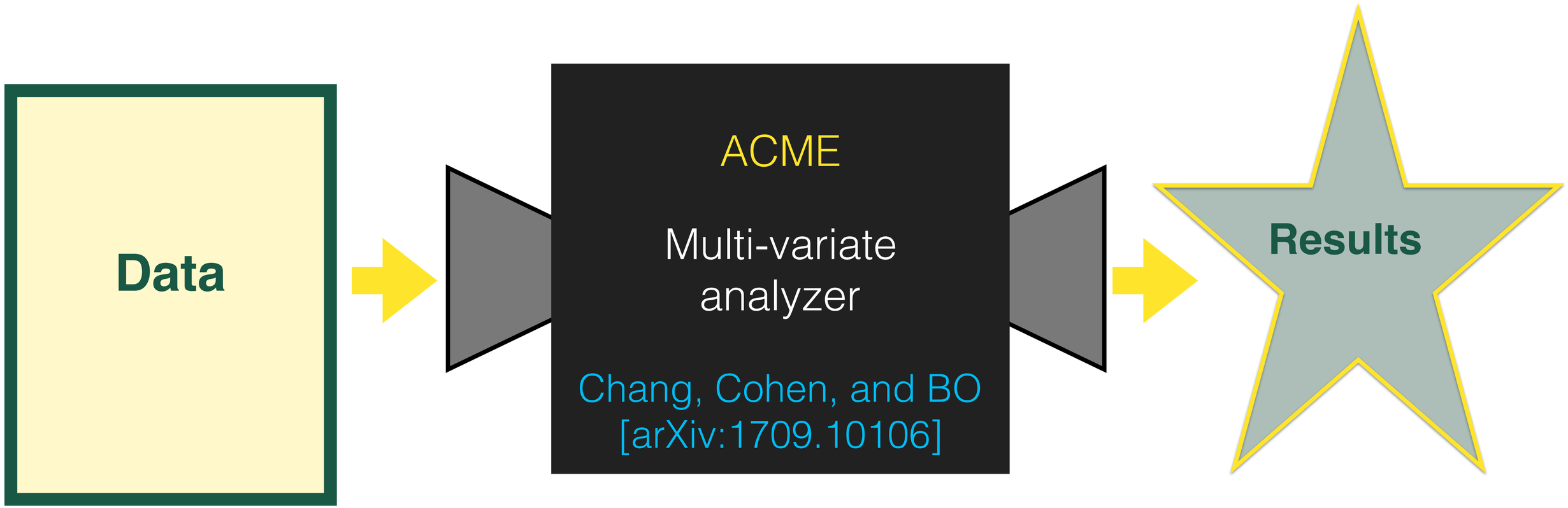
"Planing" to expose what the machine is learning

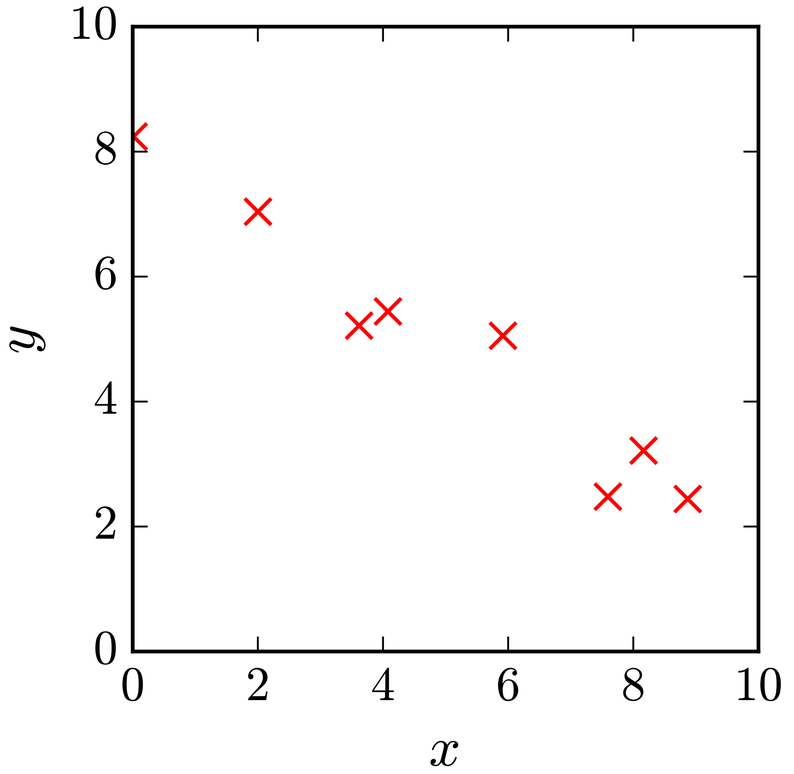


Bryan Ostdiek
Machine Learning for Jet Physics Workshop
Lawrence Berkeley National Laboratory
December 13, 2017

Review: Linear Regression

How to fit data

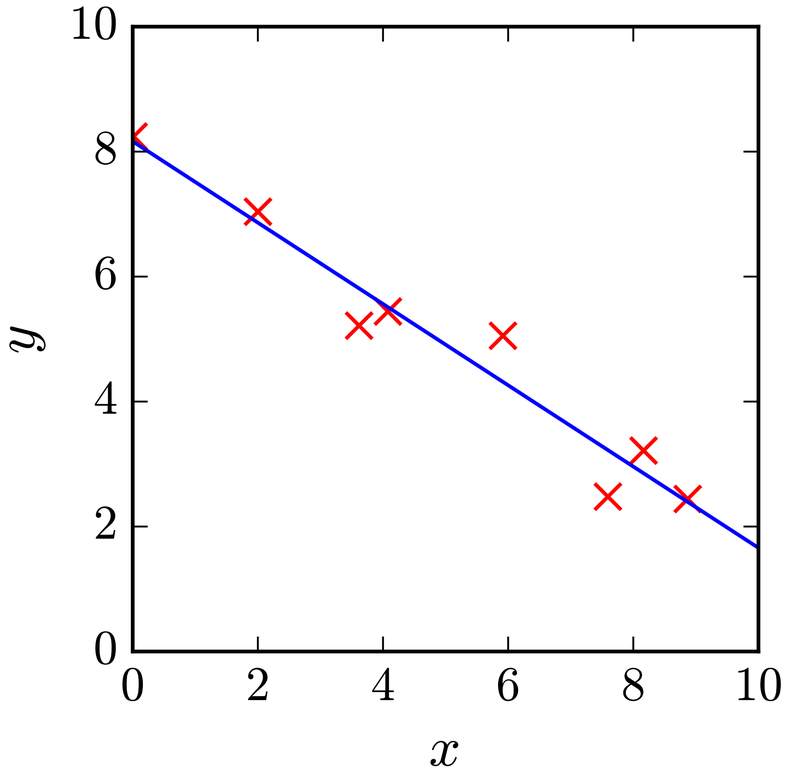
1. Plot the data
2. Define the function
   - $f(x, \vec{a}) = a_0 + a_1 x$
3. Choose how to know what fits best
   - a.k.a. *Loss Function*
   - MSE: $L(x, y, \vec{a}) = \frac{1}{N} \sum_{i=1}^{N} \left(f(x_i, \vec{a}) - y_i \right)^2$
5. Find the minimum error (loss) (cost)
   - $a_{\text{best}} = a$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



Review: Linear Regression

How to fit data

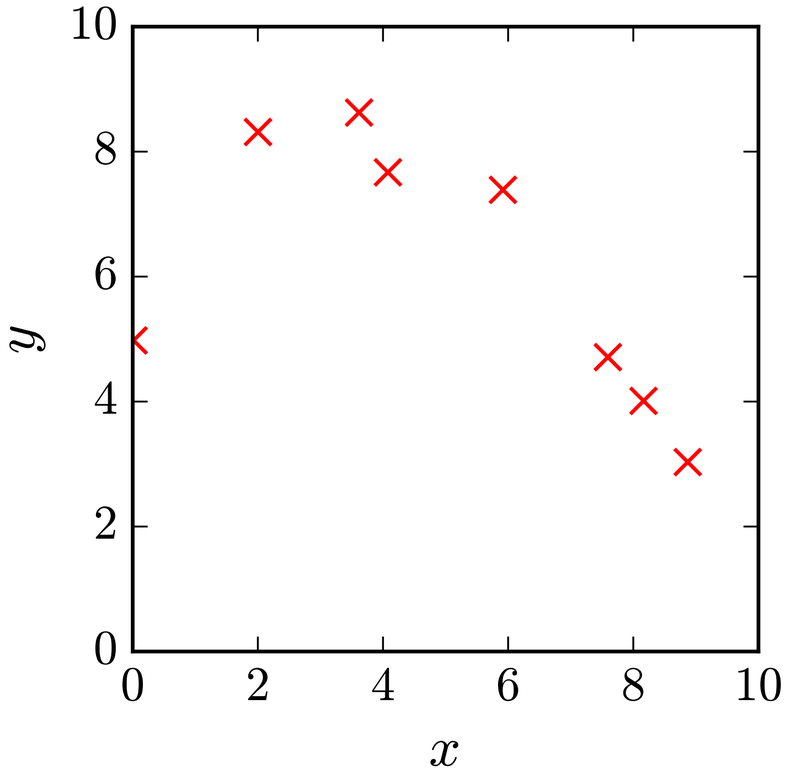
1. Plot the data
2. Define the function
   - $f(x, \vec{a}) = a_0 + a_1 x$
3. Choose how to know what fits best
   - a.k.a. *Loss Function*
   - MSE: $L(x, y, \vec{a}) = \frac{1}{N} \sum_{i=1}^{N} \left(f(x_i, \vec{a}) - y_i \right)^2$
5. Find the minimum error (loss) (cost)
   - $a_{\text{best}} = a$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



Review: Linear Regression

How to fit data

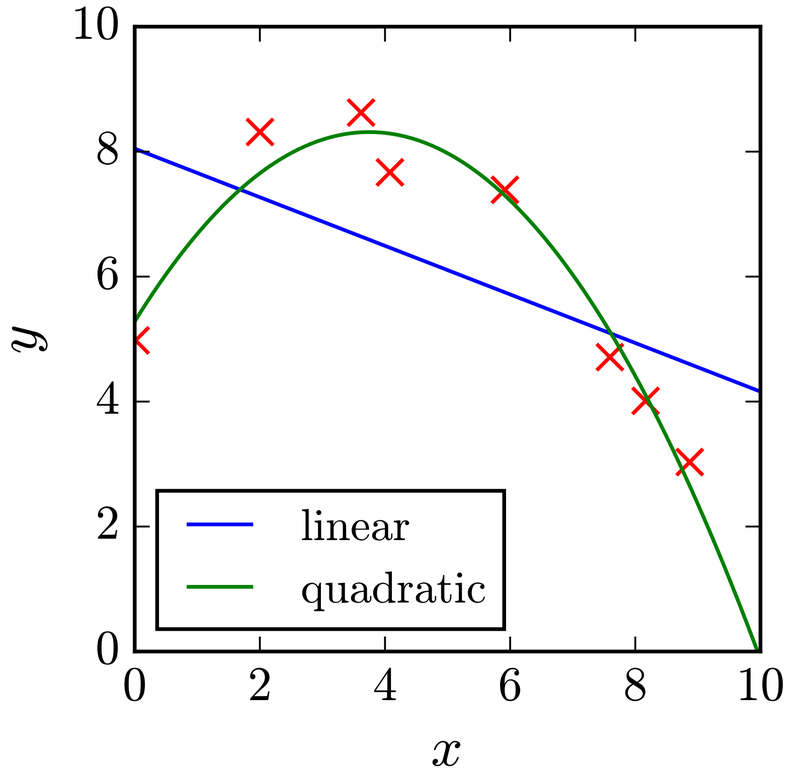
1. Plot the data
2. Define the function
   - $f(x, \vec{a}) = a_0 + a_1 x + a_2 x^2$
3. Choose how to know what fits best
   - a.k.a. *Loss Function*
   - MSE: $L(x, y, \vec{a}) = \frac{1}{N} \sum_{i=1}^{N} \left(f(x_i, \vec{a}) - y_i \right)^2$
5. Find the minimum error (loss) (cost)
   - $a_{\text{best}}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



