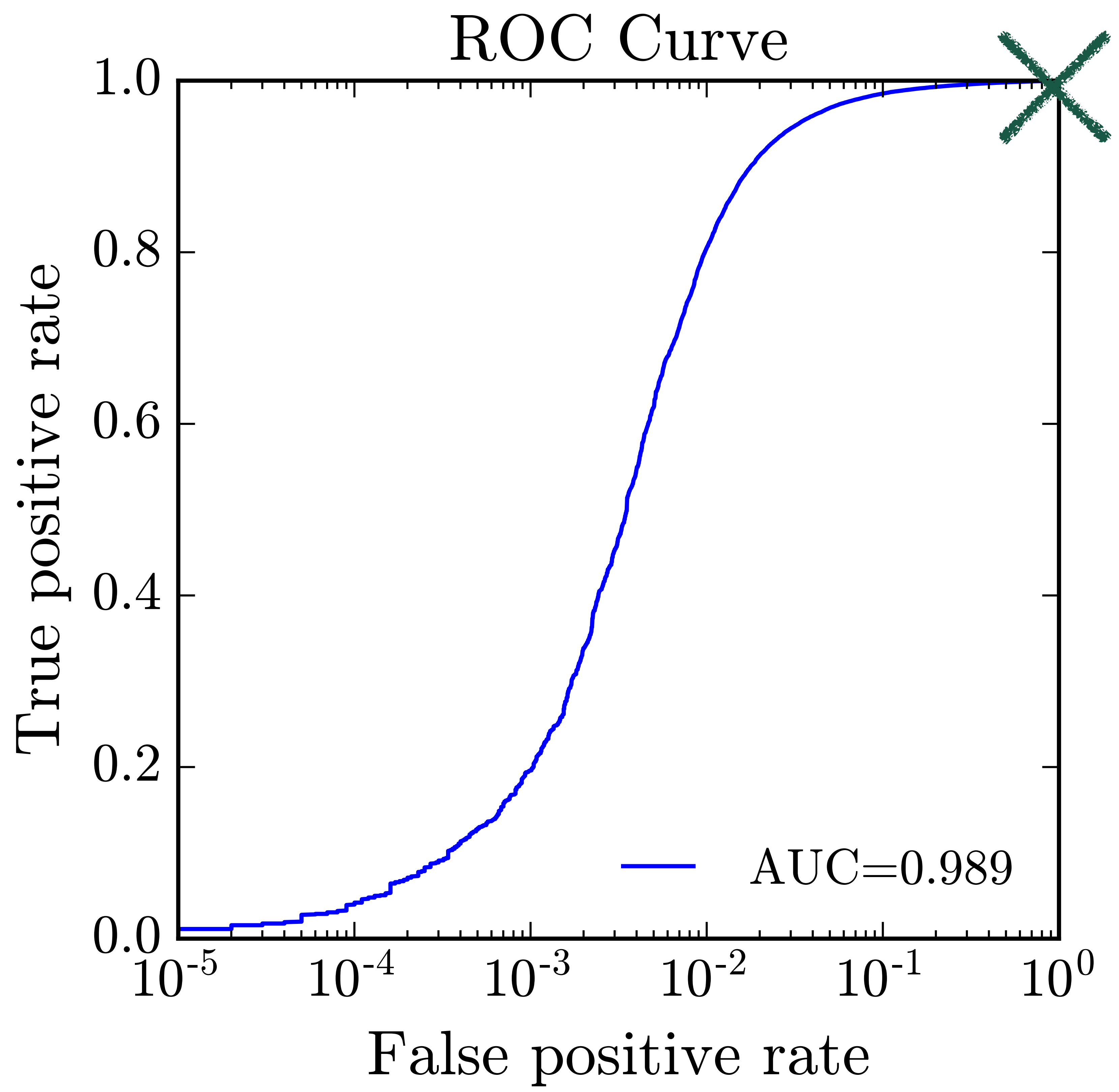




Measuring Classifiers

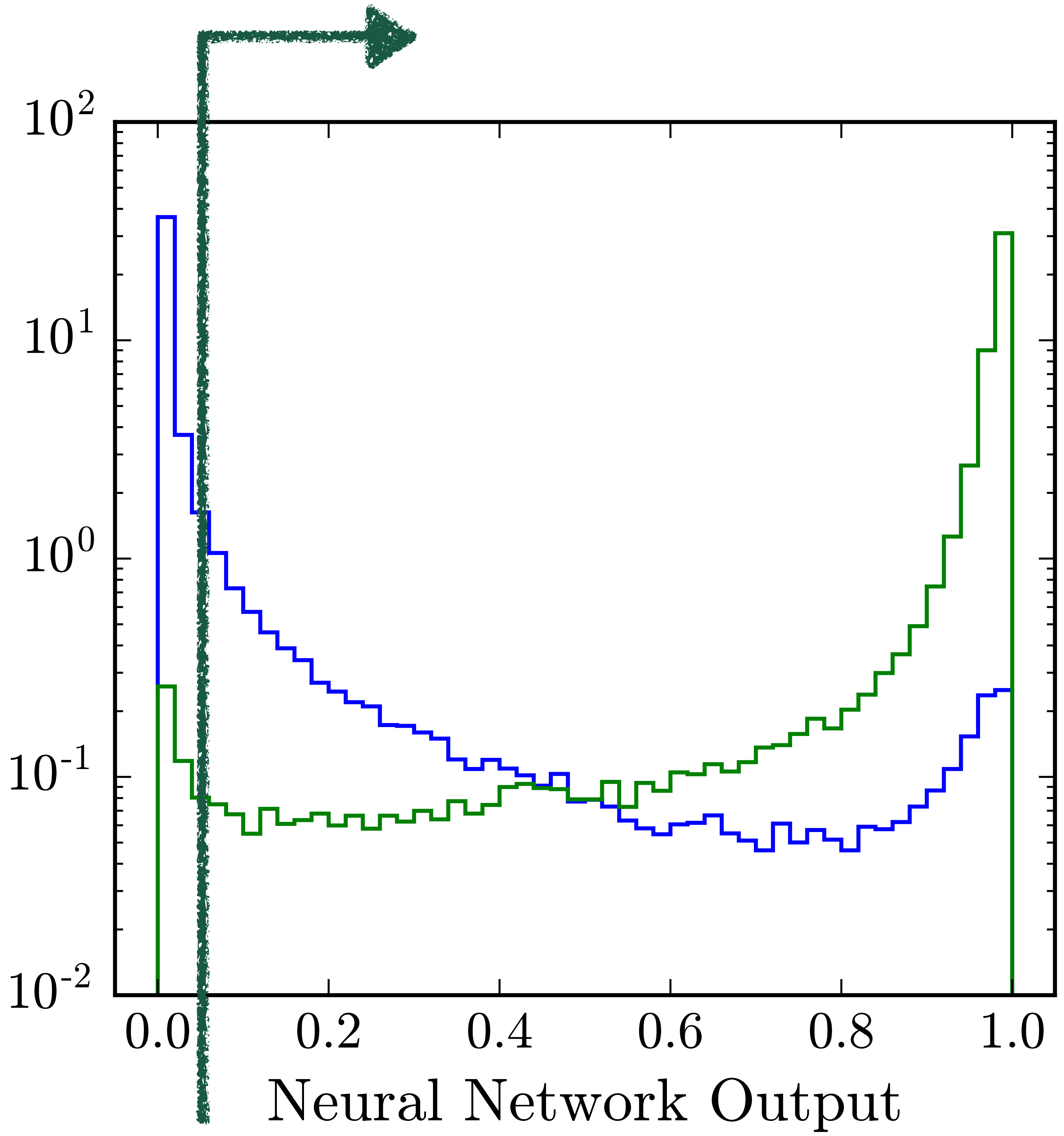
How good is the classifier?

