CS 6140 - Machine Learning

CS 6140: Machine Learning - Fall 2021

Time & Location:

2:50 - 4:30pm Eastern Time, Mondays and Wednesdays, Location: Mugar 201

Staff

Instructor: Paul Hand
Email: p.hand@northeastern.edu   Office Hours: Mondays 1:30 - 2:30 PM and by appointment. These will be held over Zoom

TA: Daniel Goldfarb
Email: goldfarb.d@northeastern.edu   Office Hours: Every other Tuesday 1-3 PM

TA: Devanshi Sanghvi
Email: sanghvi.d@northeastern.edu   Office Hours: Thursdays 11 AM

TA: Jorio Cocola
Email: cocola.j@northeastern.edu   Office Hours: Every other Tuesday 1-3 PM

Course Description and Goals

Machine learning is the study and design of algorithms, which enable computers/machines to learn from data. This course is an introduction to machine learning. It provides a broad view of models and algorithms, discusses their methodological foundations, as well as issues of practical implementation, use, and techniques for assessing the performance. At the end of the course the students will (1) understand and implement common machine learning methods, (2) recognize the problems that are amenable to machine learning, and perform appropriate data analysis, and (3) recognize failure points and threats to validity of the results.

Student Work:

Student work will involve the following four categories:

- Homework Assignments
- Midterm I
- Midterm II
- Project

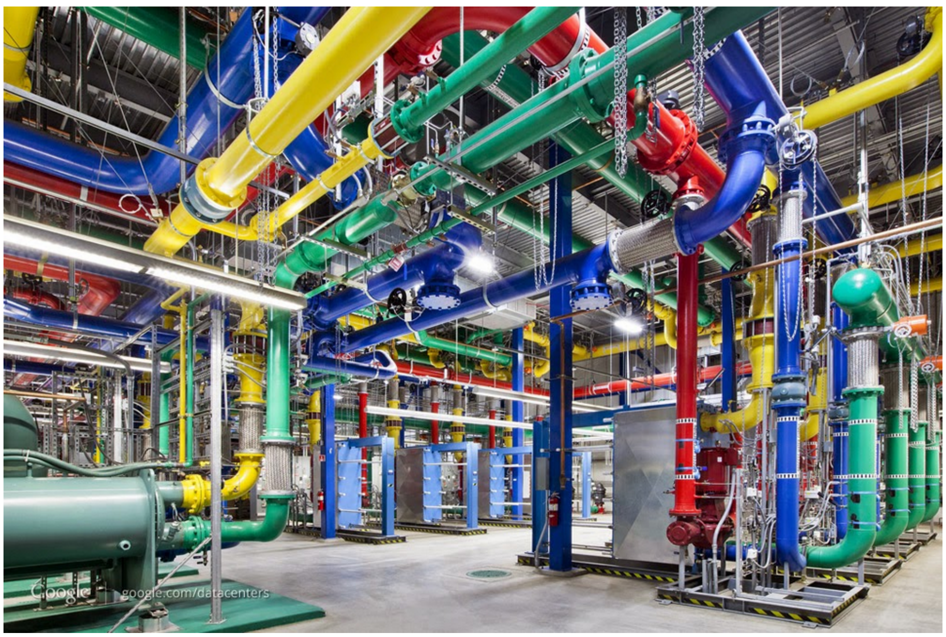
Details:

There will be eight homework assignments. Odd numbered assignments will be theoretical in nature. Even numbered assignments will be computational in nature. Assignments must be uploaded to Gradescope. You may type your responses or handwrite them. You can upload a pdf or individual photographs of each page of hand written work. For computational assignments, you must write your solutions as a Jupyter notebook and include a pdf of the Jupyter notebook as part of your submission.