Review: Linear Regression

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)
   - $a_{\text{best}} = a$ when $\left(\frac{\partial L(x, y, \vec{a})}{\partial \vec{a}} \Big|_{x,y} = 0 \right)$

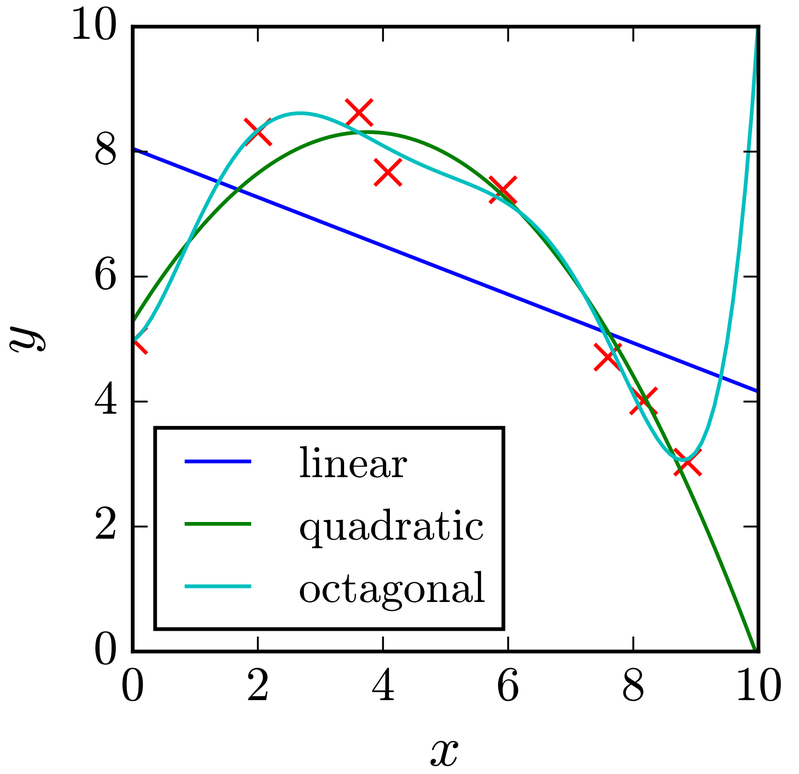


Is that good enough?
How many parameters can we add?

Review: Linear Regression

How to fit data

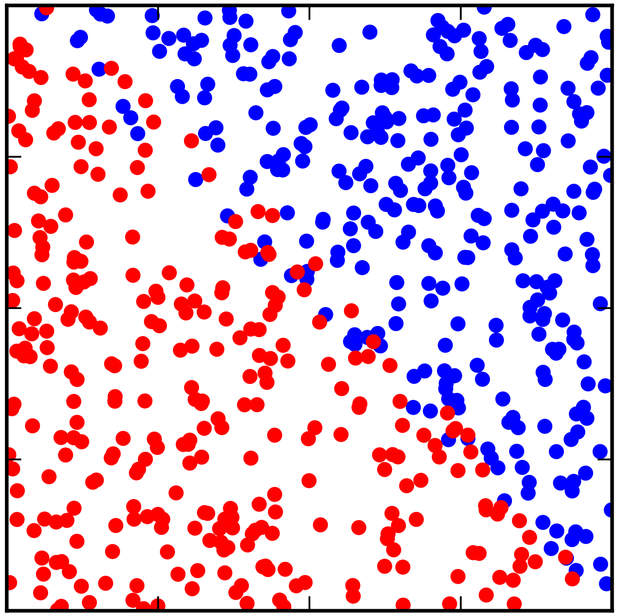
1. Plot the data
2. Define the function
   - $f(x, \vec{a}) = a_0 + a_1 x + a_2 x^2$
3. Choose how to know what fits best
   - a.k.a. *Loss Function*
   - MSE: $L(x, y, \vec{a}) = \frac{1}{N} \sum_{i=1}^{N} \left(f(x_i, \vec{a}) - y_i \right)^2$
5. Find the minimum error (loss) (cost)
   - $a_{\text{best}}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

What is the machine learning?

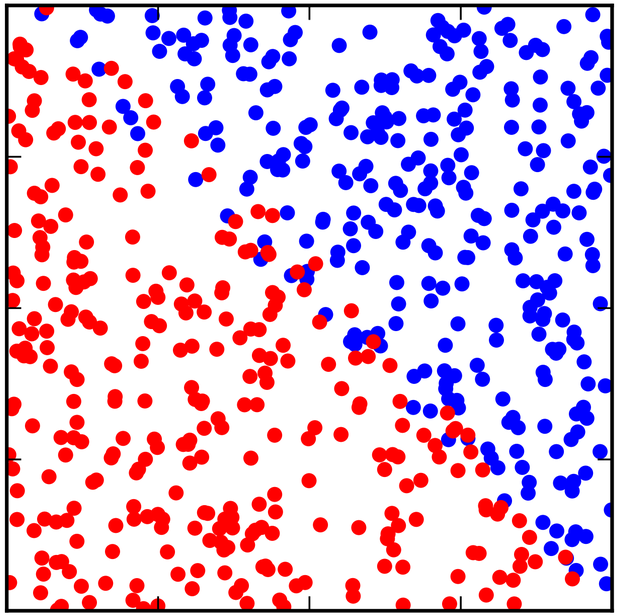
Logistic regression similar, but intuition is off/different

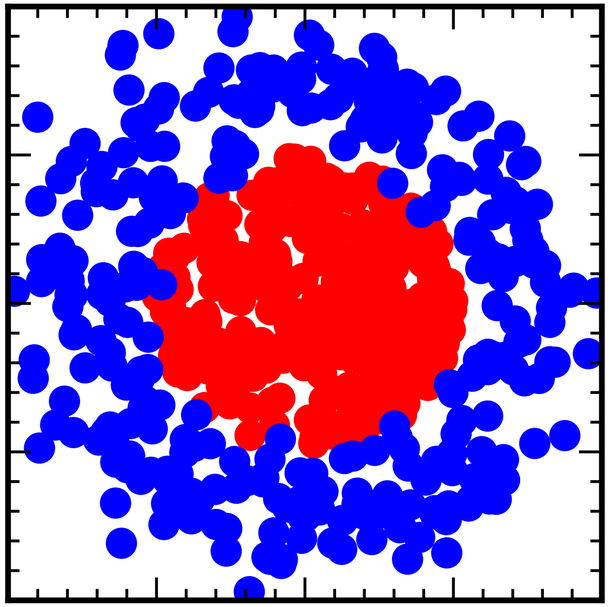


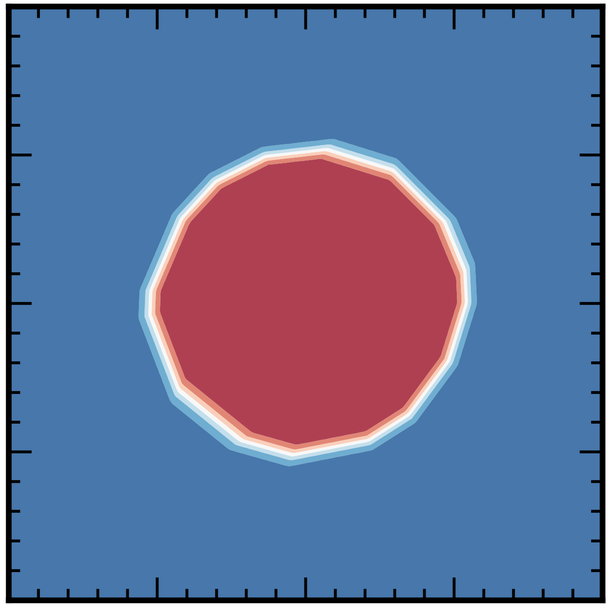
Chang, Cohen, and BO [arXiv:1709.10106]

What is the machine learning?