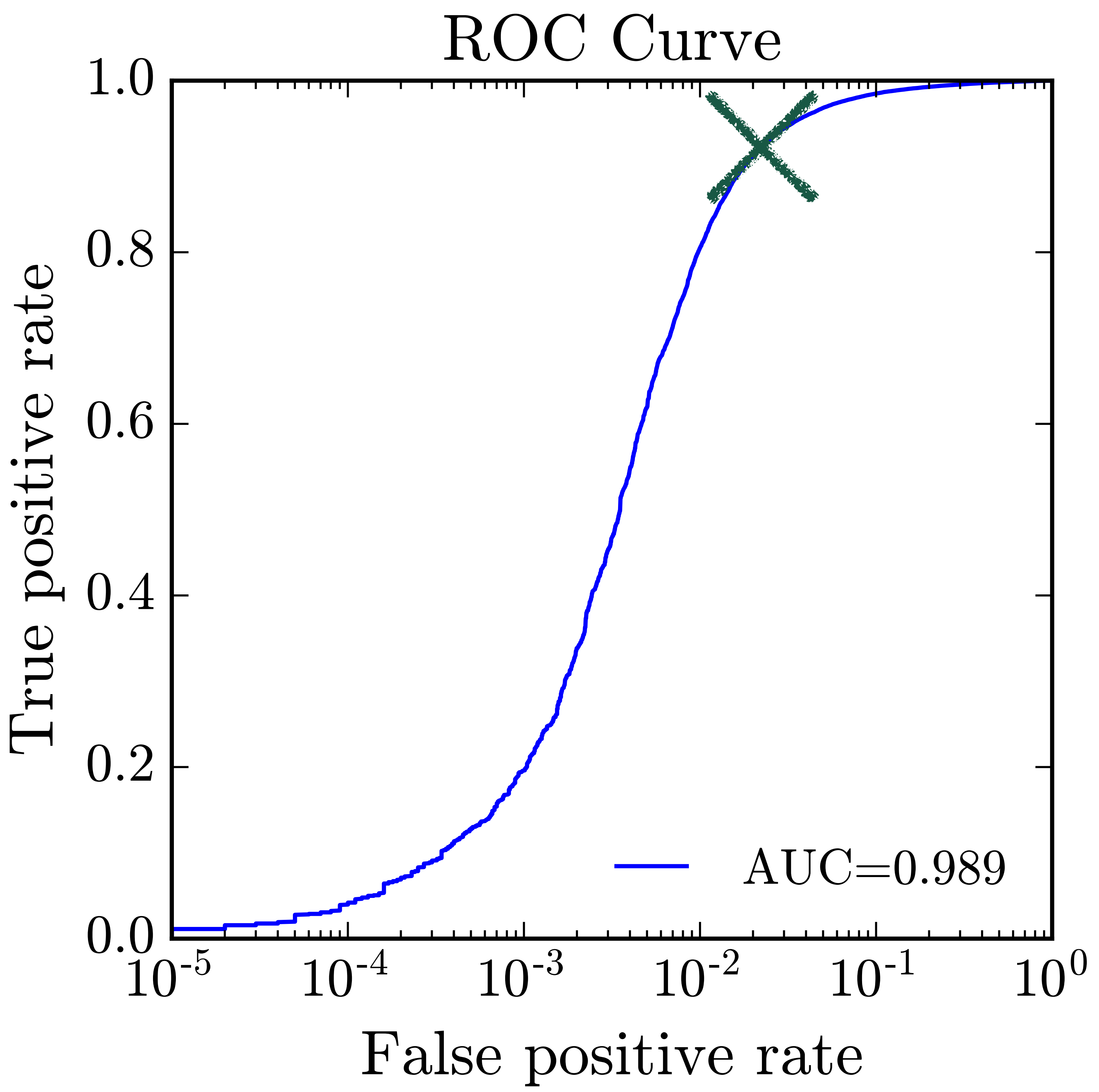




Measuring Classifiers

