

# Intro to Deep Learning

- 1) The neuron model
- 2) Gradient descent
- 3) Architectures (mostly for vision)
- 4) **Model evaluation**

---

# Training Error and Generalization Error

---

- Training error: model error on the training data
- Generalization error: model error on new data
- Example: practice a future exam with past exams
  - Doing well on past exams (training error) doesn't guarantee a good score on the future exam (generalization error)
  - Student A learns 100% of the answers by heart
  - Student B understands the reasons for 90% of answers

---

# Validation Dataset and Test Dataset

---

- Validation dataset: a dataset used to select the model
  - E.g. Take out 50% of the training data
  - Should not be mixed with the training data (#1 mistake)
- Test dataset: a dataset that can be used once, e.g.
  - A future exam
  - Dataset used in private leaderboard in Kaggle

# Validation



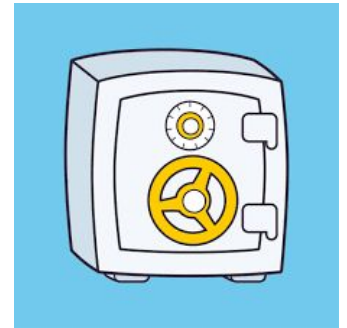
Training data

Validation data

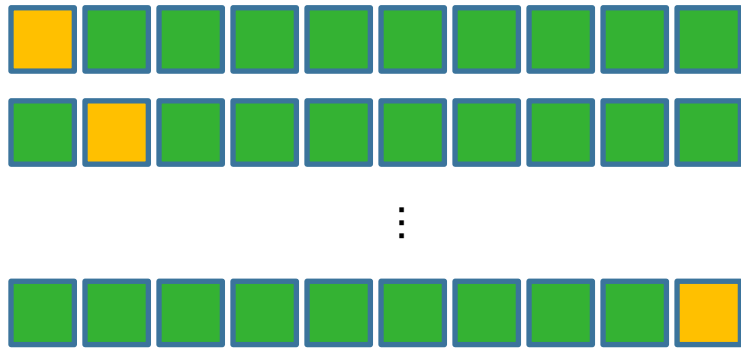


Model chosen based on its performance here

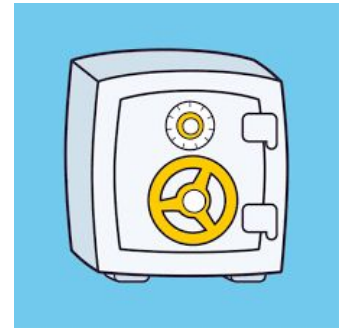
Test data



