

NEURAL NETWORKS

Nauman Nayyar

COURSE

PRE-WORK

PRE-WORK REVIEW

- ▶ Understand Logistic Regression and link functions
- ▶ Be familiar with training and testing classifiers and regressors

OPENING

ARTIFICIAL NEURAL NETWORKS

OPENING

- ▶ Neural networks were first studied in the 1940s (!) as a model of biological neural networks
- ▶ Many advances since then have improved the ability to train and apply neural networks
- ▶ Good for both classification and regression but difficult to interpret model behaviors
- ▶ Deep learning in the past few years has been highly successful for otherwise difficult problems

OPENING

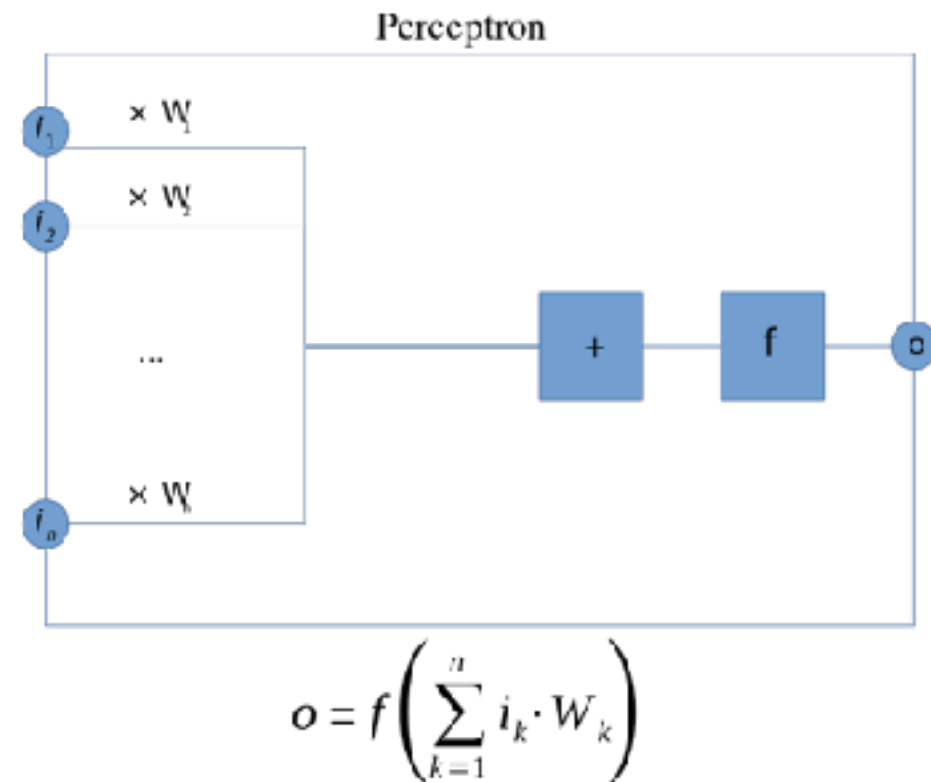
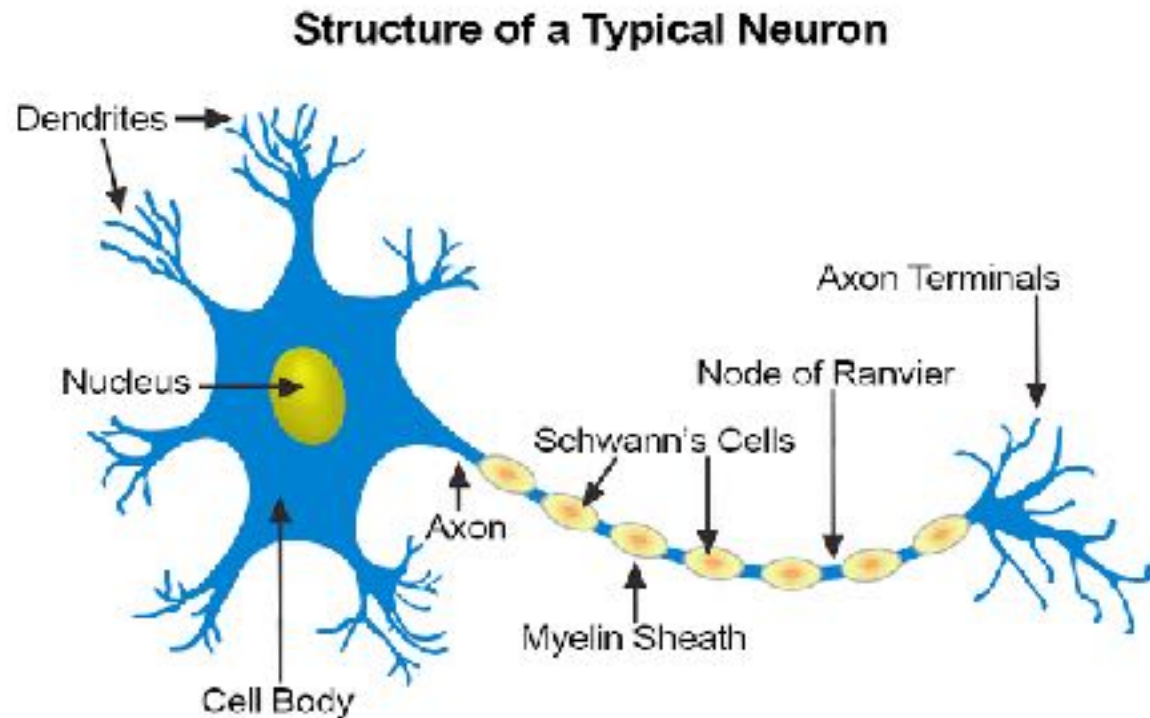
- ▶ Today we will focus on types of neural networks and their applications, and skip some of the more technical details
- ▶ Specifically we'll skip training neural networks -- there are many methods in various situations and the details can be tedious (but not particularly difficult)
- ▶ Methods include backpropagation, gradient descent, and Hessian-free learning

