Lecture 01

What are Machine Learning and Deep Learning? An Overview.

STAT 479: Deep Learning, Spring 2019

Sebastian Raschka

http://stat.wisc.edu/~sraschka/teaching/stat479-ss2019/

But first ...
a course overview

Main Topics

- History of neural networks and what makes deep learning different from “classic machine learning”
- Introduction to the concept of neural networks by connecting it to familiar concepts such as logistic regression and multinomial logistic regression
- Modeling and deriving non-convex loss function through computation graphs
- Introduction to automatic differentiation and PyTorch for efficient data manipulation using GPUs
- Convolutional neural networks for image analysis
- Tricks of the trade for neural network design and training
- 1D convolutions for sequence analysis
- Sequence analysis with recurrent neural networks
- Generative models to sample from input distributions
  - Autoencoders
  - Variational autoencoders
  - Generative Adversarial Networks

Course Material

- Field is relatively new and evolves quickly => No "good" textbook

- Main course material: Mainly slides
  + assigned reading (online references and papers)

- 3 lectures per week, so mainly slides instead of lecture notes (but I will make them more wordy compared to the FS2018 ML class)

Course Logistics

When

- Mon 11:00-11:50 am
- Wed 11:00-11:50 am
- Fri 11:00-11:50 am

Where

- Psychology 121

Instructors

- Instructor: Sebastian Raschka
- Teaching Assistant: Youran Qi

Office Hours

- Sebastian Raschka:
  - Wed 2:00-3:00 pm, Room MSC 1171
- Youran Qi:
  - TBD

Course Website

- Course Logistics
- Course Description
- Resources
- Grading
- Class Project
- Other Important Course Information
- Schedule
  - Topics Summary
  - Calendar

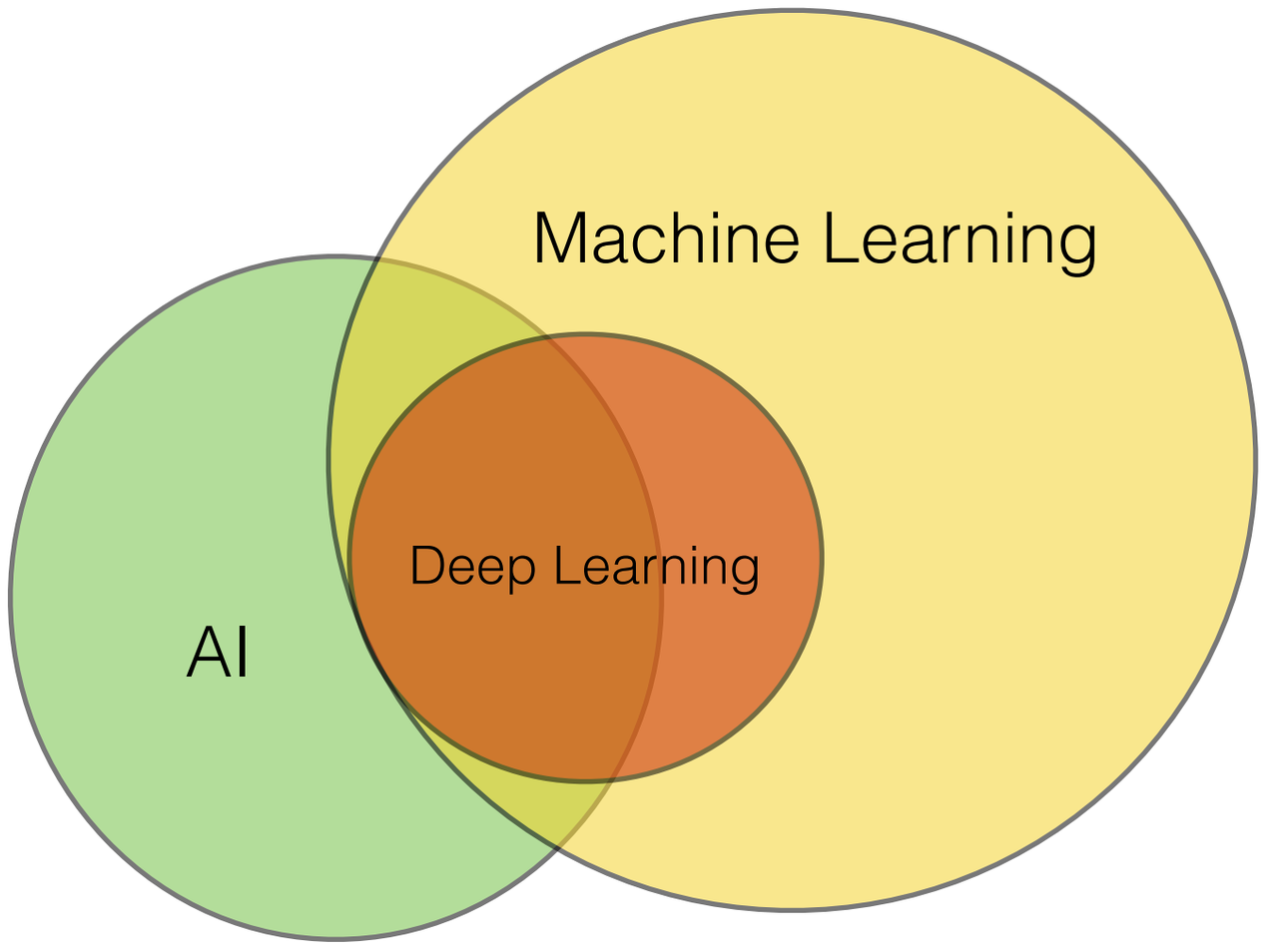
Please check regularly!

http://pages.stat.wisc.edu/~sraschka/teaching/stat479-ss2019/

What is Machine Learning?