# K-fold Cross Validation



Test data

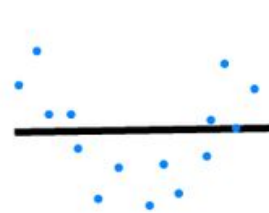


Model chosen based on its average performance across all folds

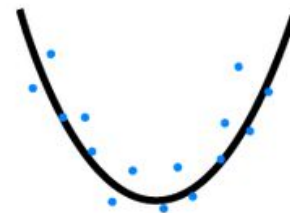
---

# Underfitting

# Overfitting



Underfitting



Desired



Overfitting

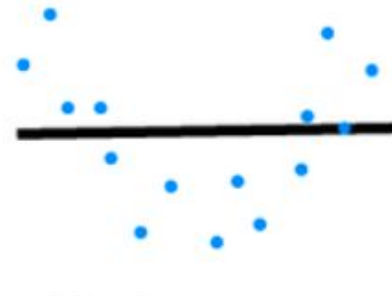
---

Image credit: hackernoon.com

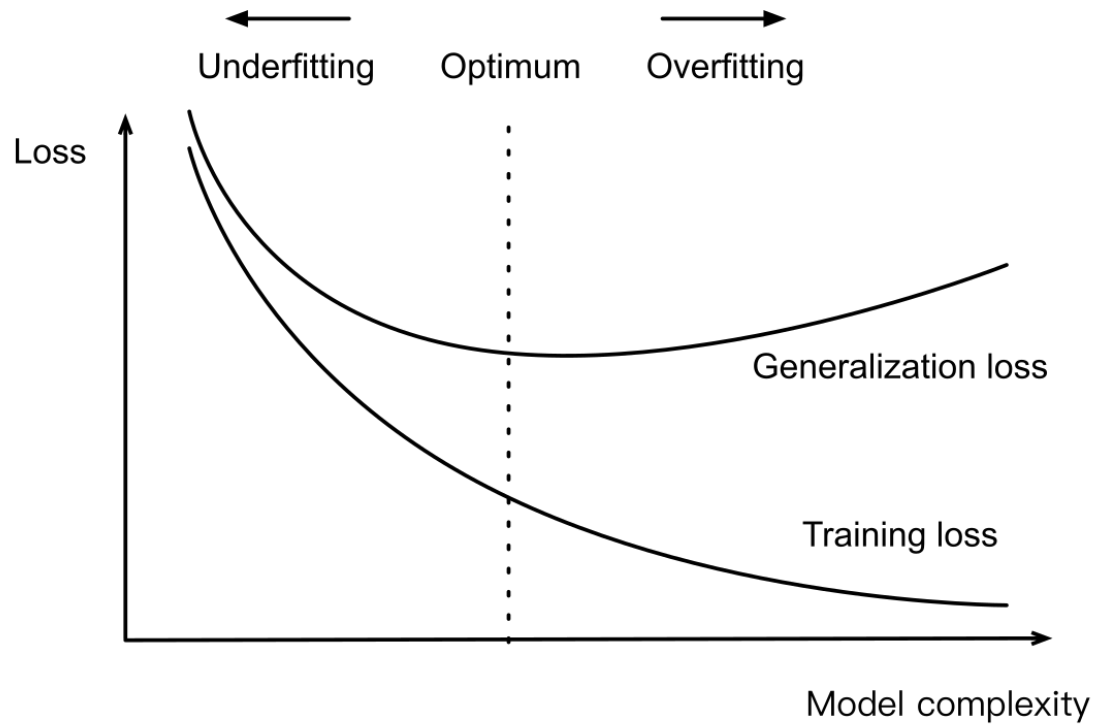
# Model Capacity

---

- The ability to fit variety of functions
- Low capacity models struggles to fit training set
  - Underfitting
- High capacity models can memorize the training set
  - Overfitting

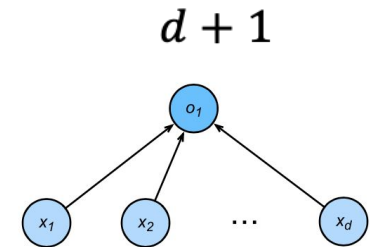


# Influence of Model Complexity

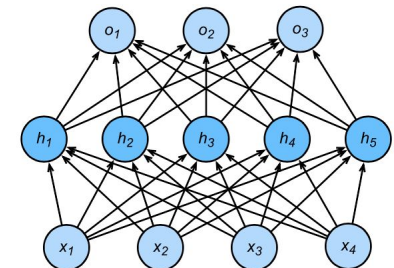


# Estimate Model Capacity

- It's hard to compare complexity between different algorithms
  - e.g. tree vs neural network
- Given an algorithm family, two main factors matter:
  - The number of parameters
  - The values taken by each parameter



$(d + 1)m + (m + 1)k$



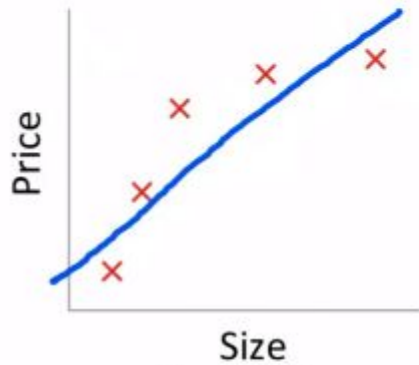
# Underfitting and Overfitting

---

		Data complexity	
		Simple	Complex
Model capacity	Low	Normal	Underfitting
	High	Overfitting	Normal

