Machine Learning CS229/STATS229

Instructors: Moses Charikar, Tengyu Ma, and Chris Re

Hope everyone stays safe and healthy in these difficult times!

A simple example: predicting electricity use

What will peak power consumption be in Pittsburgh tomorrow?

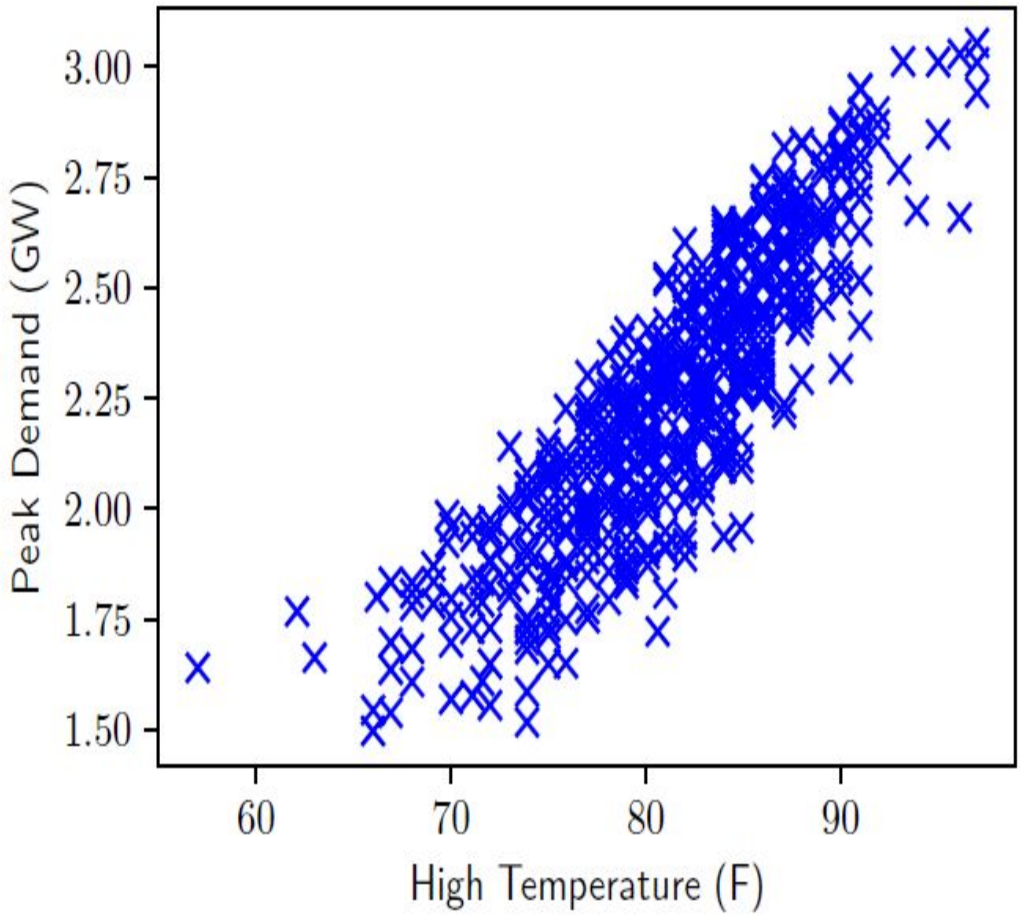
Difficult to build an "a priori" model from first principles to answer this question

But, relatively easy to record past days of consumption, plus additional features that affect consumption (i.e., weather)

Date	High Temperature (F)	Peak Demand (GW)
2011-06-01	84.0	2.651
2011-06-02	73.0	2.081
2011-06-03	75.2	1.844
2011-06-04	84.9	1.959
…	…	…

Plot of consumption vs. temperature

Plot of high temperature vs. peak demand for summer months (June – August) for past six years



Hypothesis: linear model

$(x1, y1)$ $(x2, y2)$ => $SQRT((X1-X2)^2 + (Y1-Y2)^2)$

$(Y_{Observed})$ (Y_{exp}) => SUM $(SQRT\ (Y_{Observed} - Y_{exp})^2)$

Let's suppose that the peak demand approximately fits a *linear model*

$Y = mx + c$

$$\text{Peak\_Demand} \approx \theta_1 \cdot \text{High\_Temperature} + \theta_2$$

Here θ_1 is the "slope" of the line, and θ_2 is the intercept

How do we find a "good" fit to the data? $Y = m1x1 + m2x2 + c$

$PeakDemand = T_1 . HT + T_2 . Humidity + T_3$

Many possibilities, but natural objective is to minimize some difference between this line and the observed data, e.g. squared loss

$E = sum\ ((mx + c - Y)^2)$

$$E(\theta) = \sum_{i \in \text{days}} \left(\theta_1 \cdot \text{High\_Temperature}^{(i)} + \theta_2 - \text{Peak\_Demand}^{(i)}\right)^2$$

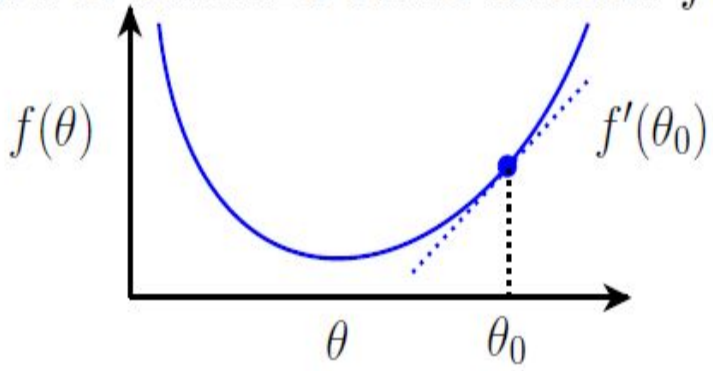
How do we find parameters?

How do we find the parameters θ_1, θ_2 that minimize the function

$$E(\theta) = \sum_{i \in \text{days}} \left(\theta_1 \cdot \text{High\_Temperature}^{(i)} + \theta_2 \ - \text{Peak\_Demand}^{(i)}\right)^2$$

$$\equiv \sum_{i \in \text{days}} \left(\theta_1 \cdot x^{(i)} + \theta_2 \ - y^{(i)}\right)^2$$

General idea: suppose we want to minimize some function $f(\theta)$



Derivative is slope of the function, so negative derivative points "downhill"

Computing the derivatives

What are the derivatives of the error function with respect to each parameter θ_1 and θ_2?

$$\frac{\partial E(\theta)}{\partial \theta_1} = \frac{\partial}{\partial \theta_1} \sum_{i \in \text{days}} \left(\theta_1 \cdot x^{(i)} + \theta_2 - y^{(i)}\right)^2$$

$$= \sum_{i \in \text{days}} \frac{\partial}{\partial \theta_1} \left(\theta_1 \cdot x^{(i)} + \theta_2 - y^{(i)}\right)^2$$

$$= \sum_{i \in \text{days}} 2\left(\theta_1 \cdot x^{(i)} + \theta_2 - y^{(i)}\right) \cdot \frac{\partial}{\partial \theta_1} \theta_1 \cdot x^{(i)}$$

$$= \sum_{i \in \text{days}} 2\left(\theta_1 \cdot x^{(i)} + \theta_2 - y^{(i)}\right) \cdot x^{(i)}$$

$$\frac{\partial E(\theta)}{\partial \theta_2} = \sum_{i \in \text{days}} 2\left(\theta_1 \cdot x^{(i)} + \theta_2 - y^{(i)}\right)$$

$$\frac{d(T_2 + T_2 - Y)^2)}{dT_2}$$

$$\frac{2(T_2 + T_2 - Y)}{T_2}$$

$$\frac{2(0 + T_2 - 0)}{dT_2}$$

Finding the best θ

To find a good value of θ, we can repeatedly take steps in the direction of the negative derivatives for each value

Repeat:

$$\theta_1 := \theta_1 - \alpha \sum_{i \in \text{days}} 2\left(\theta_1 \cdot x^{(i)} + \theta_2 \ - y^{(i)}\right) \cdot x^{(i)}$$

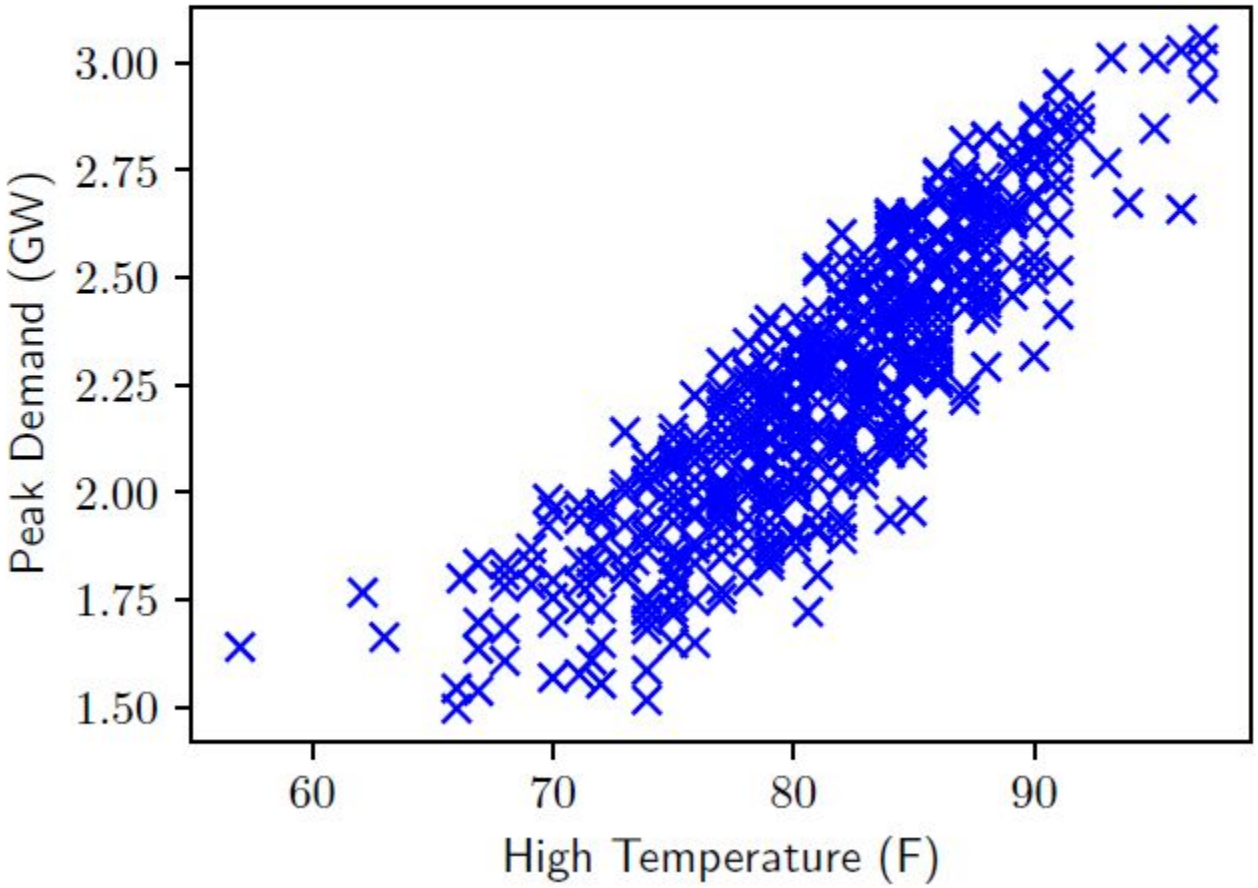
$$\theta_2 := \theta_2 - \alpha \sum_{i \in \text{days}} 2\left(\theta_1 \cdot x^{(i)} + \theta_2 \ - y^{(i)}\right)$$