Example about data center energy efficiency

Machine Learning Applications for Data Center Optimization
Jim Gao, Google

Mechanical Plant at a Google Data Center:



Energy Efficiency of the data center (Ideal value is 1)

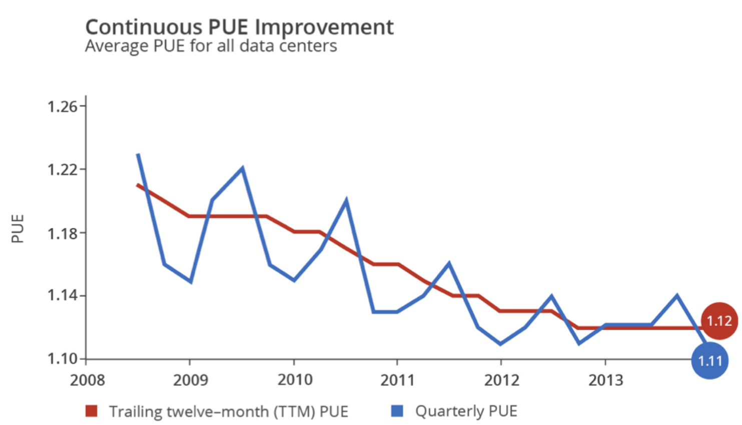


Fig 1. Historical PUE values at Google.

Features that affect the energy efficiency of the data center

1. Total server IT load [kW]
2. Total Campus Core Network Room (CCNR) IT load [kW]
3. Total number of process water pumps (PWP) running
4. Mean PWP variable frequency drive (VFD) speed [%]
5. Total number of condenser water pumps (CWP) running
6. Mean CWP variable frequency drive (VFD) speed [%]
7. Total number of cooling towers running
8. Mean cooling tower leaving water temperature (LWT) setpoint [F]
9. Total number of chillers running
10. Total number of drycoolers running
11. Total number of chilled water injection pumps running
12. Mean chilled water injection pump setpoint temperature [F]
13. Mean heat exchanger approach temperature [F]
14. Outside air wet bulb (WB) temperature [F]
15. Outside air dry bulb (DB) temperature [F]
16. Outside air enthalpy [kJ/kg]
17. Outside air relative humidity (RH) [%]
18. Outdoor wind speed [mph]
19. Outdoor wind direction [deg]

Question:
Some of these features can be controlled by the data center. Some can not. How would you use measurements of all of these features in order to make the data center as energy efficient as possible?

