SIGGRAPH THINK BEYOND

2020 S2020.SIGGRAPH.ORG

WALT DISNEY ANIMATION STUDIOS

JUNE 2020 WEBINAR

Hands-on Workshop: Machine Learning and Neural Networks

SIGGRAPHNOW

RAJESH SHARMA

SOFTWARE ENGINEER

Walt Disney Animation Studios

1

Machine Learning

Rajesh Sharma

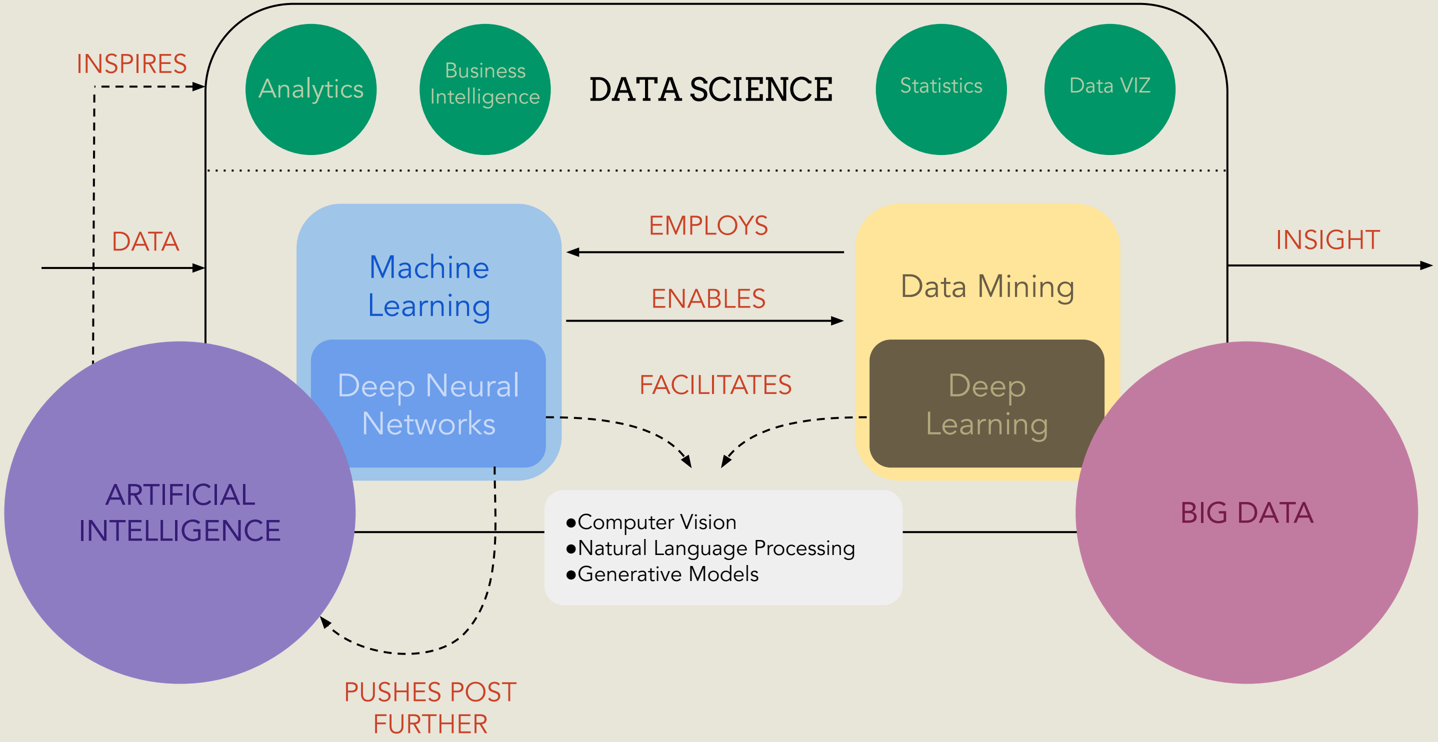
Today

- Introduction
- What is Machine Learning?
- What are Neural Networks?
- How do I ‘solve’ a neural network?
- What are Autoencoders?
- What is CNN?

Housekeeping

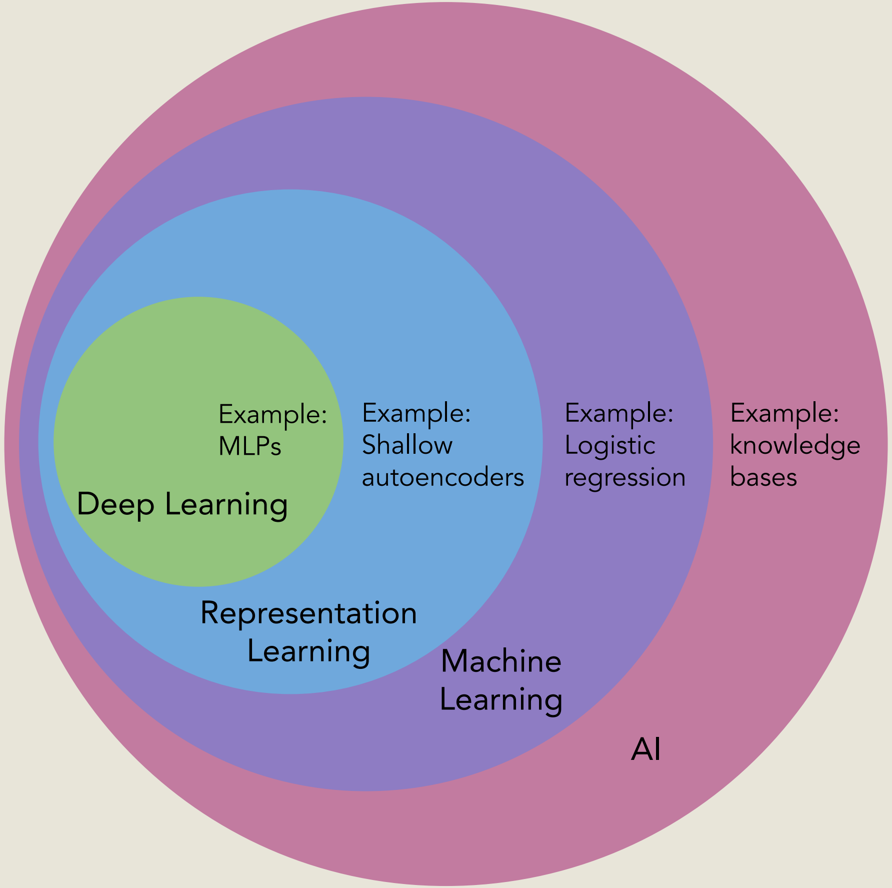
- Link to today's slides and Colab notebooks:
  - Log in to your google drive
  - Find the shared folder 'Disney Machine Learning Webinar'
- Use the 'Ask a question' feature
- After the lecture: @xarmalarma, #siggraphNOW