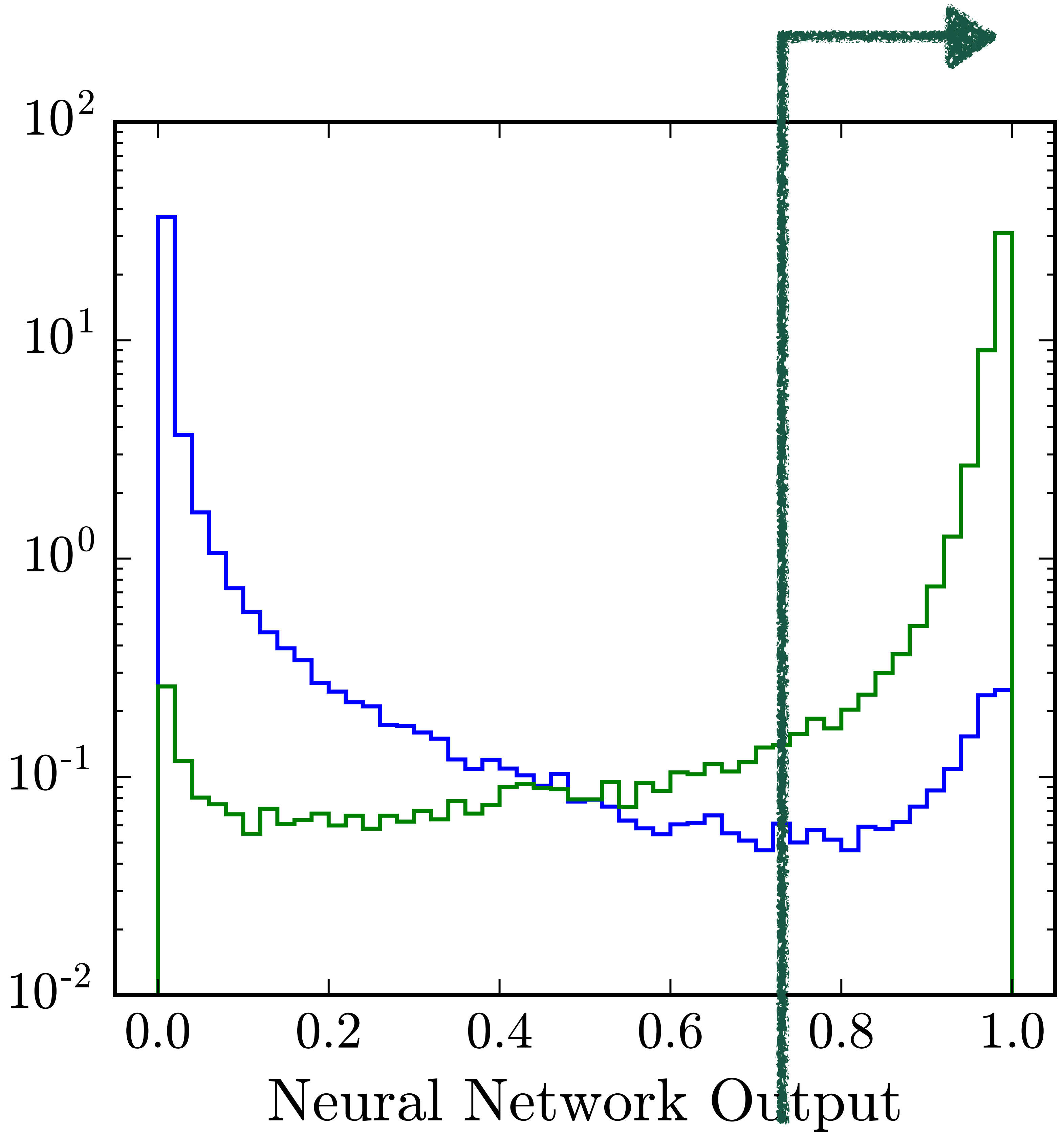
How good is the classifier?