A short overview before we jump into Deep Learning

The Connection Between Fields

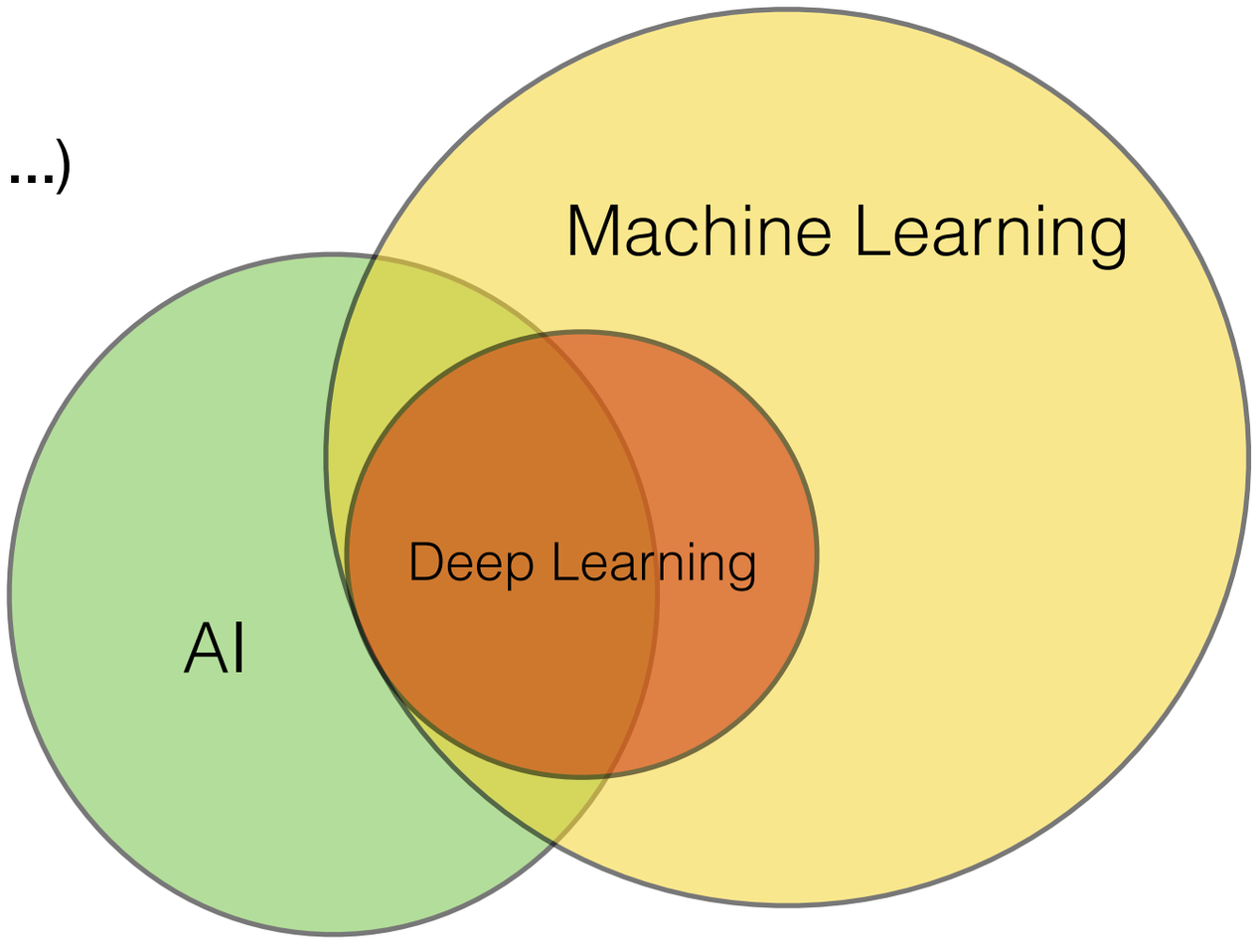


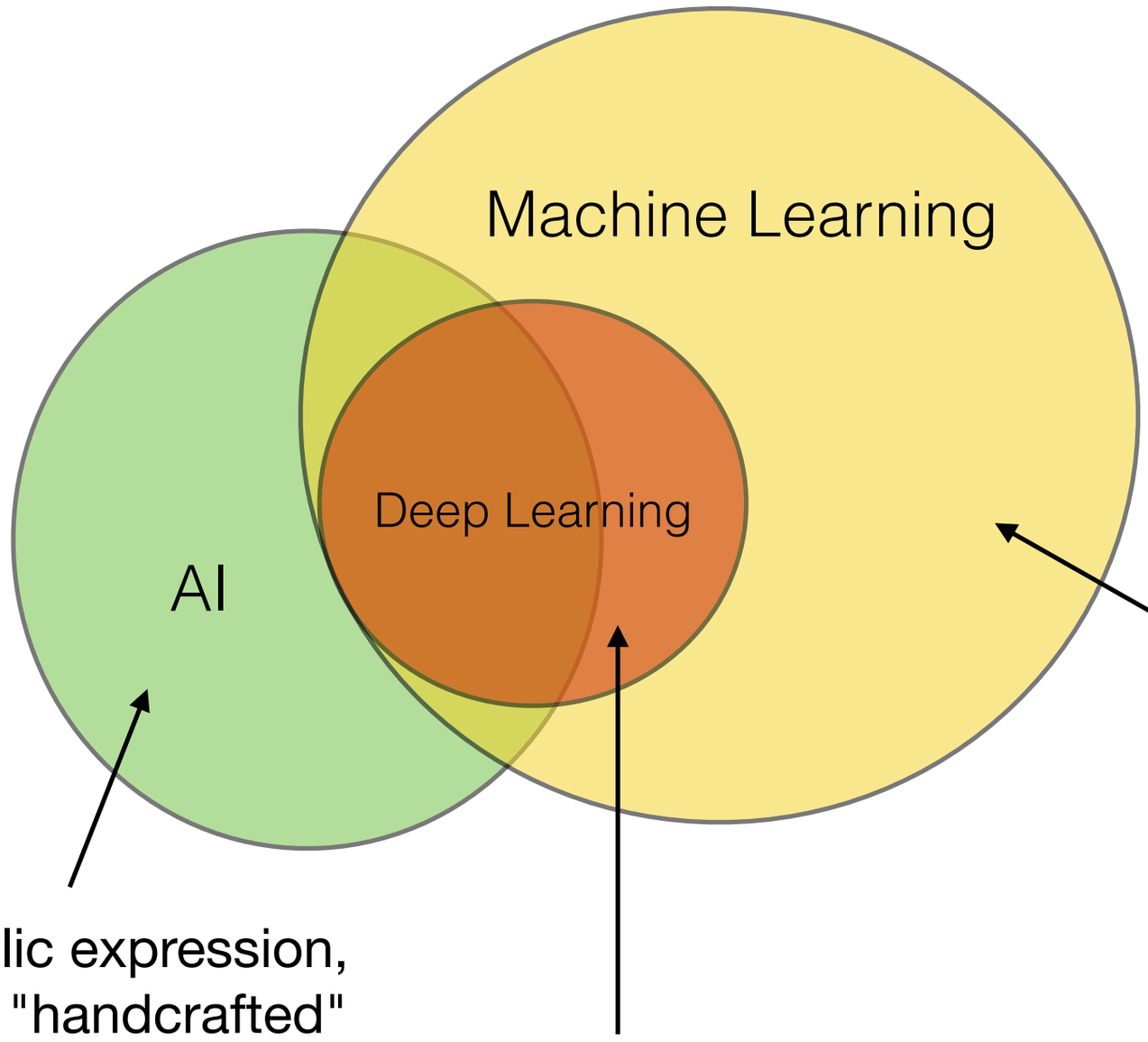
The Connection Between Fields

Artificial Intelligence (AI):
orig. subfield of computer science, solving tasks humans are good at (natural language, speech, image recognition, ...)

Artificial General Intelligence (AGI):
multi-purpose AI mimicking human intelligence across tasks

Narrow AI:
solving "a" task (playing a game, driving a car, ...)

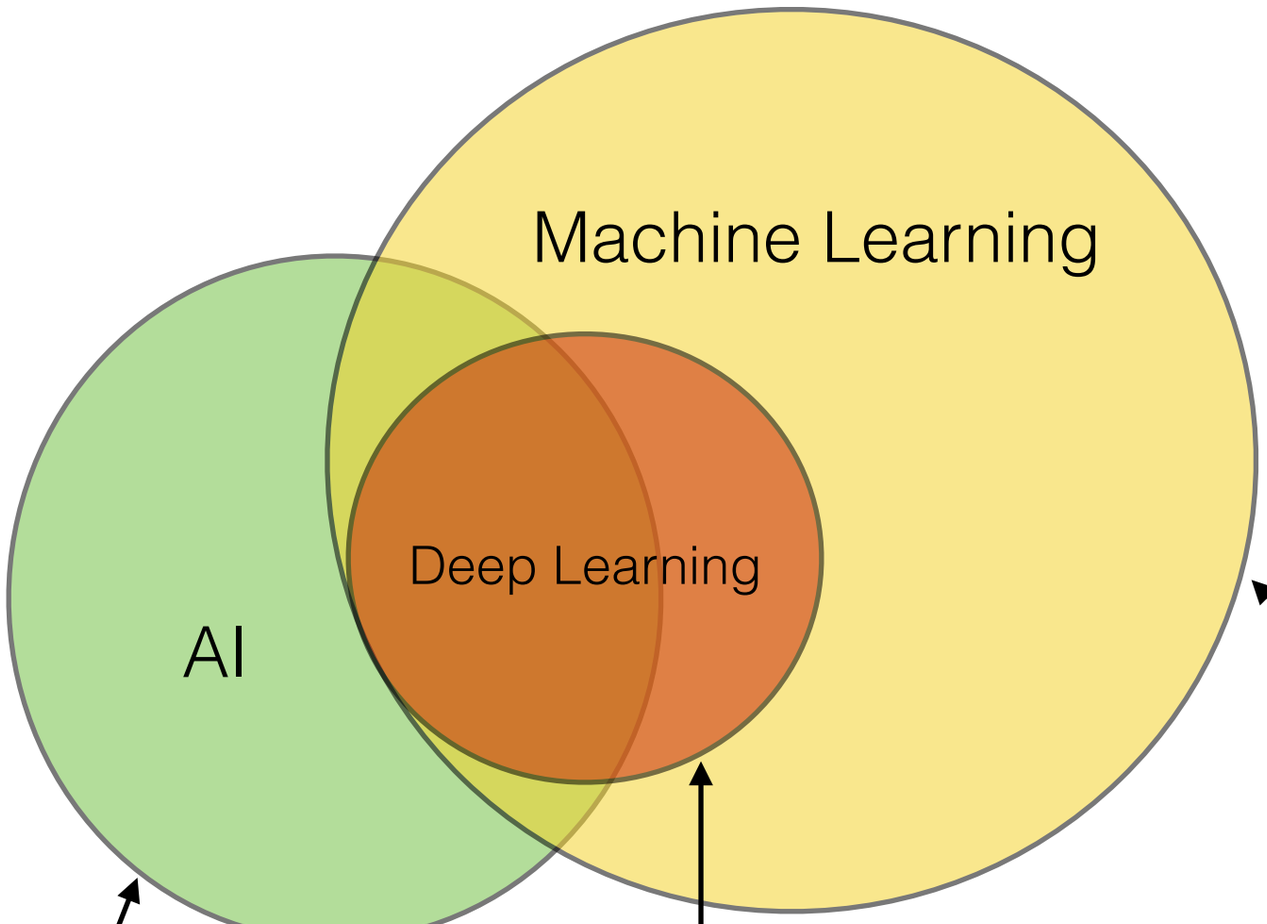




E.g., symbolic expression, logic rules / "handcrafted" nested if-else programming statements ...

Main focus of the course

E.g.,
generalized linear models, tree-based methods, "shallow" networks, SVM, nearest neighbors, ...



= a non-biological system
that is intelligent
through rules

= algorithms that parameterize multi-layers
neural networks that then learn
representations of data with multiple layers
of abstraction

= algorithms that learn
models/representations/
rules automatically
from data/examples

What is Machine Learning?

"Machine learning is the field of study that gives computers the ability to learn without being explicitly programmed"

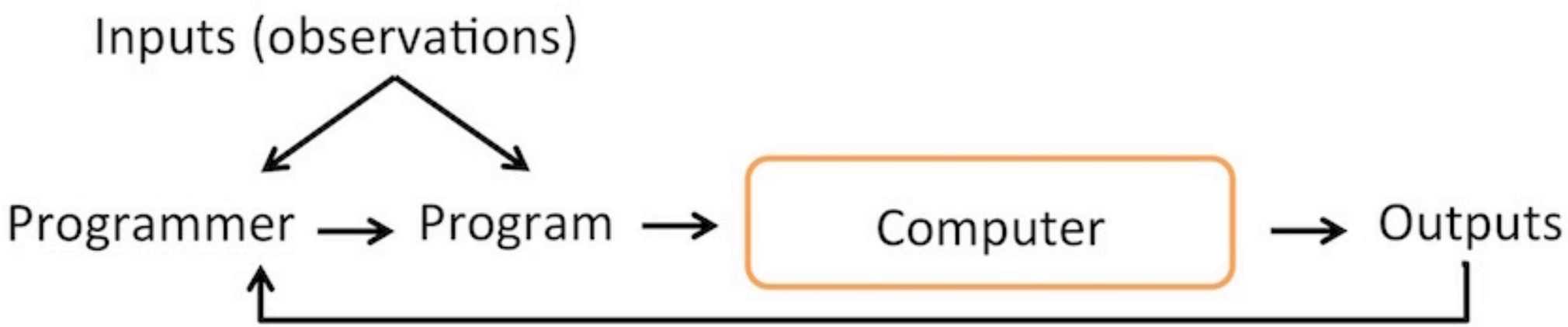
— Arthur L. Samuel, AI pioneer, 1959

[probably the first, and undoubtedly the most popular definition]

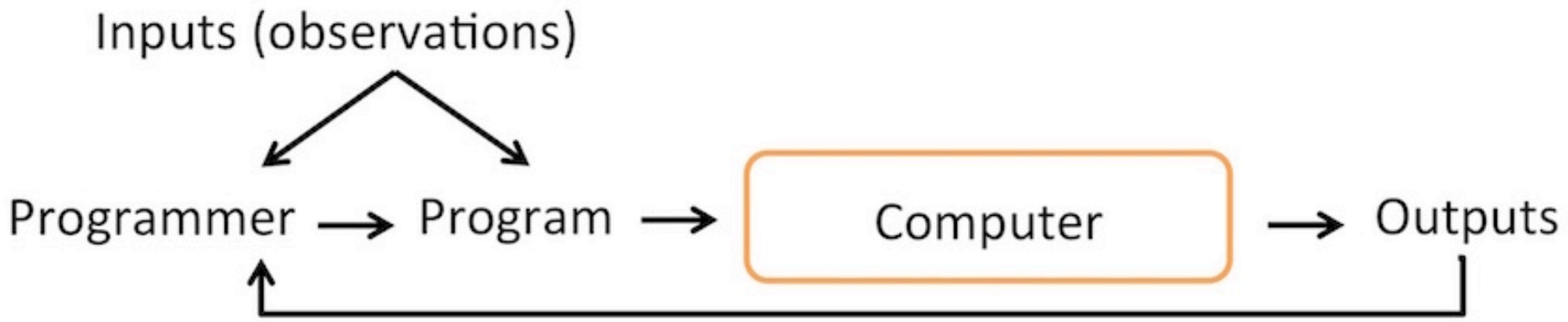
(This is likely not an original quote but a paraphrased version of Samuel's sentence "Programming computers to learn from experience should eventually eliminate the need for much of this detailed programming effort.")