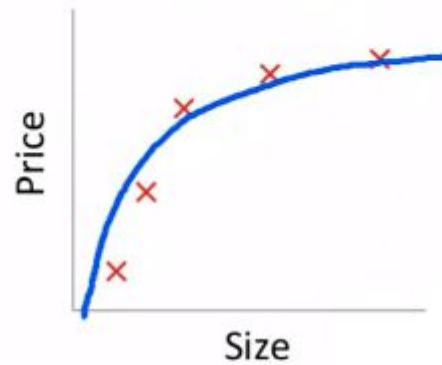
# Regularization with weight decay

## Bias/variance



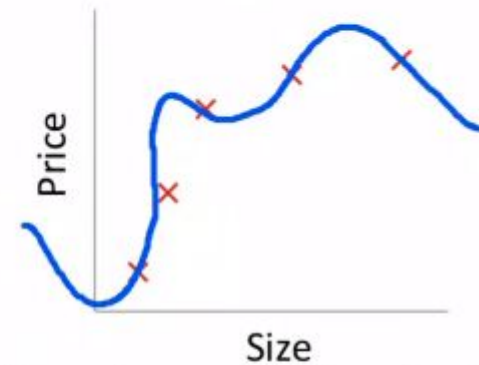
$$f(x) = b + w_1x$$

High bias  
(underfit)  
 $d=1$



$$f(x) = b + w_1x + w_2x^2$$

“Just right”  
 $d=2$

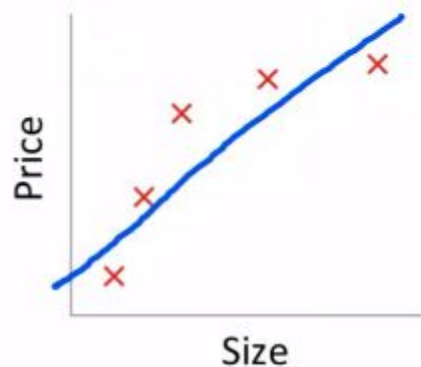


$$f(x) = b + w_1x + w_2x^2 + w_3x^3 + w_4x^4$$

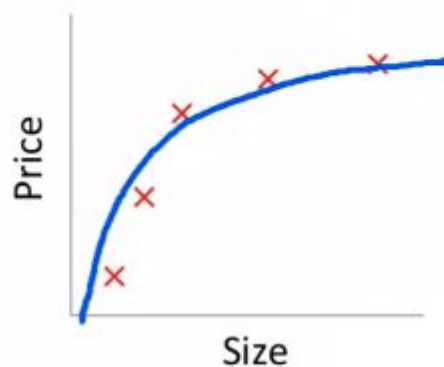
High variance  
(overfit)  
 $d=4$

# Regularization with weight decay

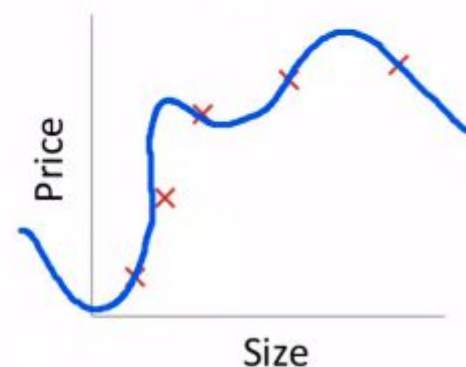
## Bias/variance



$$f(x) = b + w_1x$$



$$f(x) = b + w_1x + w_2x^2$$

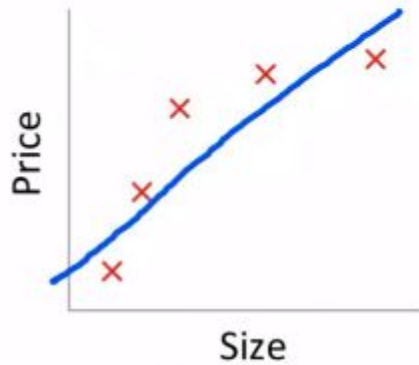


$$f(x) = b + w_1x + w_2x^2 + w_3x^3 + w_4x^4$$

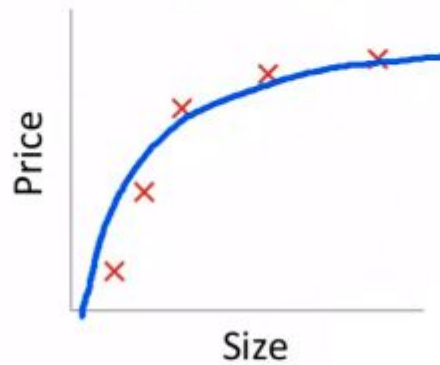
$$L(\mathbf{w}, b) = \sum_{i=1}^d (f(x) - y)^2$$

# Regularization with weight decay

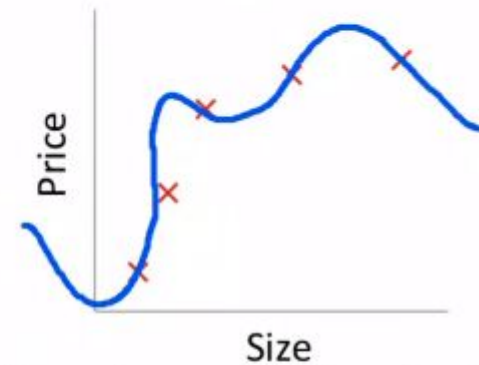
## Bias/variance



$$f(x) = b + w_1x$$