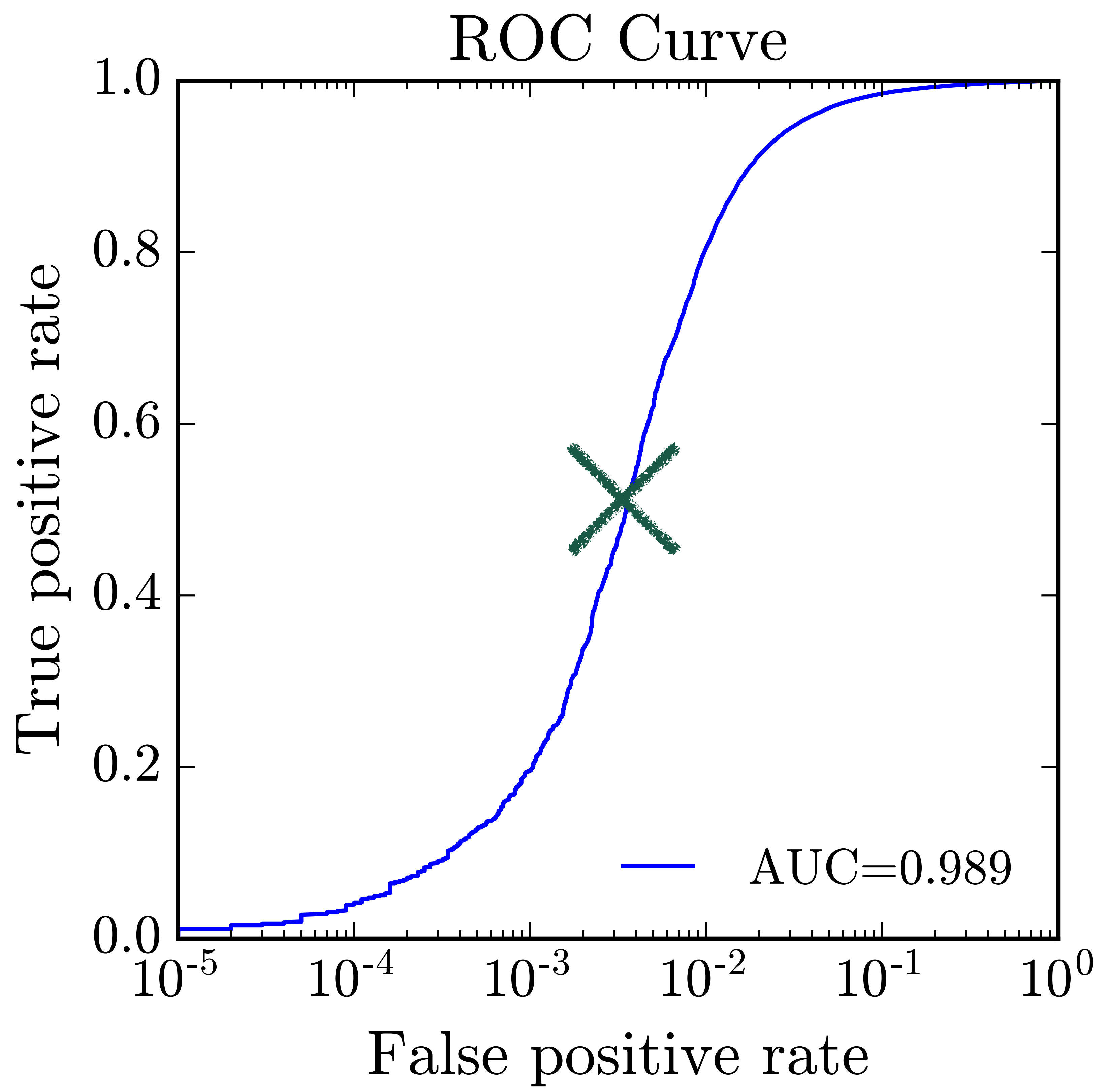




Measuring Classifiers