Arthur L Samuel. "Some studies in machine learning using the game of checkers". In: *IBM Journal of research and development* 3.3 (1959), pp. 210–229.

The Traditional Programming Paradigm



The Traditional Programming Paradigm



Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed
– Arthur Samuel (1959)

Machine Learning

Some Applications of Machine Learning/Deep Learning

- Email spam detection
- Face detection and matching (e.g., iPhone X)
- Web search (e.g., DuckDuckGo, Bing, Google)
- Sports predictions
- Post office (e.g., sorting letters by zip codes)
- ATMs (e.g., reading checks)
- Credit card fraud
- Stock predictions

Some Applications of Machine Learning/Deep Learning

- Smart assistants (Apple Siri, Amazon Alexa, ...)
- Product recommendations (e.g., Netflix, Amazon)
- Self-driving cars (e.g., Uber, Tesla)
- Language translation (Google translate)
- Sentiment analysis
- Drug design
- Medical diagnoses
- ...

Categories of ML/DL

Supervised Learning

- Labeled data
- Direct feedback
- Predict outcome/future

Unsupervised Learning

- No labels/targets
- No feedback
- Find hidden structure in data

Reinforcement Learning

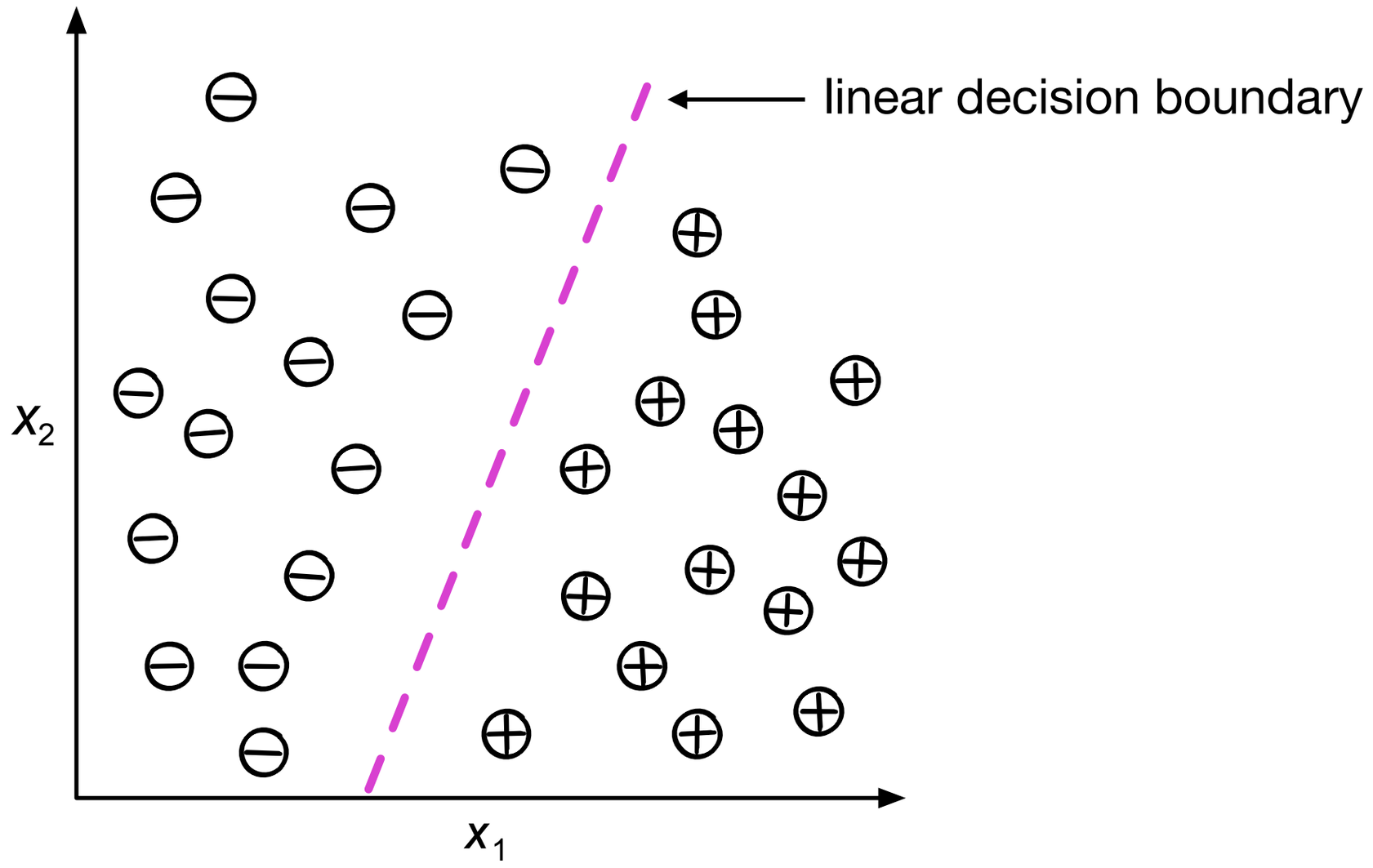
- Decision process
- Reward system
- Learn series of actions

Supervised Learning

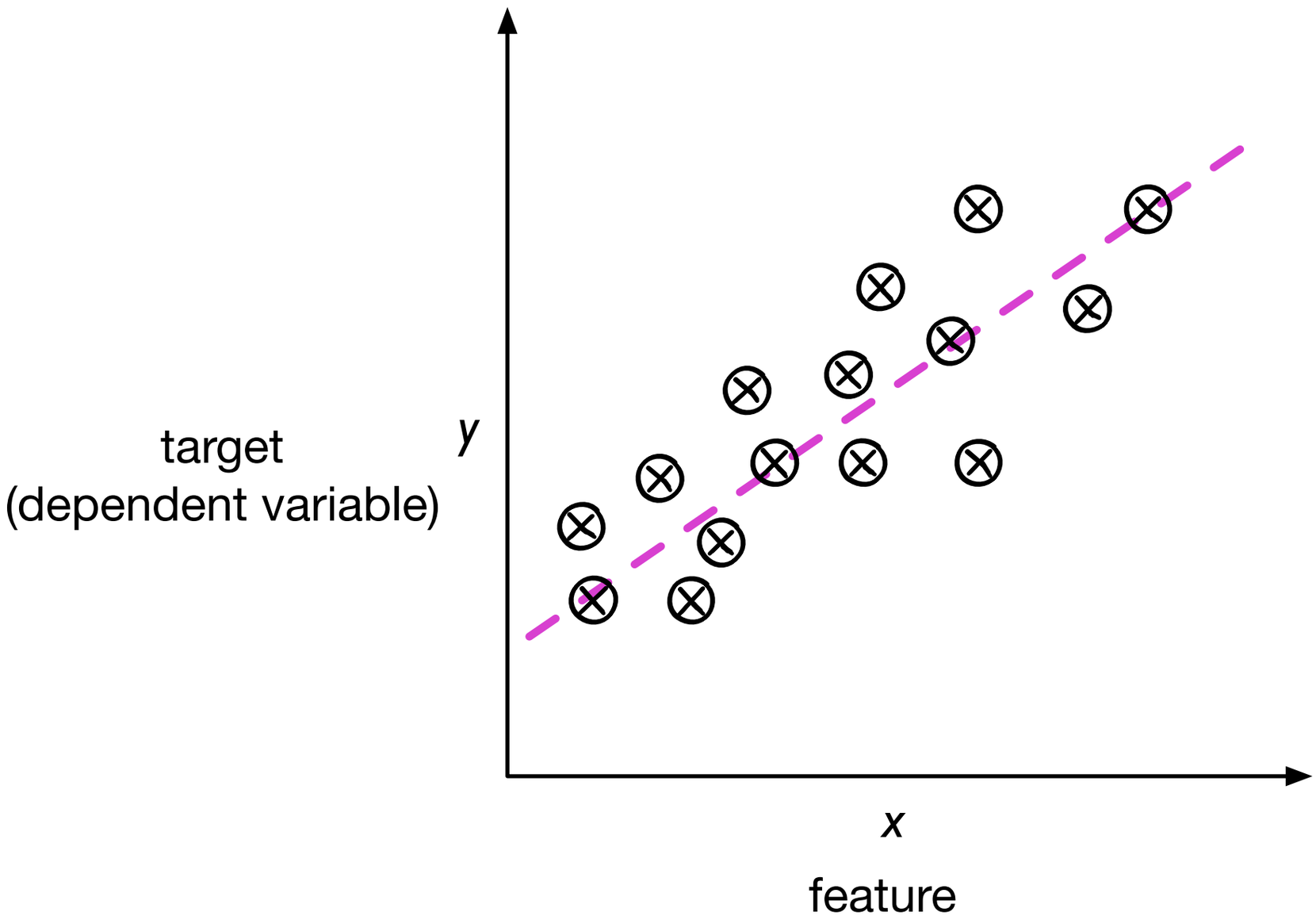
- Labeled data
- Direct feedback
- Predict outcome/future

Supervised Learning: Classification

Binary classification example with two *features* ("independent" variables, predictors)



Supervised Learning: Regression



Supervised Learning: Ordinal Regression

- Ordinal regression also called *ordinal classification* or *ranking* (although ranking is a bit different)

Order dependence like in metric regression, but no metric distance

discrete values like in classification, but order dependence

$$r_K \succ r_{K-1} \succ \ldots \succ r_1$$

E.g., movie ratings: *great* $\succ$ *good* $\succ$ *okay* $\succ$ *for genre fans* $\succ$ *bad*

Supervised Learning: Ordinal Regression

- Ranking: Correct order matters
  (0 loss if order is correct, e.g., rank a collection of movies by "goodness")

- Ordinal Regression: Correct label matters
  (E.g., age of a person in years; here, regard aging as a non-stationary process)