$$f(x) = b + w_1x + w_2x^2$$



$$f(x) = b + w_1x + w_2x^2 + w_3x^3 + w_4x^4$$

$$L(\mathbf{w}, b) = \sum_{i=1}^d (f(x) - y)^2 + \lambda \|\mathbf{w}\|^2$$

---

# Regularization with weight decay

---

- L2-norm regularization:

$$\lambda \|\mathbf{w}\|^2 = \lambda(w_1^2 + w_2^2 + w_3^2 + w_4^2)$$

- L1-norm regularization:

$$\lambda \|\mathbf{w}\| = \lambda(|w_1| + |w_2| + |w_3| + |w_4|)$$

---

# Regularization with weight decay

---

- L2-norm regularization:

$$\lambda \|\mathbf{w}\|^2 = \lambda(w_1^2 + w_2^2 + w_3^2 + w_4^2)$$

- L1-norm regularization:

$$\lambda \|\mathbf{w}\| = \lambda(|w_1| + |w_2| + |w_3| + |w_4|)$$

<http://playground.tensorflow.org/>

# Regularization with weight decay

Epoch 000,000    Learning rate 0.03    Activation ReLU    **Regularization L1**    **Regularization rate 0.01**    Problem type Classification

DATA  
Which dataset do you want to use?