INTRODUCTION

PERCEPTRON

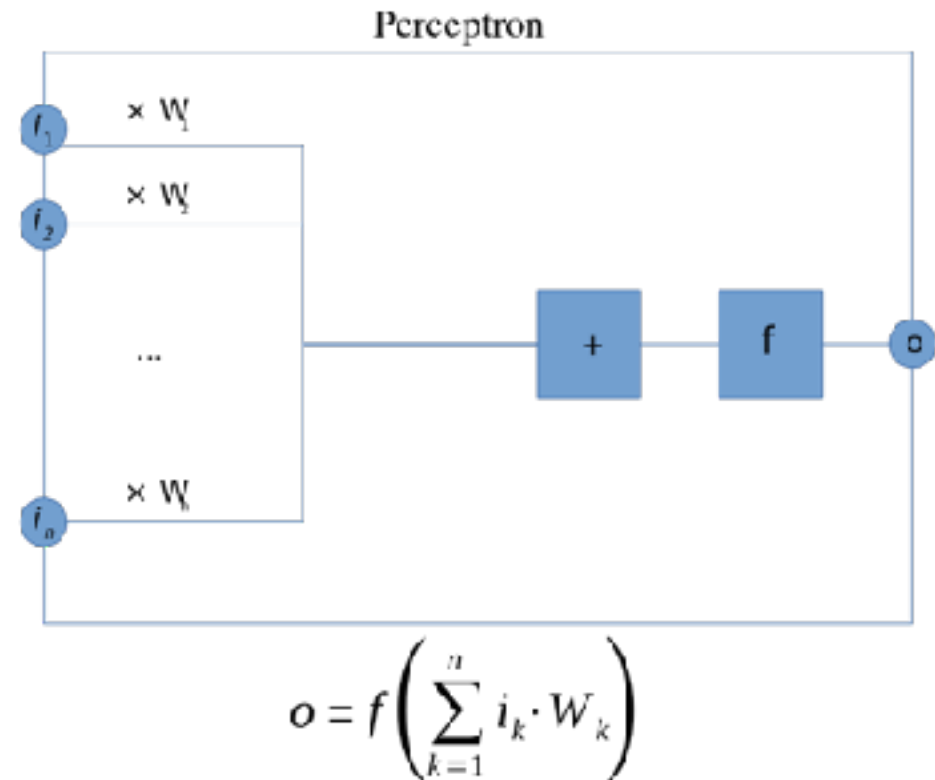
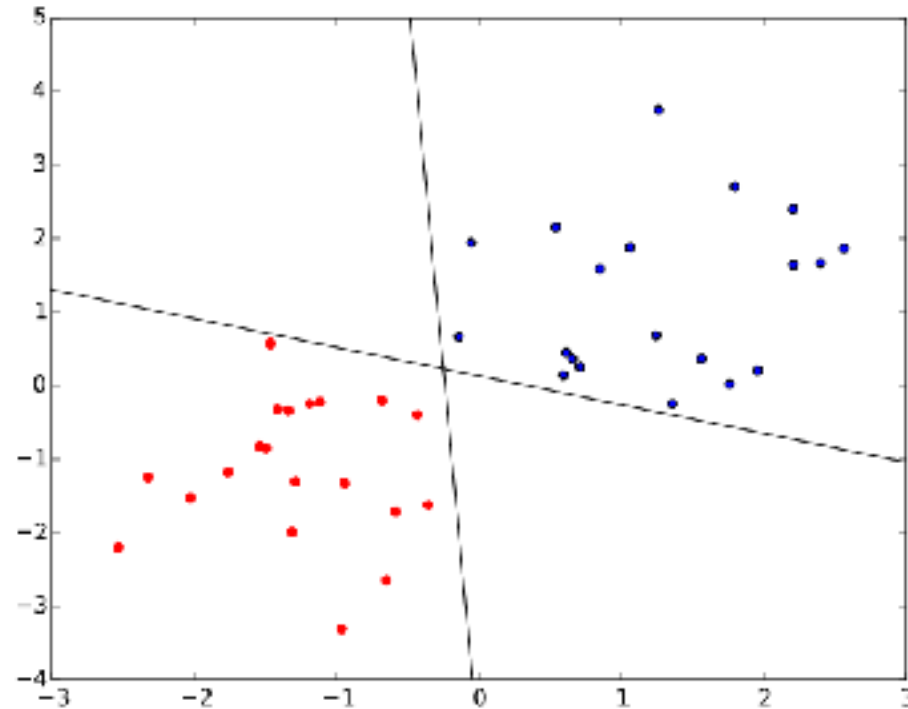
PERCEPTRON

- ▶ Perceptrons are the simplest example of a neural network
- ▶ The idea is to emulate a single neuron



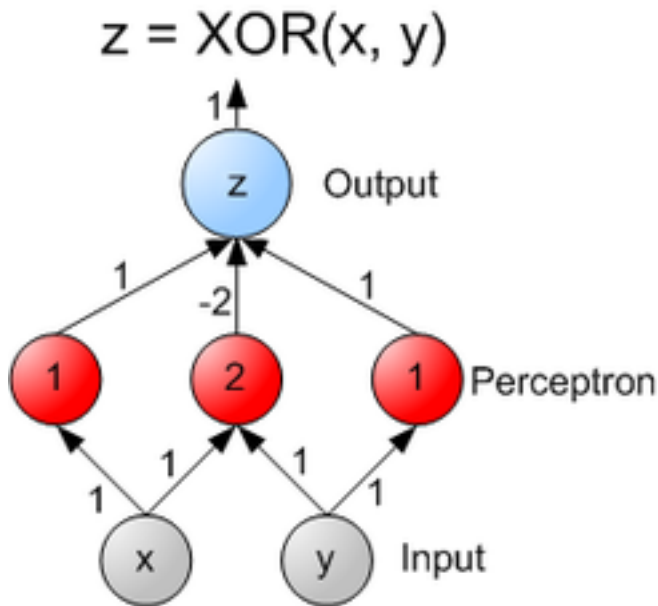
PERCEPTRON

- ▶ Perceptrons are the simplest example of a neural network
- ▶ Given n inputs and an activation or link function f
- ▶ The perceptron computes a linear separating curve



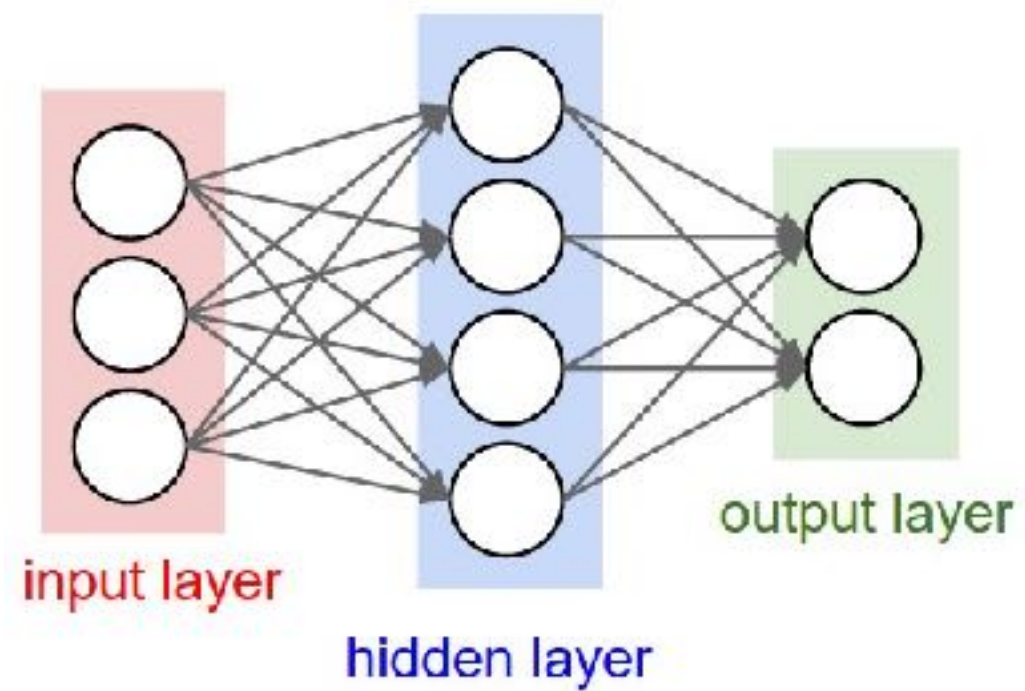
PERCEPTRON

- ▶ Common [activation functions](#) are linear, logistic, tanh, and [softmax](#)
- ▶ We'll see shortly that some are better for classification, some for regression
- ▶ Perceptrons can be combined into multilayer perceptrons or feed-forward network