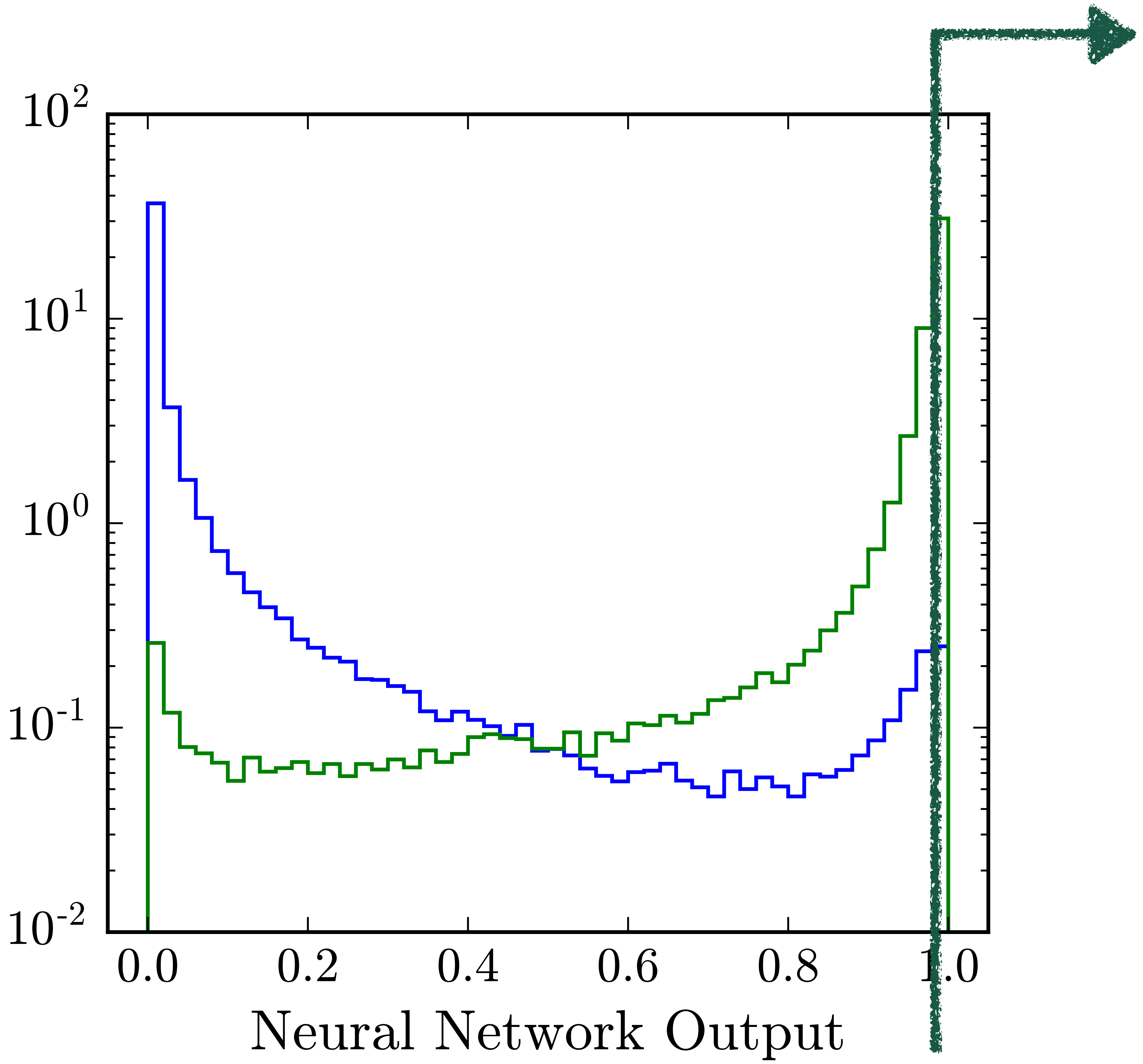
How good is the classifier?