The BIG Picture



(c) Goodfellow-et-al-2016, Deep Learning, Ian Goodfellow and Yoshua Bengio and Aaron Courville, MIT Press, http://www.deeplearningbook.org, 2016

AI & Machine Learning



(c) Goodfellow-et-al-2016, Deep Learning, Ian Goodfellow and Yoshua Bengio and Aaron Courville, MIT Press, http://www.deeplearningbook.org, 2016

What problem are you solving?

Question	AI/ML Task	Healthcare	Retail	Finance
Yes or **No**?	Detection	Cancer Detection	Targeted Ads	Cybersecurity
What **type**?	*Classification*	Image Classification	Basket Analysis	Credit Scoring
What **size**?	Segmentation	Tumor size	Customer Types	Risk Analysis
What **result**?	*Prediction*	Survivability	Sentiment/Behavior	Fraud Detection
What **action**?	Recommendation	Therapy	Recommendation	Fast Trading

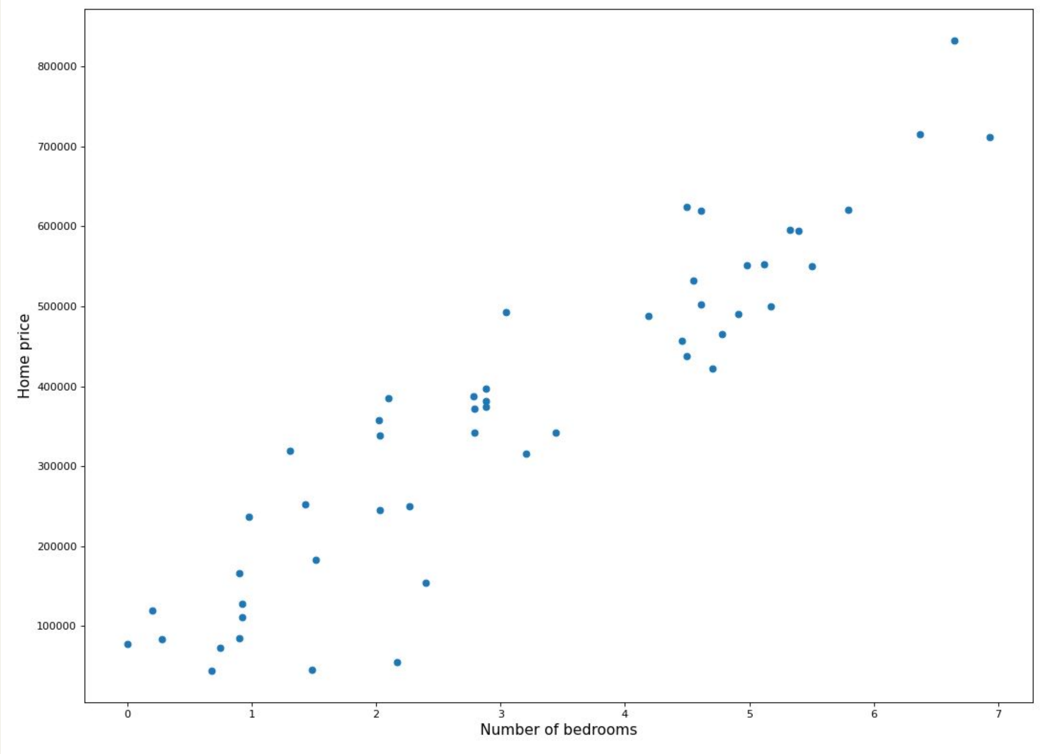
Not a solution to every type of problem

When NOT to use	When to use
✗ Can use computations, or algorithms or simple rules that can be programmed	✓ Problem cannot be solved using rule-based solutions
	✓ Model is complex or has too many factors
	✓ Need to scale to large number of inputs or factors