Ratio of training to test data: 10%

Noise: 50

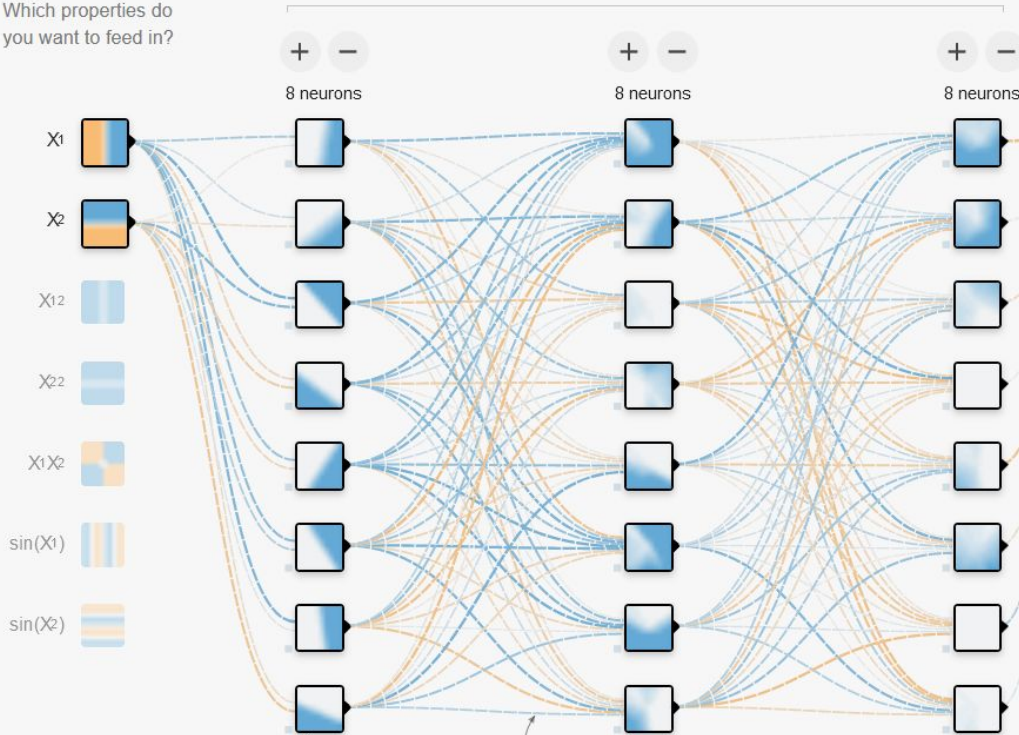
Batch size: 10

REGENERATE

FEATURES  
Which properties do you want to feed in?

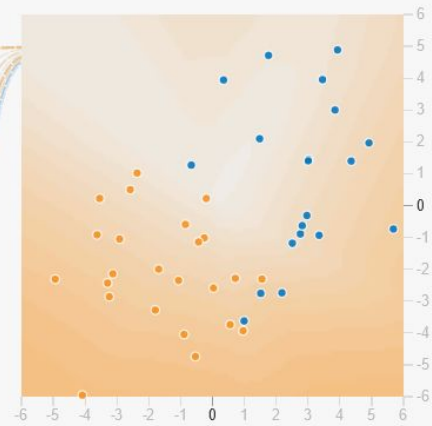
- $X_1$
- $X_2$
- $X_{12}$
- $X_{22}$
- $X_1 X_2$
- $\sin(X_1)$
- $\sin(X_2)$

+ - 3 HIDDEN LAYERS



OUTPUT

Test loss 0.476  
Training loss 0.465



Colors shows data, neuron and weight values.

Show test data     Discretize output

# Regularization with weight decay

⏪ ▶ 000,000 Epoch

Learning rate: 0.03

Activation: ReLU

Regularization: L1

Regularization rate: 0.01

Problem type: Classification

## DATA

Which dataset do you want to use?



Ratio of training to test data: 10%

Noise: 50

Batch size: 10

REGENERATE

- What is the effect of very strong regularization?
- What is the difference of L1-norm and L2-norm in terms of the resulting MLP model?

# L2 regularization



Epoch  
000,038

Learning rate  
0.03

Activation  
ReLU

Regularization  
L2

Regularization rate  
0.3

Problem type  
Classification

## DATA

Which dataset do you want to use?