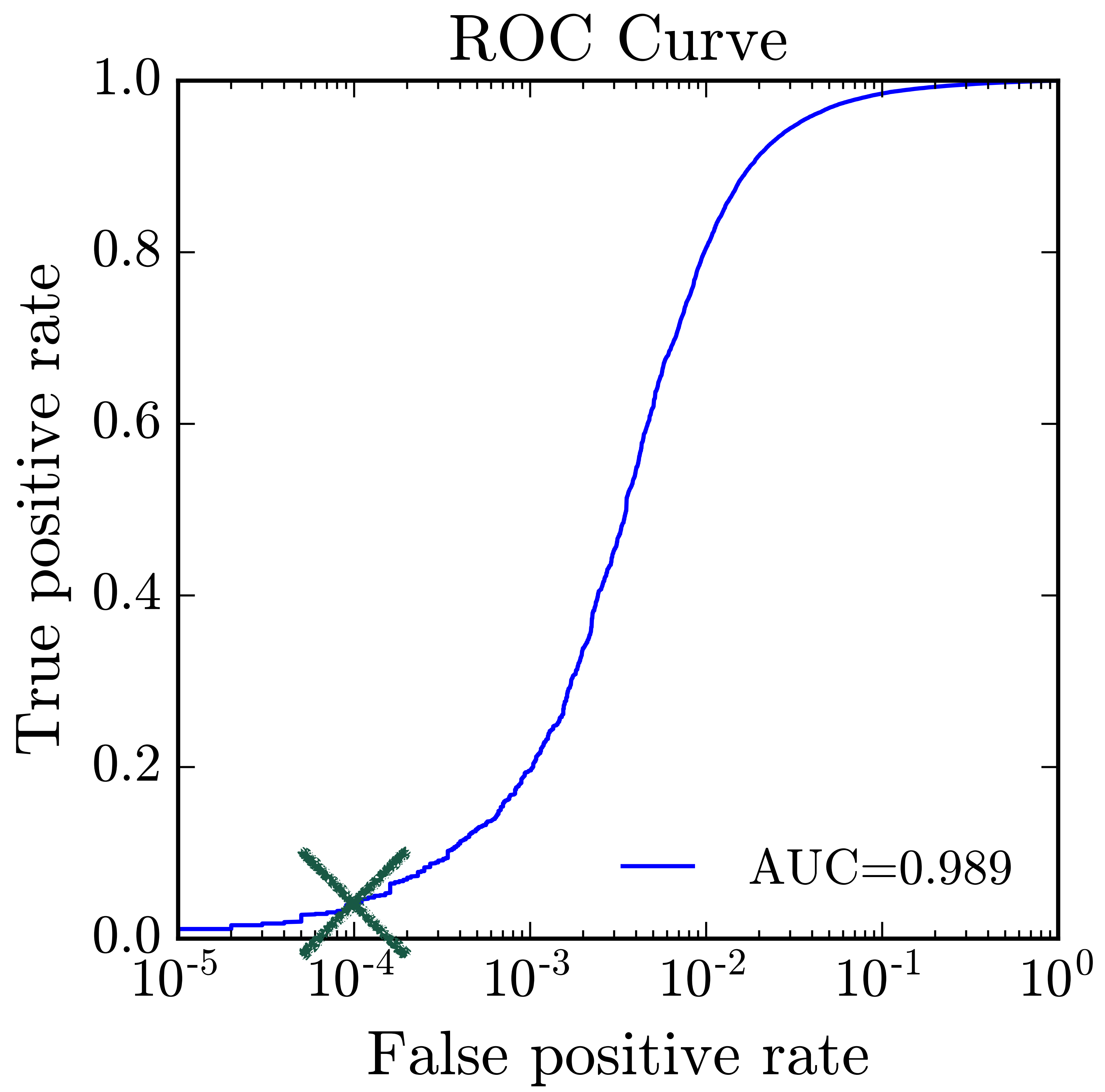




Measuring Classifiers