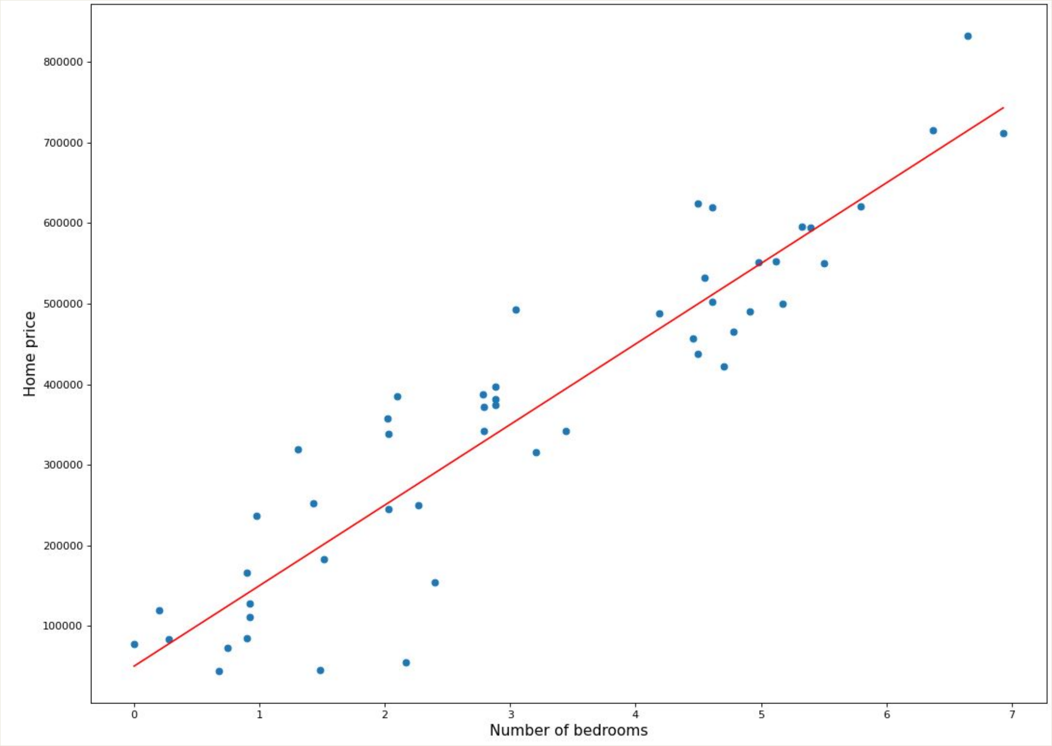
How

- FRAMING: What is observed & what answer you want to predict
- DATA COLLECTION: Collect, clean, and prepare data
- DATA ANALYSIS: Visualize & analyze the data
- FEATURE PROCESSING: Transform raw data for better predictive input
- MODEL BUILDING: Design and build the learning algorithm
- TRAINING: Feed data to the model and evaluate the quality of the models
- PREDICTION: Use model to generate predictions for new data instances

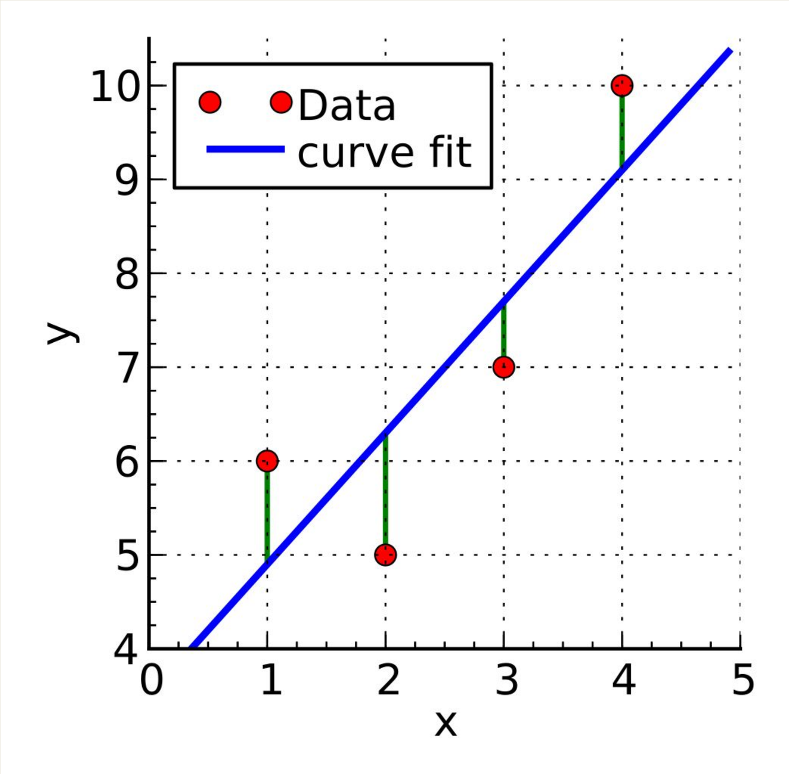
Example (Linear Regression)



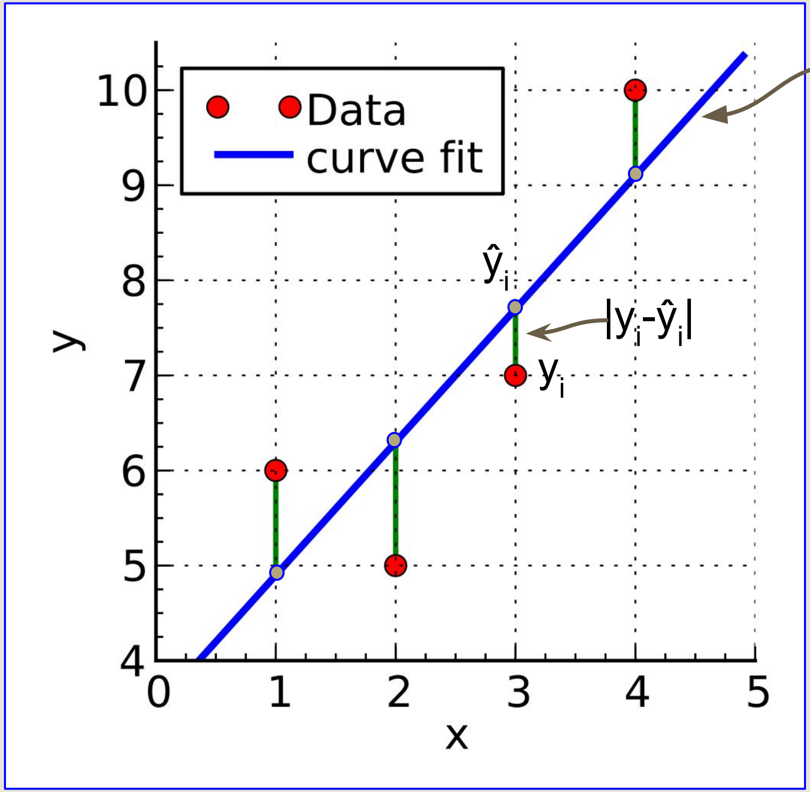
Example (Linear Regression)



Example (Regression)



Example (Regression) - Sum of least squares



Prediction: $\hat{y} = ax + b$

Actual: y_i

Error: $|y_i - \hat{y}_i|$

Total Squared Error:

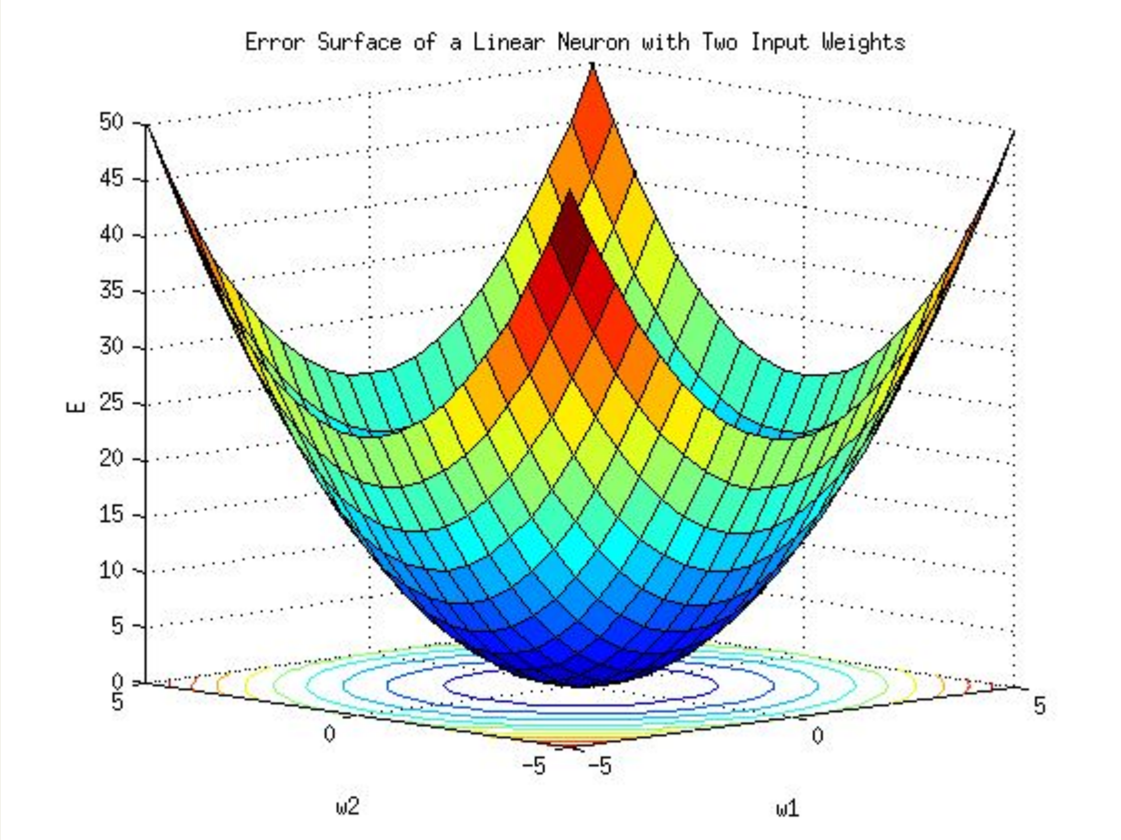
$\sum (y_i - \hat{y}_i)^2$, for i=(1, n)

Minimize Total Squared Error:

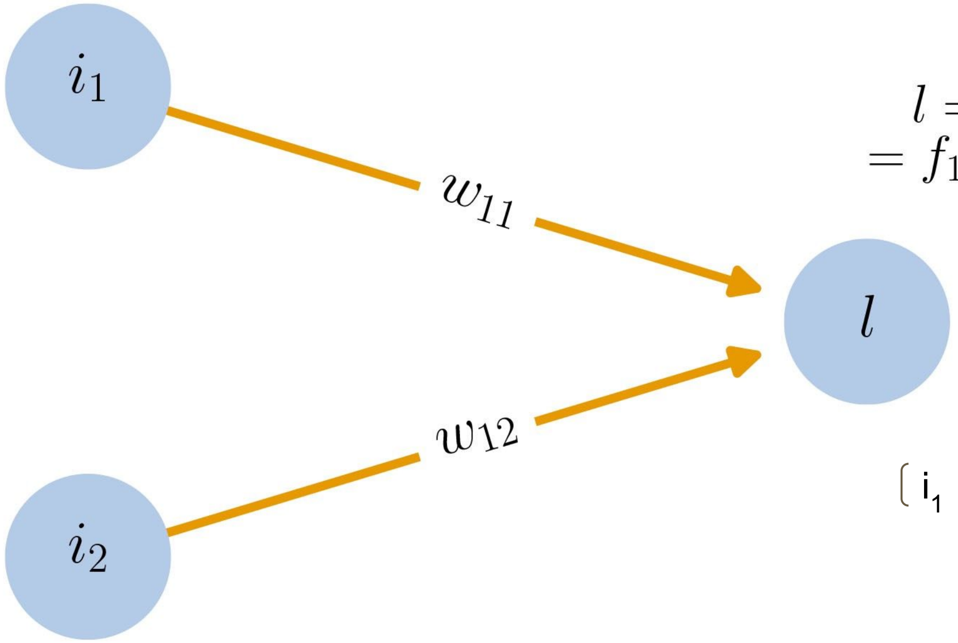
$E(a,b) = \sum(y_i - ax_i - b)^2$

(a,*b*) are the parameters (weights)

Regression - Minimize Error (Cost) via Gradient Descent



Linear function as a Network & a Matrix op



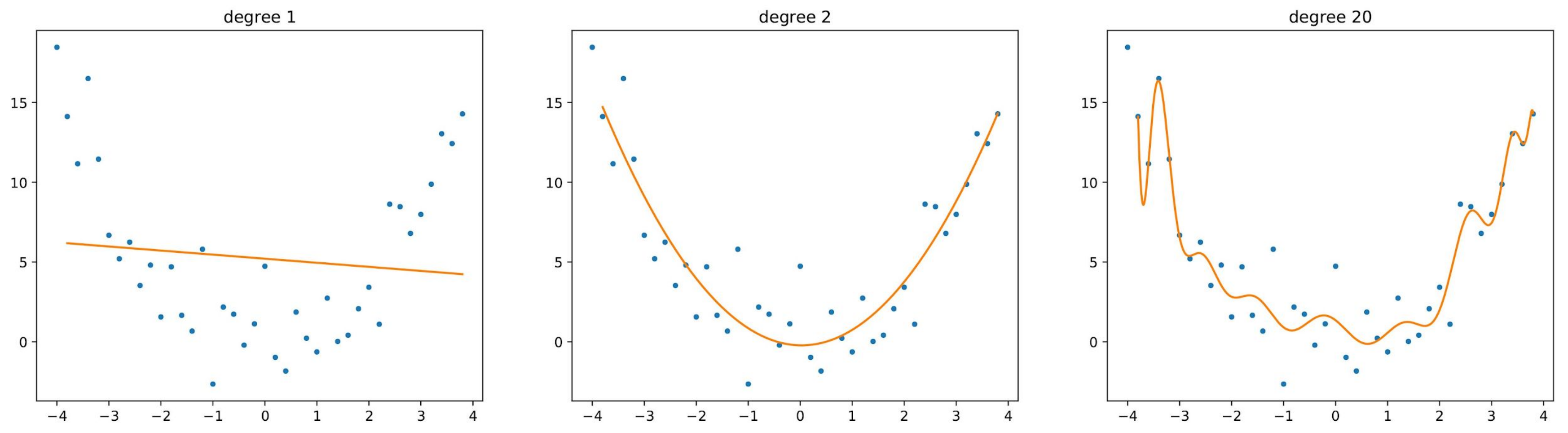
$$l = w_{11} \times i_1 + w_{12} \times i_2$$
$$= f_1(i_1, i_2)$$