Ratio of training to test data: 40%



Noise: 25



Batch size: 10



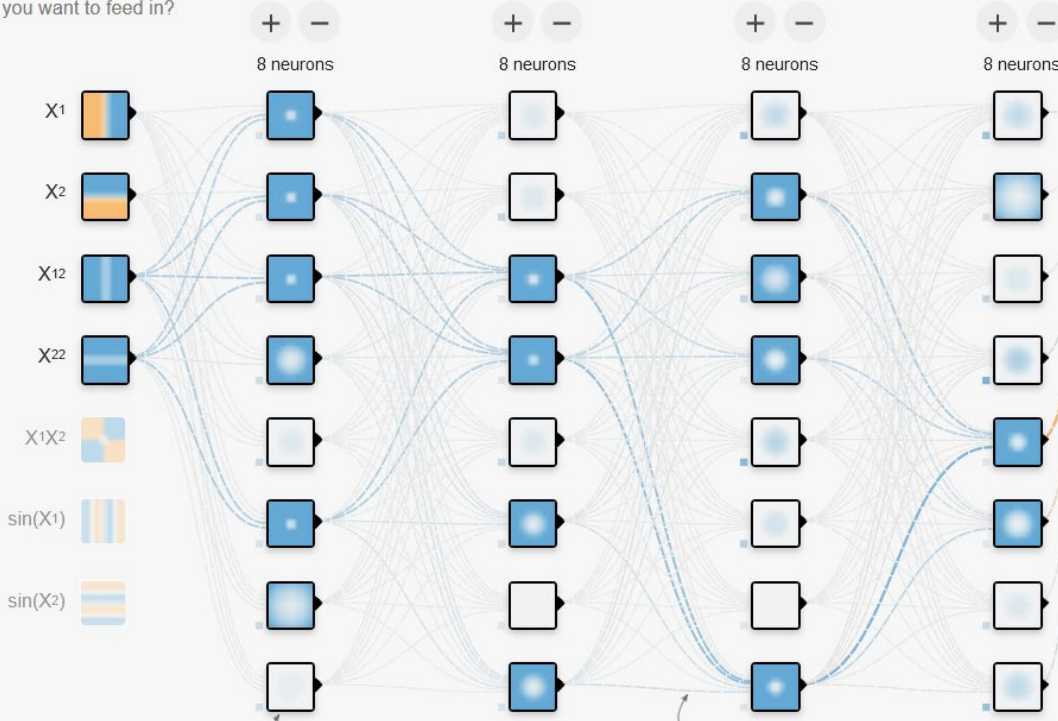
REGENERATE

## FEATURES

Which properties do you want to feed in?

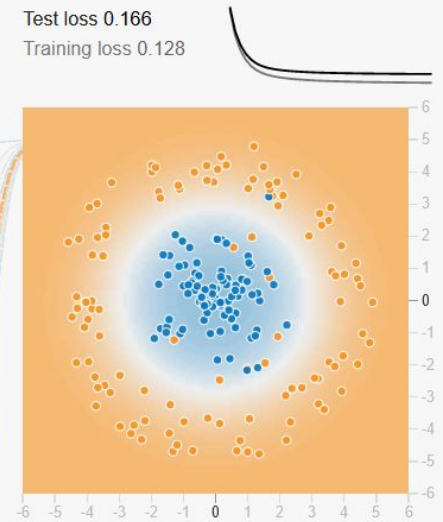
- X1
- X2
- X12
- X22
- X1X2
- sin(X1)
- sin(X2)

## 4 HIDDEN LAYERS



## OUTPUT

Test loss 0.166  
Training loss 0.128



- Show test data
- Discretize output

# L1 regularization

⏪  ⏩ Epoch **000,066** Learning rate **0.03** Activation **ReLU** Regularization **L1** Regularization rate **0.1** Problem type **Classification**

## DATA

Which dataset do you want to use?