Excerpt from the UTKFace dataset
https://susanqq.github.io/UTKFace/

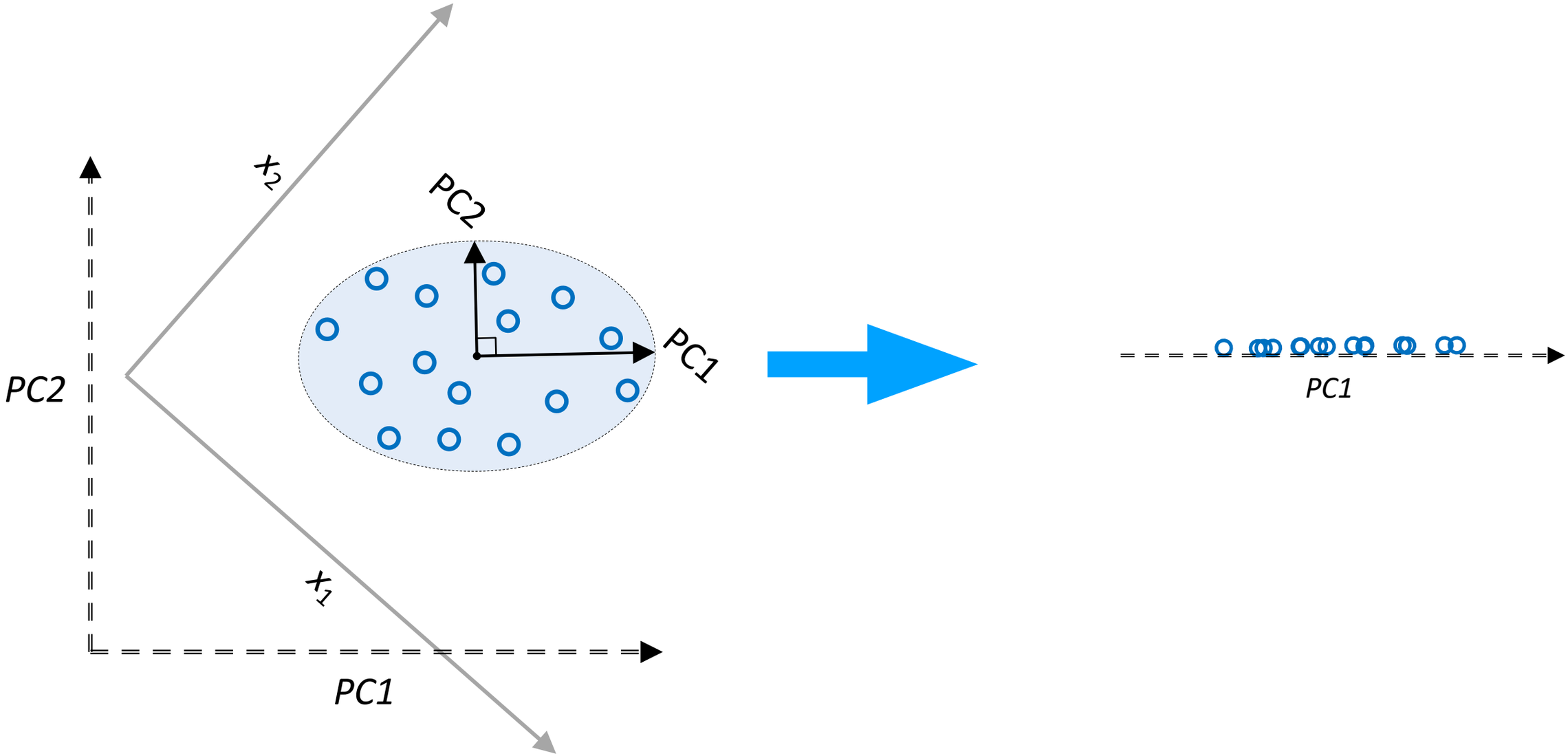
18 29 41

Unsupervised Learning

- No labels/targets
- No feedback
- Find hidden structure in data

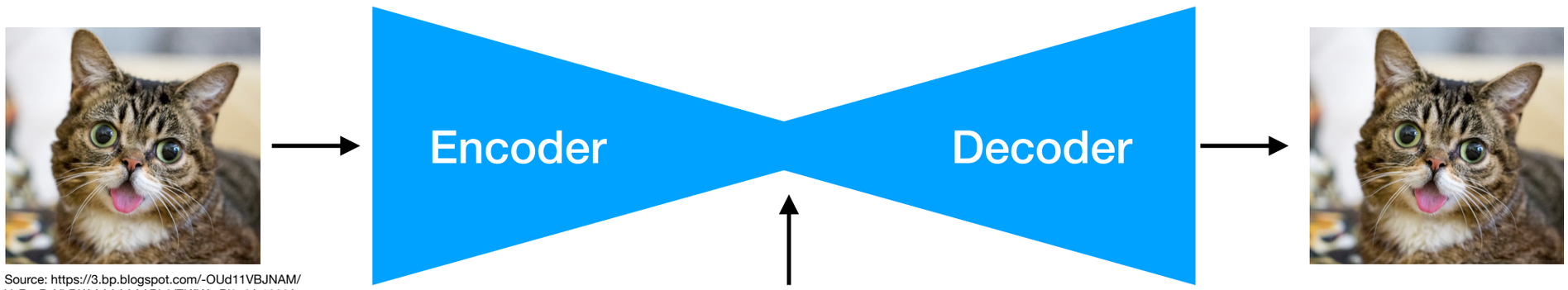
Unsupervised Learning: Representation Learning/Dimensionality Reduction

E.g., Principal Component Analysis



Unsupervised Learning: Representation Learning/Dimensionality Reduction

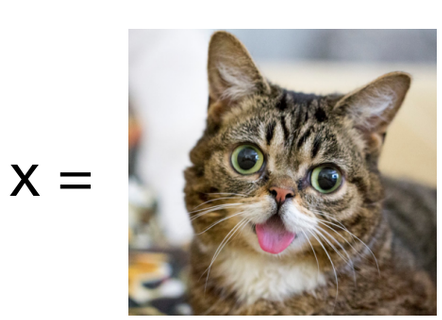
E.g., Autoencoders



Source: https://3.bp.blogspot.com/-OUd11VBJNAM/VsFacR_YhBI/AAAAAAAABh0/ZKfKAnRj3x0/s1600/cannot%2Bresist.jpg

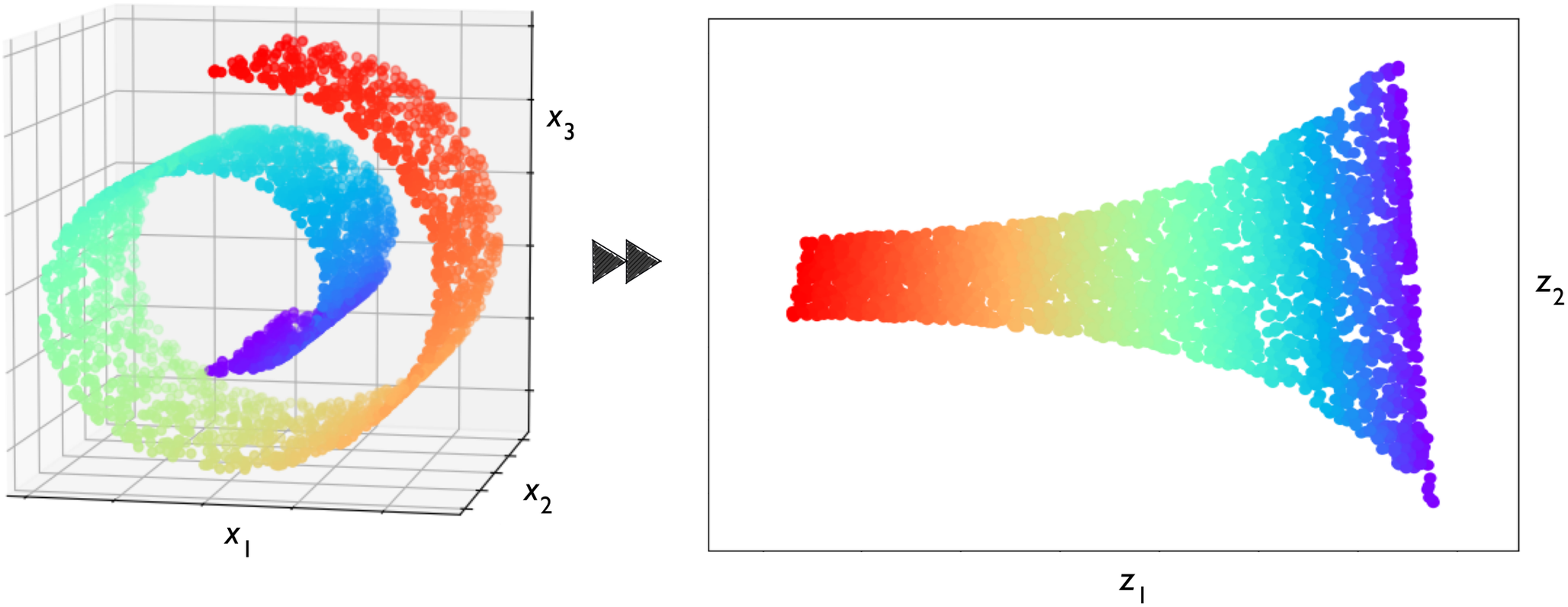
latent representation/
feature embedding

Reminder: Classification works like this



Source: https://3.bp.blogspot.com/-OUd11VBJNAM/VsFacR_YhBI/AAAAAAAABh0/ZKfKAnRj3x0/s1600/cannot%2Bresist.jpg