Results from a machine learning approach using neural networks

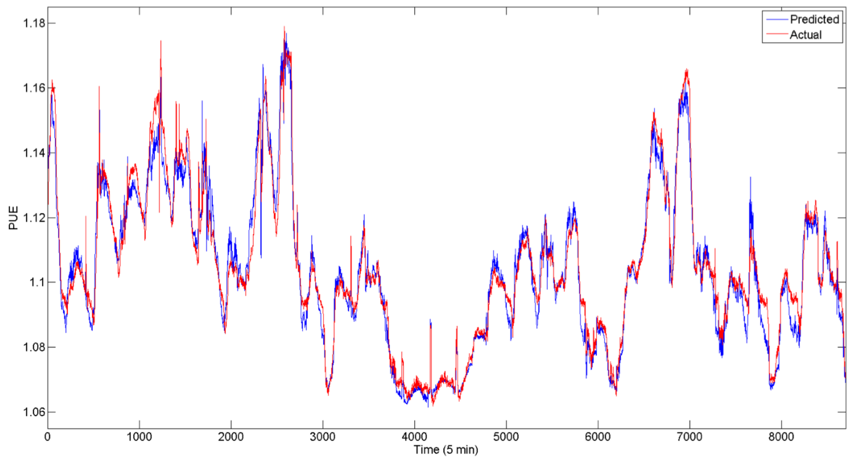
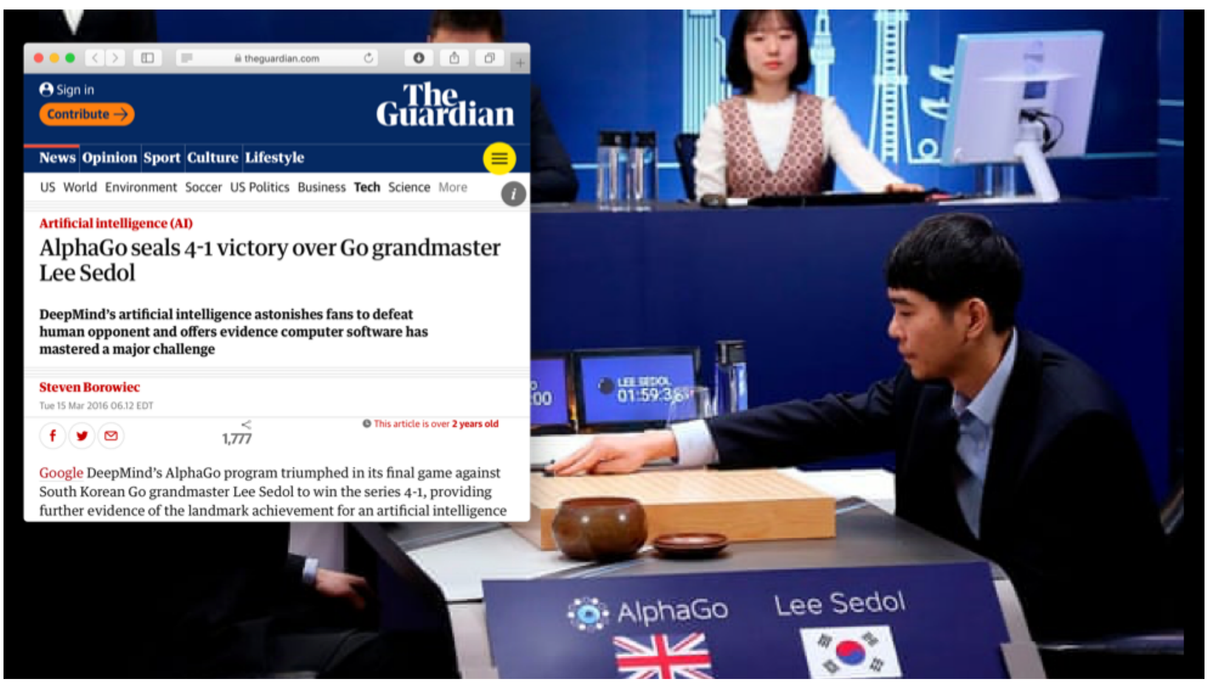


Fig. 3 Predicted vs actual PUE values at a major DC.