Logistic regression similar, but intuition is off/different



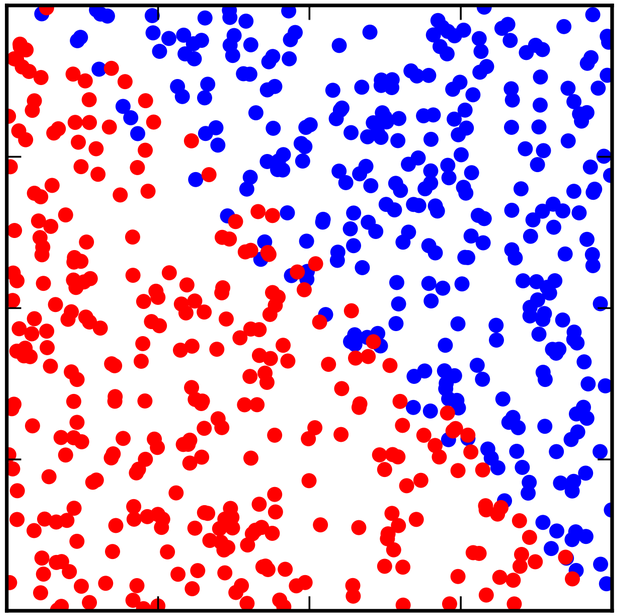


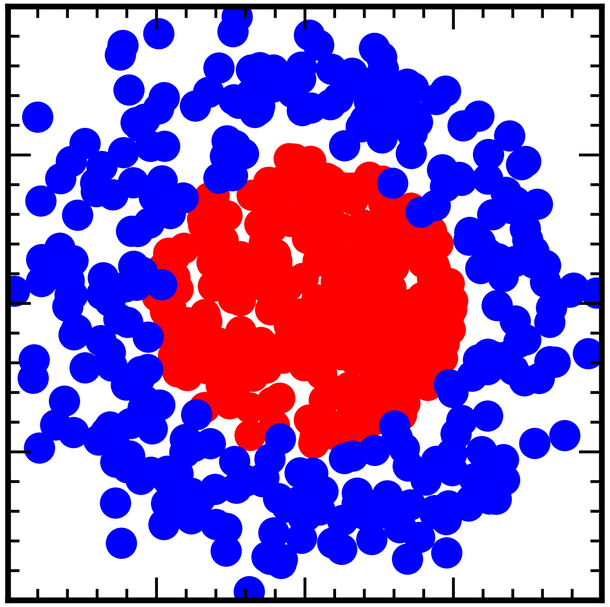


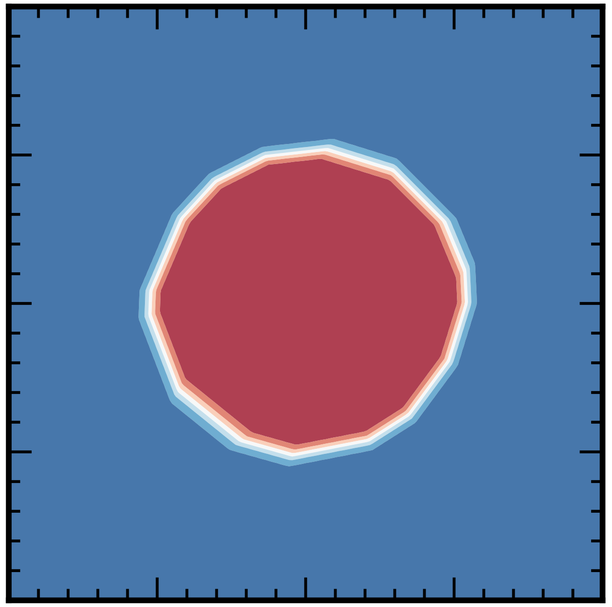
Chang, Cohen, and BO [arXiv:1709.10106]

What is the machine learning?

Logistic regression similar, but intuition is off/different







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 weighed (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]

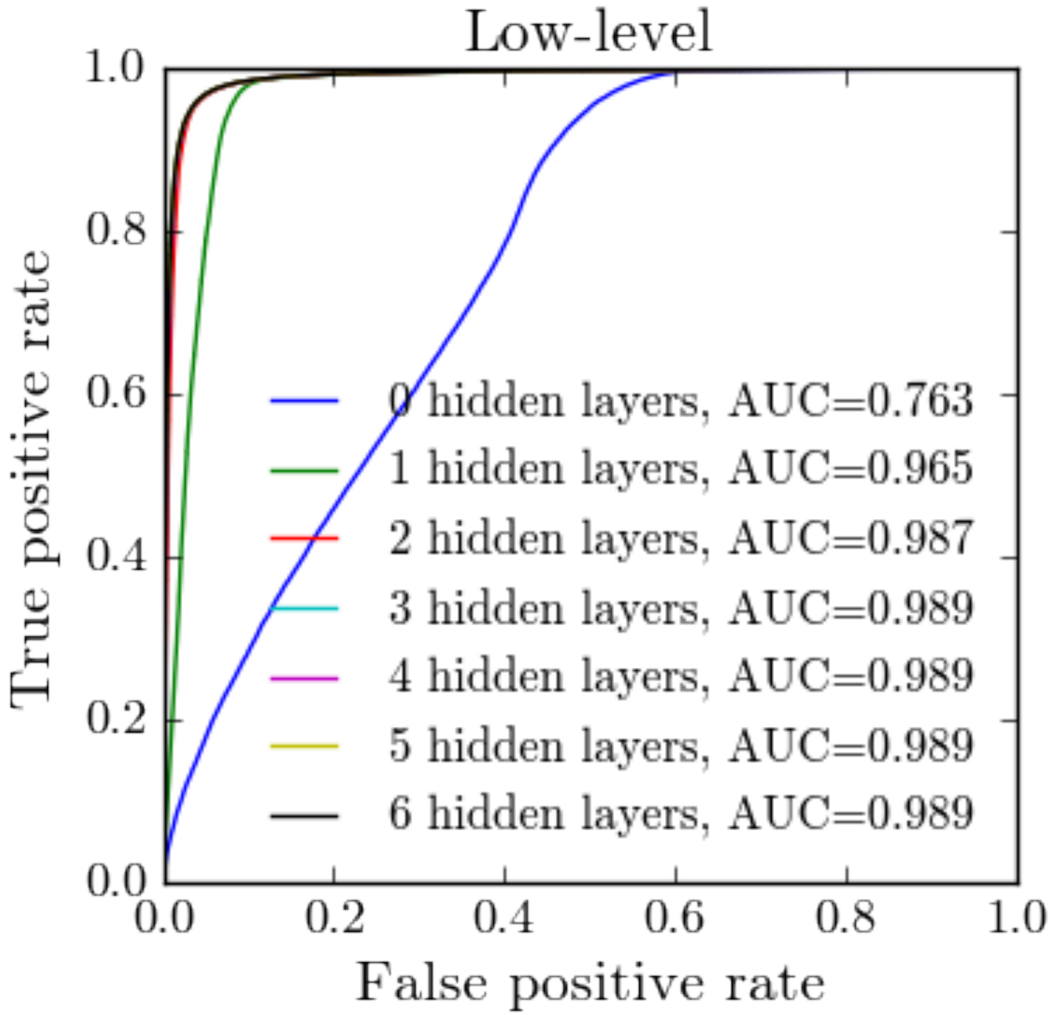
(a) Train machine on low level data
(b) Compute low level AUC
(c) Choose a variable: compute (planing) weights
(d) Train machine on weighed (planed) data
(e) Compute planed AUC
(f) Compare: looking for significant performance drop

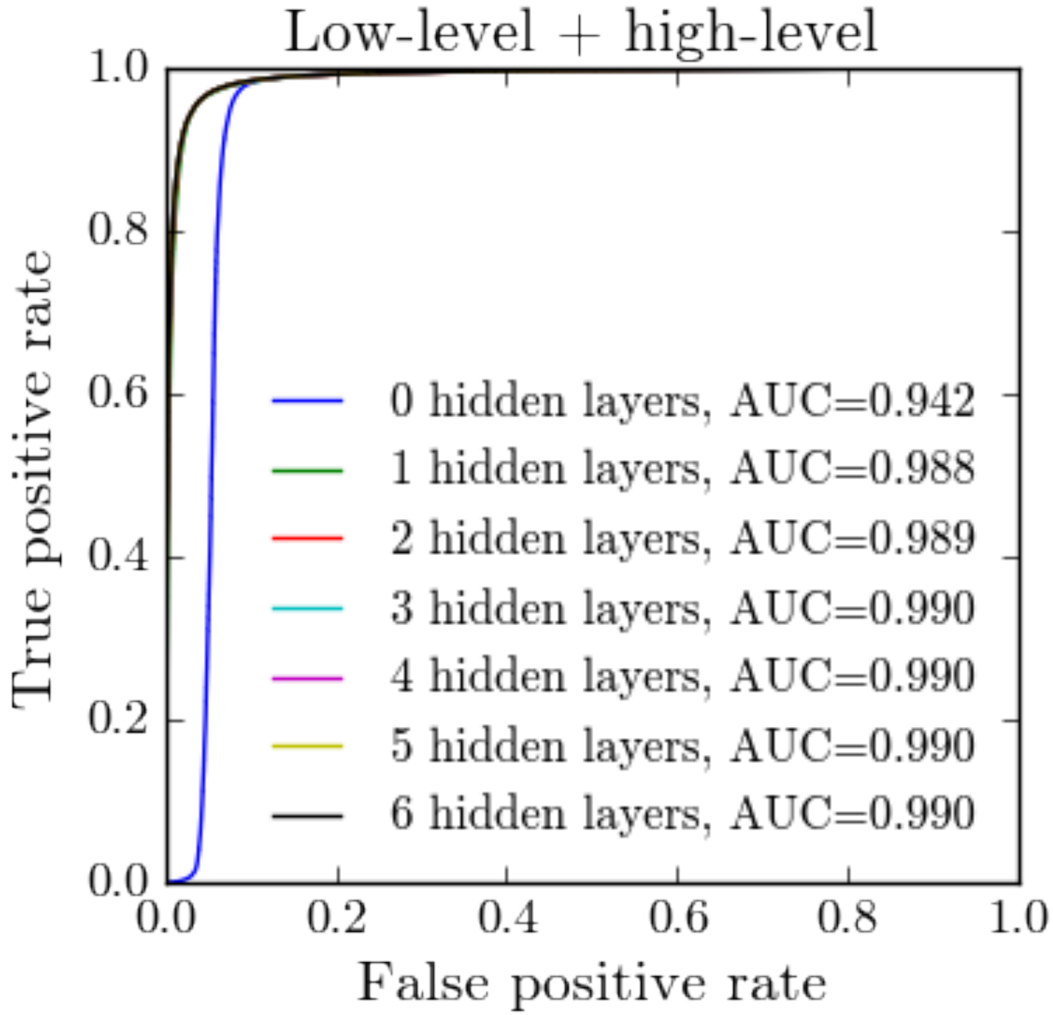
Saturation

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]

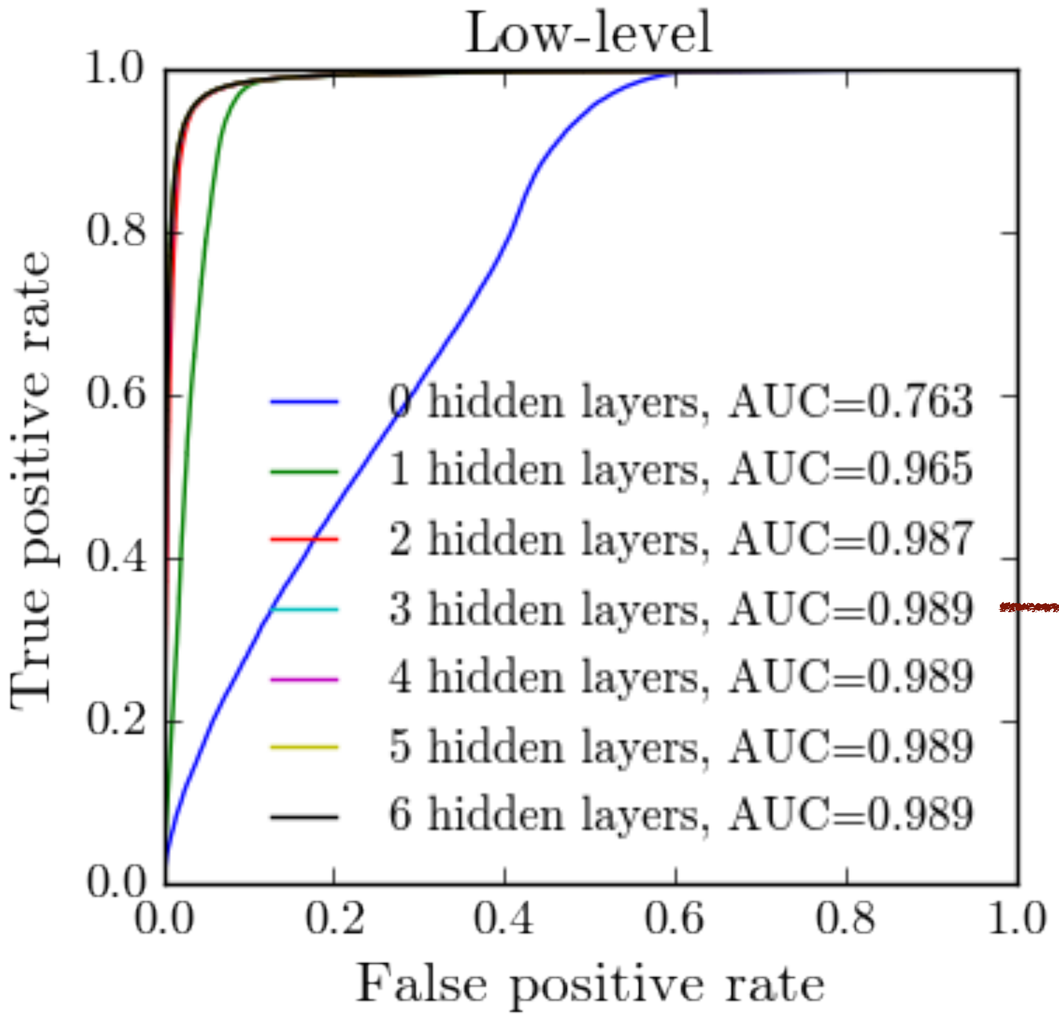
(a) Train network on low level data
(b) Compute low level AUC
(c) Add high level variable
(d) Train new machine using low + high level info
(e) Compute AUC
(f) No performance gain implies information has been learned