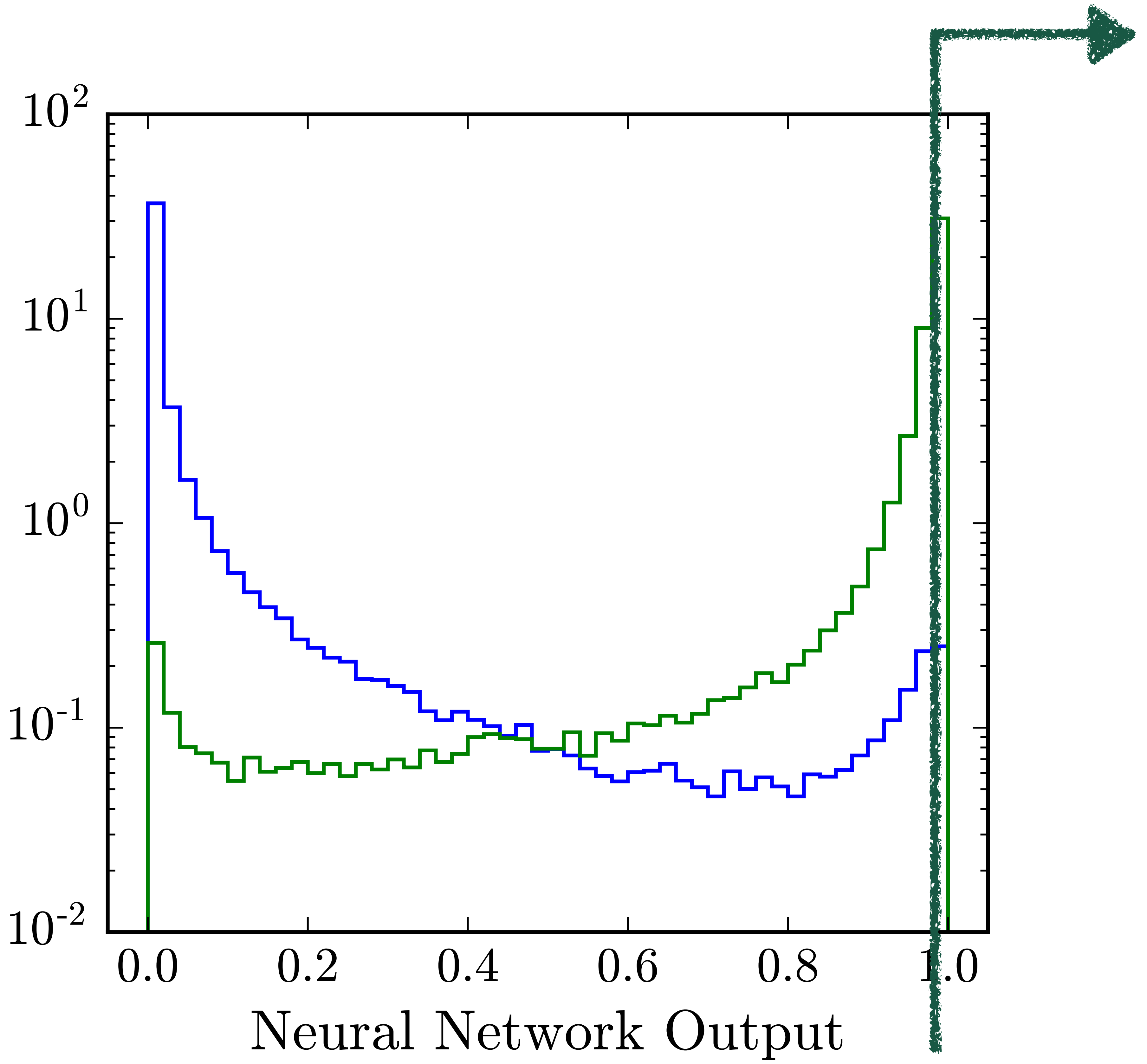
How good is the classifier?