where α is some small positive number called the *step size*

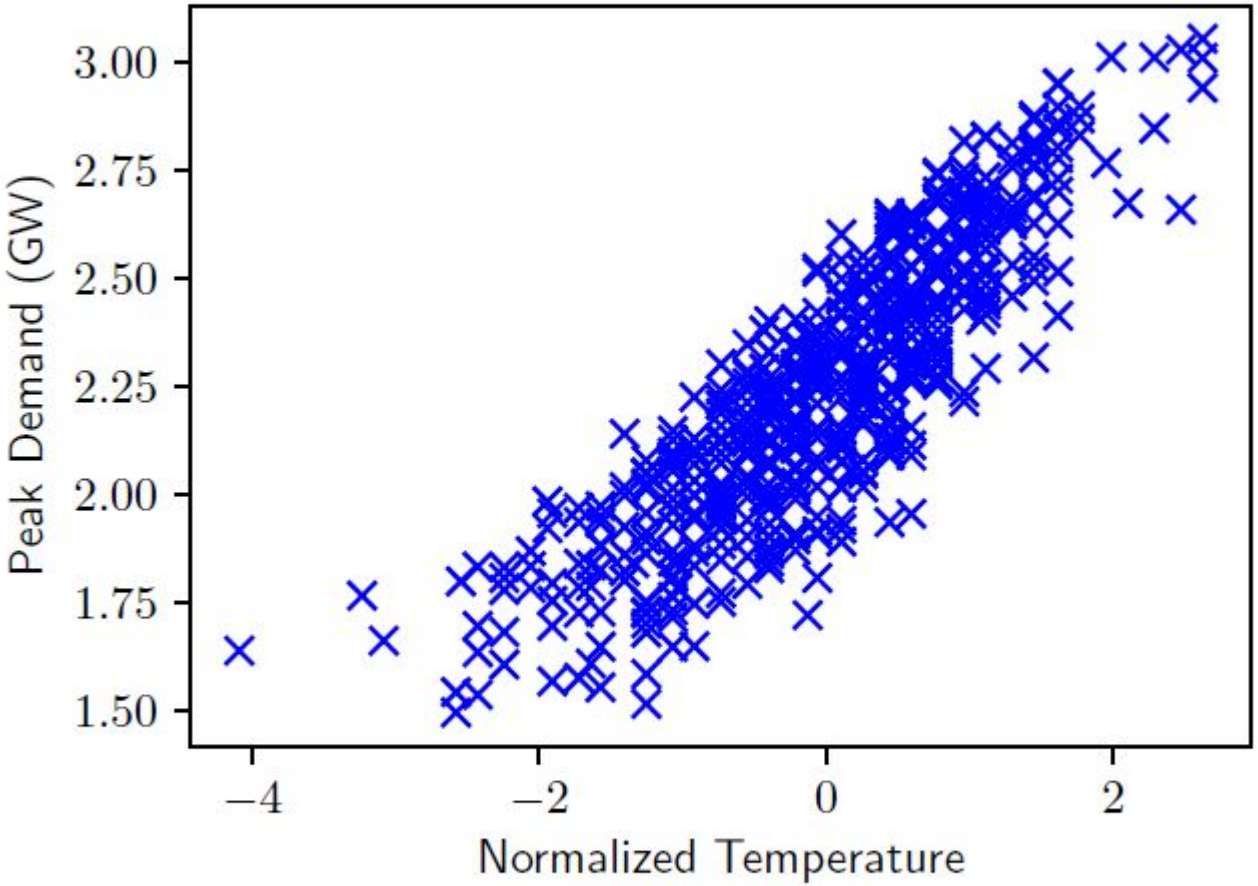
This is the *gradient decent algorithm*, the workhorse of modern machine learning

T1	T2	E	T_E
3	2	5	0
2.5 Dec	2.5	6	
3.5 Inc	2.5	5.2	
2.5 up to 3.5 3.1			

Gradient descent

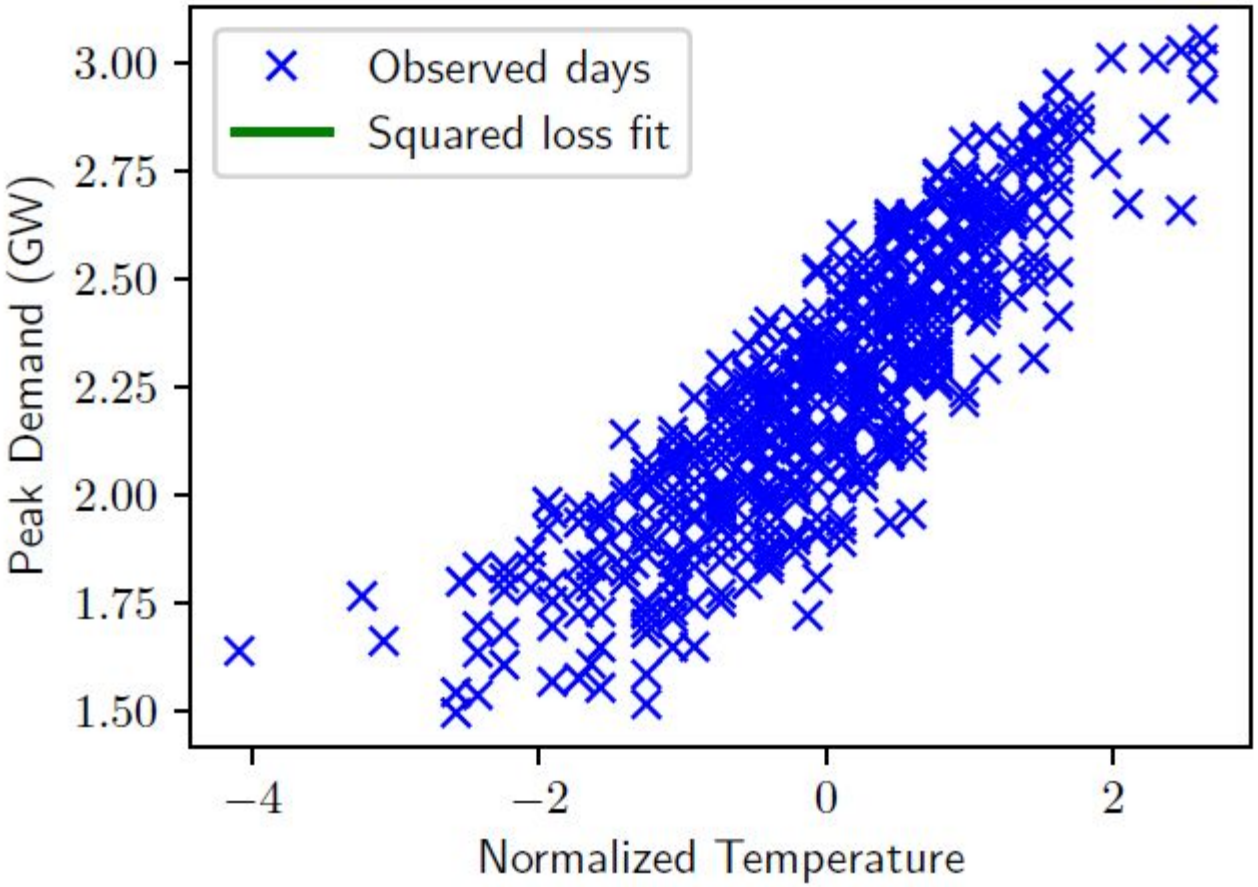


Gradient descent



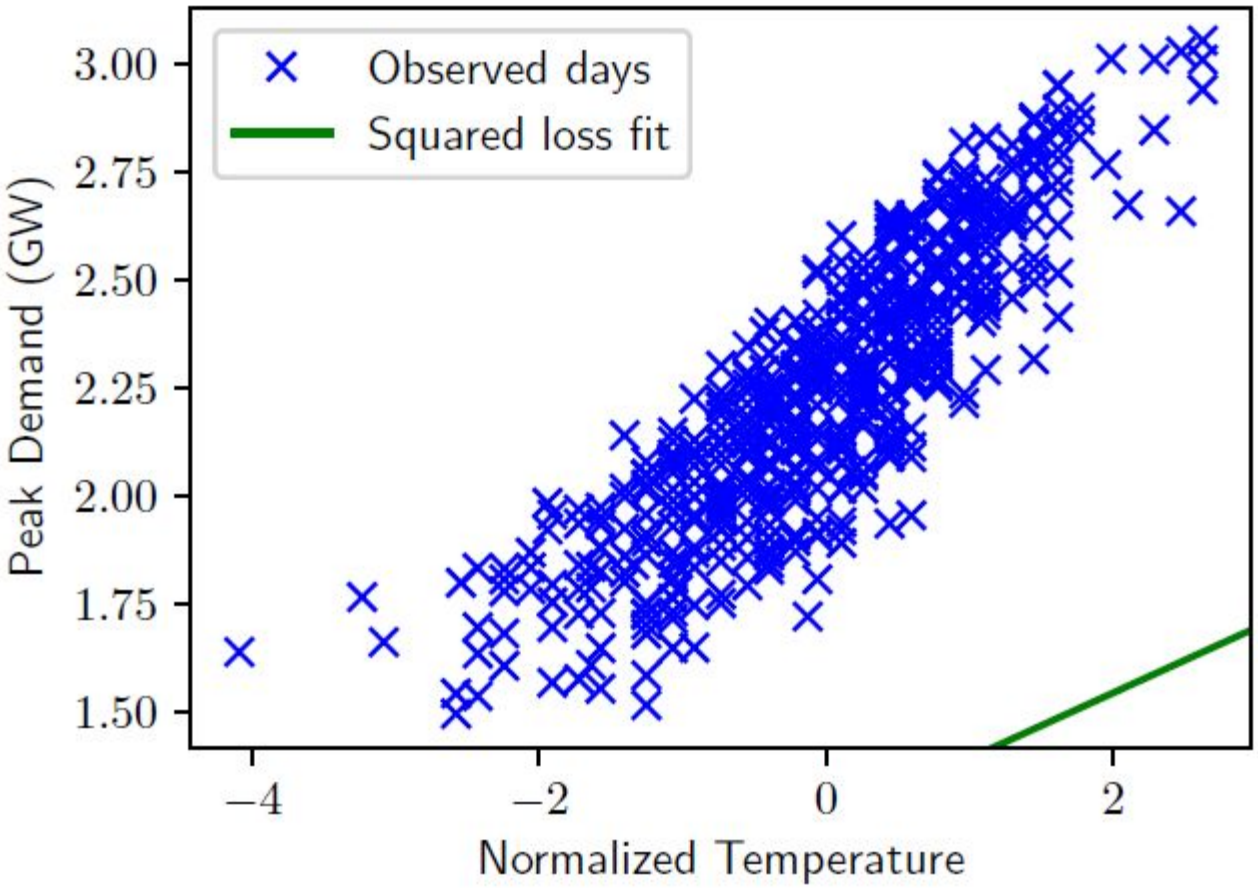
Normalize input by subtracting the mean and dividing by the standard deviation