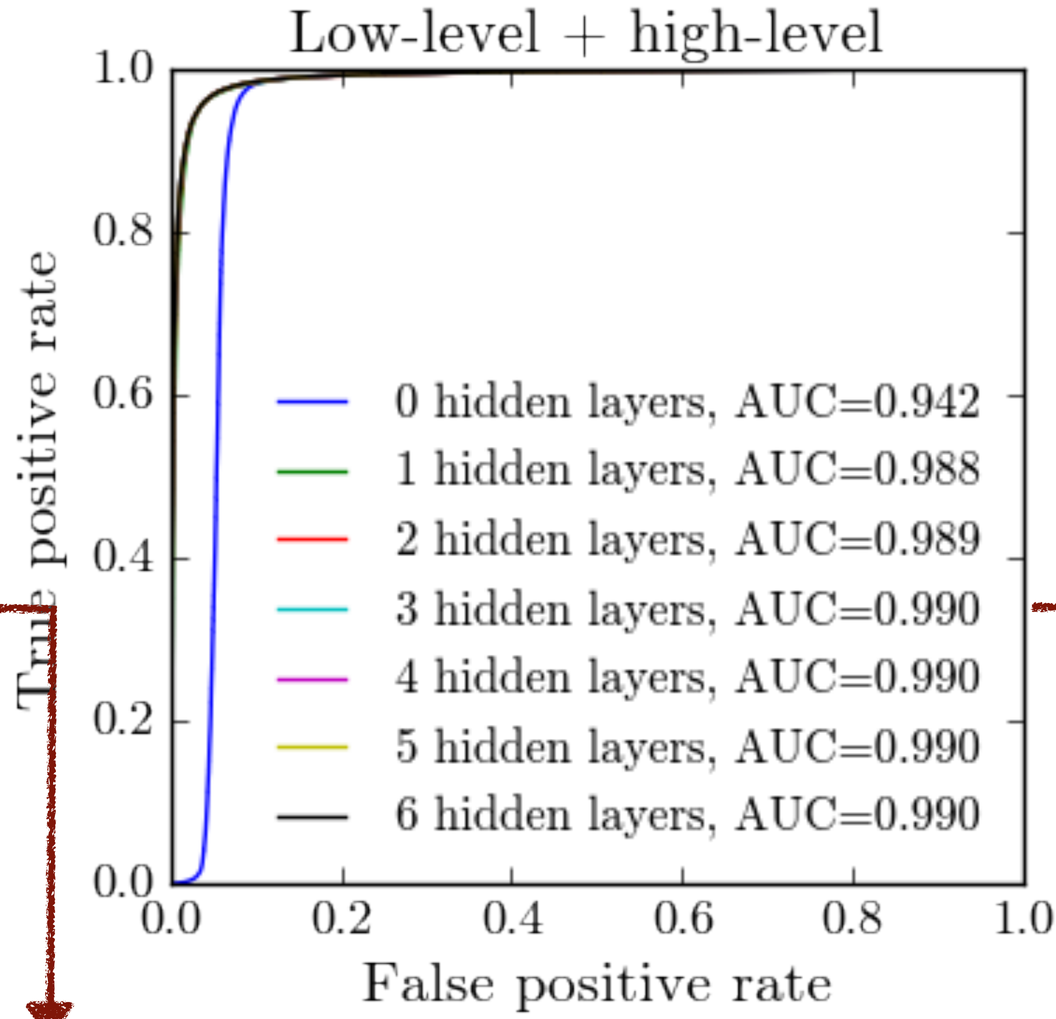
Using saturation to pick the network





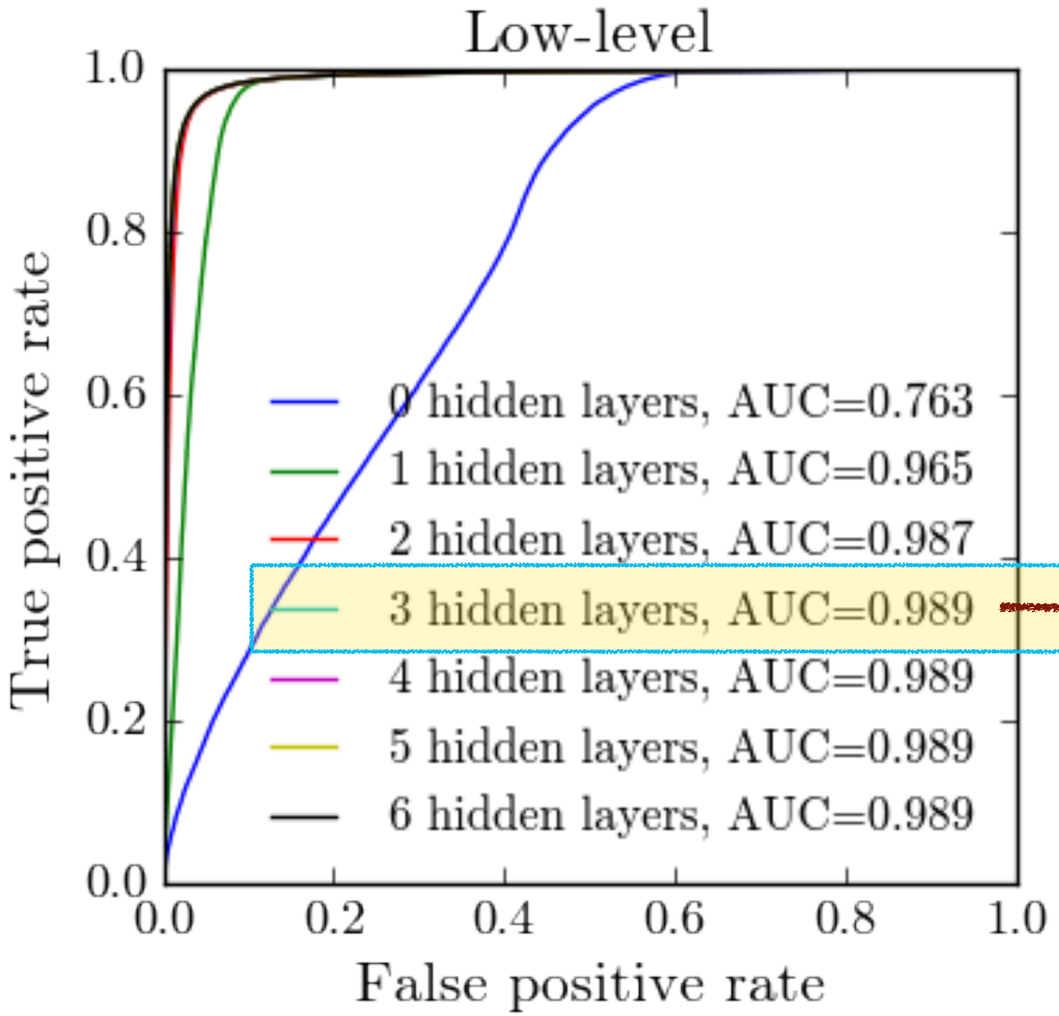
Using saturation to pick the network

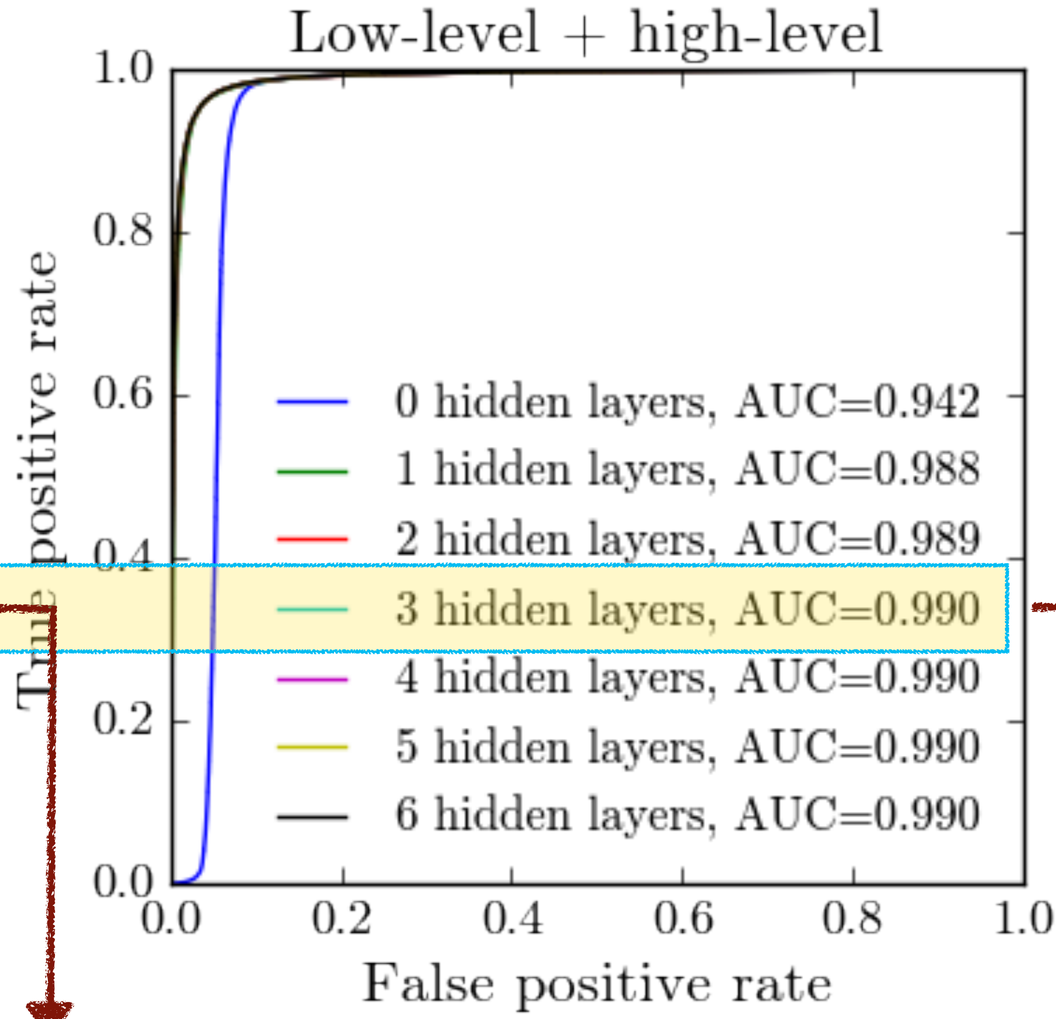




Minimal improvement

Using saturation to pick the network

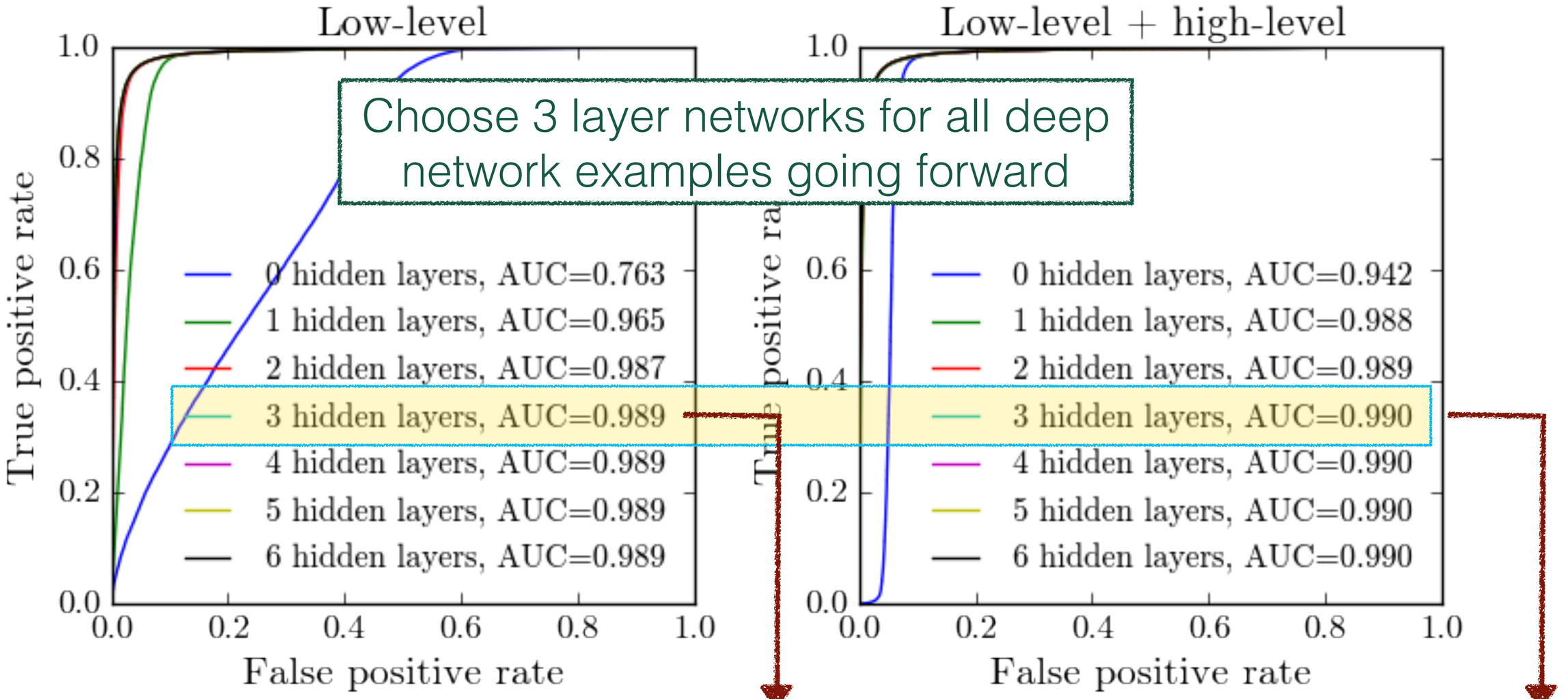




high-level not adding much

Minimal improvement

Using saturation to pick the network

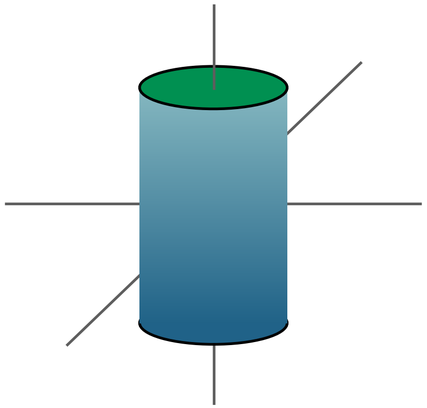


Choose 3 layer networks for all deep network examples going forward

high-level not adding much

Minimal improvement

Example with a toy model

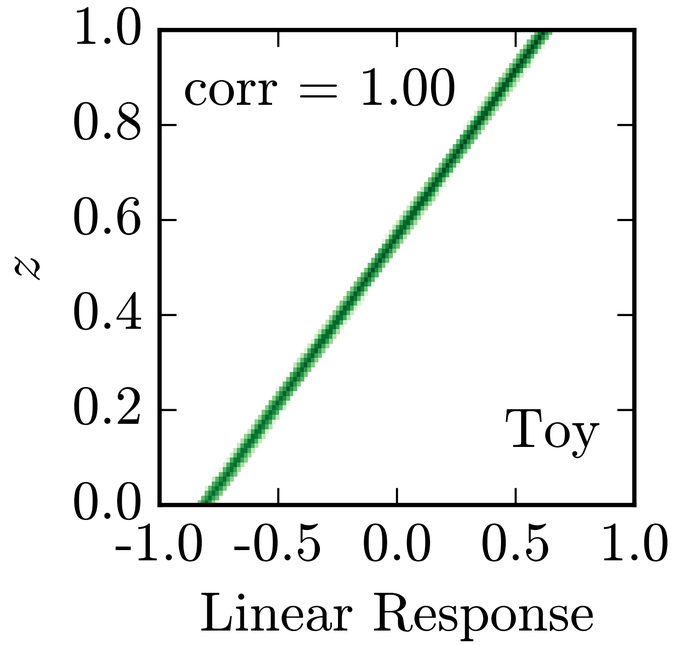