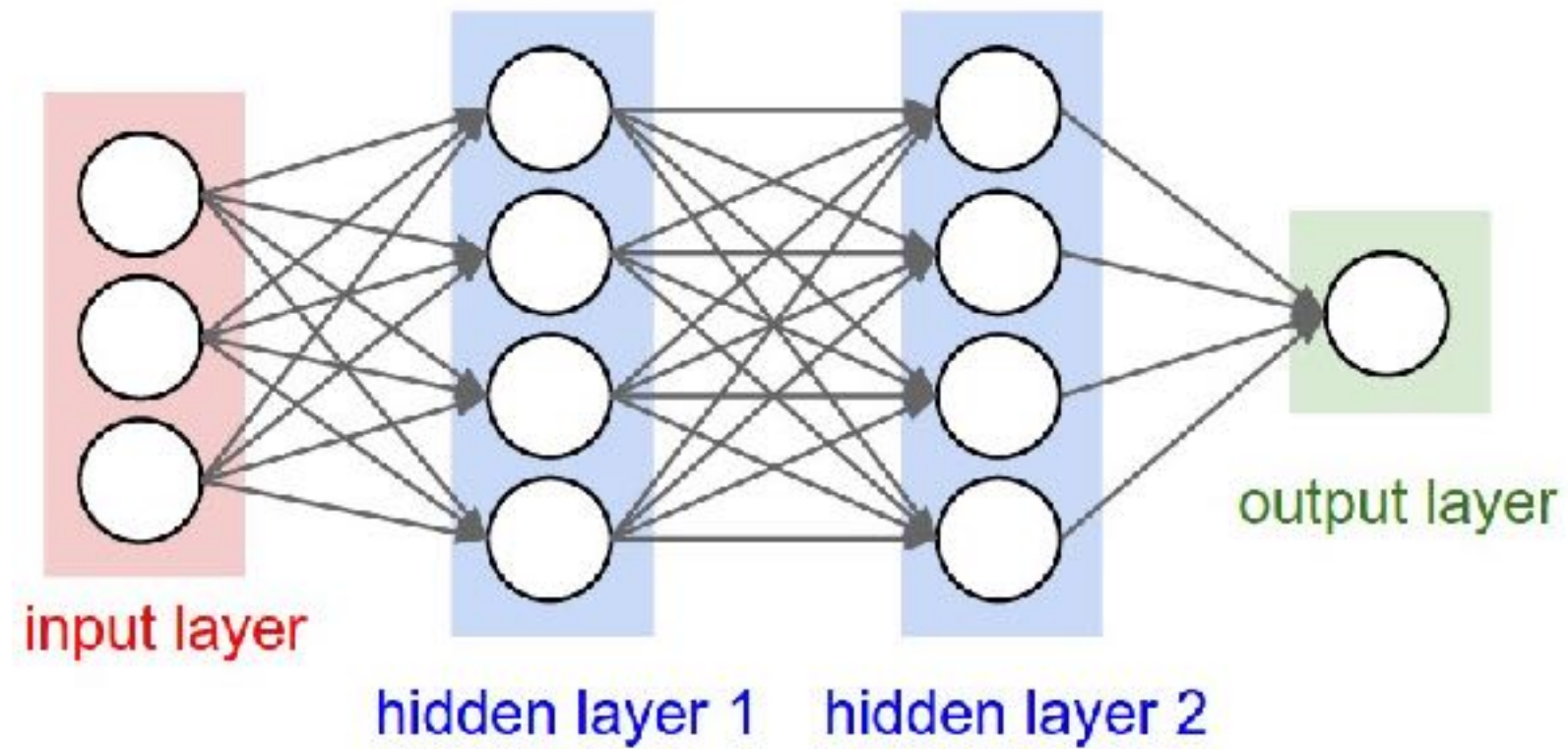
FEED FORWARD NN

► [Source](#)



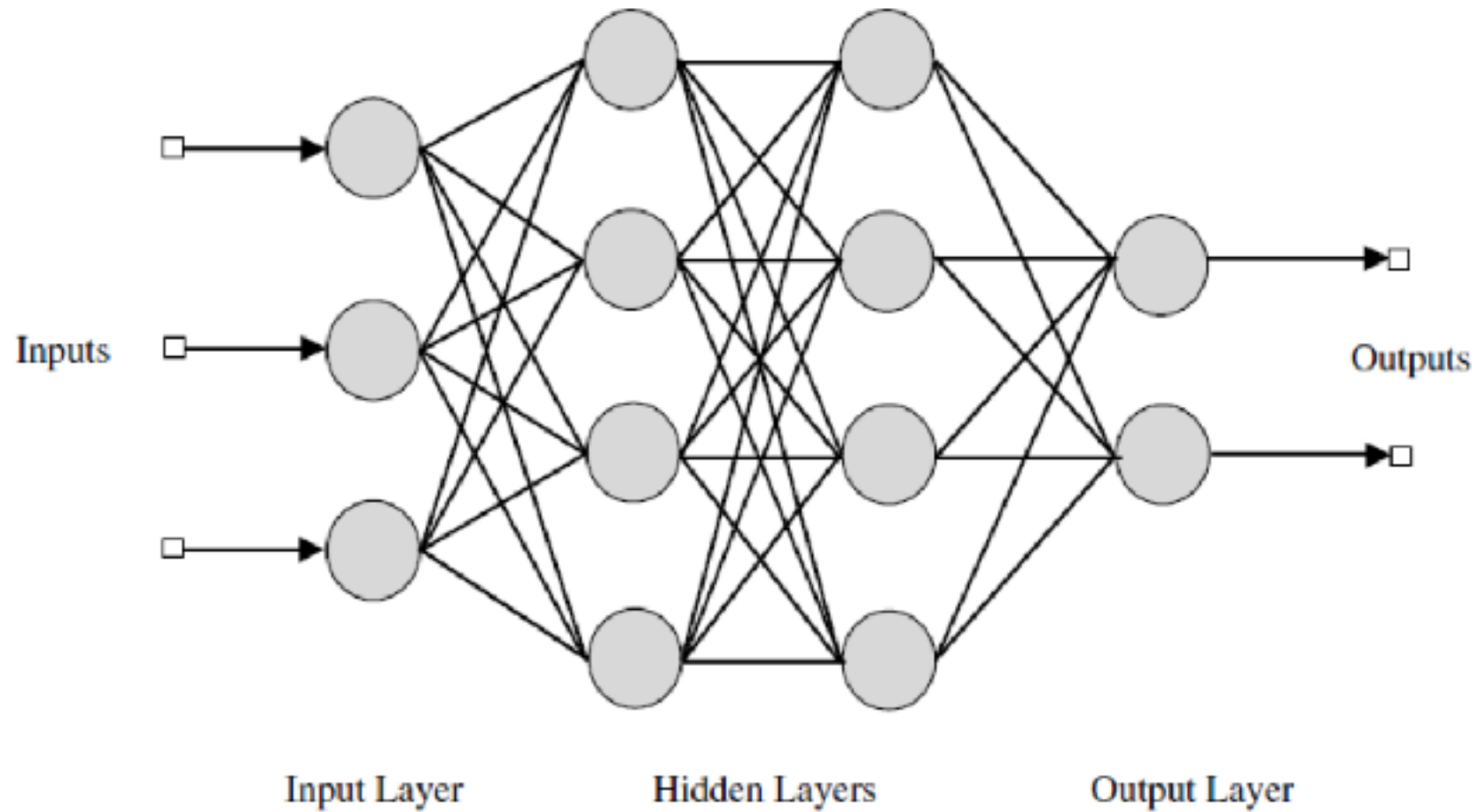
FEED FORWARD NN

► [Source](#)