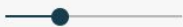
Ratio of training to test data: 40%



Noise: 25



Batch size: 10



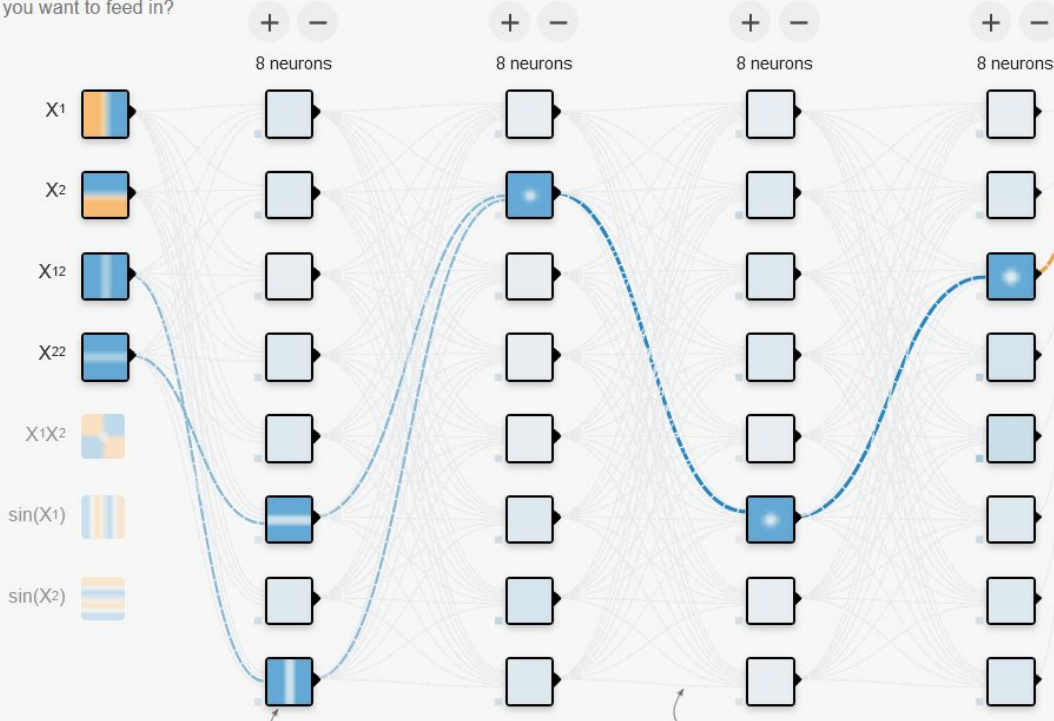
REGENERATE

## FEATURES

Which properties do you want to feed in?

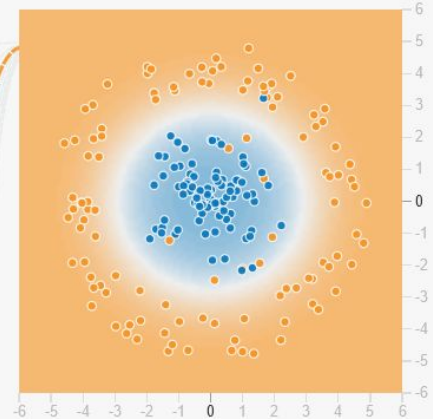
- X1
- X2
- X12
- X22
- X1X2
- sin(X1)
- sin(X2)

## 4 HIDDEN LAYERS



## OUTPUT

Test loss 0.139  
Training loss 0.094



Colors shows data, neuron and weight values.

- Show test data
- Discretize output

# No regularization



Epoch  
000,056

Learning rate  
0.03

Activation  
ReLU

Regularization  
None

Regularization rate  
0

Problem type  
Classification

## DATA

Which dataset do you want to use?