Gradient descent – Iteration 1



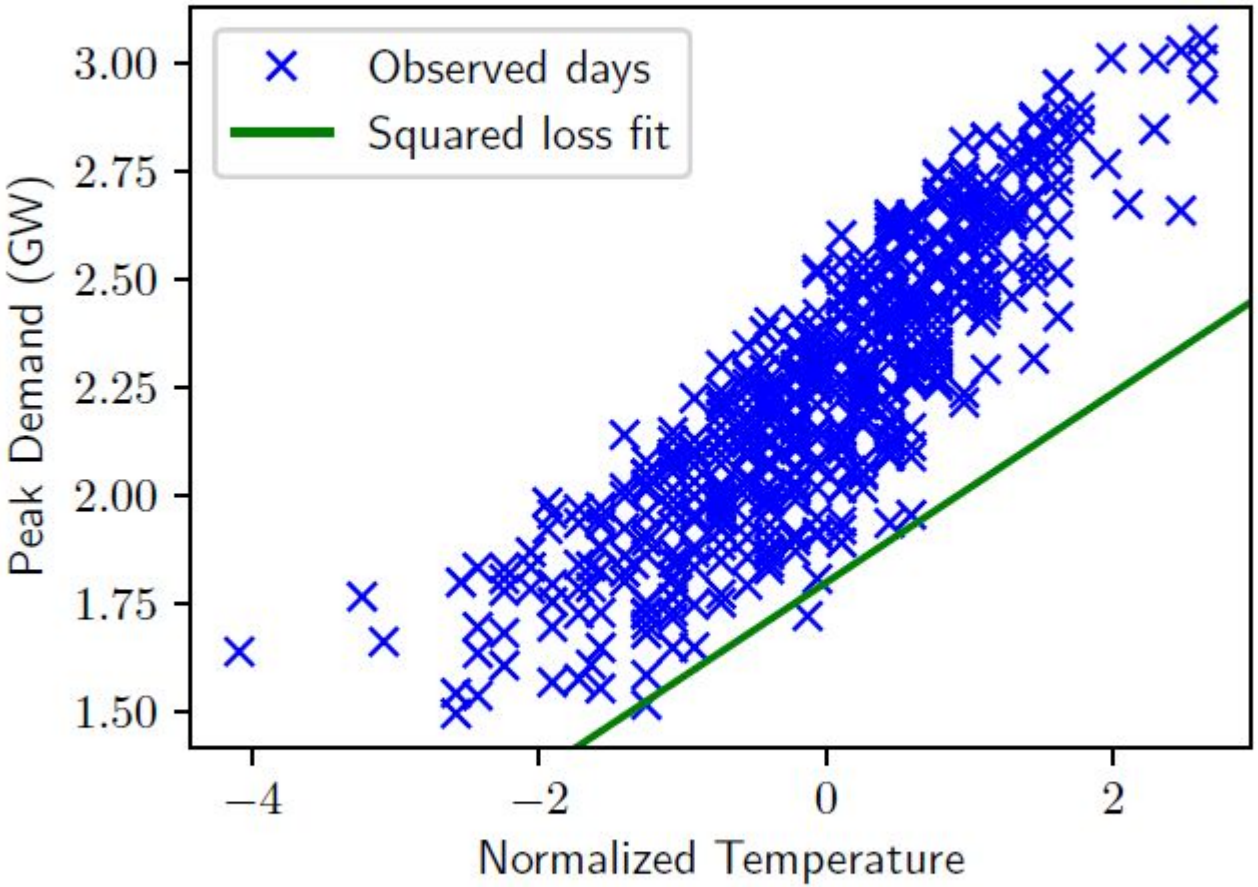
$\theta = (0.00, 0.00)$
$E(\theta) = 1427.53$
$(\frac{\partial E(\theta)}{\partial \theta_1}, \frac{\partial E(\theta)}{\partial \theta_2}) = (-151.20, -1243.10)$

Gradient descent – Iteration 2



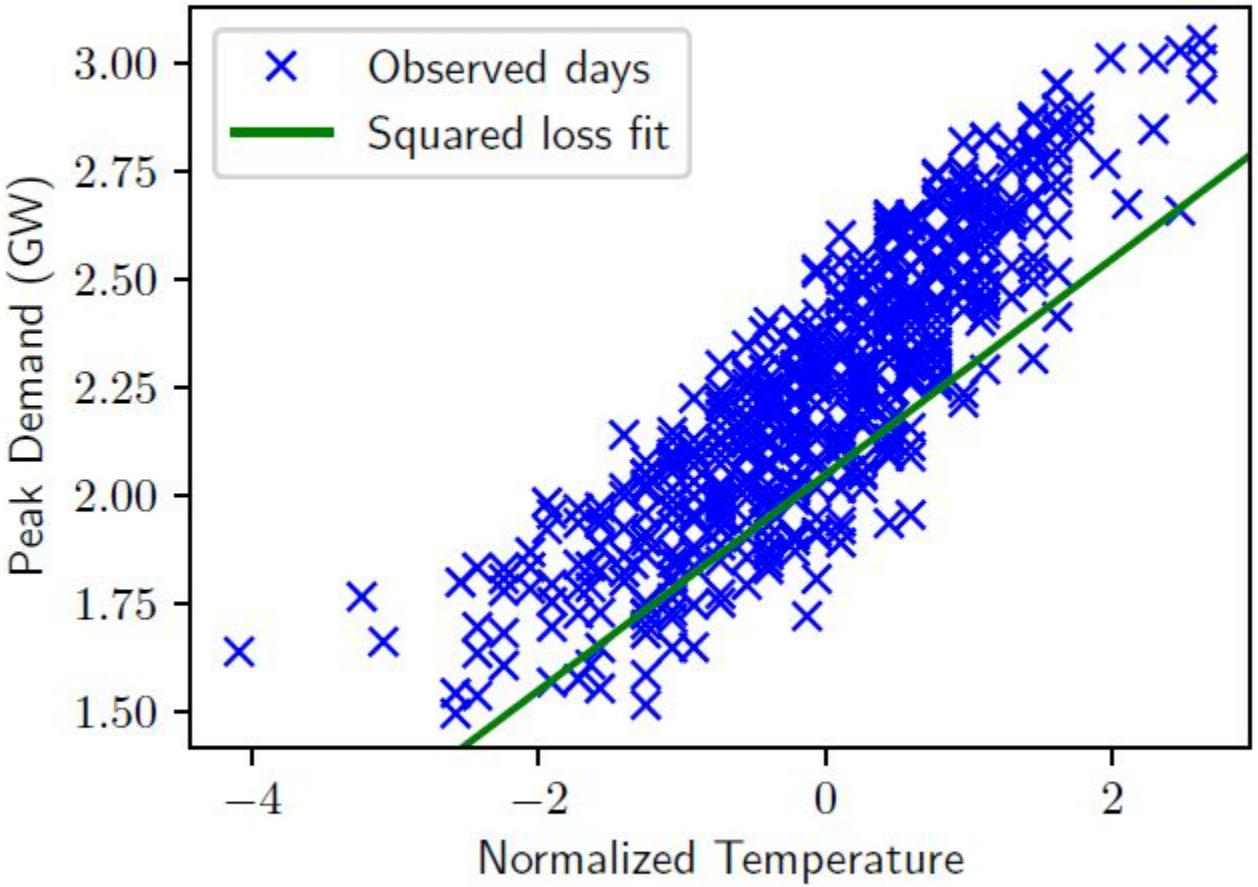
$\theta = (0.15, 1.24)$
$E(\theta) = 292.18$
$(\frac{\partial E(\theta)}{\partial \theta_1}, \frac{\partial E(\theta)}{\partial \theta_2}) = (-67.74, -556.91)$

Gradient descent – Iteration 3



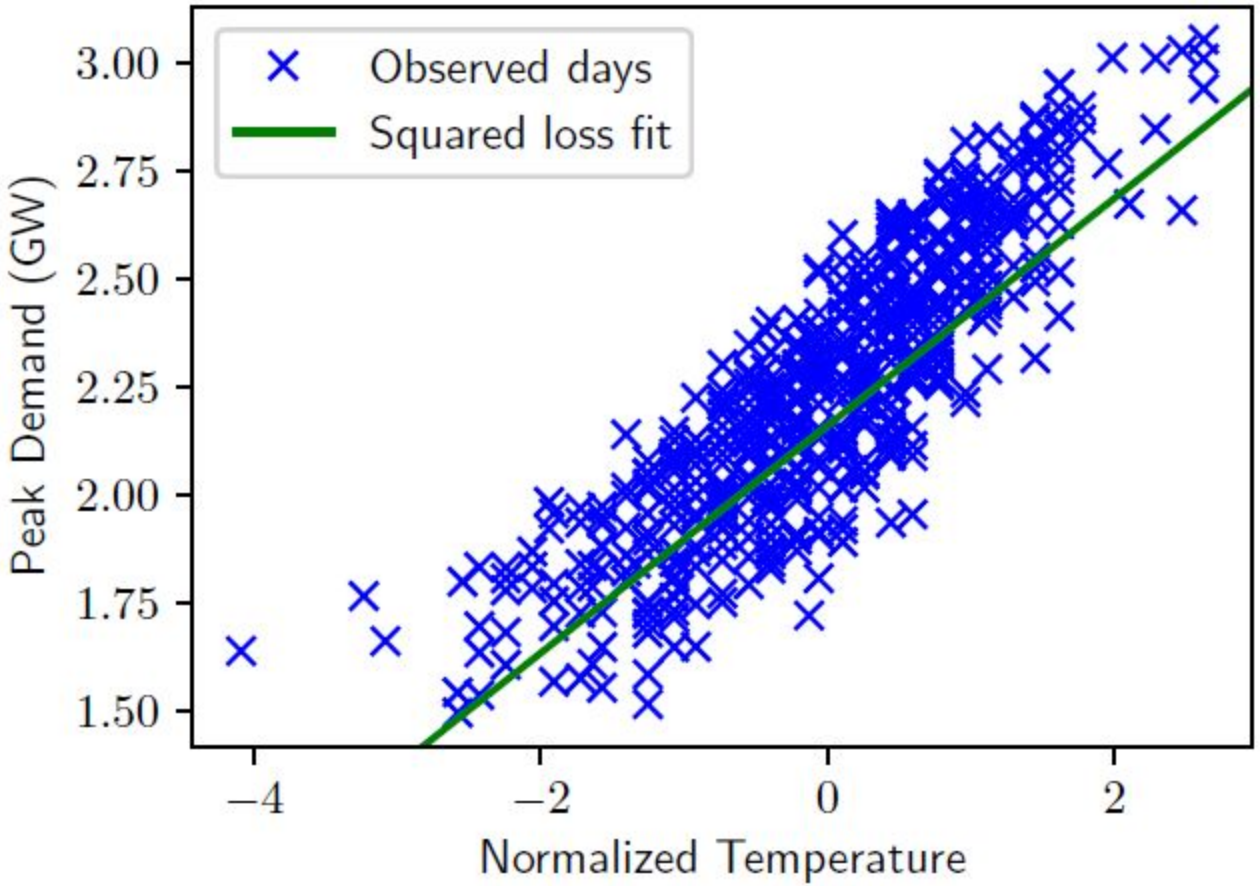
$\theta = (0.22, 1.80)$
$E(\theta) = 64.31$
$(\frac{\partial E(\theta)}{\partial \theta_1}, \frac{\partial E(\theta)}{\partial \theta_2}) = (-30.35, -249.50)$