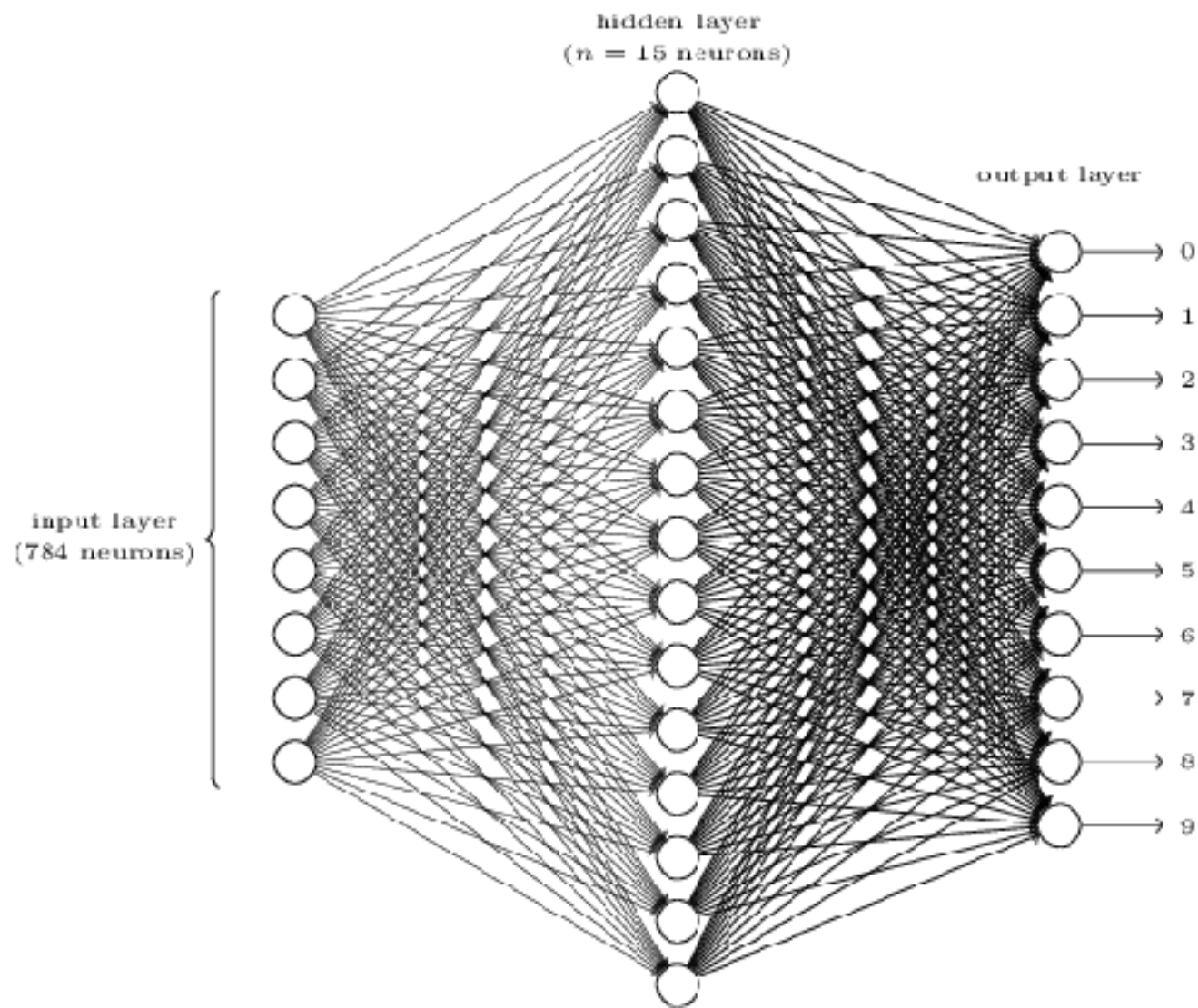
FEED FORWARD NN

► [Source](#)



FEED FORWARD NN

► [Source](#)



FEED FORWARD NN

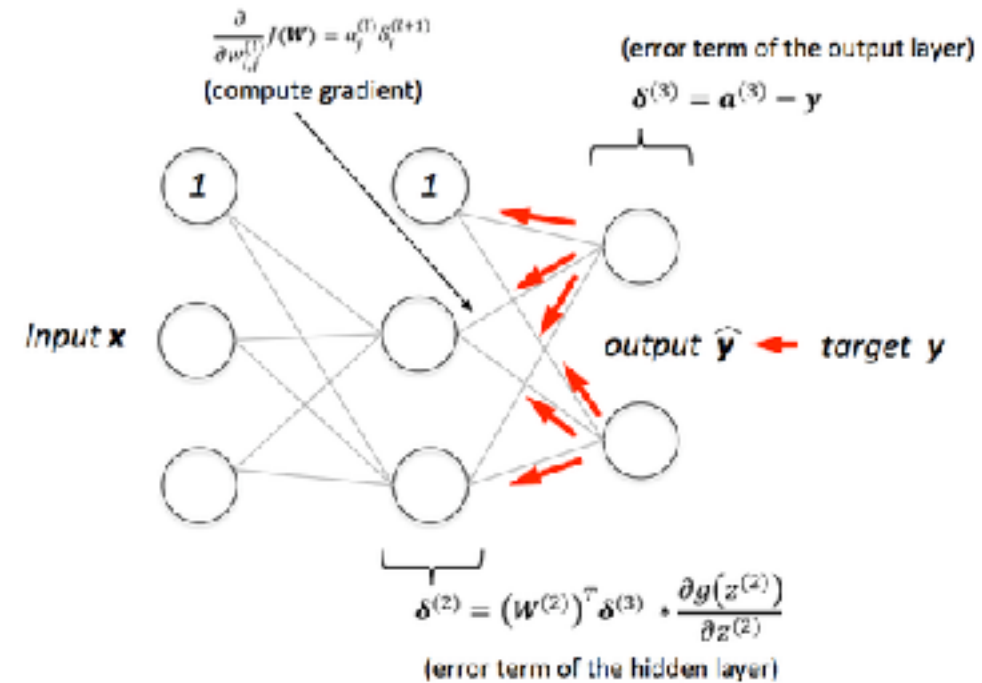
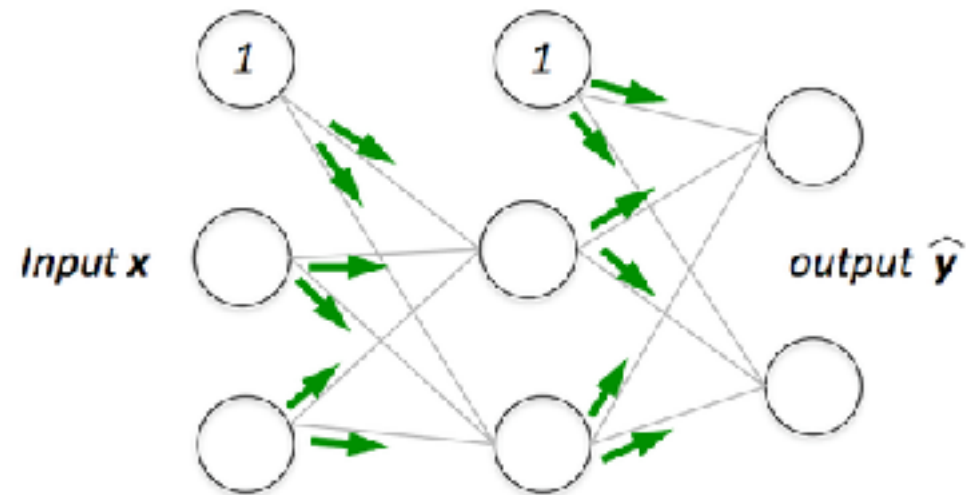
- ▶ Typically we use
 - ▶ Input layers have no activation. Simply inputs!!
 - ▶ Linear layers for regression output
 - ▶ Logistic or Tanh for binary output
 - ▶ Softmax for n-class output (yields probabilities)