Signal Distribution: $f(\vec{x}) = [\Theta(r_0 - r) + C_r] \cdot [z \cdot B_z + C_z]$

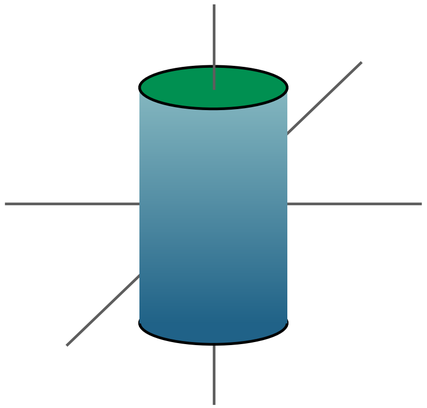
Background Distribution: Uniform

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



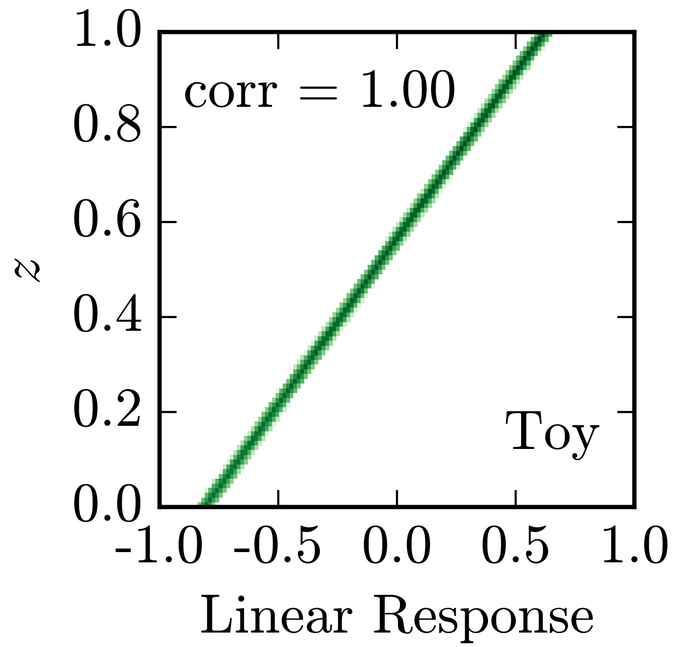
Signal Distribution: $f(\vec{x}) = [\Theta(r_0 - r) + C_r] \cdot [z \cdot B_z + C_z]$

Background Distribution: Uniform

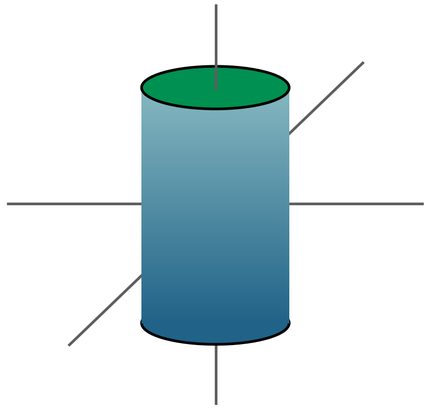
Check Saturation

Results

	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



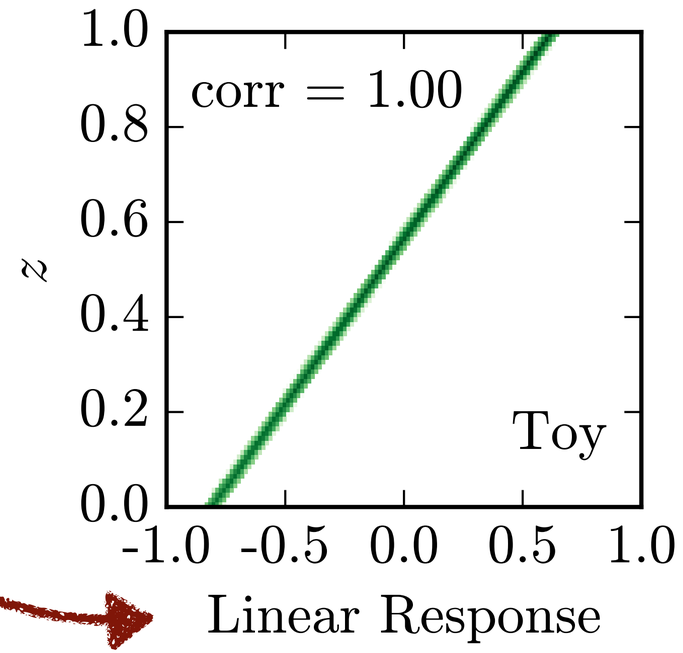
Signal Distribution: $f(\vec{x}) = [\Theta(r_0 - r) + C_r] \cdot [z \cdot B_z + C_z]$