Gradient descent – Iteration 4



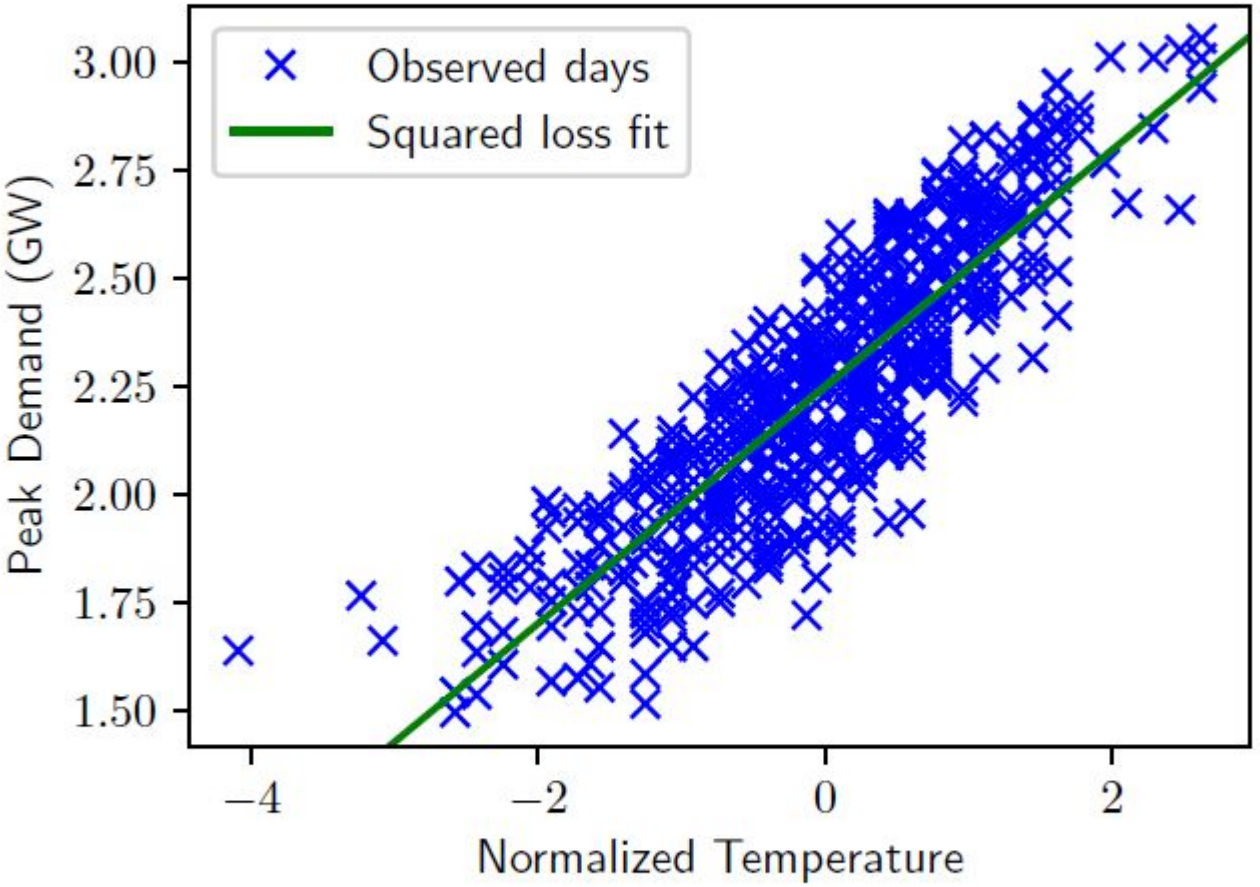
$\theta = (0.25, 2.05)$
$E(\theta) = 18.58$
$(\frac{\partial E(\theta)}{\partial \theta_1}, \frac{\partial E(\theta)}{\partial \theta_2}) = (-13.60, -111.77)$

Gradient descent – Iteration 5



$\theta = (0.26, 2.16)$
$E(\theta) = 9.40$
$(\frac{\partial E(\theta)}{\partial \theta_1}, \frac{\partial E(\theta)}{\partial \theta_2}) = (-6.09, -50.07)$

Gradient descent – Iteration 10

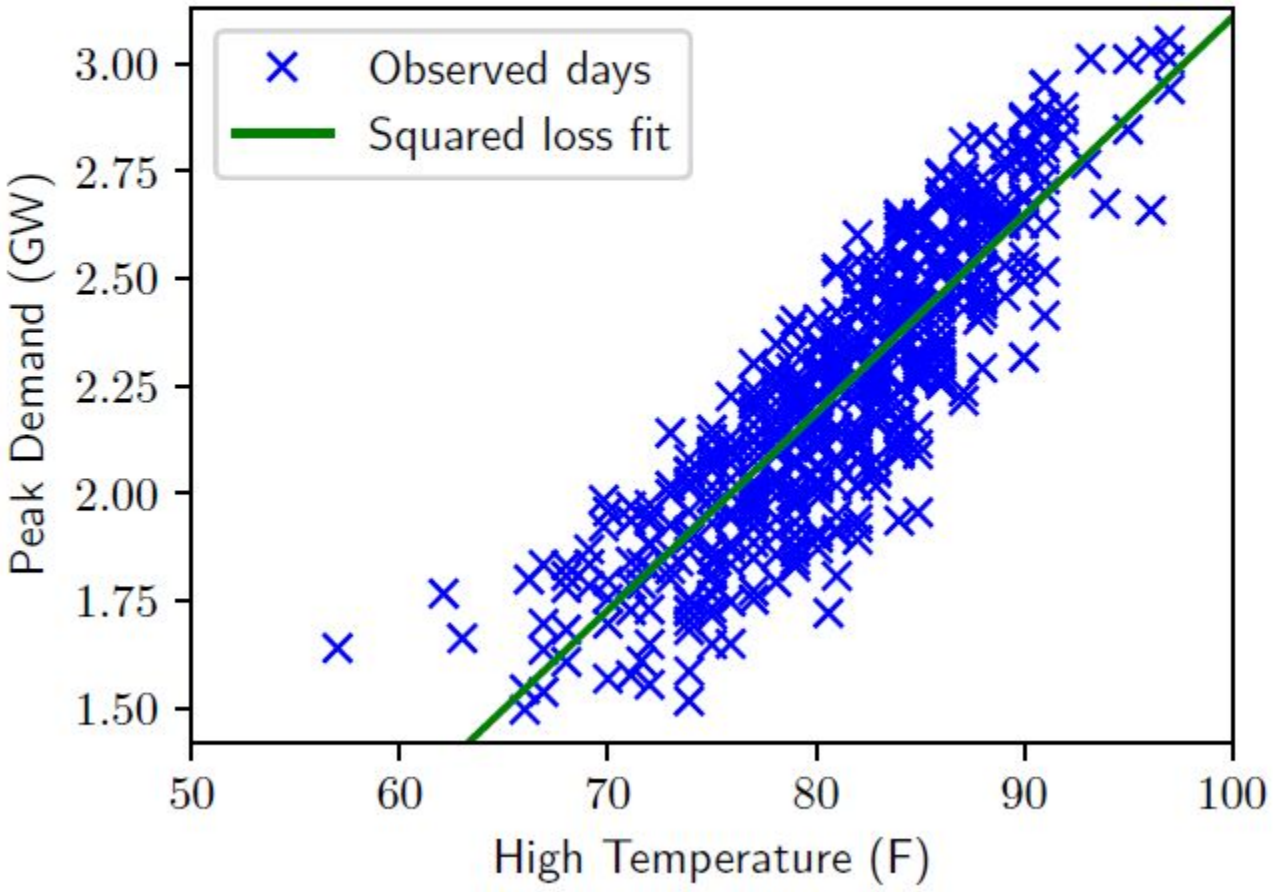


$\theta = (0.27, 2.25)$

$E(\theta) = 7.09$

$(\frac{\partial E(\theta)}{\partial \theta_1}, \frac{\partial E(\theta)}{\partial \theta_2}) = (-0.11, -0.90)$

Fitted line in “original” coordinates



Making predictions

Importantly, our model also lets us make *predictions* about new days

What will the peak demand be tomorrow?

If we know the high temperature will be 72 degrees (ignoring for now that this is *also* a prediction), then we can predict peak demand to be:

$$\text{Predicted\_demand} = \theta_1 \cdot 72 + \theta_2 = 1.821 \text{ GW}$$

(requires that we rescale θ after solving to "normal" coordinates)

Equivalent to just "finding the point on the line"

Extensions

What if we want to add additional features, e.g. day of week, instead of just temperature?

What if we want to use a different loss function instead of squared error (i.e., absolute error)?

What if we want to use a non-linear prediction instead of a linear one?

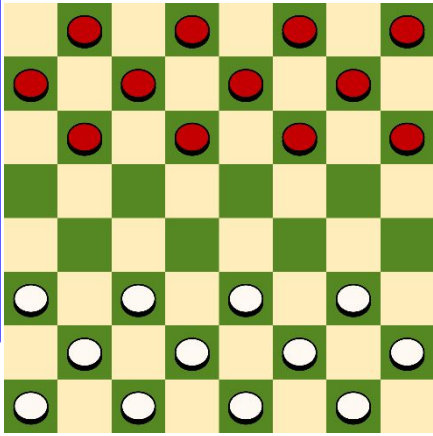
We can easily reason about all these things by adopting some additional notation…

Definition of Machine Learning

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

A. L. Samuel*

Some Studies in Machine Learning
Using the Game of Checkers. II—Recent Progress



Photos from Wikipedia

Definition of Machine Learning

Tom Mitchell (1998): a computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Experience (data): games played by the program (with itself)