GUIDED PRACTICE

TRAINING

TRAINING

- ▶ Feed forward neural networks can be trained with [backpropagation](#)
- ▶ [Source](#)



TRAINING

- ▶ Key Parameters

- ▶ Learning Rate (gradient descent for training)
- ▶ Epochs: number of backpropagation passes (over entire dataset)
- ▶ Batch size: how many training points used at a time to update weights

- ▶ Model others behaves as usual with

- ▶ `model.predict`
- ▶ `model.predict_classes`

TRAINING

► Tips

- If the error jumps around per epoch, decrease the learning rate
- Taking too long to train: use higher learning rate or batch_size
- High error after convergence?
 - More hidden layers / neurons
 - Normalize data or use PCA

INTRODUCTION

UNIVERSAL APPROXIMATION THEORY

UNIVERSAL APPROXIMATION

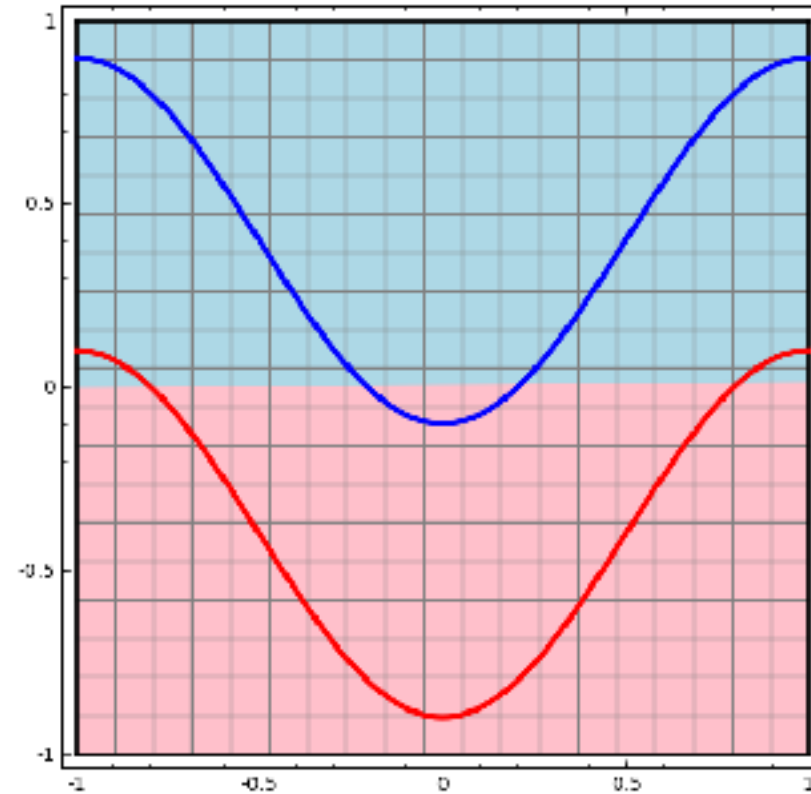
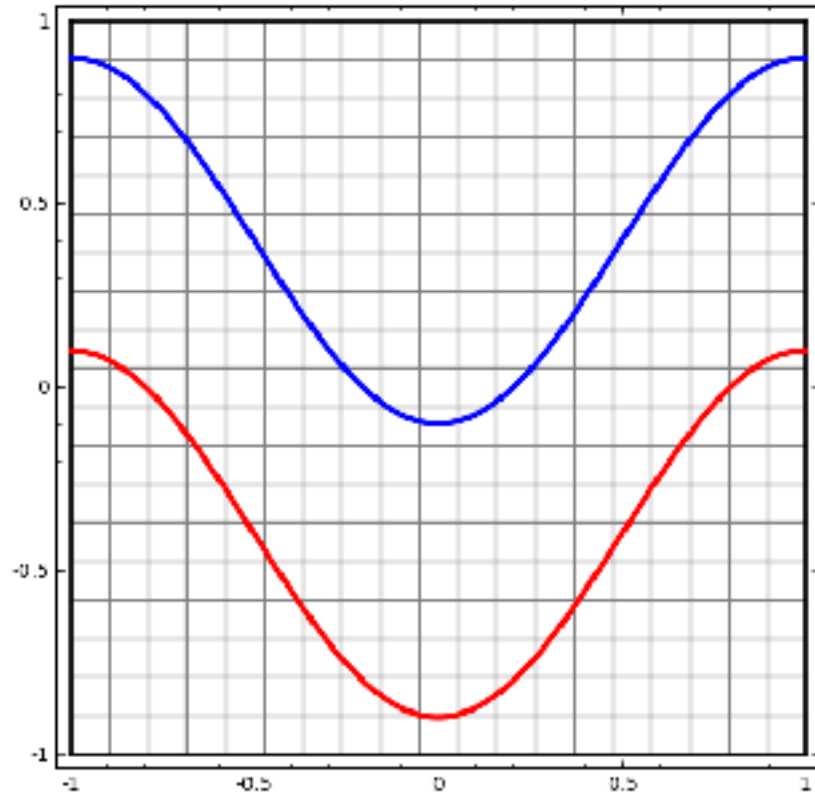
- ▶ One major reason that neural networks are useful is the [Universal Approximation Theorem](#)
- ▶ The result basically says that many real vector-valued functions can be approximated arbitrarily well with *some* feed-forward neural network
- ▶ This is why neural networks are useful for regression -- given enough data and the right network structure they can fit many common data sets