Line: y = a + b*x

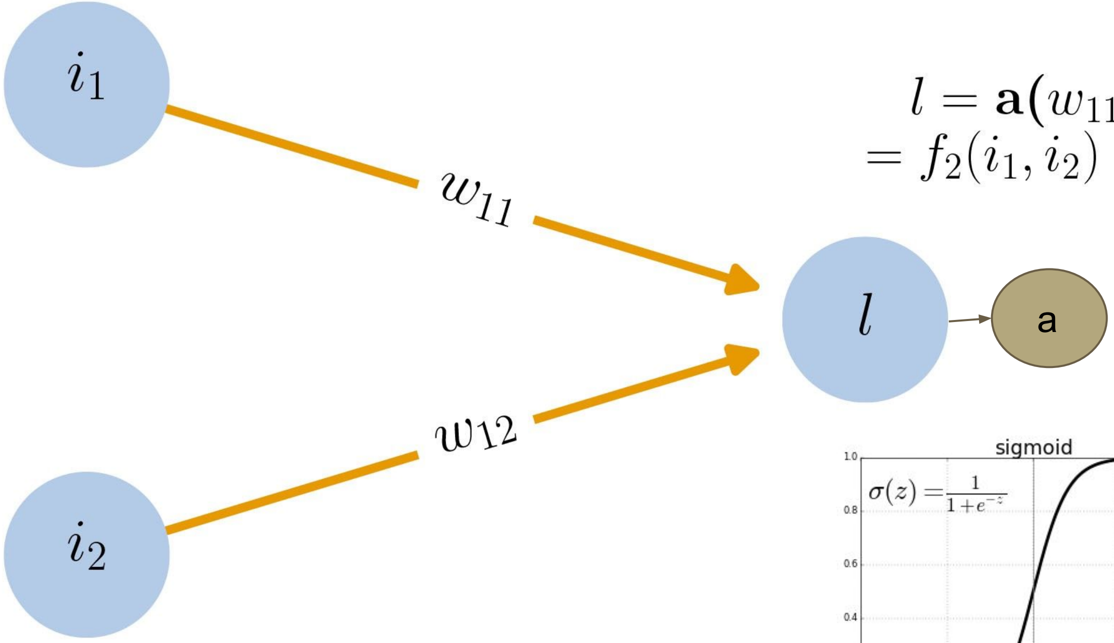
i_1=x, i_2=1, w_{11}=b, w_{12}=a

$$\begin{pmatrix} i_1 & i_2 \end{pmatrix} \cdot \begin{pmatrix} w_{11} \\ w_{12} \end{pmatrix}$$