Performance measure: winning rate

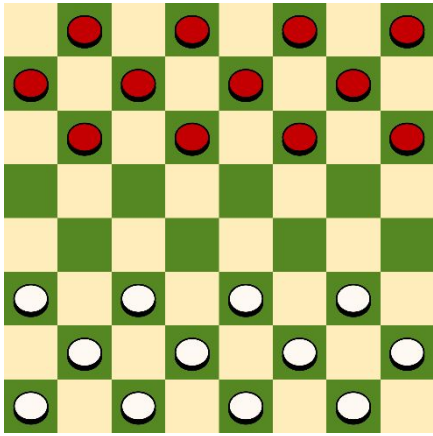
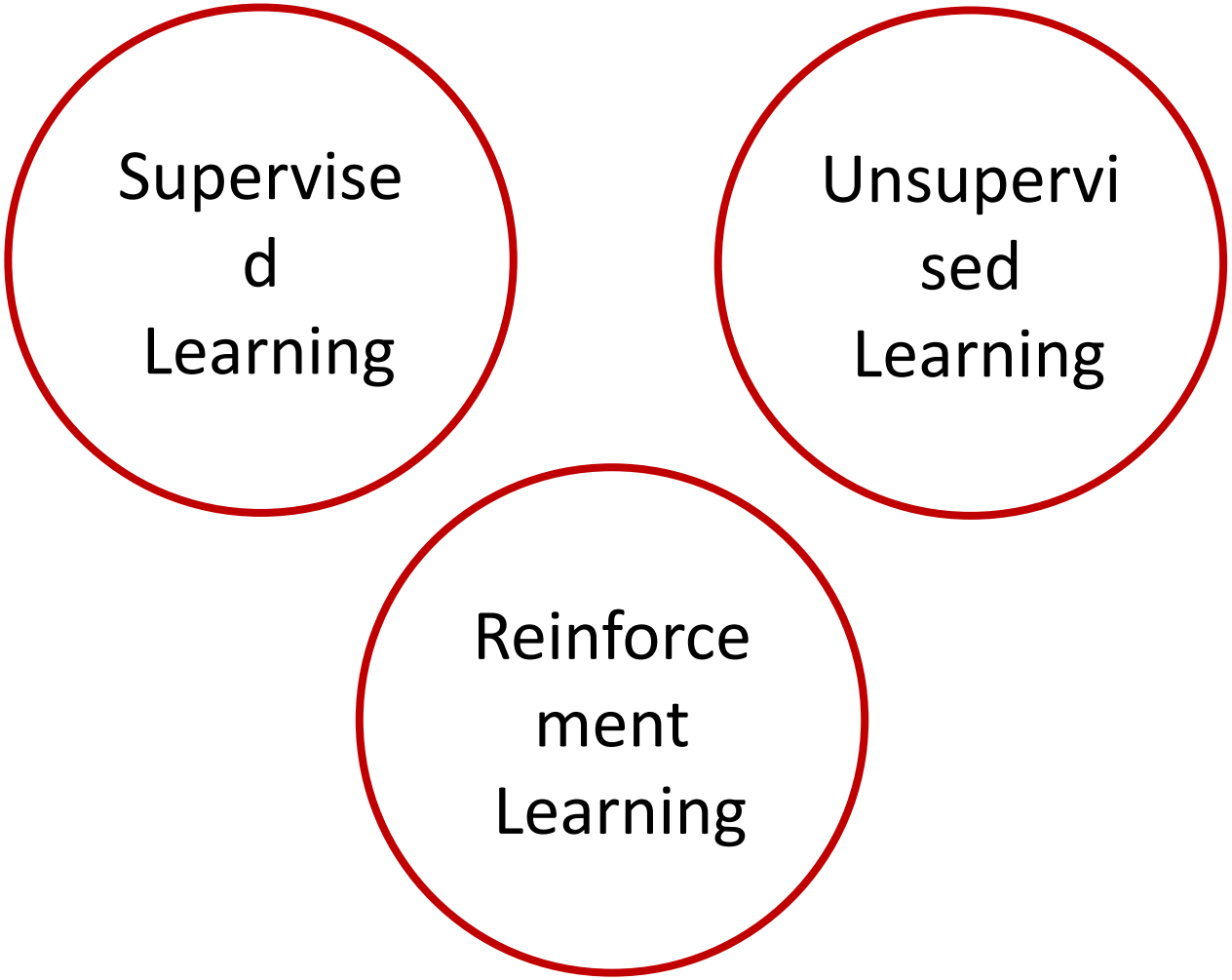
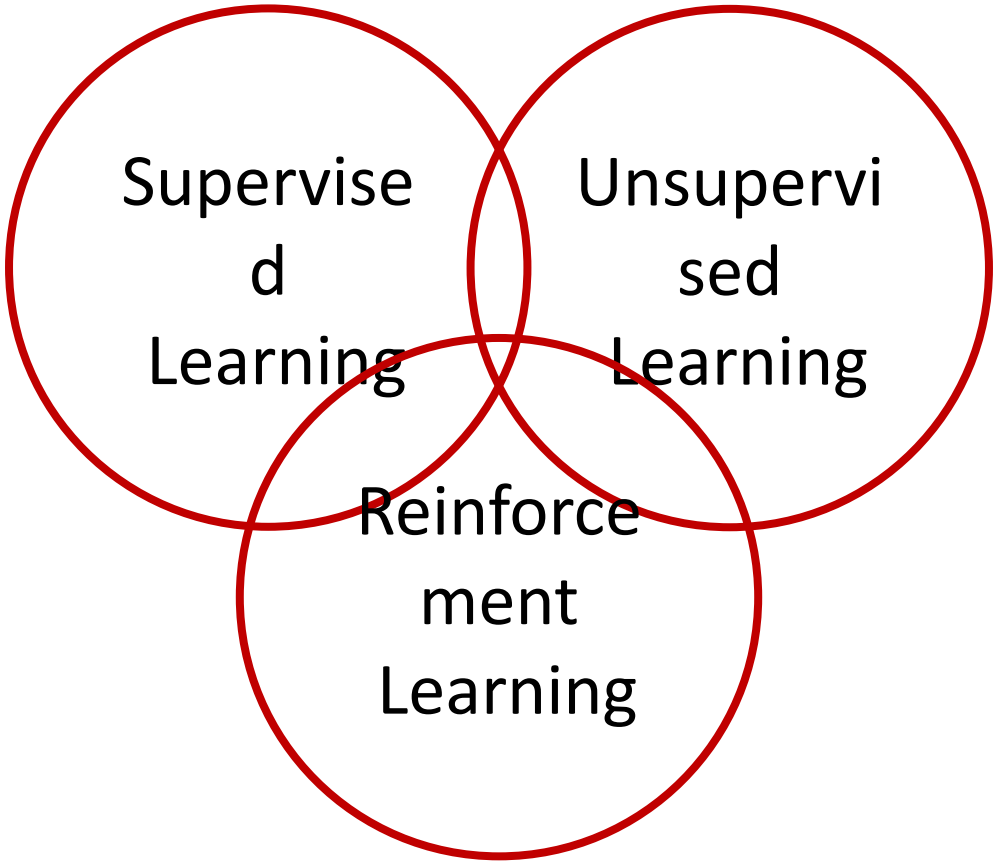


Image from Tom Mitchell's homepage

Taxonomy of Machine Learning (A Simplistic View Based on Tasks)



Taxonomy of Machine Learning (A Simplistic View Based on Tasks)



can also be viewed as tools/methods

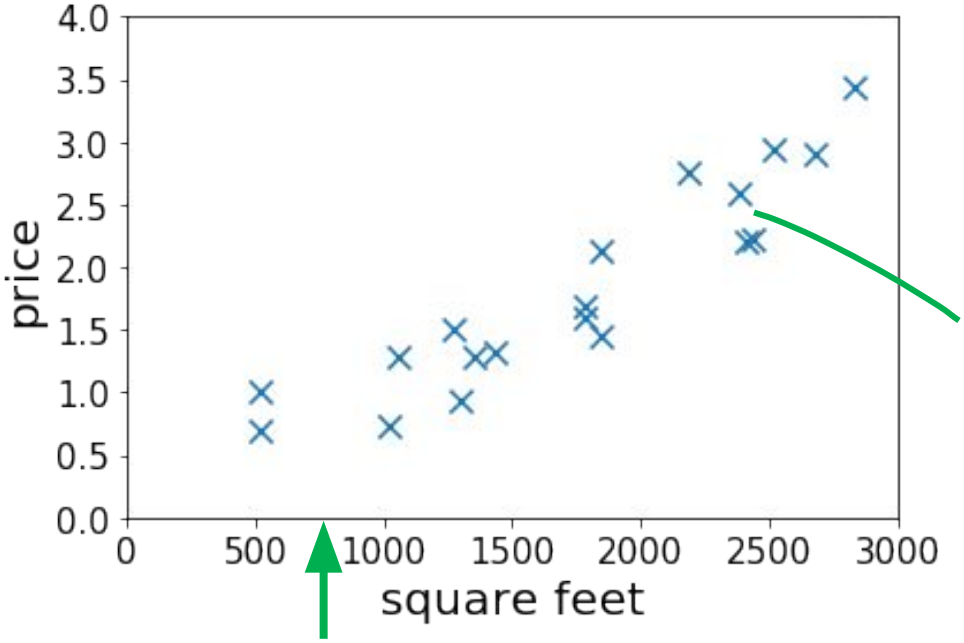
Supervised Learning

Housing Price Prediction

- Given: a dataset that contains n samples

$$(x^{(1)}, y^{(1)}), \ldots (x^{(n)}, y^{(n)})$$

- Task: if a residence has x square feet, predict its price?



15th sample
$(x^{(15)}, y^{(15)})$

$x = 800$

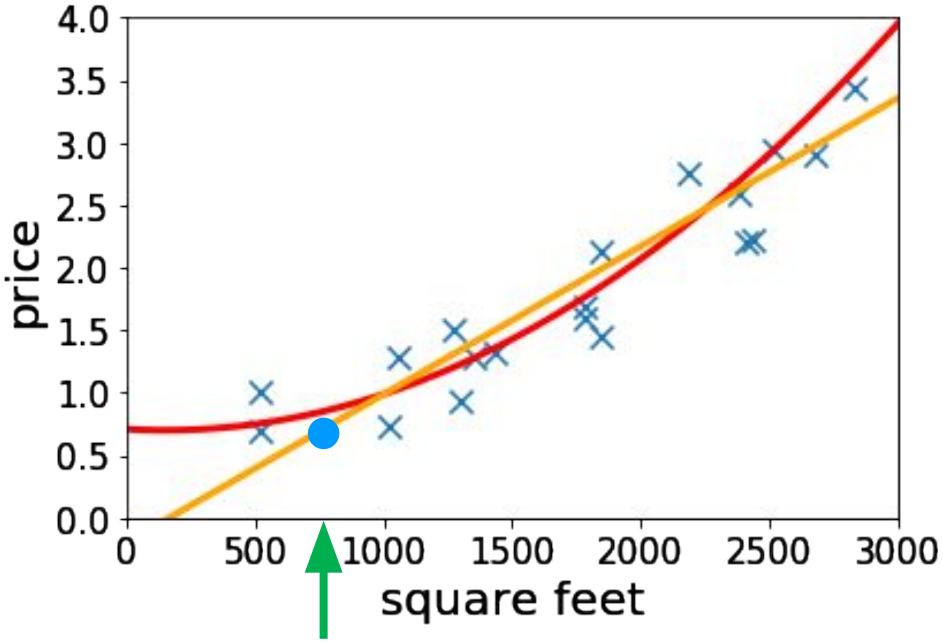
$y = ?$

Housing Price Prediction

- Given: a dataset that contains n samples

$$\left(x^{(1)}, y^{(1)}\right), \ldots \left(x^{(n)}, y^{(n)}\right)$$

- Task: if a residence has x square feet, predict its price?



- Lecture 2&3: fitting linear/qaudratic functions to the dataset

More Features

- Suppose we also know the lot size
- Task: find a function that maps

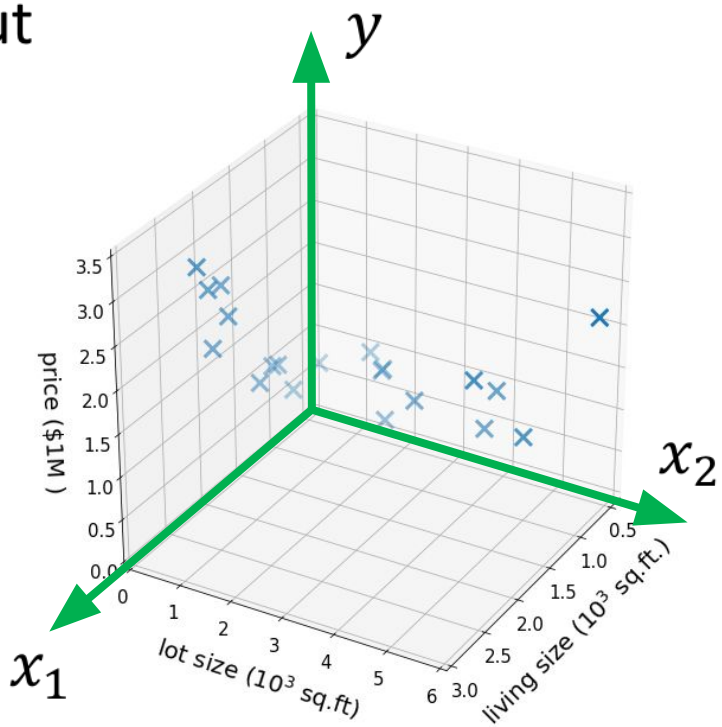
$$\underbrace{(\text{size, lot size})}_{\text{features/input}} \rightarrow \underbrace{\text{price}}_{\text{label/output}}$$

$$x \in \mathbb{R}^2 \qquad y \in \mathbb{R}$$

- Dataset: $\left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(n)}, y^{(n)}\right)$