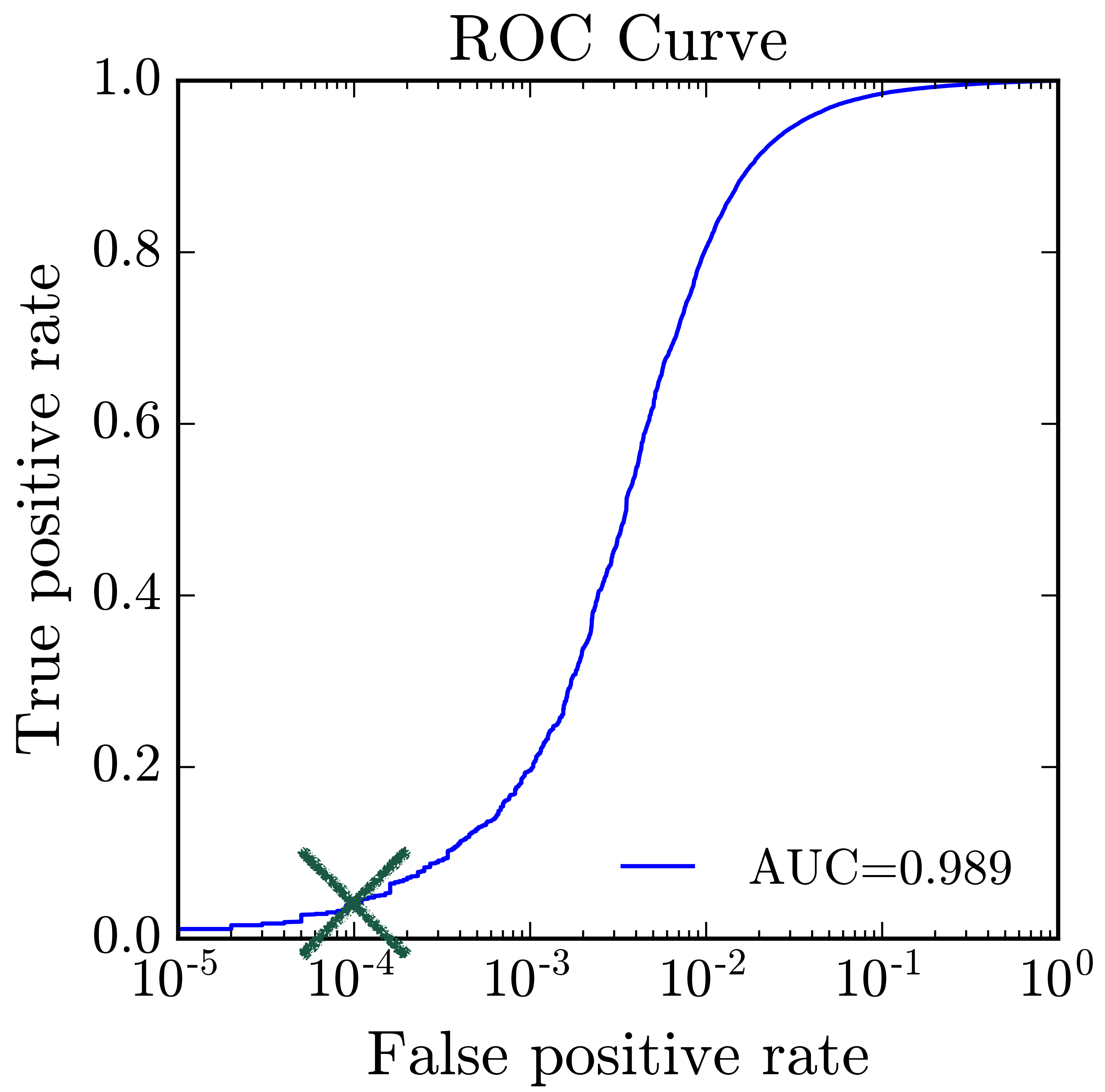




Measuring Classifiers

How good is the classifier?





AUC = 1.0 is perfect, this is not attainable for most problems

What is machine learning (for)?

Getting information from data

Labeled data

Supervised Learning

- Classification
- Numerical Predictions
- etc

Unlabeled data

Unsupervised Learning

- Clustering
- Anomaly Detection
- Generative adversarial networks
- etc

Hybrid?

- Learning from label proportions
- Classification without labels