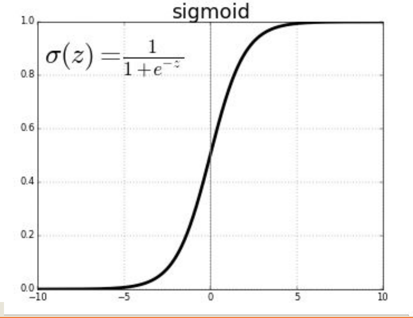
Example

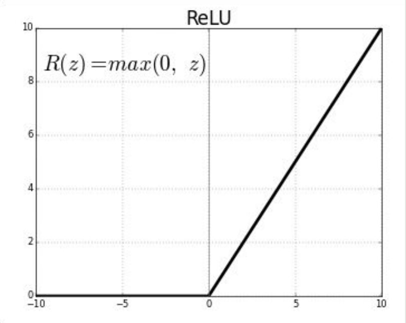


Adding non-linearity via an activation function

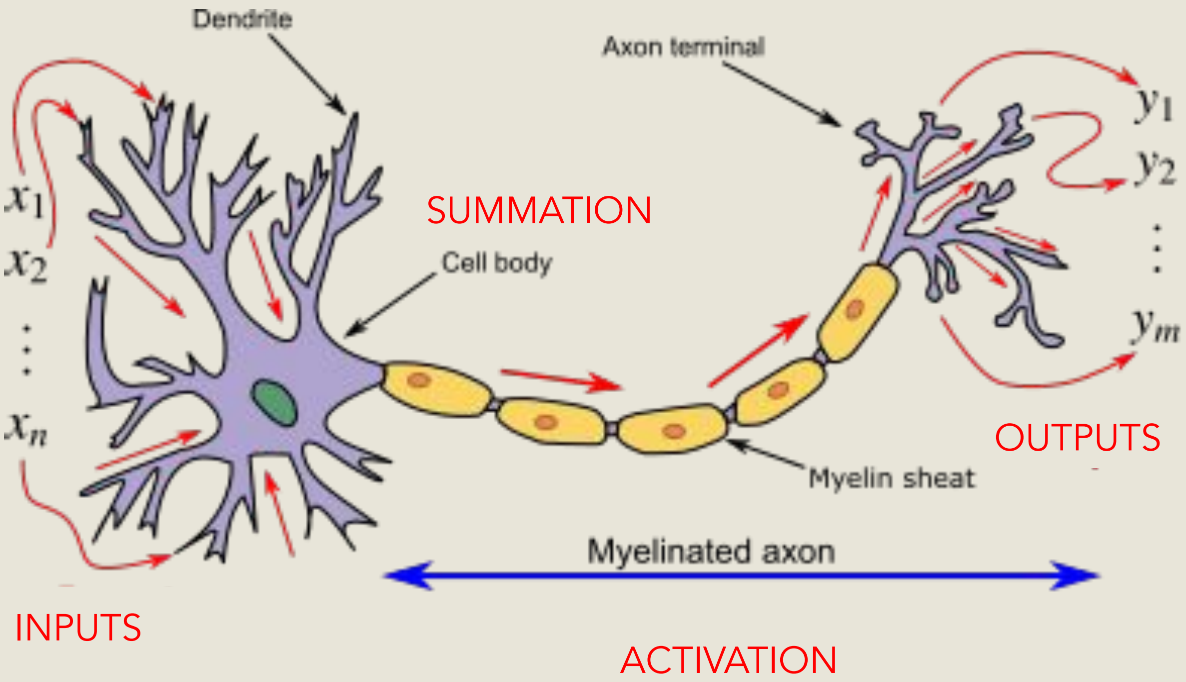


$$l = \mathbf{a}(w_{11} \times i_1 + w_{12} \times i_2)$$
$$= f_2(i_1, i_2)$$

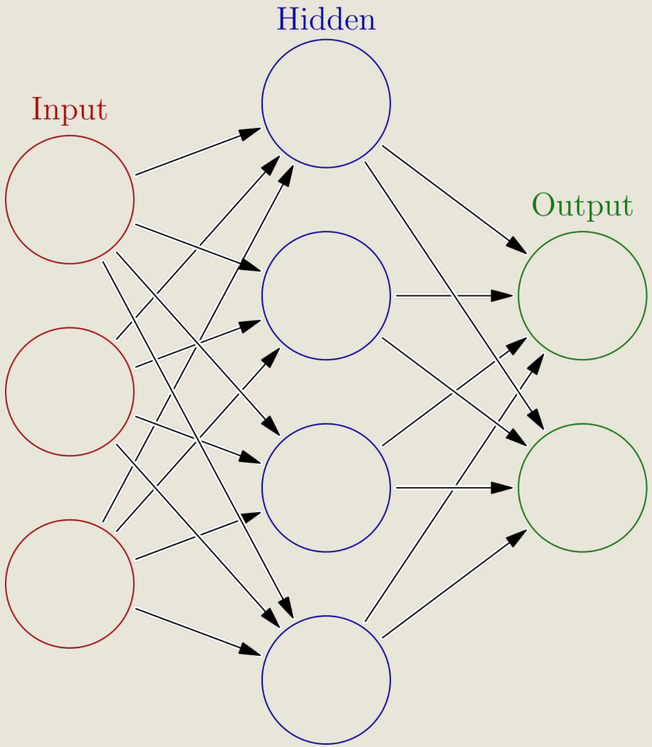




Network Node == Artificial Neuron



Adding complexity via a layer:



Universal Approximation Theorem:

In the mathematical theory of artificial neural networks, the **universal approximation theorem** states[1] that a feed-forward network with a single hidden layer containing a finite number of neurons can approximate arbitrary well real-valued continuous functions on compact subsets of $\mathbf{R}^n$.

But, No Free Lunch Theorem:

For optimization problems... if an algorithm performs well on a certain class of problems then it necessarily pays for that with degraded performance on the set of all remaining problems.

Solving the network

- *Set the initial weights of the network randomly*
- *Make a forward pass through the network and compute output*
- *Compare the output with expected result and compute loss*
- *Change the weights by a small amount (Gradient descent via back prop)*
- *Repeat* until desired minimization of error (cost) is achieved

Summarizing

- *Given: <u>Features</u> (X, Attributes), Output (Y, <u>Labels</u>, Ground Truth): Y=f(X)*
- *Network (<u>Model</u>)*
- *Loss Function (Metric, <u>Cost</u>)*
- *<u>Activation</u> Function (adds non-linearity, Ex: sigmoid, ReLU)*
- *<u>Training</u> (<u>fit</u>, Optimization to minimize Loss Function)*
- *Evaluate (performance, correctness)*
- *Predict (<u>Inference</u>, on new data)*

Computer Graphics Applications

- *Scheduling Optimization*
- *Character AI*
- *Style Transfer*
- *Slow Motion*
- *Up-Res*
- *Denoising*
- *Story Sentiment*
- *Rough to Fine*
- *Body Tracking*
- *Image Generation*

Hands-on (software and environment)

- ★ *We'll be using a python virtual environment: Colab*
- ★ *Colab: Jupyter derived python IDE with tensorflow support*
- ★ Software and tools:
  - Python 3.x - programming
  - Tensorflow 2.1.0 - machine learning
  - Numpy - numerical mathematics, linear algebra
  - Pandas - data analysis
  - Matplotlib - plotting
  - Seaborn - advanced plotting

Hands-on

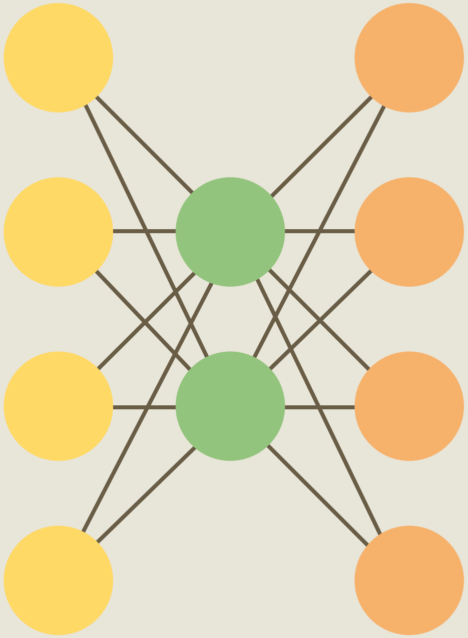
- *Log in to your google drive*
- Find the shared folder 'Disney Machine Learning Webinar'
- Make a copy of:
  - Housing.ipynb,
  - *Let's take a look at the Housing.ipynb*
    - *Analysis of data and possible transformations*