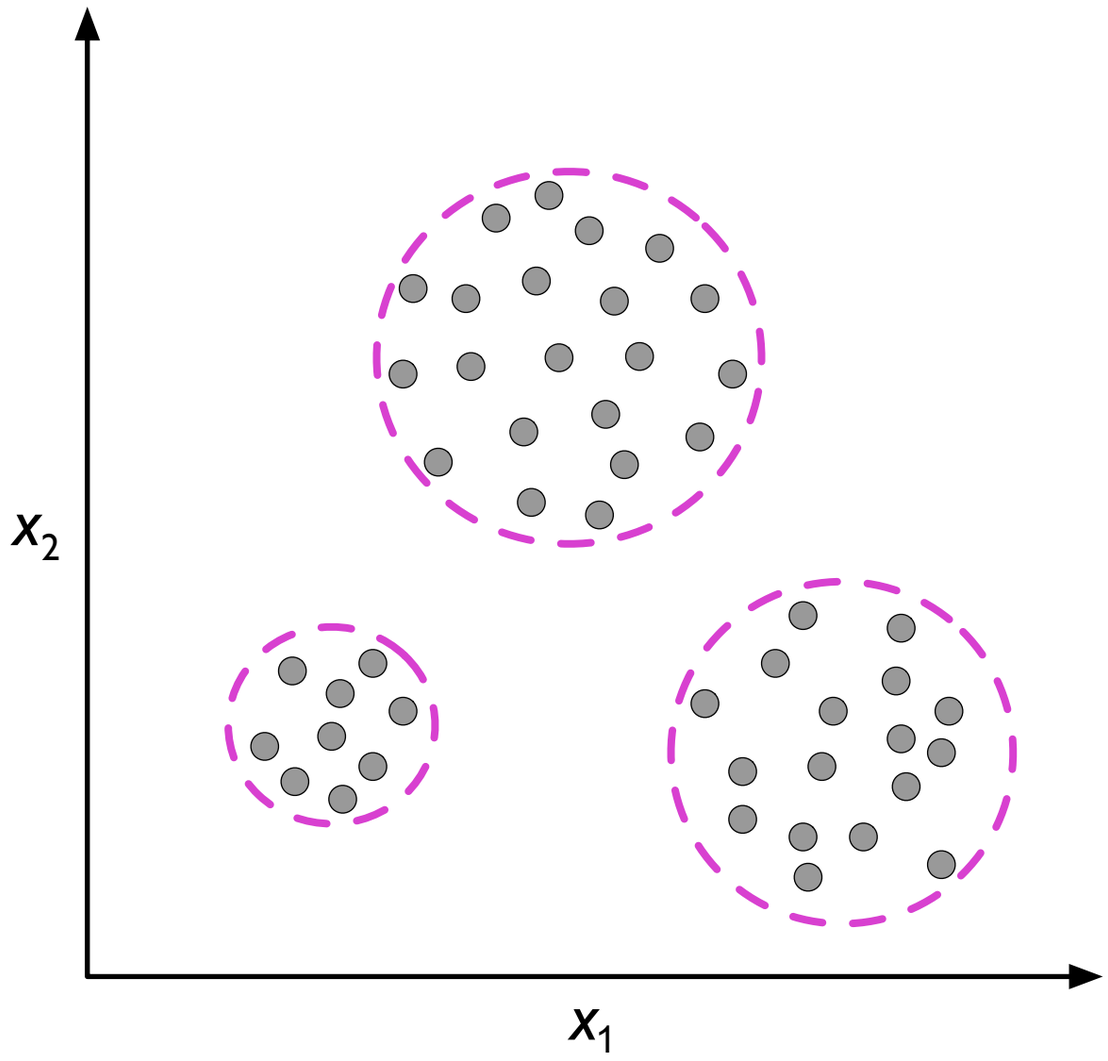
Unsupervised Learning: Representation Learning/Dimensionality Reduction



Example of manifold learning using kernel PCA

Unsupervised Learning: Clustering

Assigning group memberships to unlabelled examples (instances, data points)

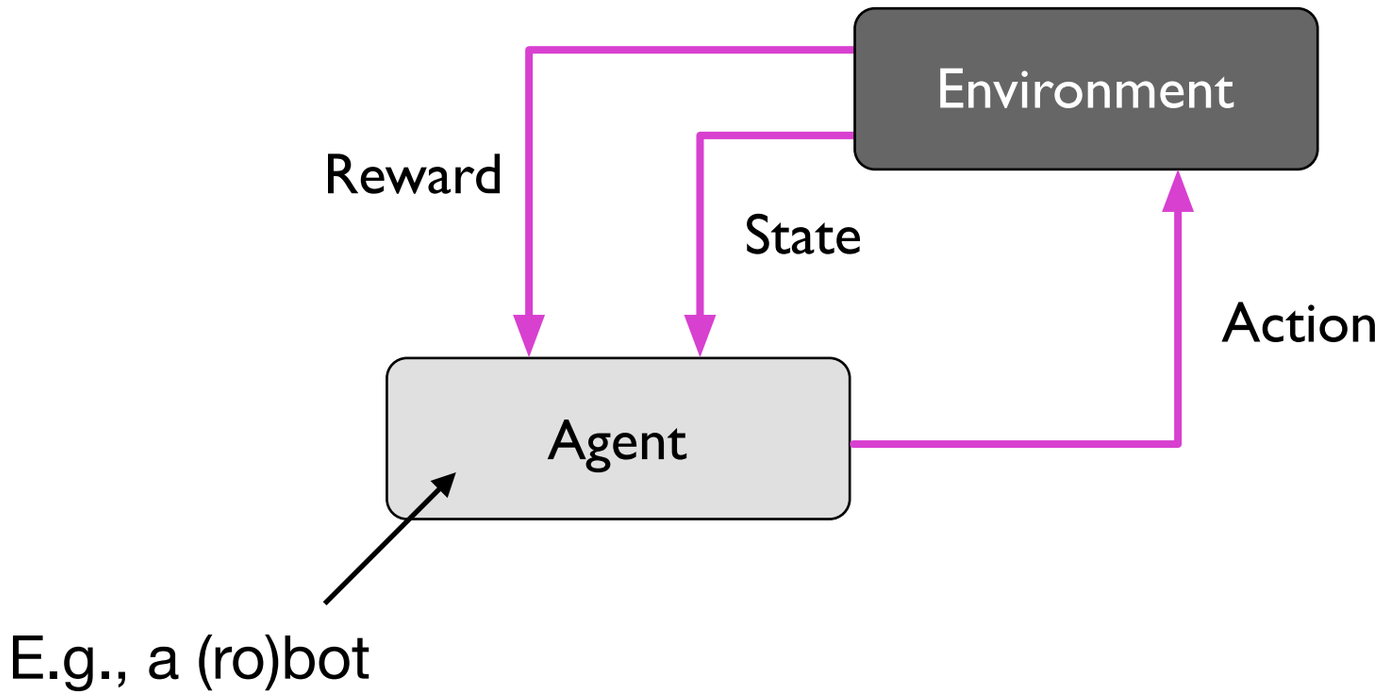


Semi-Supervised Learning

- mix between supervised and unsupervised learning
- some training examples contain outputs, but some do not
- use the labeled training subset to label the unlabeled portion of the training set, which we then also utilize for model training

Reinforcement Learning

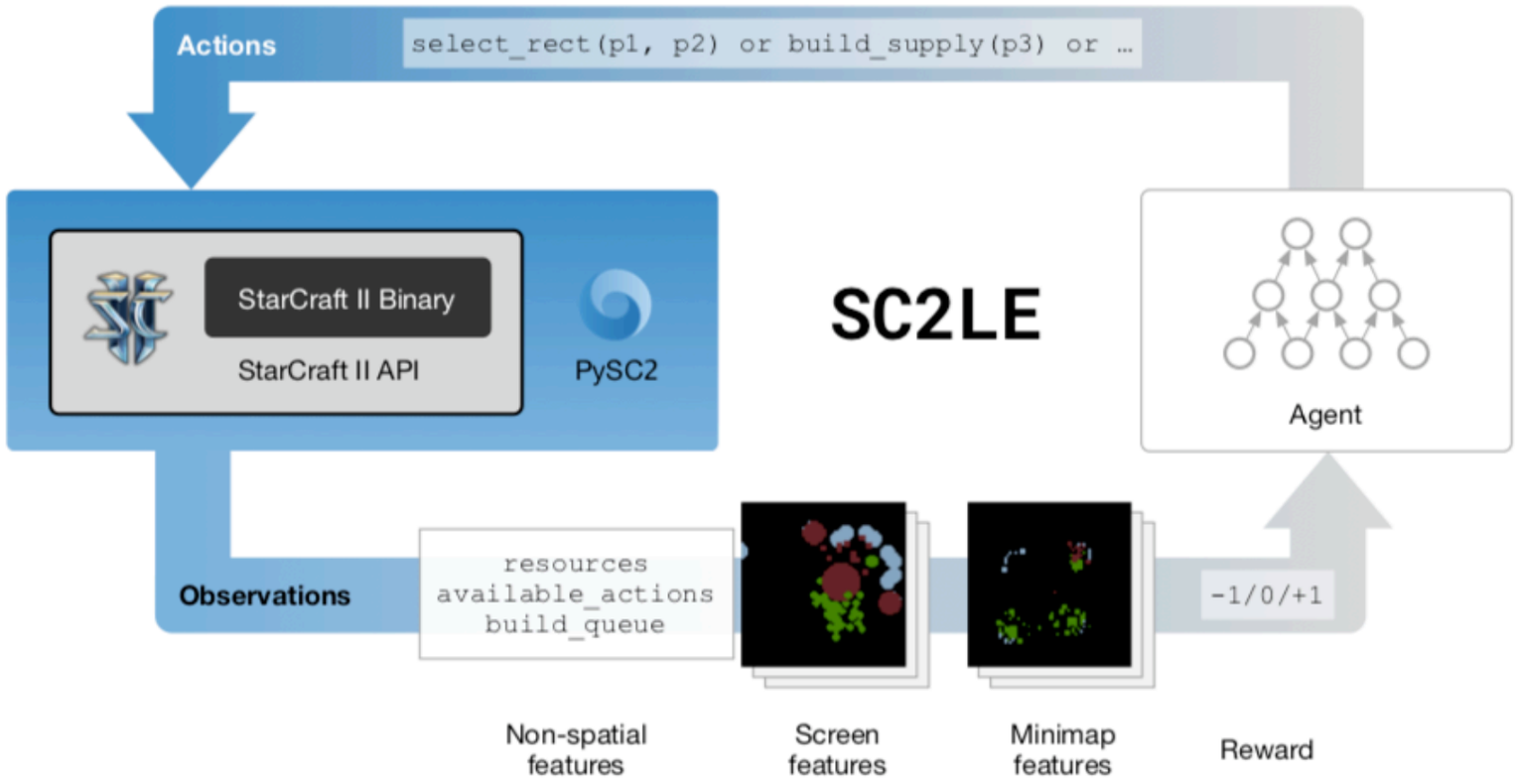
maximize the reward for a *series* of actions



(Probably won't cover this in this course)

Reinforcement Learning

Current state-of-the-art benchmark: StarCraft II



Vinyals, Oriol, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani et al. "Starcraft II: A new challenge for reinforcement learning." *arXiv preprint arXiv:1708.04782* (2017).