Background Distribution: Uniform

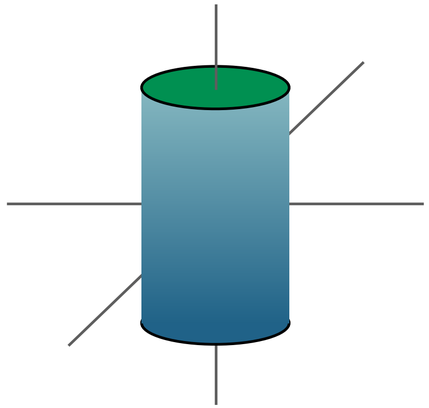
Check Saturation

Results

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

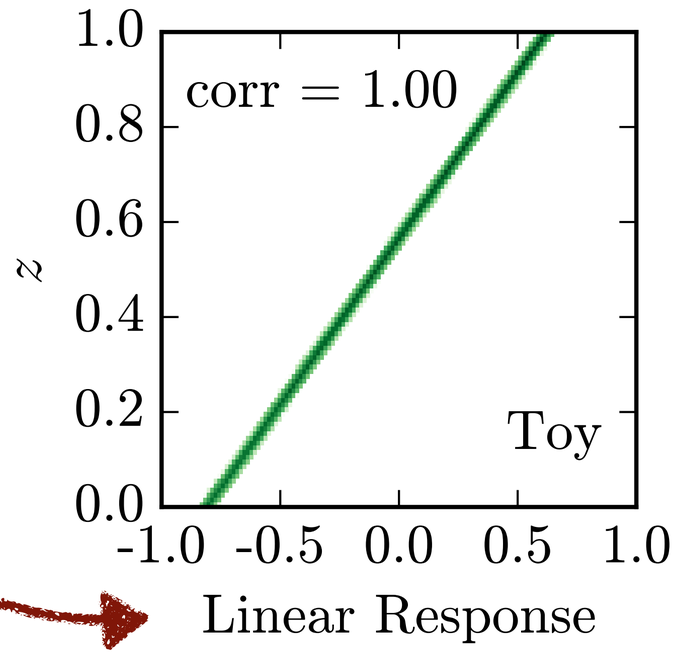
Background Distribution: Uniform

Check Saturation

Results

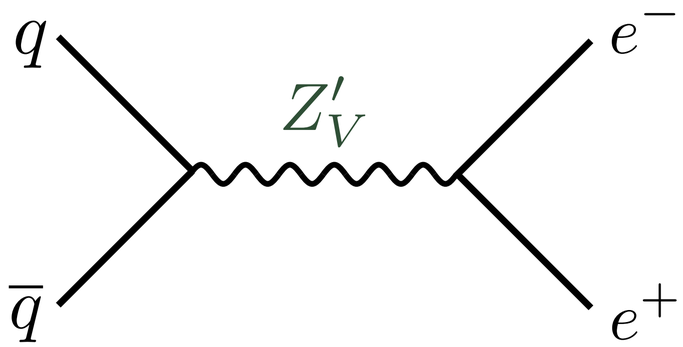
	r	Planed	Linear AUC	Deep AUC
	✘	✘	0.61275(01)	0.81243(45)
✓	✓	✘	0.79672(01)	0.81388(23)
✓	✘	r	0.61030(01)	0.61026(02)
✓	✘	(r, z)	0.5081(16)	0.49998(03)

No discrimination left



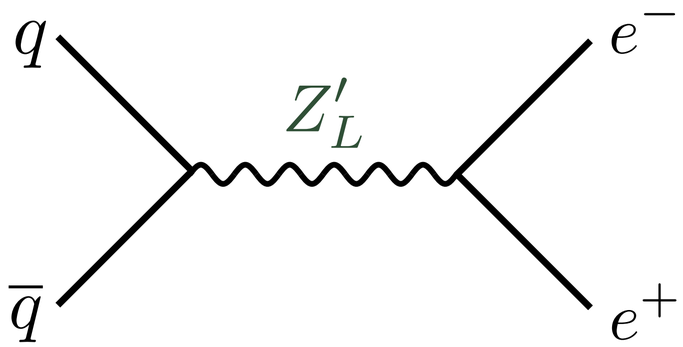
BSM Models

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



Vector couplings

$$g_L = g_R$$



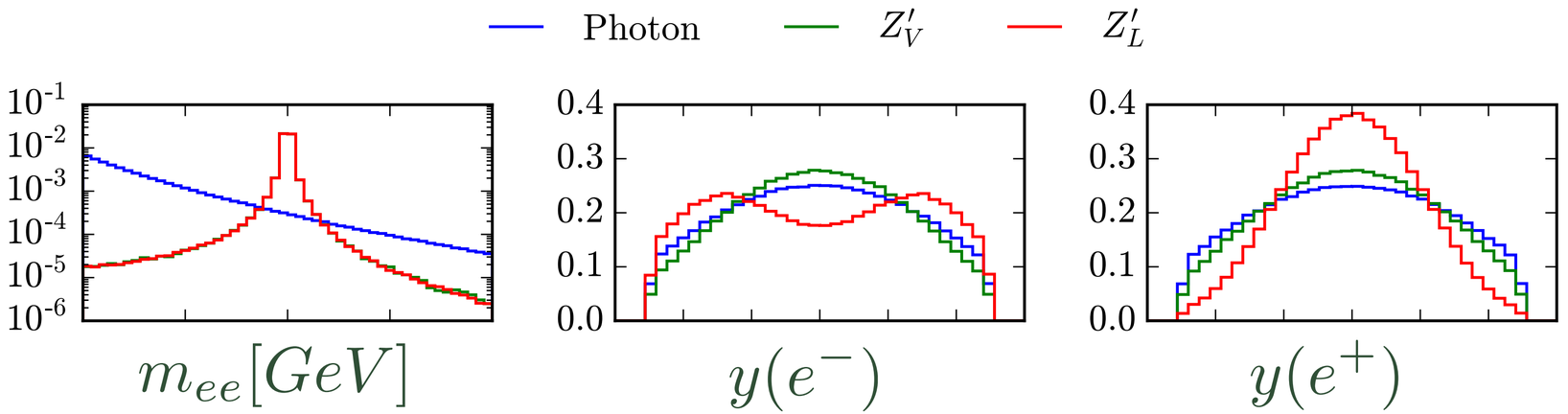
Left-handed couplings

$$g_R = 0$$

What is the machine learning?

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

Original distributions



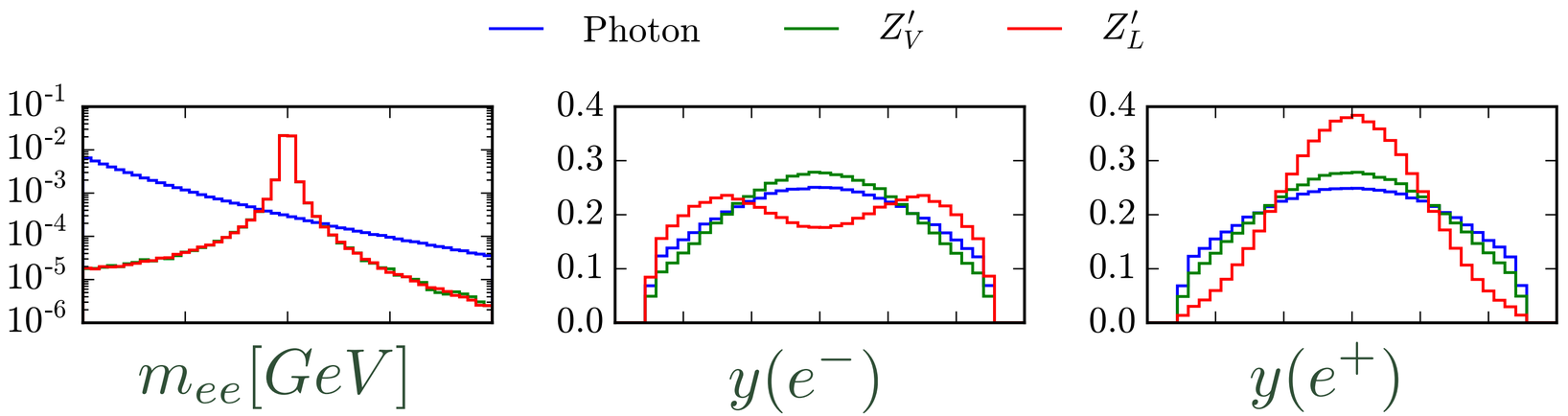
Planed distributions

$(E, \vec{p})$	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

$(E,\vec{p})$	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)

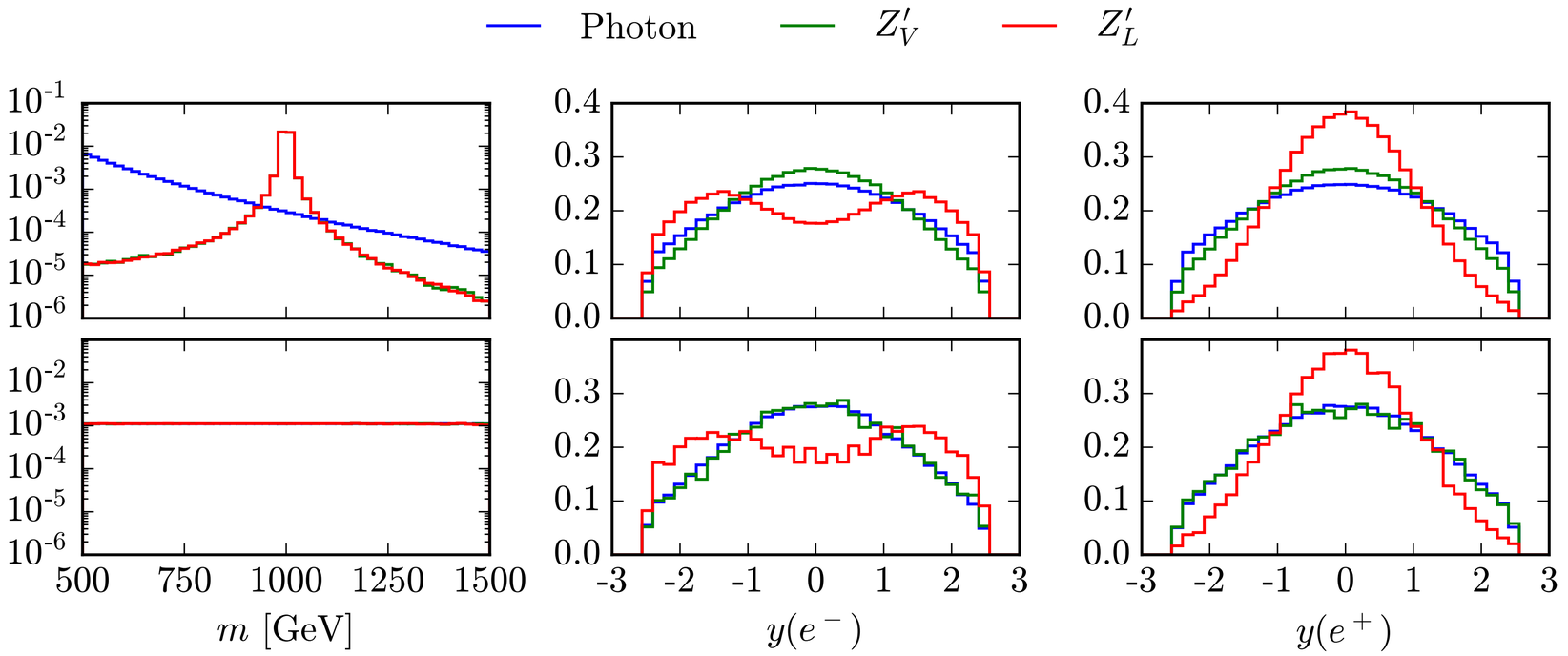
Something more than m_{ee}?