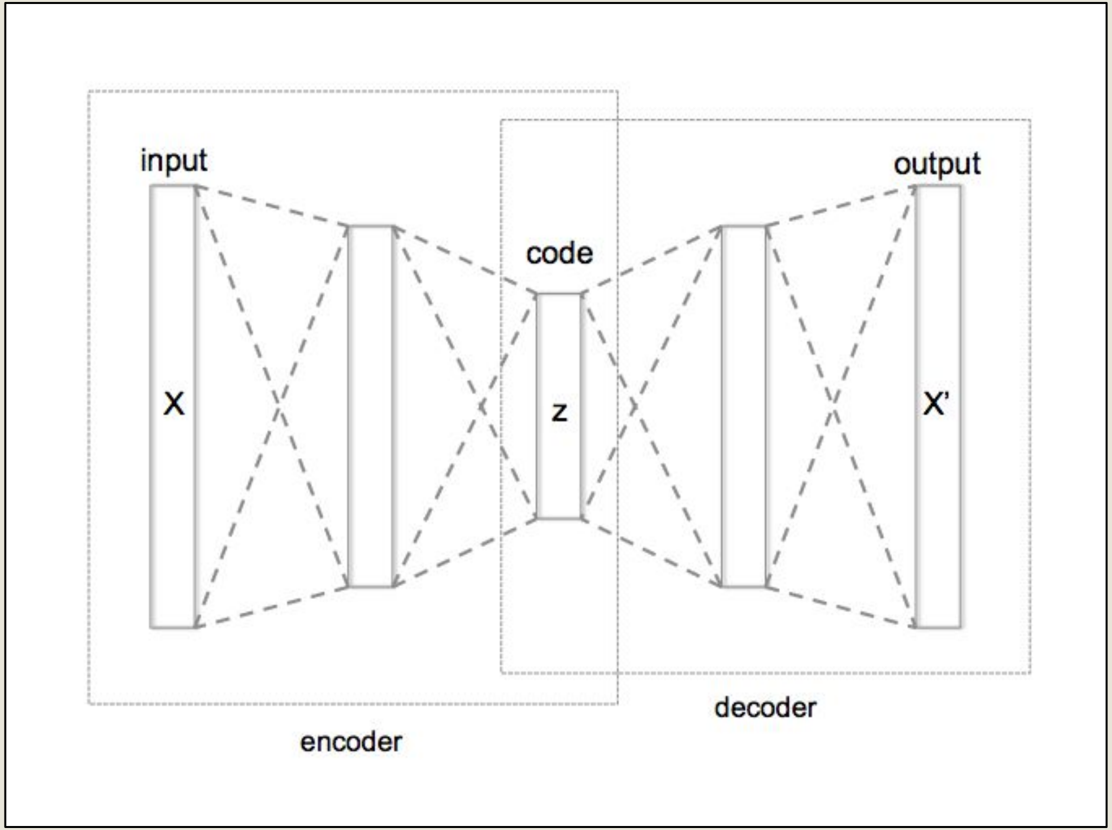
Hands-on

```
# -*- coding: utf-8 -*-
"""Housing.ipynb
"""
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
#----------DATA READING
filename = 'https://download.mlcc.google.com/mledu-datasets/california_housing_train.csv'
# read comma separated values (csv) from the file
csv_data = pd.read_csv(filename, sep=',')

print(csv_data.shape) # matrix of data
print(csv_data.head()) # first five data points
print(csv_data.head().transpose())
print(csv_data.describe()) # simple statistics about the data

print(csv_data[['latitude', 'longitude']]) # pick two columns to print

# plot two variables, the third (total_bedrooms) is the size of the dots
sns.relplot(x='latitude', y='longitude', size='total_bedrooms', alpha=0.5, palette='muted',
data=csv_data)

sns.pairplot(csv_data.head(1000)[['longitude', 'latitude', 'total_bedrooms', 'median_house_value',
'population']])

# pick random sample of data
sns.pairplot(csv_data.sample(n=1000)[['longitude', 'latitude', 'total_bedrooms',
'median_house_value', 'population']])
```

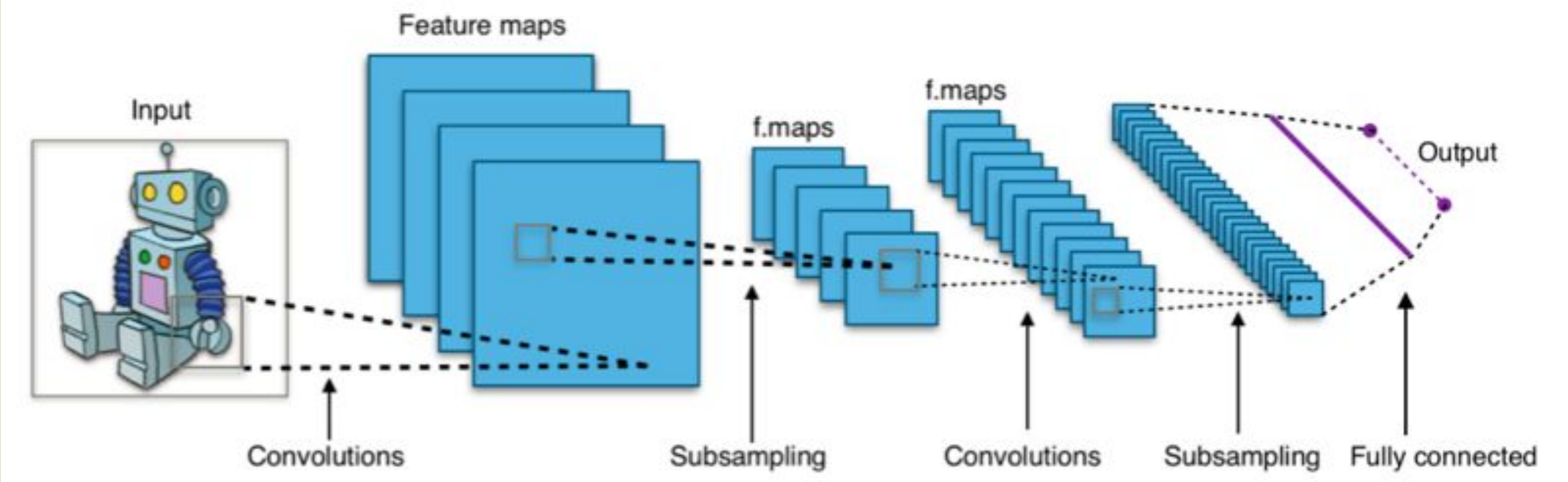
Kinds of Neural Networks

Auto Encoder (AE)