CLASSIFICATION

CLASSIFICATION WITH NEURAL NETWORKS

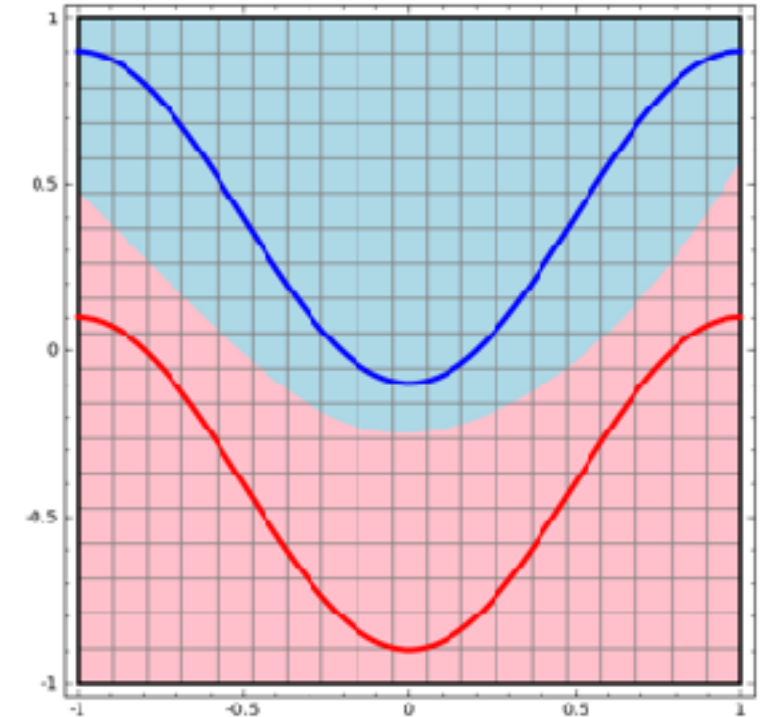
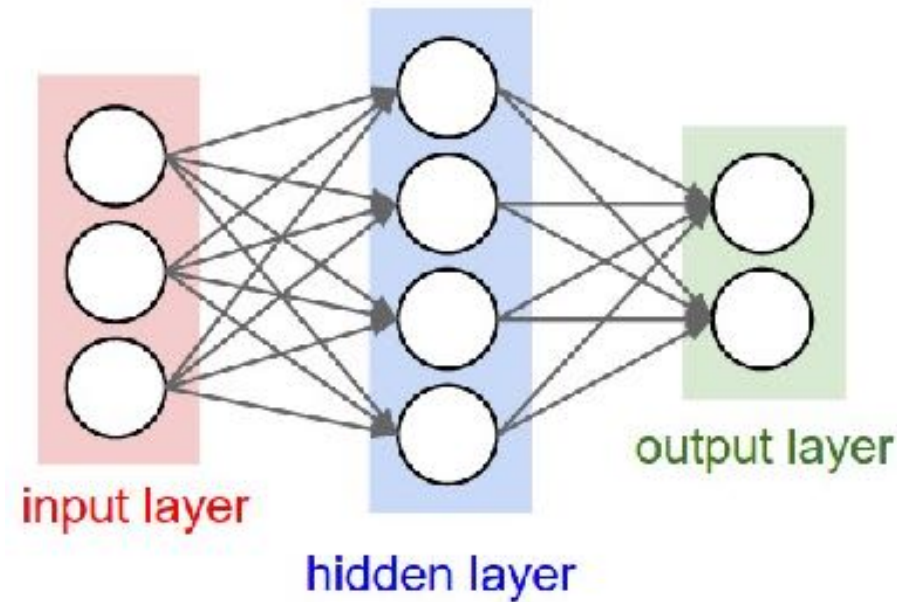
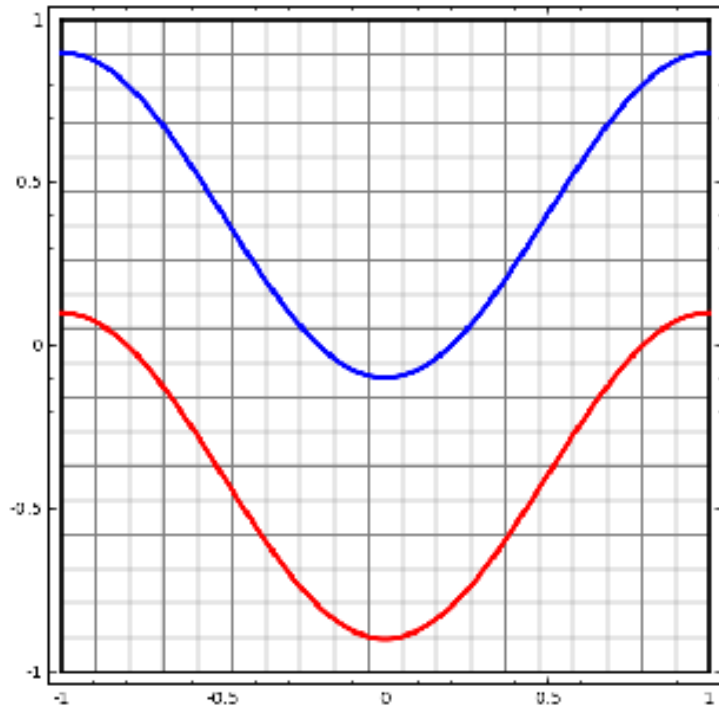
CLASSIFICATION

- ▶ Neural Networks are also extremely useful for classification ([source](#))
- ▶ No hidden layers:



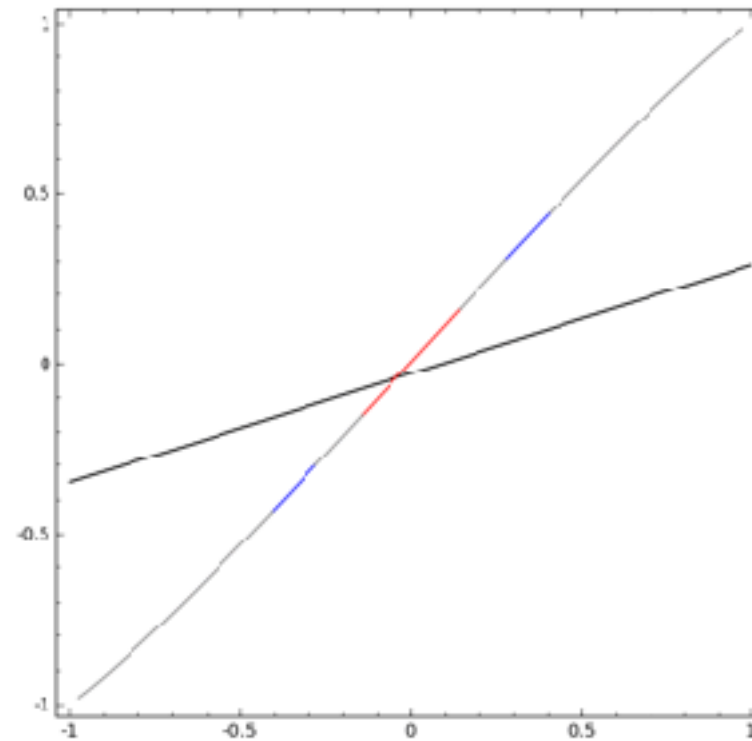
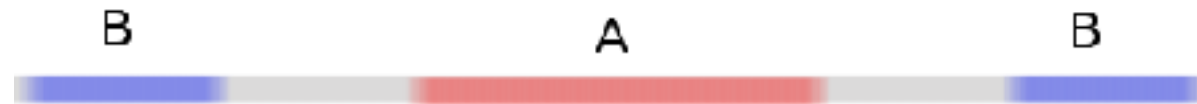
CLASSIFICATION

- ▶ Neural Networks are also extremely useful for classification ([source](#))
- ▶ One hidden layer:



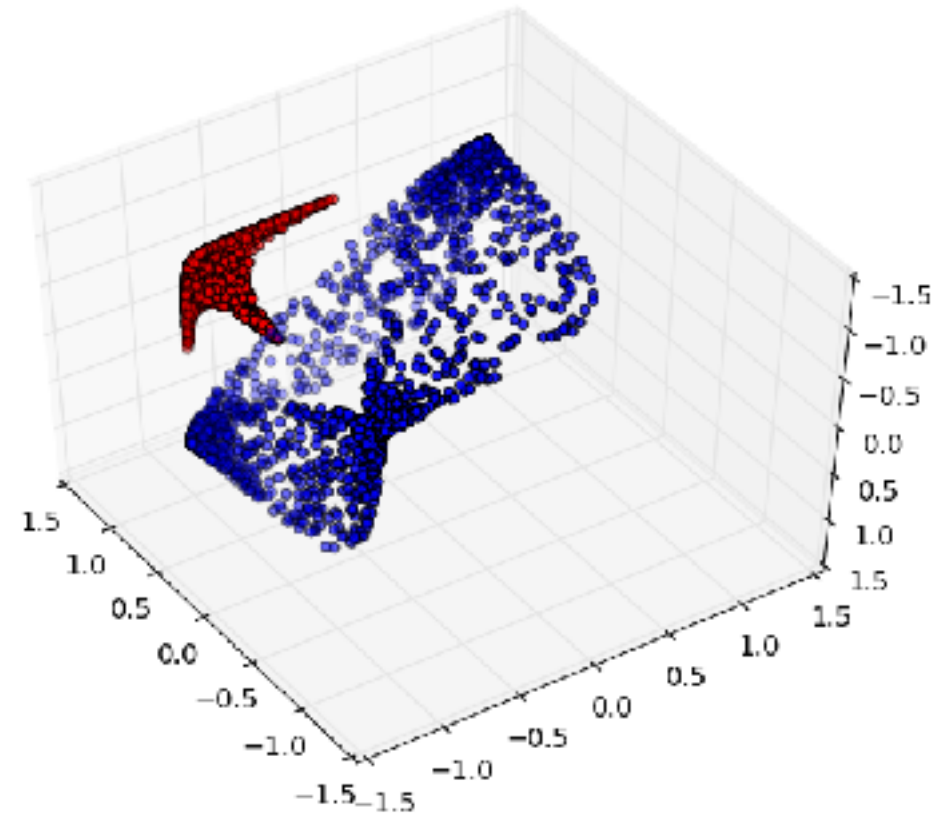
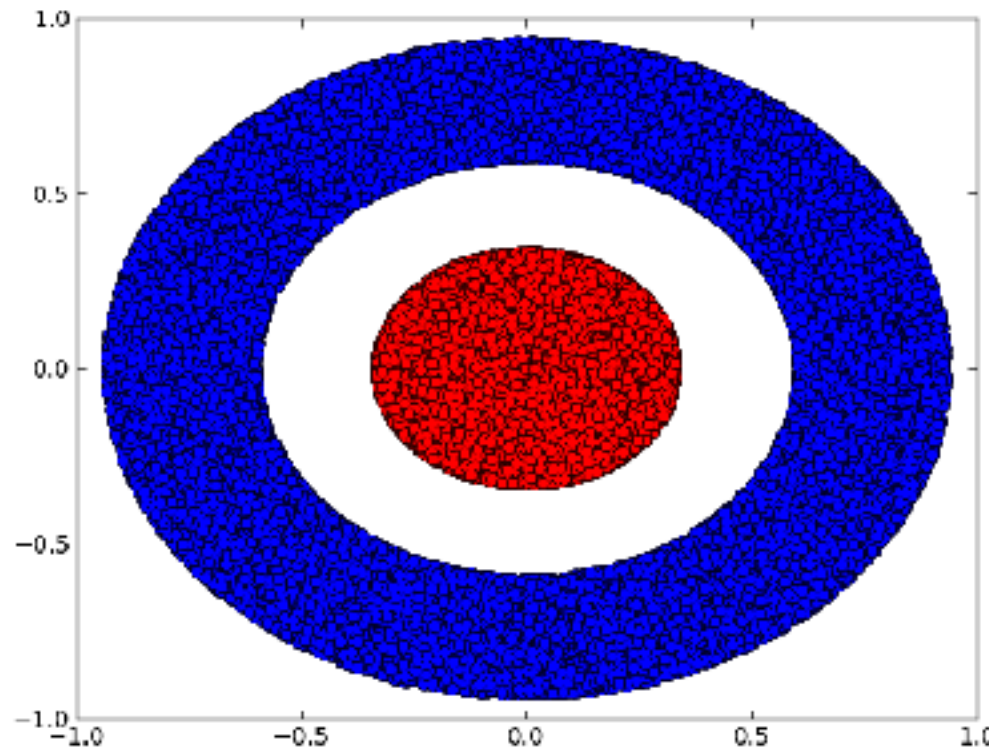
CLASSIFICATION

- Neural Networks are also extremely useful for classification ([source](#))



CLASSIFICATION

- Neural Networks are also extremely useful for classification ([source](#))



CLASSIFICATION

- ▶ The neural network transforms the data topologically (no tears or breaks) and then separates the data with a hyperplane
- ▶ NNs are capable of handling difficult data sets, including:
 - ▶ Image processing: recognizing hand-written characters
 - ▶ Image compression
 - ▶ Financial forecasting
 - ▶ Many others

ACTIVITY: KNOWLEDGE CHECK



EXERCISE

ANSWER THE FOLLOWING QUESTIONS

1. Let's practice using [neural networks for classification](#). For each of the four datasets, experiment with the number of layers and neurons to find the best model
2. Also take a look at this [visualization](#)

DELIVERABLE

Answers to the above questions

GUIDED PRACTICE

NEURAL NETWORKS IN PYTHON

NN IN PYTHON

- There are many NN libraries for python and other languages
- Python
 - Theano
 - Keras
 - Lasagne
 - TensorFlow
 - Scikit Learn support for NN coming in 0.18
- Lua
 - Torch
- Some of these libraries utilize GPUs for (much) faster training

NN IN PYTHON

- ▶ Let's look at some examples in Keras
 - ▶ Regression
 - ▶ Classification

GUIDED PRACTICE

DESIGNING NEURAL NETWORKS

NN IN PYTHON

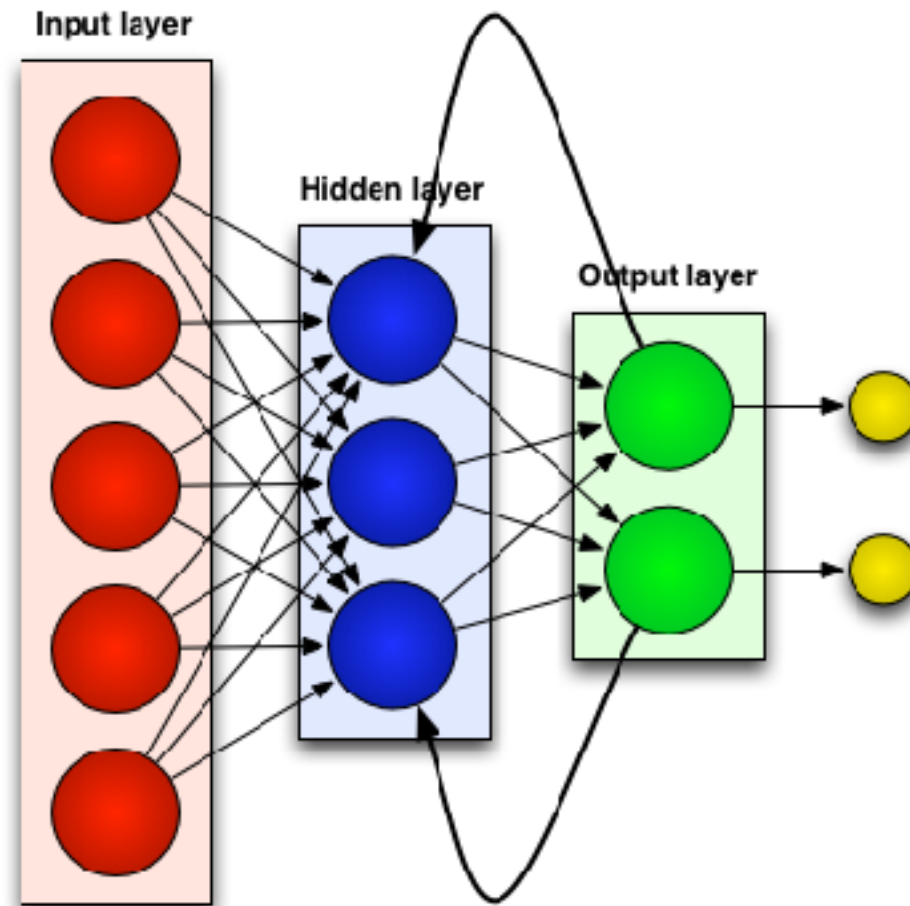
- Network design is a hard problem
 - Experience helps
 - Evolutionary algorithms are [useful](#) for [design](#)
 - Nice (free) book [available](#)
- Tips
 - Number of input layer nodes = number of features (obviously!)
 - Number of output layers nodes
 - 1 if regression
 - 1 if binary classification
 - N (number of classes) if softmax is used as activation for multi-label classification
 - Number of hidden layers
 - If data is linearly separable, no hidden layer needed
 - If not, one suffices in a lot of cases. More make the number of parameters exponentially fewer, but are more difficult to train
 - For more details, look at <http://www.faqs.org/faqs/ai-faq/neural-nets/part1/preamble.html>
 - Number of neurons
 - Between input and output nodes, e.g. mean
- Additional info: <https://stats.stackexchange.com/questions/181/how-to-choose-the-number-of-hidden-layers-and-nodes-in-a-feedforward-neural-netw>