What is the machine learning?

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

Original distributions

Planed distributions



$(E,\vec{p})$	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)

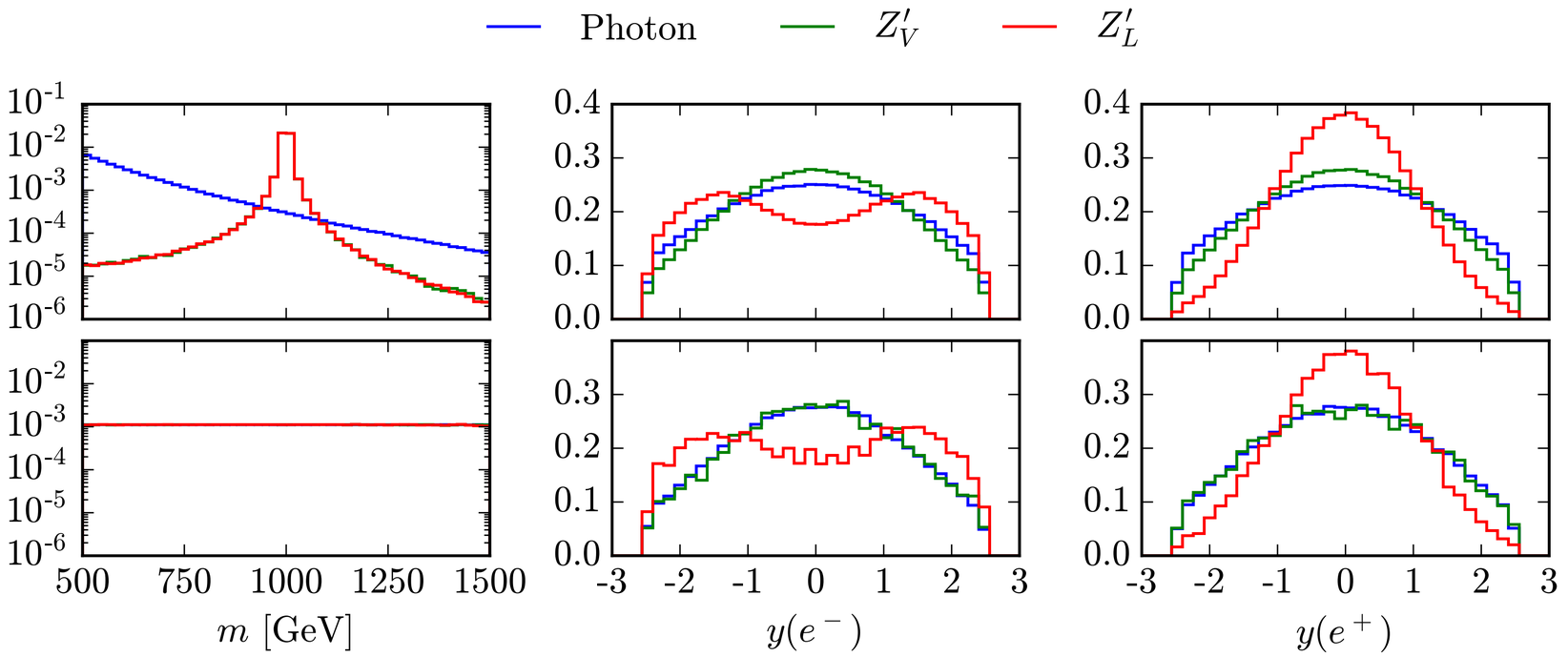
Something more than m_{ee}?

What is the machine learning?

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

Original distributions

Planed distributions



$(E,\vec{p})$	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}?

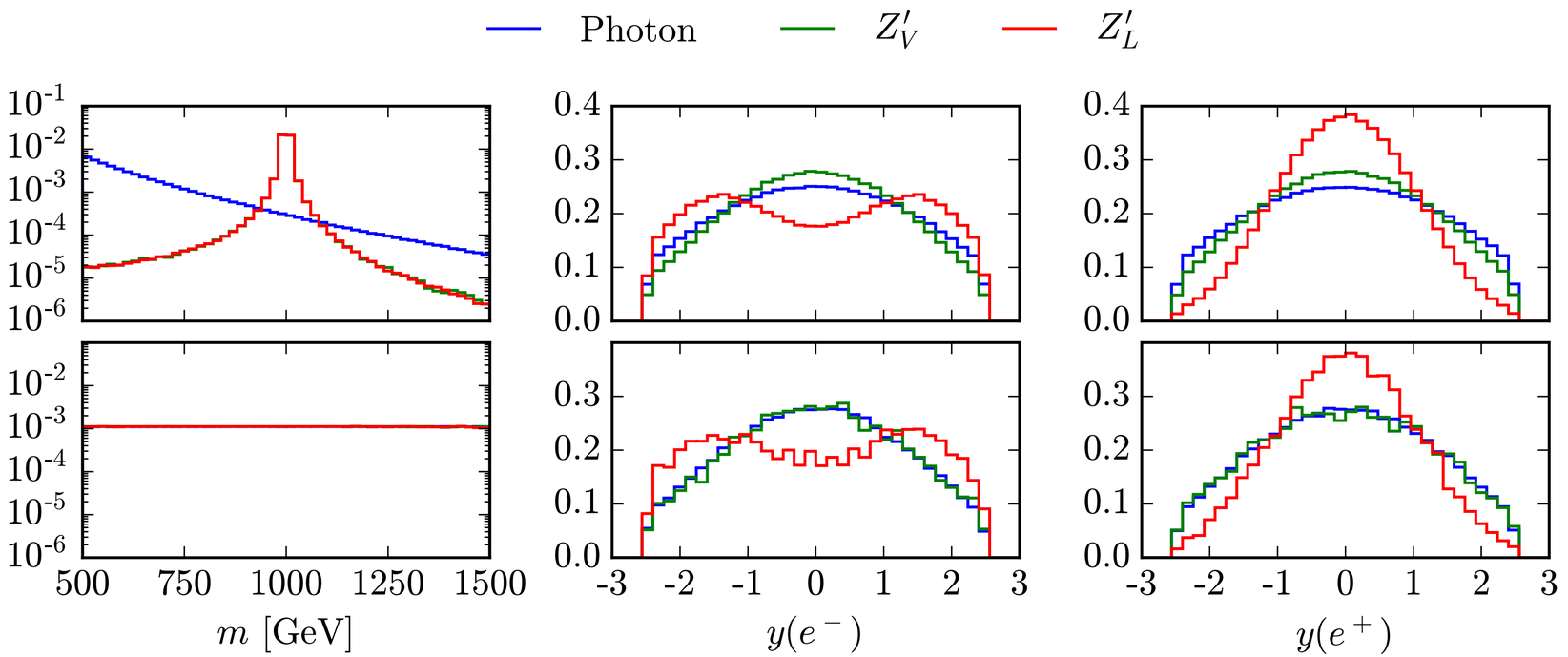
NO!

What is the machine learning?

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

Original distributions

Planed distributions



$(E,\vec{p})$	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)

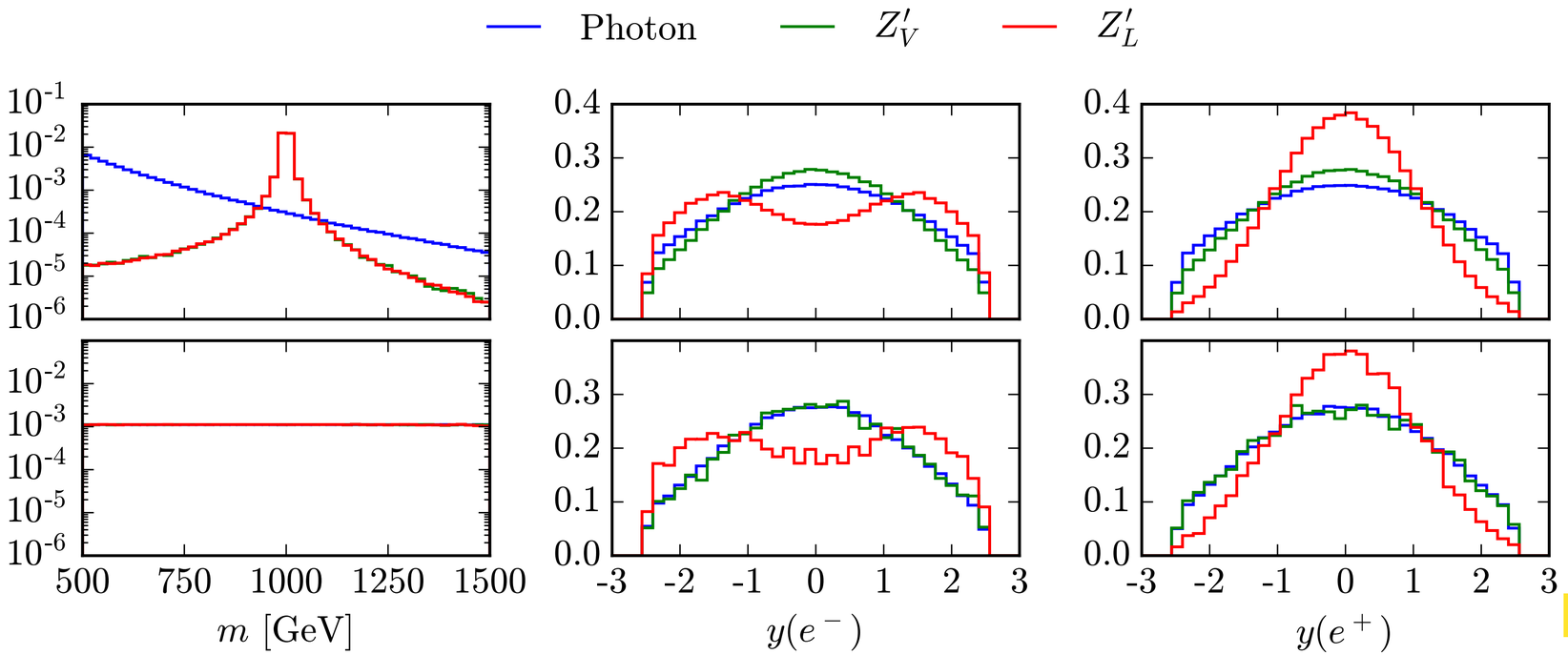
$(E,\vec{p})$	m	Planed	Linear AUC	Deep AUC
✓	✘	✘	0.763280(05)	0.989353(59)
✓	✓	✘	0.942004(02)	0.989826(10)
✓	✘	m	0.626648(28)	0.6258(24)
✓	✘	$(m, \Delta\lvert y\rvert)$	0.52421(15)	0.5320(25)

What is the machine learning?