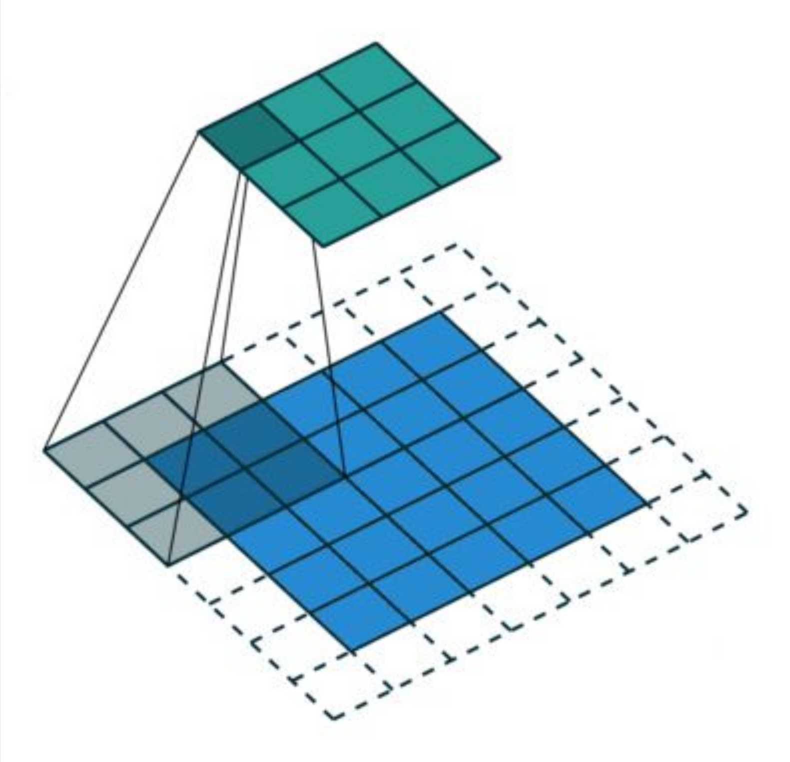




Kinds of Neural Networks (CNN)

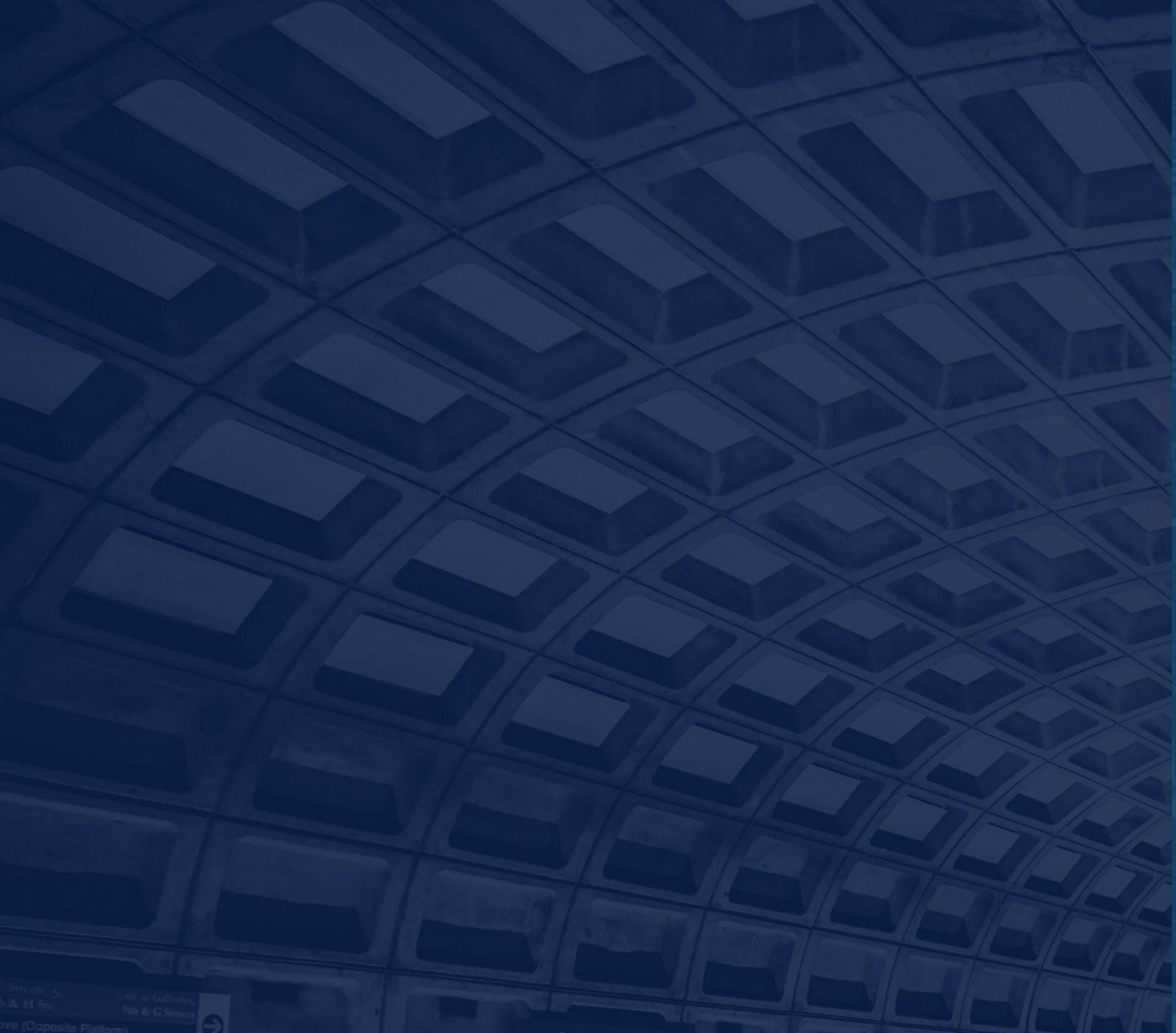


Convolution (Extract High-Level Features)



Next Class

- Tensorflow
- Neural Net for Regression, Classification
- Homework:
  - Play with the data, try other plots
- @xarmalarma, #siggraphNOW



QUESTIONS?

Submit now!

THANK YOU

WANT TO HEAR MORE FROM SIGGRAPH AND DISNEY?

Check out SIGGRAPH Now on YouTube

YouTube.com/user/ACMSIGGRAPH

Subscribe to the SIGGRAPH Spotlight podcast