Machine Learning Jargon 1/2

- supervised learning:
  learn function to map input *x* (features) to output *y* (targets)

- structured data:
  databases, spreadsheets/csv files

- unstructured data:
  features like image pixels, audio signals, text sentences
  (previous to DL, extensive feature engineering required)

Supervised Learning (more formal notation)

"training examples"

Training set: $\mathscr{D} = \{\langle \mathbf{x}^{[i]}, y^{[i]} \rangle, i = 1, \ldots, n\},$

Unknown function: $f(\mathbf{x}) = y$

Hypothesis: $h(\mathbf{x}) = \hat{y}$ sometimes *t* or *o*

Classification

$h : \mathbb{R}^m \rightarrow \mathscr{Y}, \quad \mathscr{Y} = \{1, \ldots, k\}$

Regression

$h : \mathbb{R}^m \rightarrow \mathbb{R}$

Data Representation

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$$

Feature vector

Data Representation

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$$

Feature vector

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix}$$

Design Matrix

$$\mathbf{X} = \begin{bmatrix} x_1^{[1]} & x_2^{[1]} & \cdots & x_m^{[1]} \\ x_1^{[2]} & x_2^{[2]} & \cdots & x_m^{[2]} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{[n]} & x_2^{[n]} & \cdots & x_m^{[n]} \end{bmatrix}$$

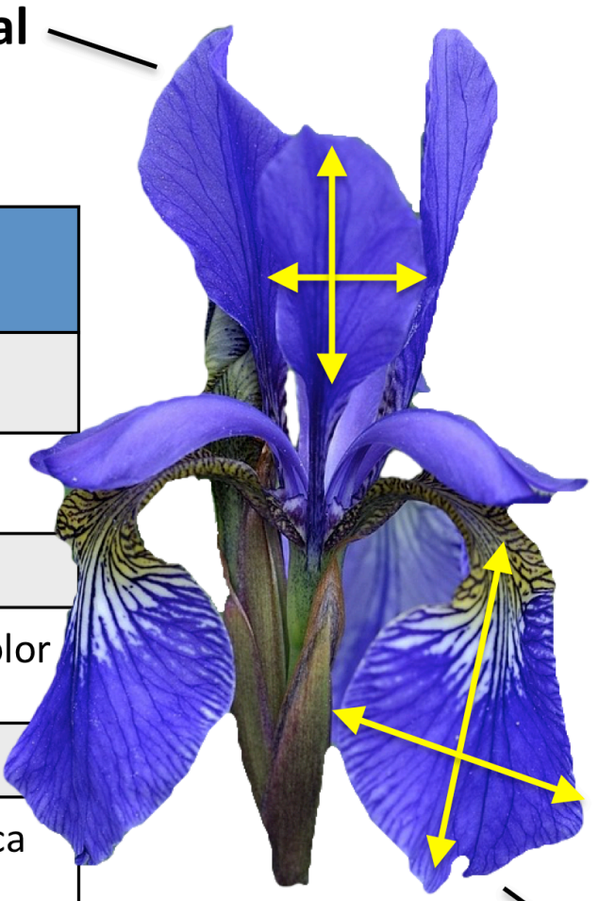
Design Matrix

Data Representation (structured data)

$m =$ ____

$n =$ ____

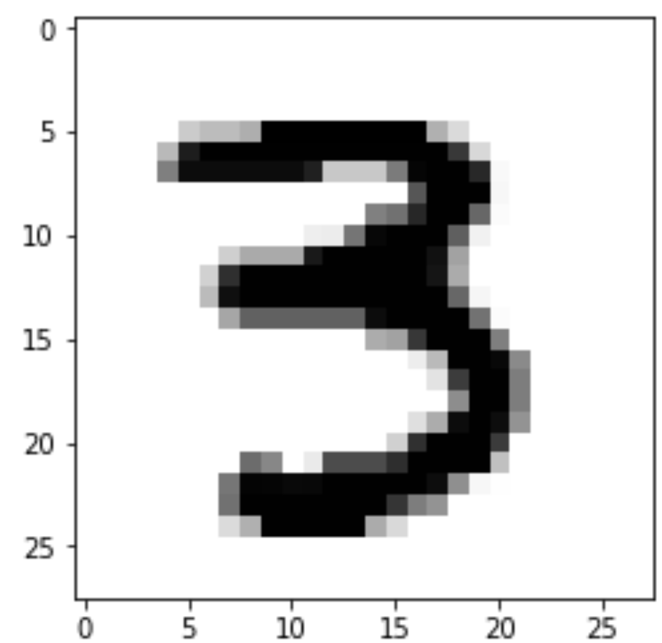
	Sepal length	Sepal width	Petal length	Petal width	
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica



Data Representation (unstructured data; images)

"traditional methods"

[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.20392157, 0.2627451, 0.2627451, 0.31764707, 0.99607843, 0.99607843, 0.99607843
, 0.99607843, 1.0, 0.99607843, 0.99607843, 0.99607843, 0.30980393, 0.14509805, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.27450982, 0.8862745, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.9
9215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.7764706, 0.15294118, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.5019608, 0.9529412, 0.9529412, 0.9529412, 0.9529412, 0.9529412,
0.9529412, 0.87058824, 0.21568628, 0.21568628, 0.21568628, 0.5176471, 0.98039216, 0.99215686, 0.99215686, 0.8392157
, 0.023529412, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.6627451, 0.99215686, 0.99215686, 0.99215686, 0.03137255, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.49803922, 0.5529412, 0.84705883, 0.99215686, 0.99
215686, 0.59607846, 0.015686275, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.06666667, 0.07450981, 0.5411765, 0.972549, 0.99215686, 0.99215686, 0.99215686, 0.627451, 0.05490196, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1882353, 0.33333334, 0.33333334, 0.33333334
, 0.8980392, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.92941177, 0.3647059, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.18039216, 0.8156863, 0.99215686, 0.99215686, 0.99215686, 0
.99215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.91764706, 0.32941177, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.26666668, 0.9411765, 0.99215686, 0.99215686, 0.99215686, 0
.99215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.59607846, 0.03529412, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.34901962, 0.61960787, 0.61960787, 0.61960787,
0.61960787, 0.61960787, 0.61960787, 0.9529412, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.54901963, 0.011764
706, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.325
4902, 0.3647059, 0.78431374, 0.99215686, 0.99215686, 0.99215686, 0.5019608, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.07058824, 0.27058825, 0.99215686, 0.99
215686, 0.96862745, 0.45882353, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.10980392, 0.7647059, 0.99215686, 0.99215686, 0.50980395, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.44705883, 0.992156
86, 0.99215686, 0.50980395, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.1254902, 0.3254902, 0.9529412, 0.99215686, 0.9490196, 0.41960785, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.18431373, 0.8117647, 0.99215686,
0.99215686, 0.99215686, 0.76862746, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.57
64706, 0.47058824, 0.0, 0.08235294, 0.7019608, 0.7019608, 0.7019608, 0.81960785, 0.99215686, 0.99215686, 0.99215686
, 0.99215686, 0.24705882, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.5372549, 0.984313
7, 0.98039216, 0.9607843, 0.9647059, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.9490196, 0.43921
57, 0.03137255, 0.003921569, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.5529412, 0.992
15686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.7921569, 0.50980395, 0.41960785, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.14117648, 0.30980393, 0.99215
686, 0.99215686, 0.99215686, 0.99215686, 0.99215686, 0.32941177, 0.14901961, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]



Data Representation (unstructured data; images)

Convolutional Neural Networks

```
Image batch dimensions: torch.Size([128, 1, 28, 28])
Image label dimensions: torch.Size([128])
```

← "NCHW" representation (more on that later)

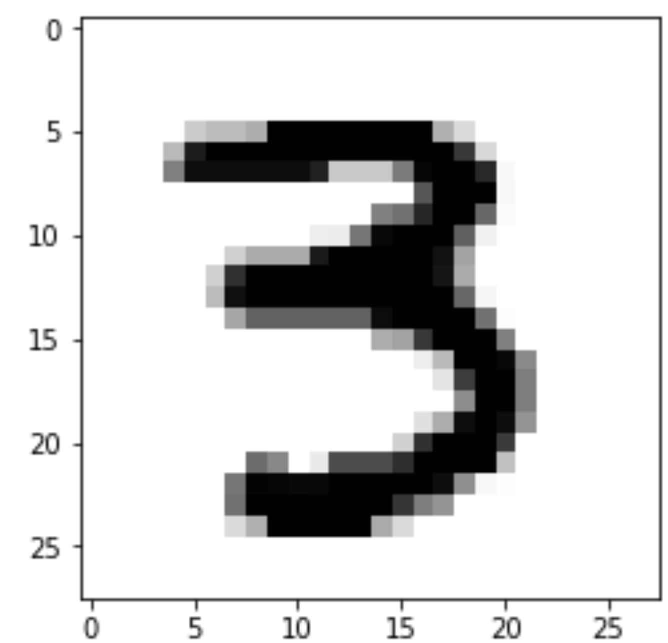
```
print(images[0].size())
```

```
torch.Size([1, 28, 28])
```

```
images[0]
```

```
tensor([[[0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],

         [0.0000, 0.0000, 0.0000, 0.0000, 0.5020, 0.9529, 0.9529, 0.9529,
          0.9529, 0.9529, 0.9529, 0.8706, 0.2157, 0.2157, 0.2157, 0.5176,
          0.9804, 0.9922, 0.9922, 0.8392, 0.0235, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.6627, 0.9922, 0.9922, 0.9922, 0.0314, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.4980, 0.5529,
          0.8471, 0.9922, 0.9922, 0.5961, 0.0157, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0000],
         [0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
          0.0000, 0.0000, 0.0000, 0.0667, 0.0745, 0.5412, 0.9725, 0.9922,
```