Ratio of training to test data: 40%



Noise: 25



Batch size: 10



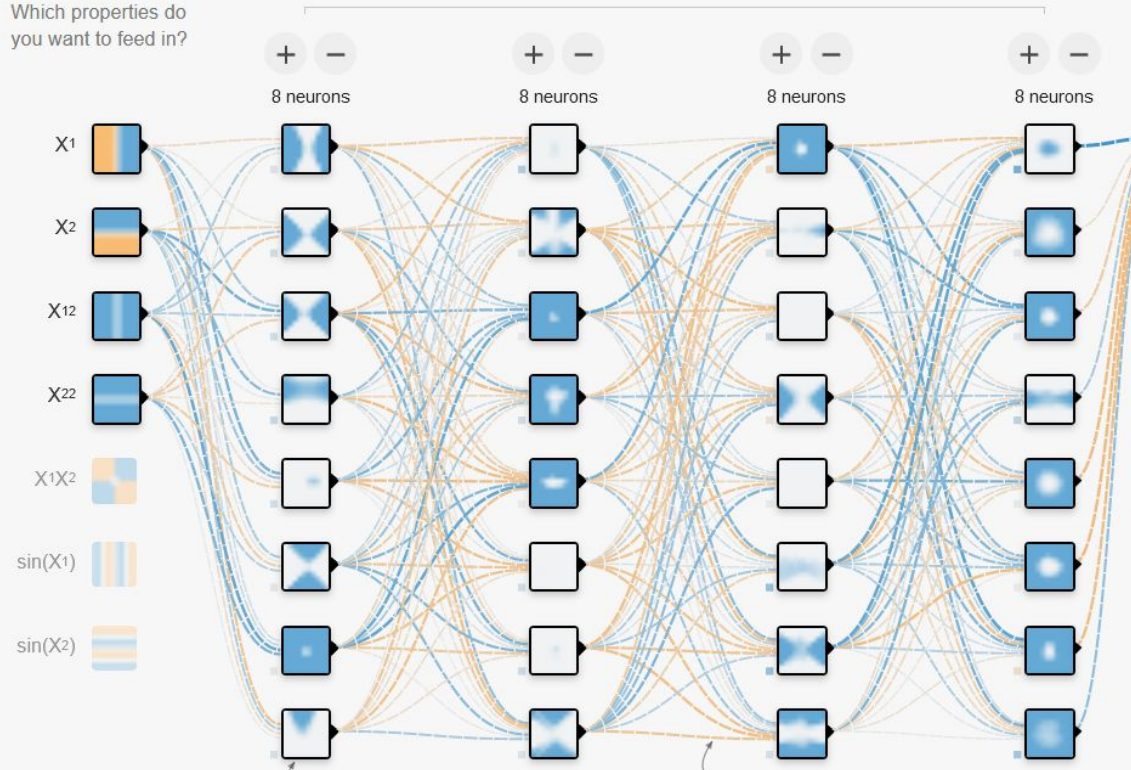
REGENERATE

## FEATURES

Which properties do you want to feed in?

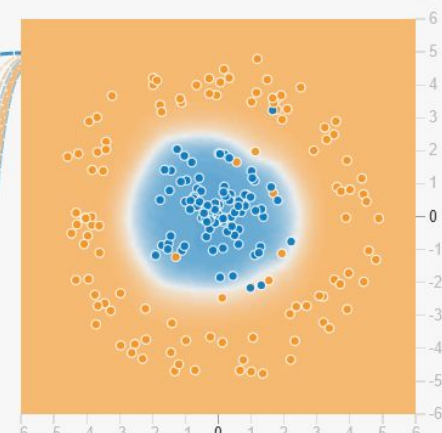
- X1
- X2
- X12
- X22
- X1X2
- sin(X1)
- sin(X2)

## 4 HIDDEN LAYERS



## OUTPUT

Test loss 0.124  
Training loss 0.057



Colors shows data, neuron and weight values.

Show test data    Discretize output

---

# Further things to explore

---

Using the same example (with the ring data):

- Do you need a hidden layer? Why?
- What happens if you use a linear activation function?
- What happens if you use too many hidden layers using a sigmoid activation?
- What happens if the learning rate is too small? And too large?