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

Original distributions

Planed distributions



More Information

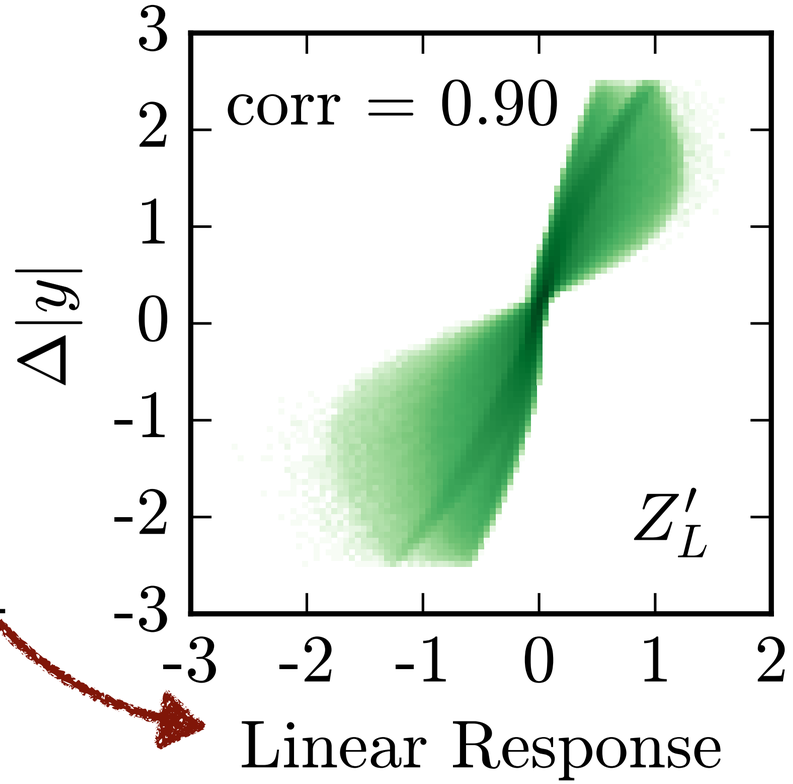
$(E,\vec{p})$	m	Planed	Linear AUC	Deep AUC
✓	✘	✘	0.746221(01)	0.988510(98)
✓	✓	✘	0.938967(01)	0.989007(03)
✓	✘	m	0.50550(29)	0.4942(48)

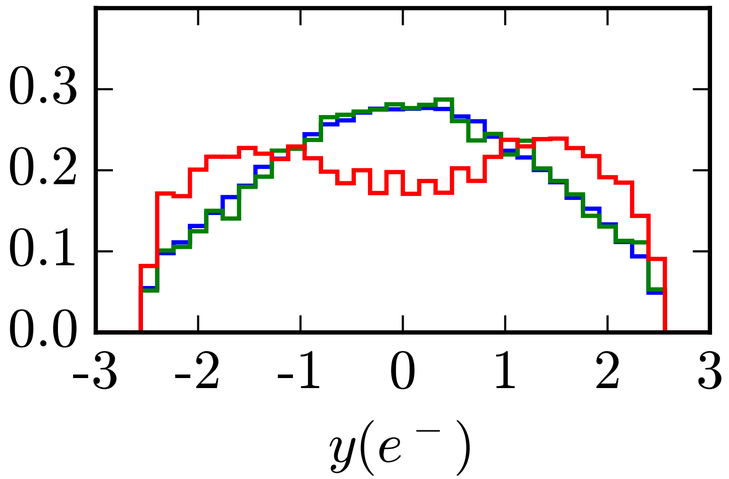
Linear

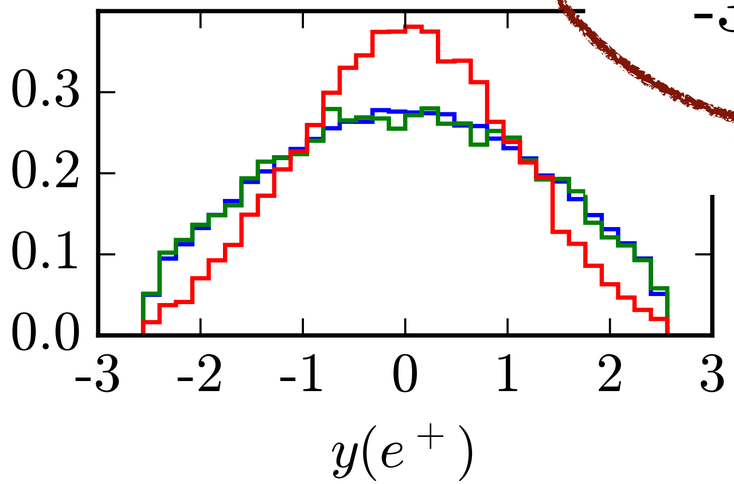
$(E,\vec{p})$	m	Planed	Linear AUC	Deep AUC
✓	✘	✘	0.763280(05)	0.989353(59)
✓	✓	✘	0.942004(02)	0.989826(10)
✓	✘	m	0.626648(28)	0.6258(24)
✓	✘	$(m, \Delta\lvert y\rvert)$	0.52421(15)	0.5320(25)

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)
✓	✘	$(m, \Delta\lvert y\rvert)$	0.52421(15)	0.5320(25)

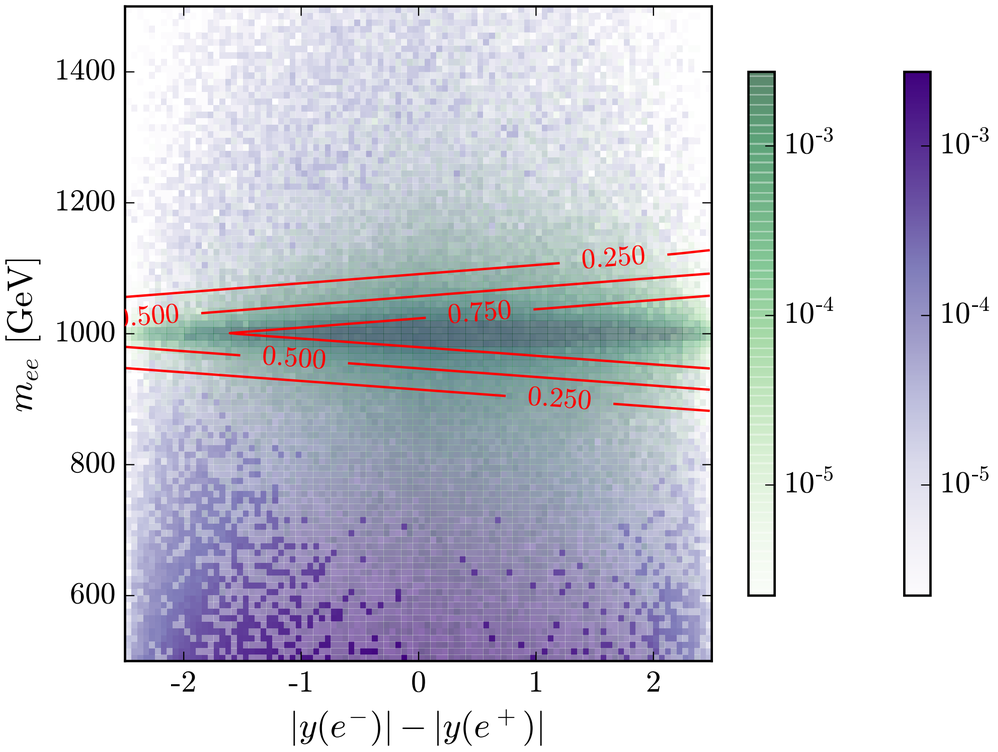


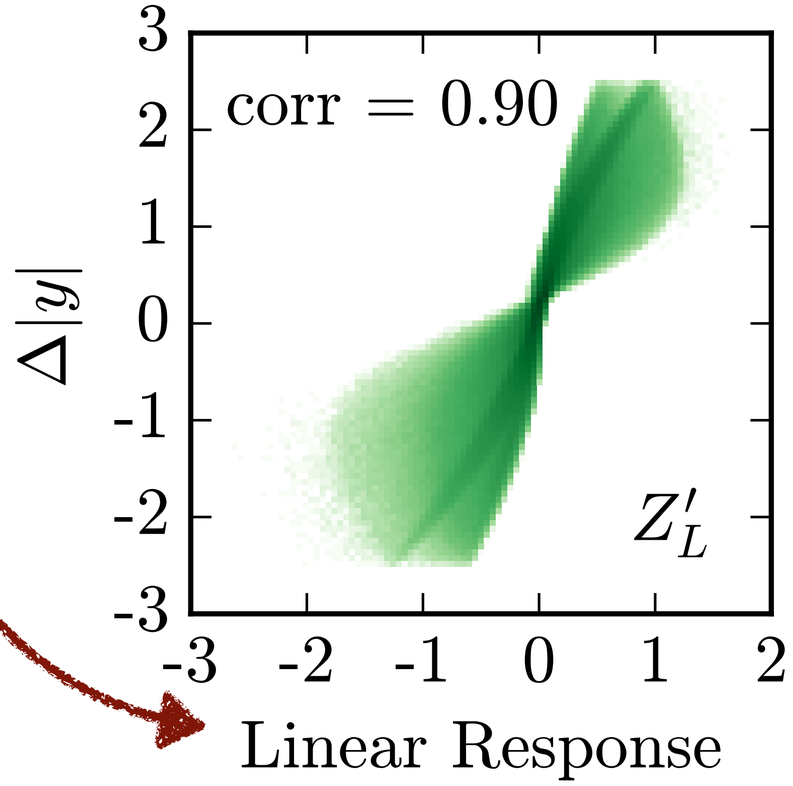




What is the machine learning?

$(E, \vec{p})$	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)
✓	✘	$(m, \Delta\lvert y\rvert)$	0.52421(15)	0.5320(25)





Using only mass: AUC = 0.939, ACC = 0.937

Using both: AUC = 0.989, ACC = 0.958

UNIVERSITY OF OREGON

Conclusion

- Iteratively remove information to see what the machine needs to learn
- Process allows for simple way to see if discriminating power is linear or not

Future Directions

- Apply to more realistic setting
- What if best variables are unknown?
- What if many iterations are needed?