where $x^{(i)} = (x_1^{(i)}, x_2^{(i)})$

- "Supervision" refers to $y^{(1)}, \ldots, y^{(n)}$



High-dimensional Features

- $x \in \mathbb{R}^d$ for large d
- E.g.,

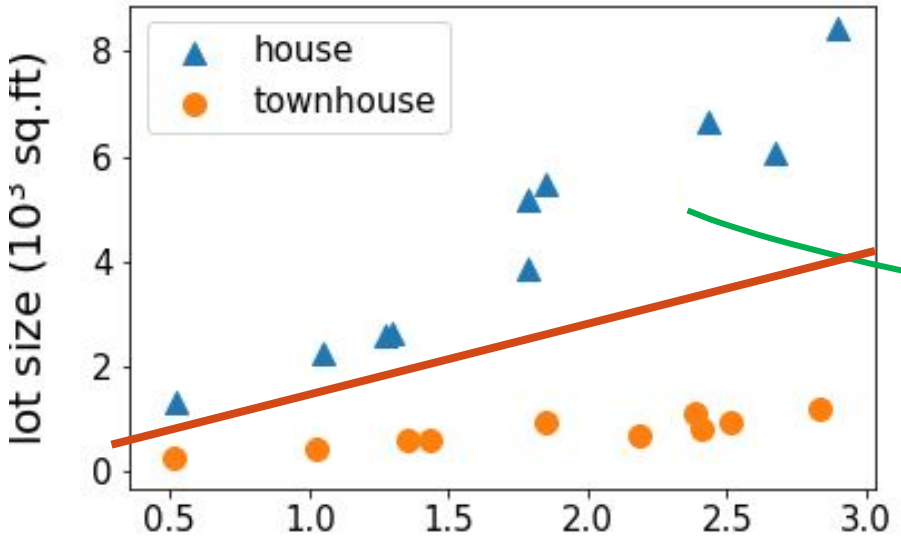
$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ \vdots \\ \vdots \\ x_d \end{bmatrix} \begin{matrix} \text{--- living size} \\ \text{--- lot size} \\ \text{--- \# floors} \\ \text{--- condition} \\ \text{--- zip code} \\ \vdots \end{matrix} \longrightarrow y \text{ --- price}$$

- Lecture 6-7: infinite dimensional features
- Lecture 10-11: select features based on the data

Regression vs Classification

- regression: if $y \in \mathbb{R}$ is a continuous variable
  - e.g., price prediction
- classification: the label is a discrete variable
  - e.g., the task of predicting the types of residence

(size, lot size) → house or townhouse?



y = house or townhouse?

Lecture 3&4: classification

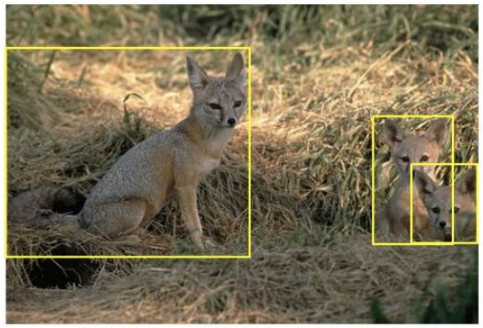
Supervised Learning in Computer Vision

- Image Classification
  - x = raw pixels of the image, y = the main object

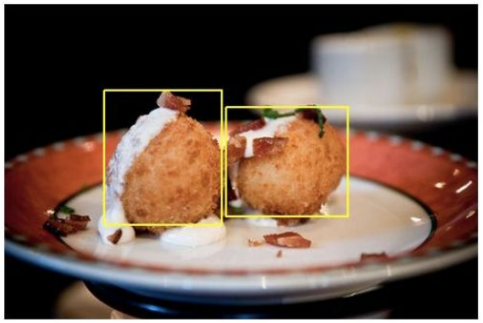
ImageNet Large Scale Visual Recognition Challenge. Russakovsky et al.'2015

Supervised Learning in Computer Vision

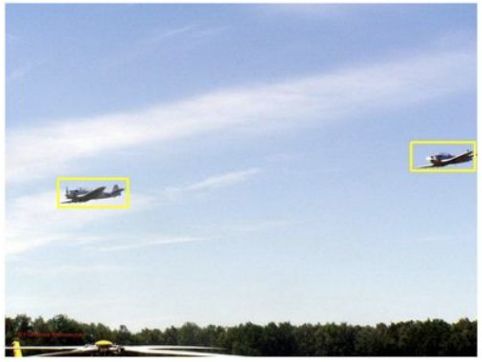
- Object localization and detection
  - x = raw pixels of the image, y = the bounding boxes



kit fox



croquette



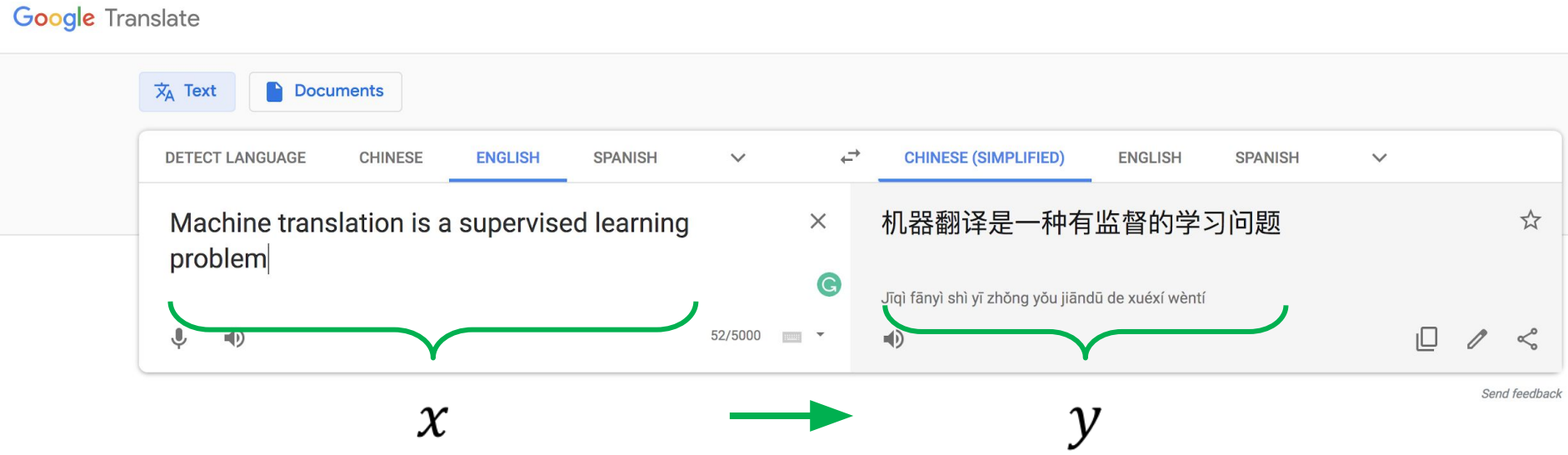
airplane

frog

ImageNet Large Scale Visual Recognition Challenge. Russakovsky et al.'2015

Supervised Learning in Natural Language Processing

- Machine translation



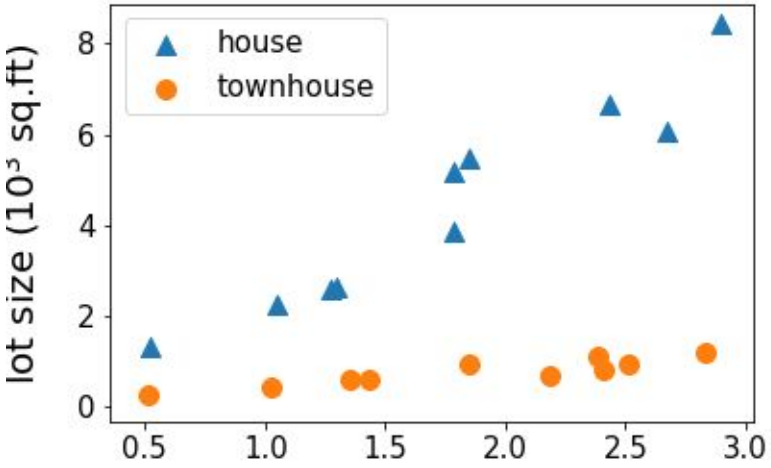
- Note: this course only covers the basic and fundamental techniques of supervised learning (which are not enough for solving hard vision or NLP problems.)
- CS224N and CS231N would be more suitable if you are interested in the particular applications

Unsupervised Learning

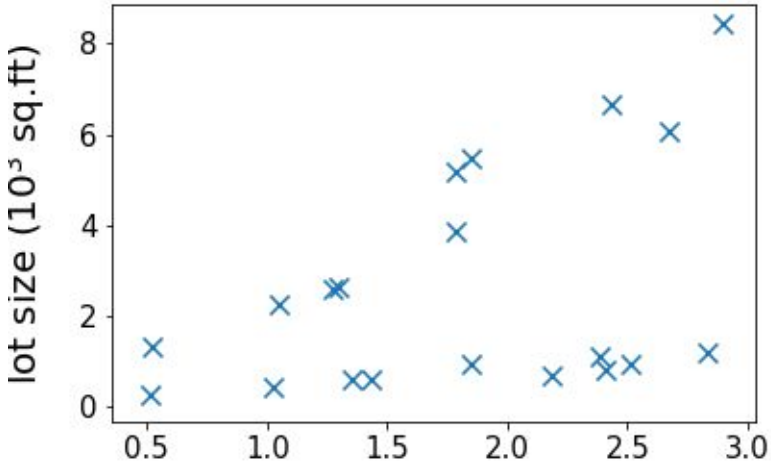
Unsupervised Learning

- Dataset contains no labels: $x^{(1)}, \ldots x^{(n)}$
- Goal (vaguely-posed): to find interesting structures in the data