RECURRENT NN

RECURRENT NEURAL NETWORKS

RECURRENT NEURAL NETWORKS

- Recurrent Neural Networks contain loops ([source](#))



RECURRENT NEURAL NETWORKS

- ▶ Recurrent Neural Networks contain loops
- ▶ This implements feedback and gives neural networks “memory” or context
- ▶ Particularly good for predicting sequences, translating text, recognizing objects in images, speech translation
- ▶ Commonly referred to as **deep learning**, involving both feature extraction and modeling
- ▶ [Nice intro here](#)

RECURRENT NEURAL NETWORKS

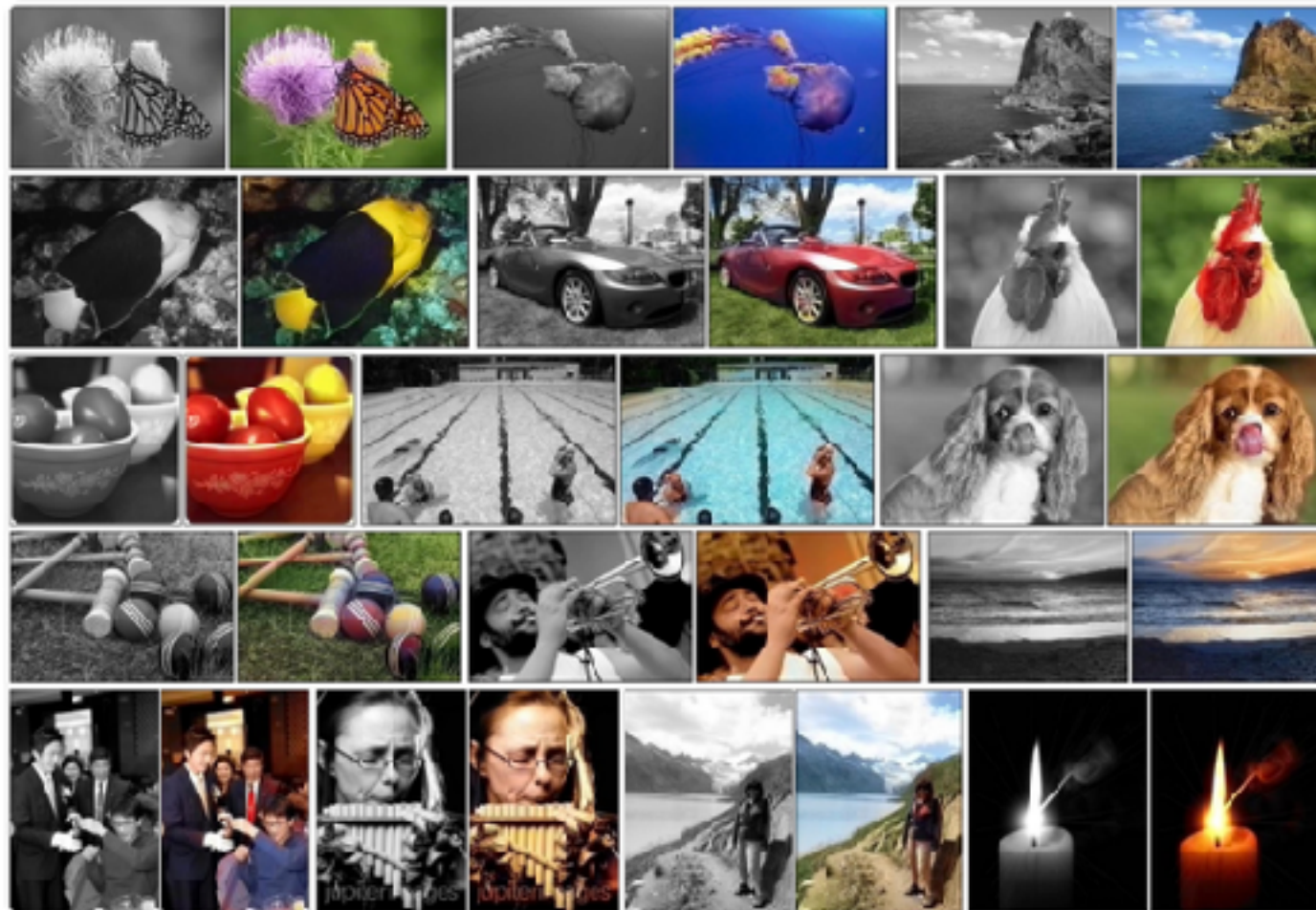
► [RNN font analysis](#)



A	B	C	D	E	F	G	H
I	J	K	L	M	N	O	P
Q	R	S	T	U	V	W	X
Y	Z	a	b	c	d	e	f
g	h	i	j	k	l	m	n
o	p	q	r	s	t	u	v
w	x	y	z	0	1	2	3
4	5	6	7	8	9		

RECURRENT NEURAL NETWORKS

► Automatic Colorization with CNN



RECURRENT NEURAL NETWORKS

- ▶ [RNN font analysis](#)
- ▶ [Automatic Colorization](#) with CNN
- ▶ Automatic translation
- ▶ [Deep Learning Applications](#)

CONCLUSION

TOPIC REVIEW

CONCLUSION: Neural Networks

Pros:

- Flexible
- Good for a variety of tasks
- Good for many types of data

Cons:

- Can require a lot of data
- Training may be slow
- Many parameters to tune
- Many layer types and activations
- Black Box model

CONCLUSION

- ▶ Many [more examples](#) for Keras available
- ▶ Recommended articles: [Convolutional NN](#),
- ▶ Advanced machine learning methods you should explore include Bayesian methods and deep learning

BOOSTING

BOOSTING AND XGBOOST

BOOSTING

- ▶ Boosting: combining multiple ‘weak learners’ to get a ‘strong learner’
- ▶ Weak learners: Model only slightly correlated with true model, but better than random
- ▶ Strong learners: Model can get arbitrarily close to true model
- ▶ Why weak learners? They are simple. E.g. a one-level decision tree
- ▶ Protip: Boosting reduces bias error!

BOOSTING

- ▶ First such algorithm: [AdaBoost](#)
- ▶ AdaBoost is a generalized methodology
- ▶ It can be applied to many different learning models
- ▶ Another such algorithm: [XGBoost](#)
- ▶ XGBoost applies boosting to decision trees (**how is it different from Random Forests?**)

ACTIVITY: KNOWLEDGE CHECK



EXERCISE

ANSWER THE FOLLOWING QUESTIONS

1. Let's practice using boosting with XGBoost. Install XGBoost with the following command:
`conda install -c aterrel xgboost=0.4.0`
2. Work through <http://machinelearningmastery.com/develop-first-xgboost-model-python-scikit-learn/>

DELIVERABLE

Answers to the above questions

LESSON

EXIT TICKET

DON'T FORGET TO FILL OUT YOUR EXIT TICKET

<https://goo.gl/forms/ZF7B45rp0zkchaRJ2>