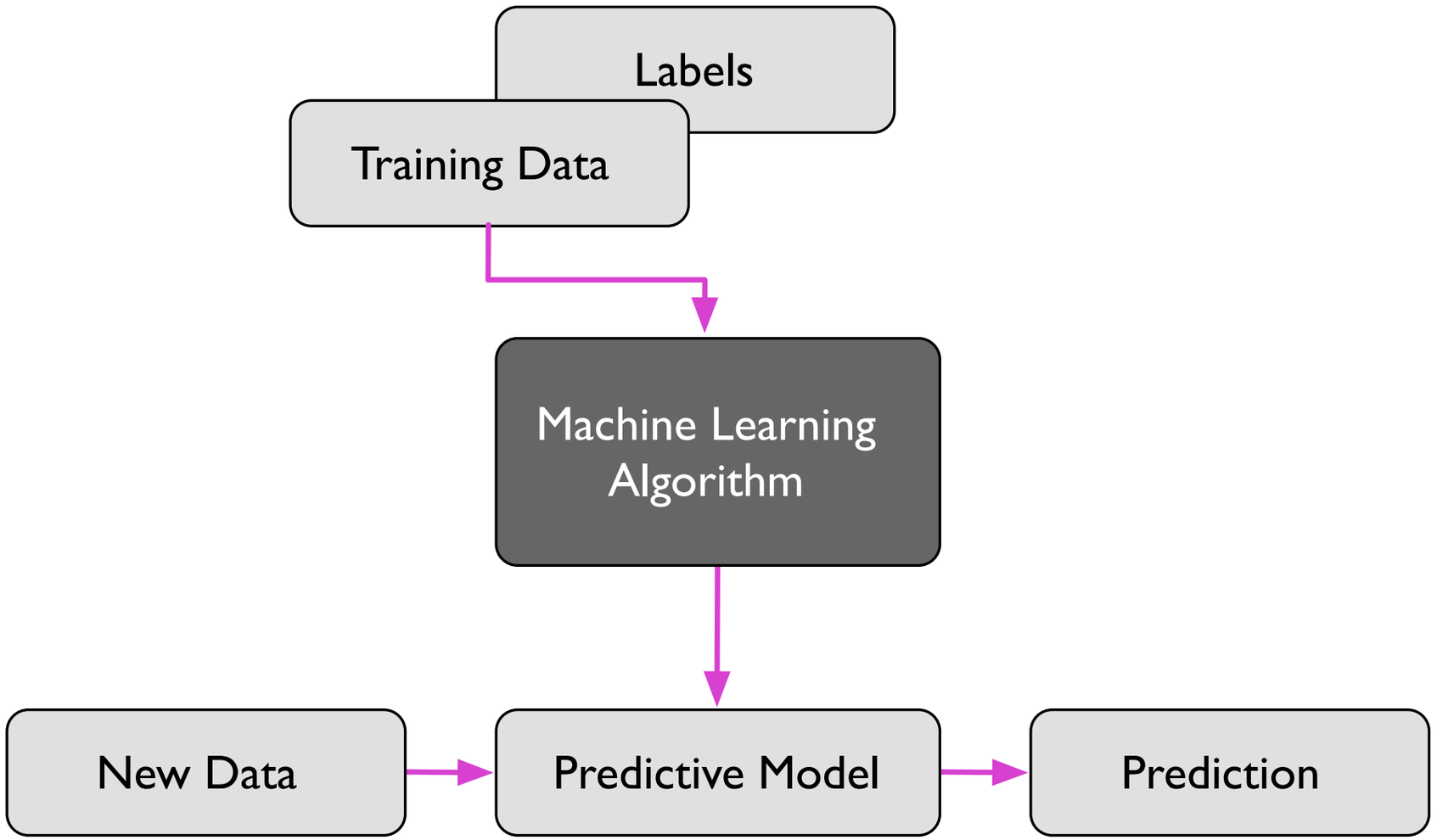
Machine Learning Jargon 2/2

- **Training example**, synonymous to
  observation, training record, training instance, training sample (in some contexts, sample refers to a collection of training examples)
- **Feature**, synonymous to
  predictor, variable, independent variable, input, attribute, covariate
- **Target**, synonymous to
  outcome, ground truth, output, response variable, dependent variable, (class) label (in classification)
- **Output / Prediction**, use this to distinguish from targets; here, means output from the model

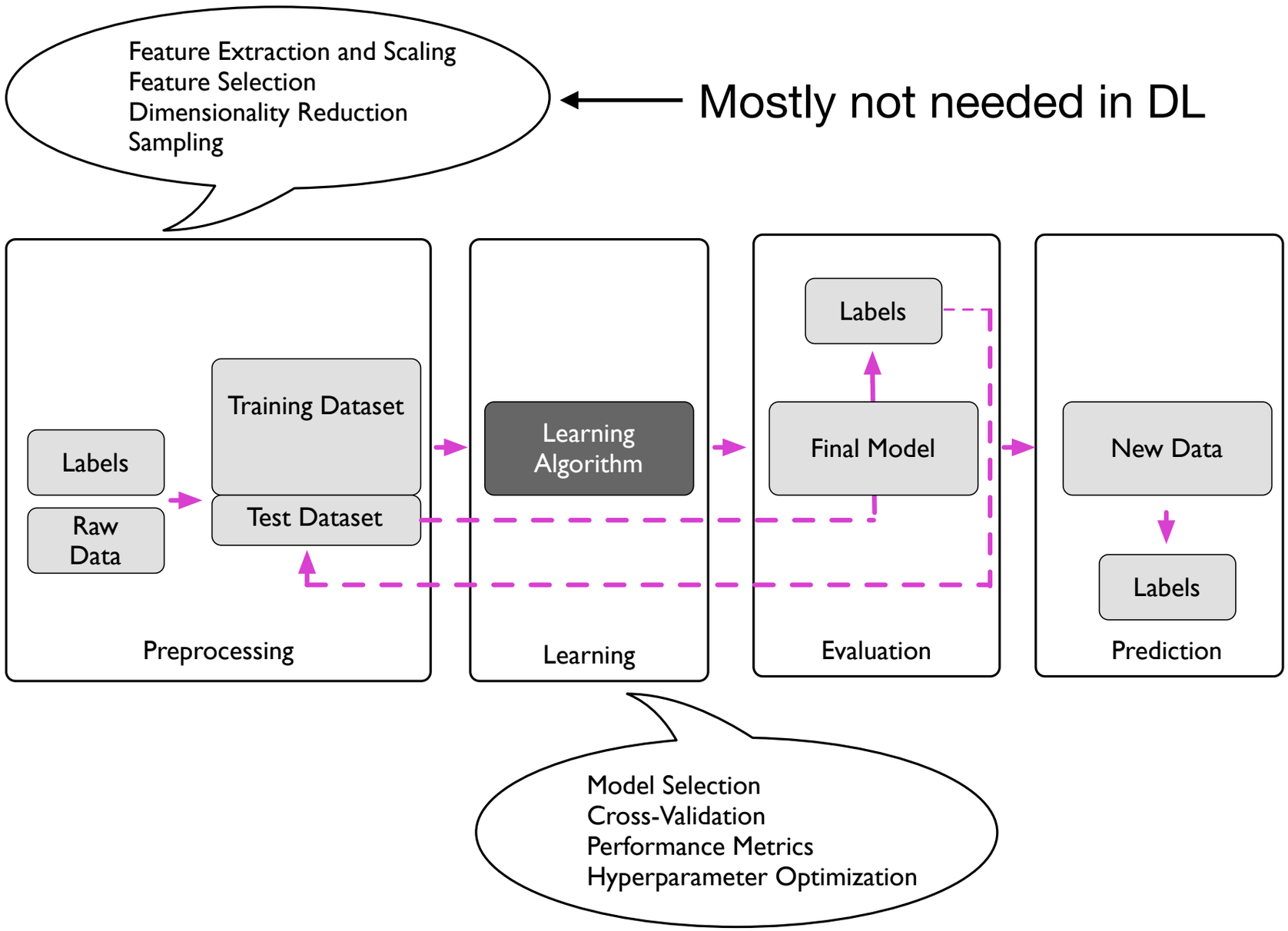
- use loss L for a single training example
- use cost C for the average loss over the training set
- use $\phi(\cdot)$, unless noted otherwise, for the activation function

(will make more sense later)

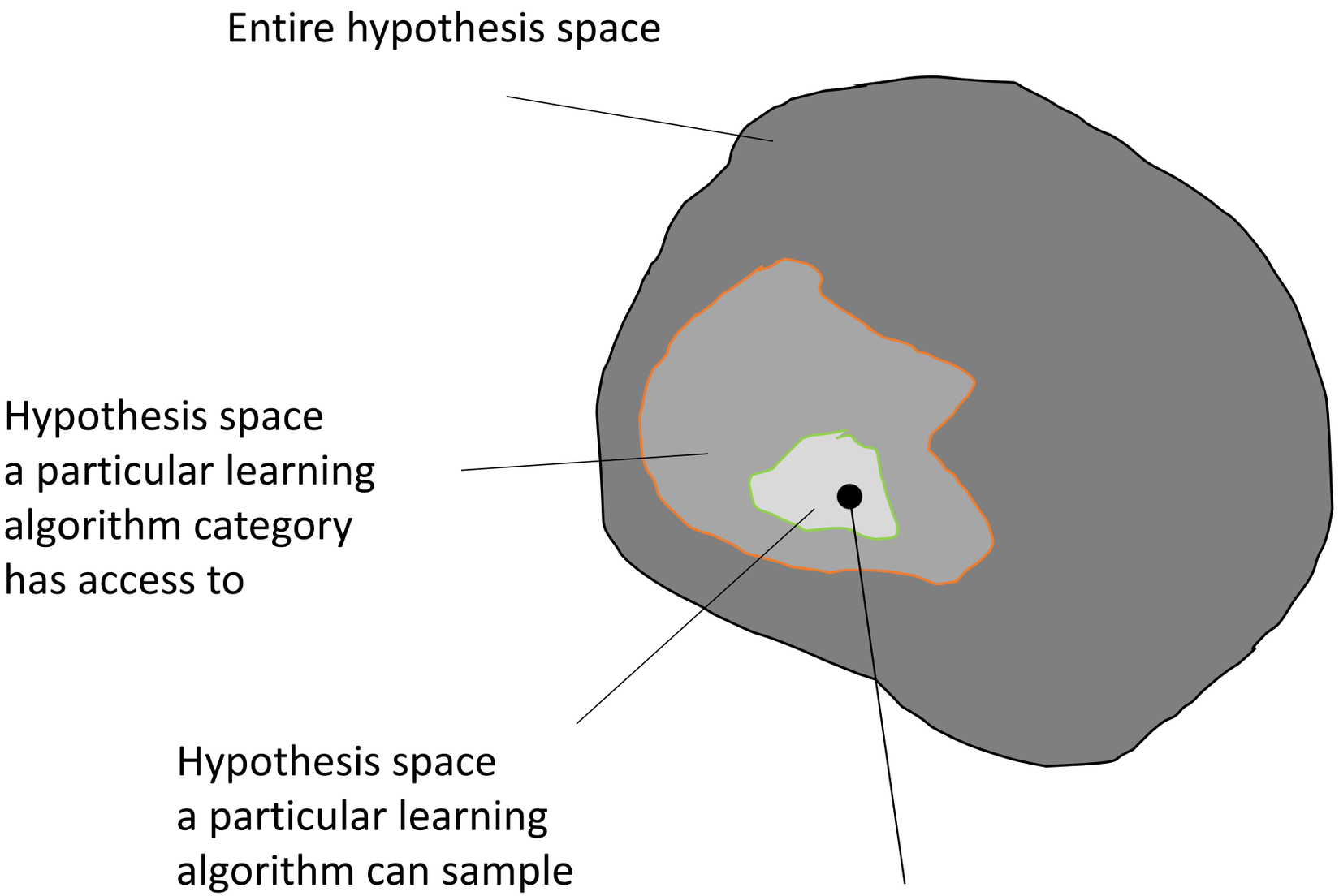
Supervised Learning Workflow



Supervised Learning Workflow (more detailed)