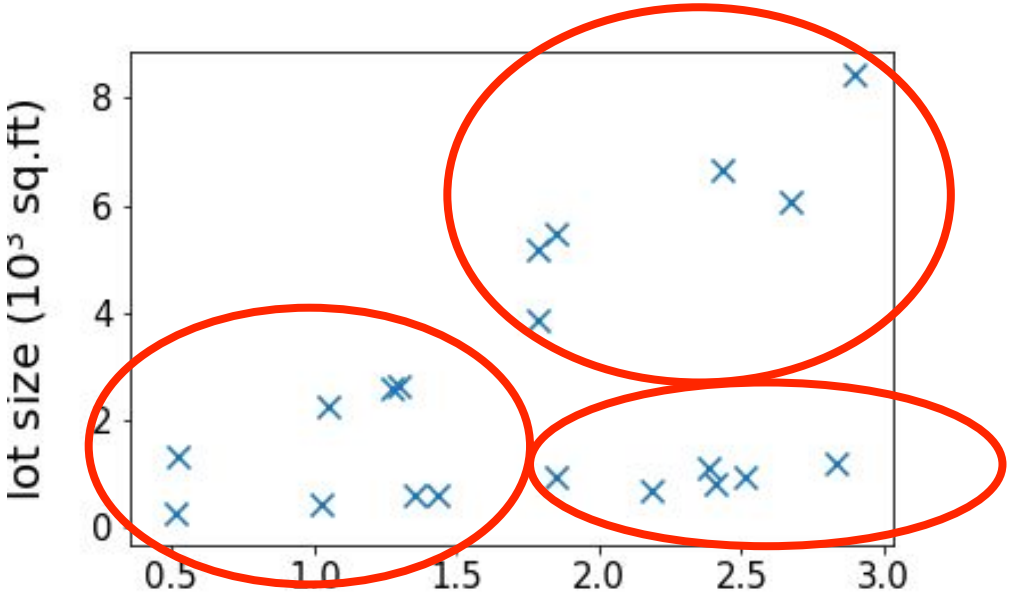
supervised



unsupervised

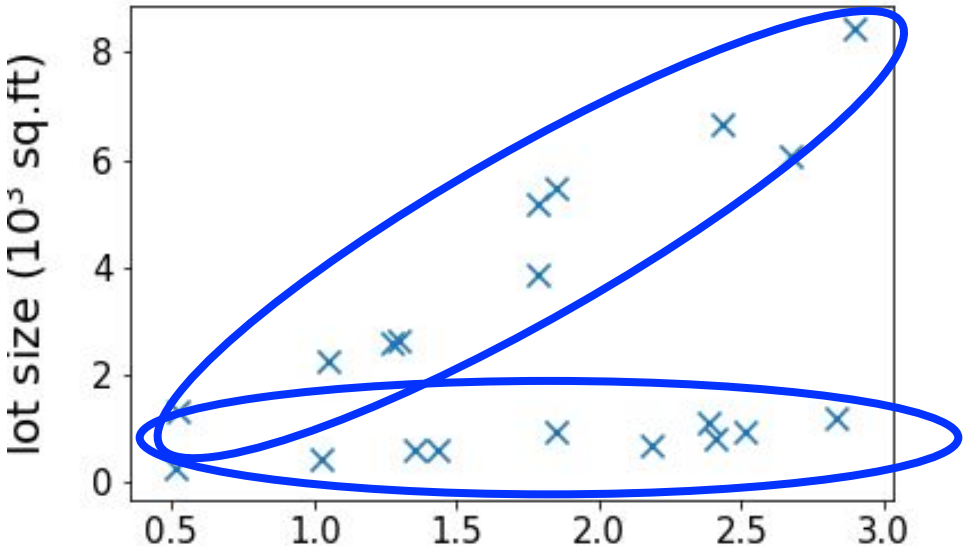


Clustering

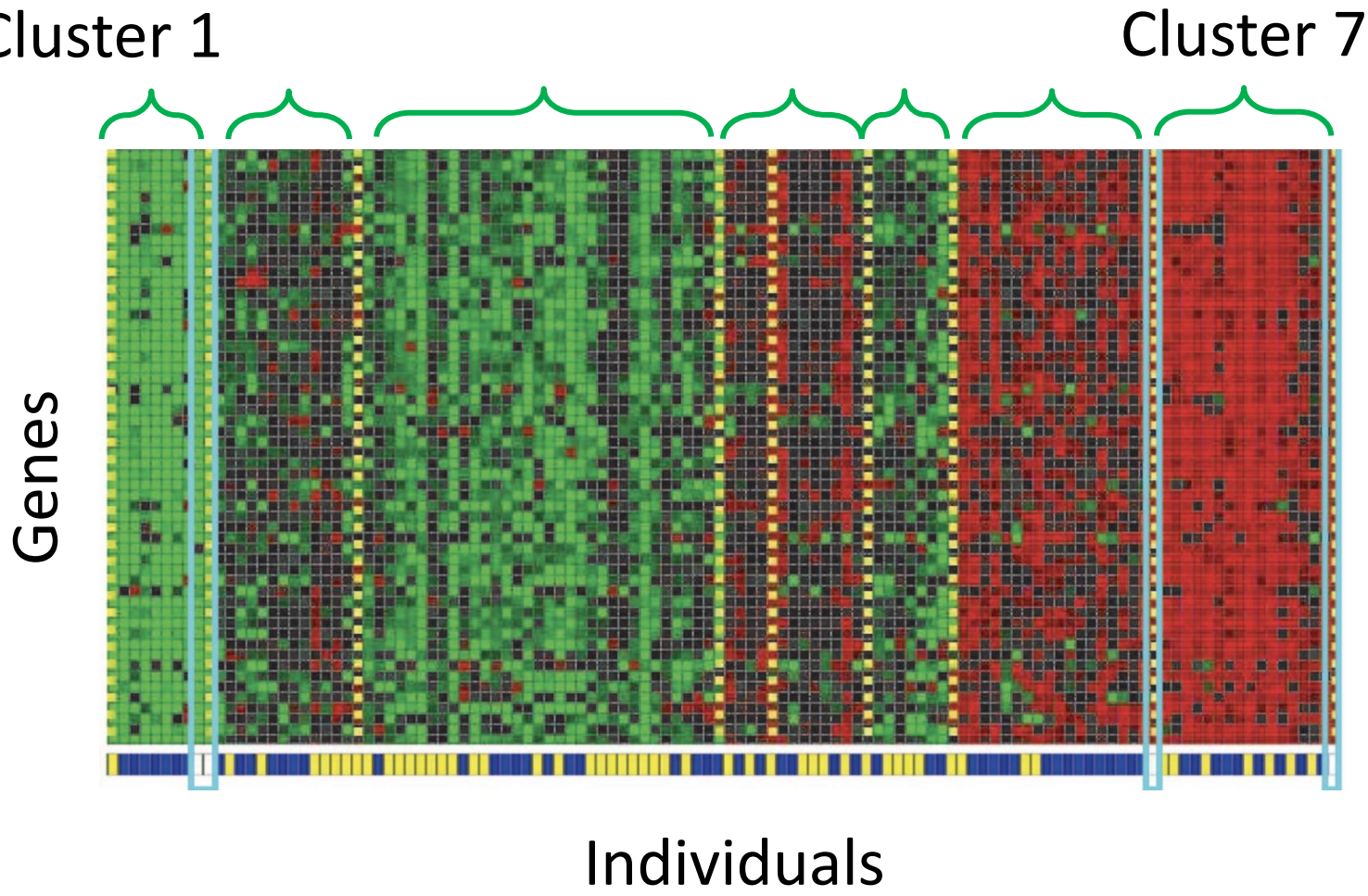


Clustering

- Lecture 12&13: k-mean clustering, mixture of Gaussians



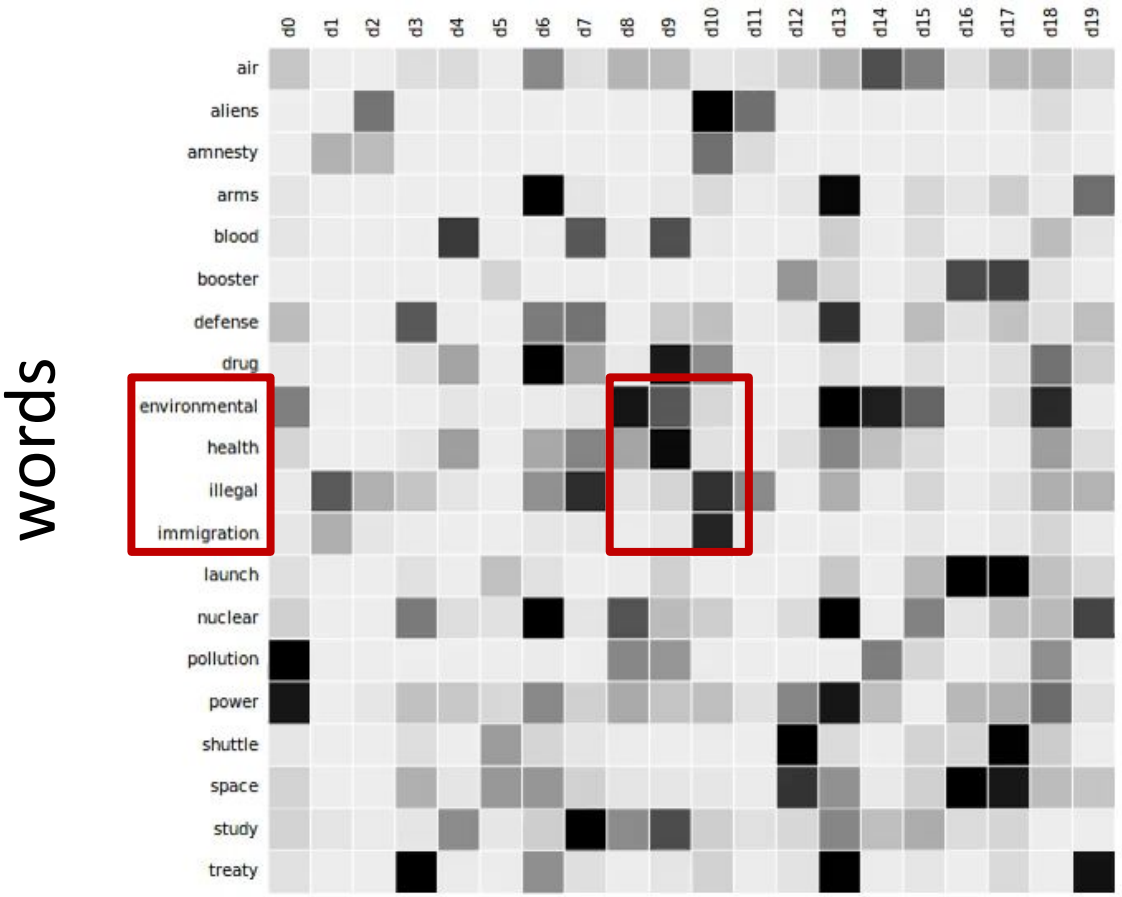
Clustering Genes



Identifying Regulatory Mechanisms using Individual Variation Reveals Key Role for Chromatin Modification. [Su-In Lee, Dana Pe'er, Aimee M. Dudley, George M. Church and Daphne Koller. ’06]

Latent Semantic Analysis (LSA)

documents



words

- Lecture 14: principal component analysis (tools used in LSA)

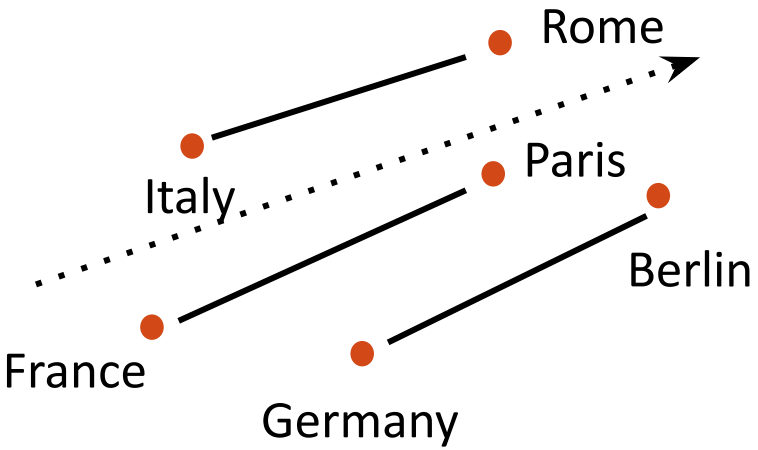
Image credit: https://commons.wikimedia.org/wiki/File:Topic_detection_in_a_document-word_matrix.gif

Word Embeddings

Unlabeled dataset

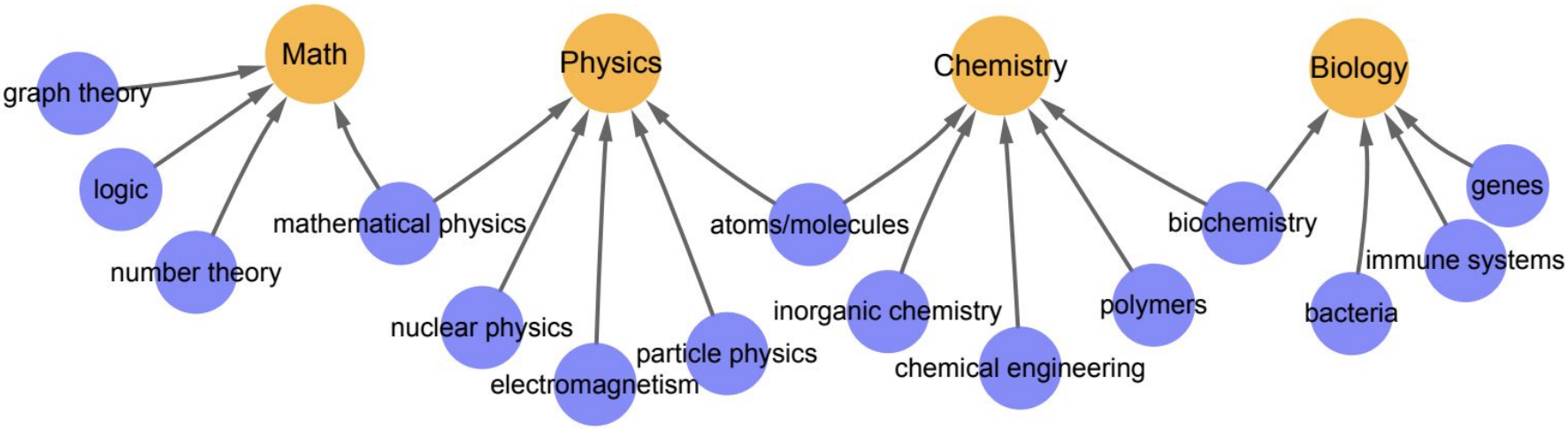
Represent words by vectors

- word $\xrightarrow{\text{encode}}$ vector
- relation $\xrightarrow{\text{encode}}$ direction