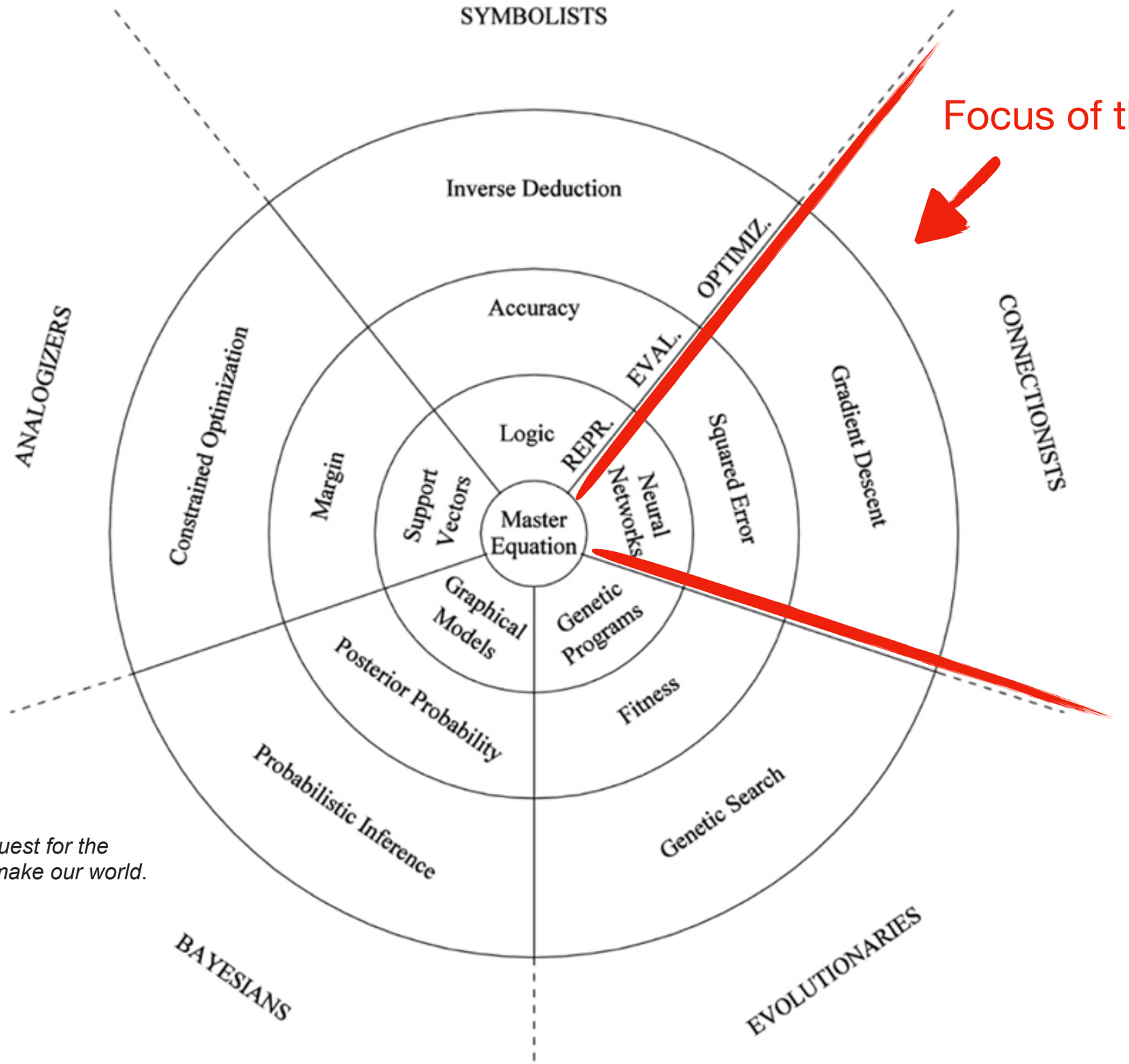
Hypothesis Space



5 Steps for Approaching an ML/DL Problem

1. Define the problem to be solved.
2. Collect (labeled) data.
3. Choose an algorithm class.
4. Choose an optimization metric for learning the model.
5. Choose a metric for evaluating the model.

Pedro Domingo's 5 Tribes of Machine Learning



Source: Domingos, Pedro.
The Master Algorithm: How the quest for the ultimate learning machine will remake our world.
Basic Books, 2015.

Learning = Representation + Evaluation + Optimization

(Pedro Domingos, *A Few Useful Things to Know about Machine Learning* https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf)

Objective Functions / Surrogate Risk/Loss

- Maximize the posterior probabilities (e.g., naive Bayes)
- Maximize a fitness function (genetic programming)
- Maximize the total reward/value function (reinforcement learning)
- Maximize information gain/minimize child node impurities (CART decision tree classification)
- Minimize a mean squared error cost (or loss) function (CART, decision tree regression, linear regression, adaptive linear neurons, ...)
- Maximize log-likelihood or minimize cross-entropy loss (or cost) function
- Minimize hinge loss (support vector machine)

Optimization Methods

- Combinatorial search, greedy search (e.g., decision trees)
- Unconstrained convex optimization (e.g., logistic regression)
- Constrained convex optimization (e.g., SVM)
- Nonconvex optimization, here: using backpropagation, chain rule, reverse autodiff. (e.g., neural networks)
- Constrained nonconvex optimization (e.g., semi-adversarial nets)

0/1 Loss, Misclassification Error

$$L(\hat{y}, y) = \begin{cases} 0 & \textbf{if } \hat{y} = y \\ 1 & \textbf{if } \hat{y} \neq y \end{cases}$$

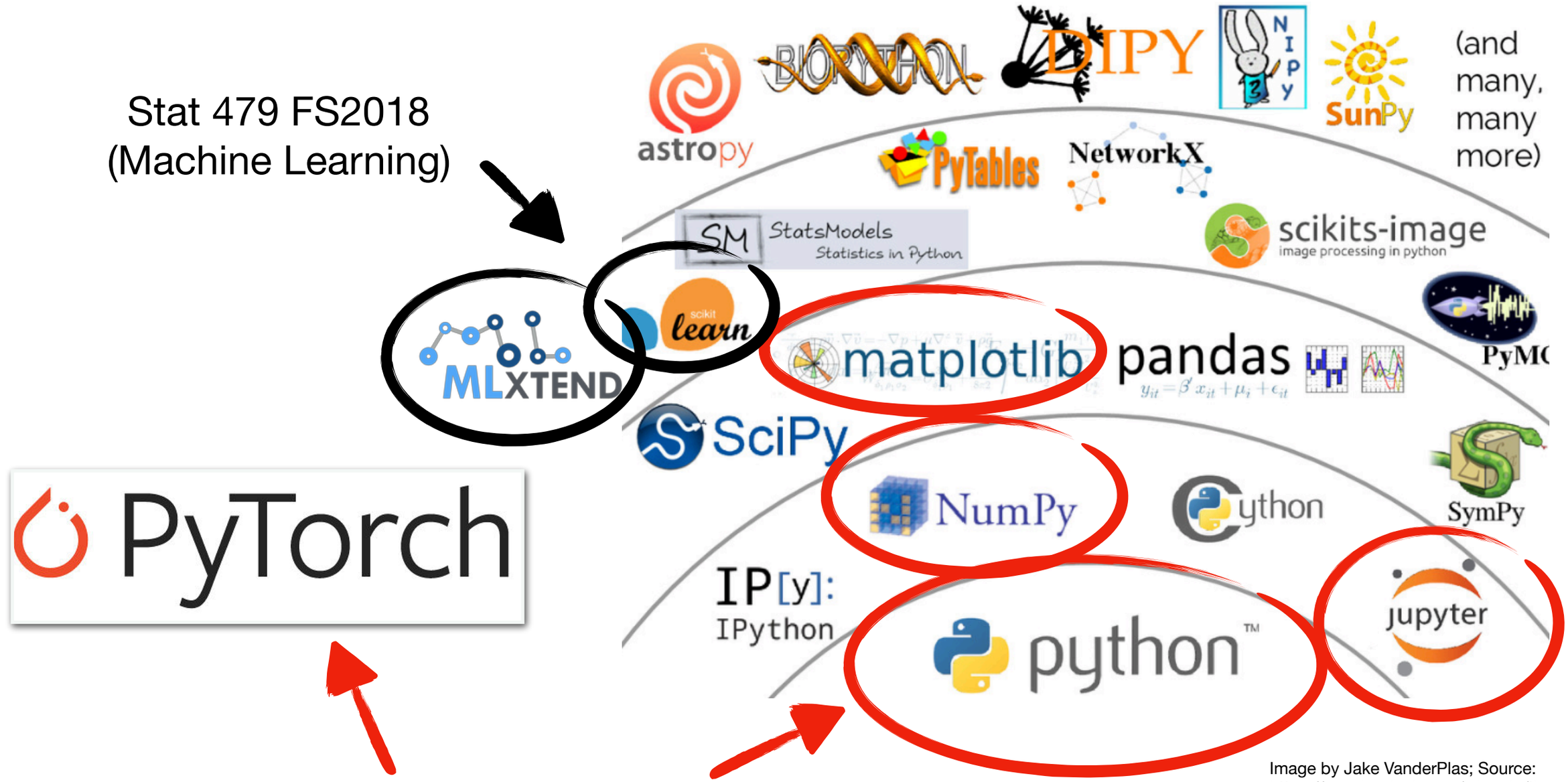
$$ERR_{\mathscr{D}}\textbf{test} = \frac{1}{n}\sum_{i=1}^{n} L\left(\hat{y}^{[i]}, y^{[i]}\right)$$

Other Performance Metrics

- Accuracy (1-Error)
- ROC AUC
- Precision
- Recall
- (Cross) Entropy
- Likelihood
- Squared Error/MSE
- L-norms
- Utility
- Fitness
- ...

Main Scientific Python Libraries

Stat 479 FS2018
(Machine Learning)



Main tools for this course

Image by Jake VanderPlas; Source: https://speakerdeck.com/jakevdp/the-state-of-the-stack-scipy-2015-keynote?slide=8)

Next Lecture:

A Brief Summary of the History of Neural Networks and Deep Learning

Reading Assignments

- Pedro Domingos, *A Few Useful Things to Know about Machine Learning* https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf

- STAT479 FS2018: Machine Learning, lecture notes 01: https://github.com/rasbt/stat479-machine-learning-fs18/blob/master/01_overview/01_ml-overview_notes.pdf

(exam questions also assume that you read the assigned reading materials)