Word2vec [Mikolov et al'13]
GloVe [Pennington et al'14]

Clustering Words with Similar Meanings (Hierarchically)

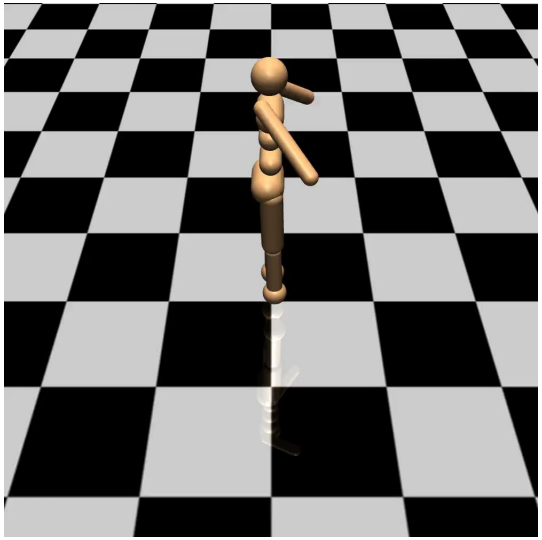


	logic deductive propositional semantics	graph subgraph bipartite vertex	boson massless particle higgs	polyester polypropylene resins epoxy	acids amino biosynthesis peptide
tag	*logic*	*graph theory*	*particle physics*	*polymer*	*biochemistry*

[Arora-Ge-Liang-M.-Risteski, TACL'17,18]

Reinforcement Learning

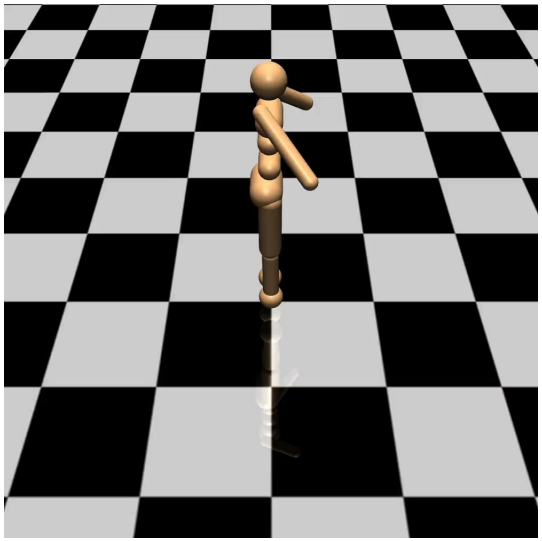
learning to walk to the right



Iteration 10

[Luo-Xu-Li-Tian-Darrell-M.'18]

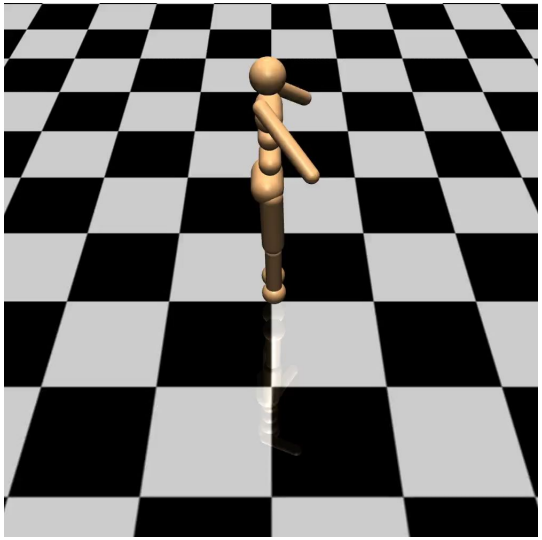
learning to walk to the right



Iteration 20

[Luo-Xu-Li-Tian-Darrell-M.'18]

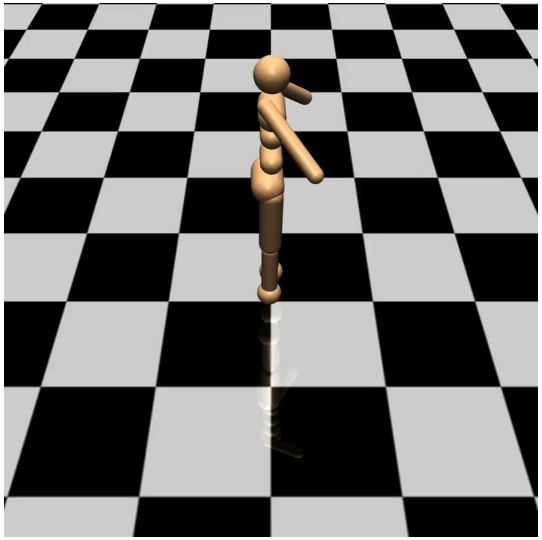
learning to walk to the right



Iteration 80

[Luo-Xu-Li-Tian-Darrell-M.'18]

learning to walk to the right



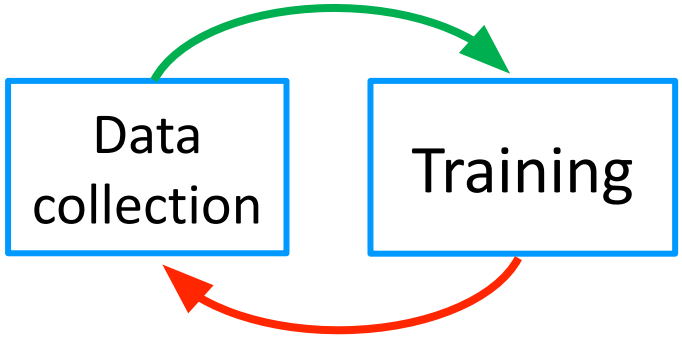
Iteration 210

[Luo-Xu-Li-Tian-Darrell-M.'18]

Reinforcement Learning

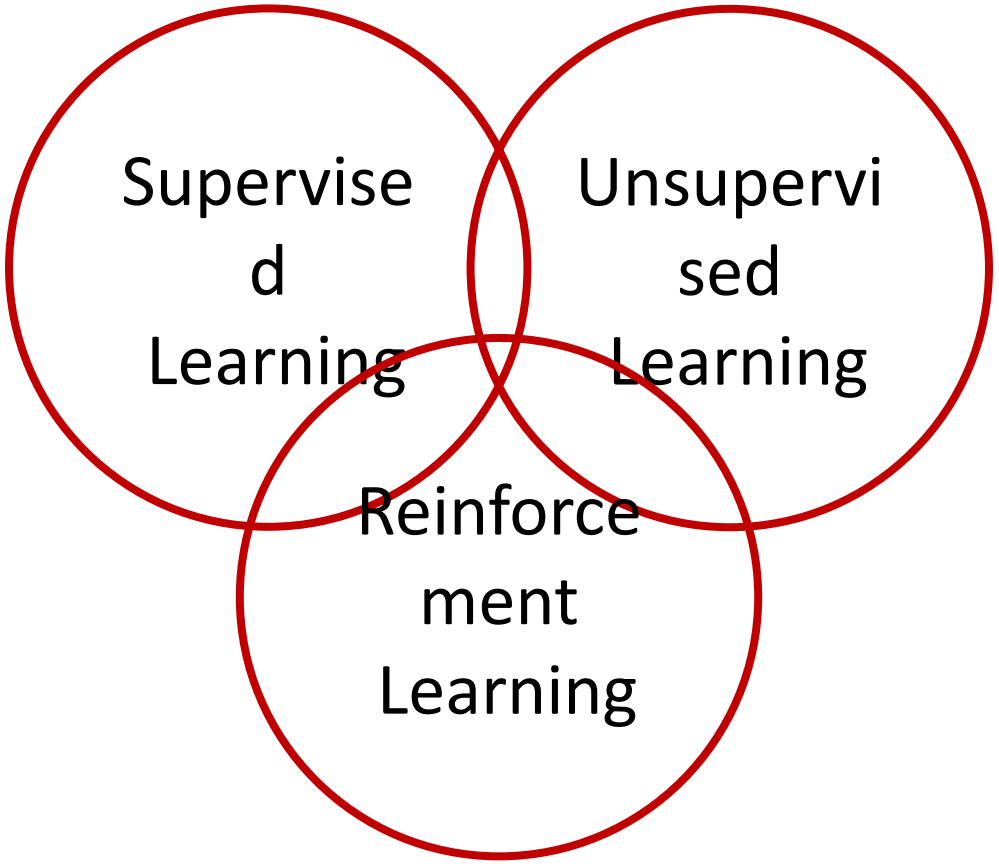
- The algorithm can collect data interactively

Try the strategy and collect feedbacks



Improve the strategy based on the feedbacks

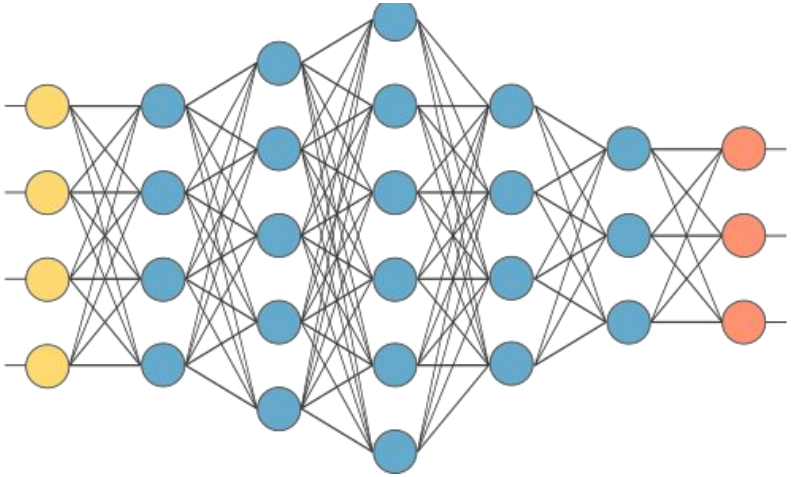
Taxonomy of Machine Learning (A Simplistic View Based on Tasks)



can also be viewed as tools/methods

Other Tools/Topics In This Course

- Deep learning basics



- Introduction to learning theory
  - Bias variance tradeoff
  - Feature selection
  - ML advice
- Broader aspects of ML
  - Robustness/fairness

Questions?

Thank you!

My Group's Research: Machine Learning Tools/Theory

How do we

- train faster?
- pick the correct model (and hyperparameters)?
- regularize the models so that they can generalize with fewer samples to unseen scenarios?
- robustify the models?

Various settings:

- supervised learning
- unsupervised learning
- reinforcement learning

Reinforcement Learning

Type equation here.