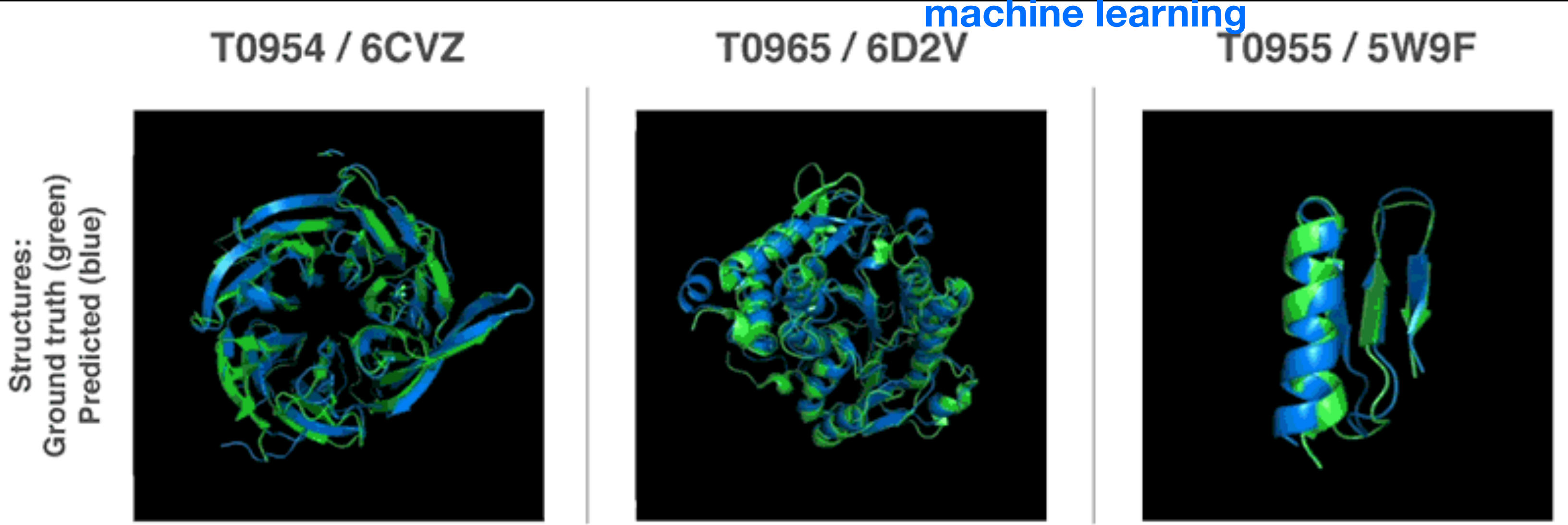

The New York Times
SUBSCRIBE NOW
LOG IN
ARCHIVES | 1997
Swift and Slashing, Computer Topples Kasparov
By BRUCE WEBER MAY 12, 1997
In brisk and brutal fashion, the I.B.M. computer Deep Blue unseated humanity, at least temporarily, as the finest chess playing entity on the planet yesterday, when Garry Kasparov, the world chess champion, resigned
e4 c6 2. d4 d5 3. Nc3 dxe4


theguardian.com
Sign in
Contribute
The Guardian
News Opinion Sport Culture Lifestyle
US World Environment Soccer US Politics Business Tech Science More
Artificial intelligence (AI)
AlphaGo seals 4-1 victory over Go grandmaster Lee Sedol
DeepMind's artificial intelligence astonishes fans to defeat human opponent and offers evidence computer software has mastered a major challenge
Steven Borowiec
Tue 15 Mar 2016 06.12 EDT
1,777
This article is over 2 years old
Google DeepMind's AlphaGo program triumphed in its final game against South Korean Go grandmaster Lee Sedol to win the series 4-1, providing further evidence of the landmark achievement for an artificial intelligence
AlphaGo
Lee Sedol

A notable achievement in AI

Example: Protein Folding by AlphaFold

A notable achievement of machine learning



Example: Image to Image Translation

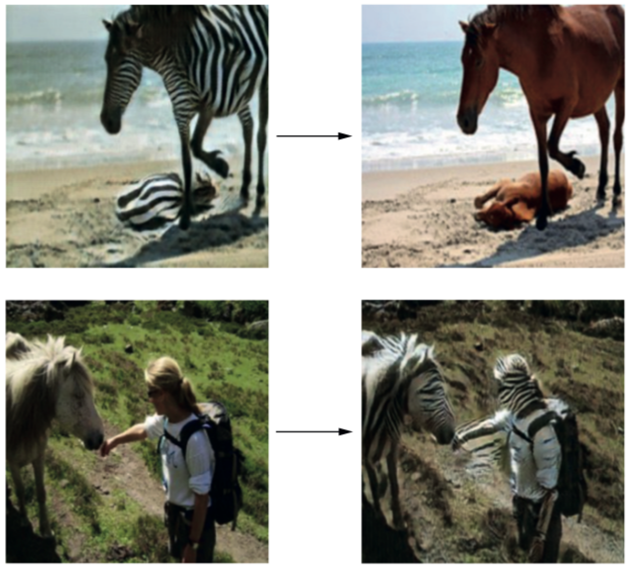
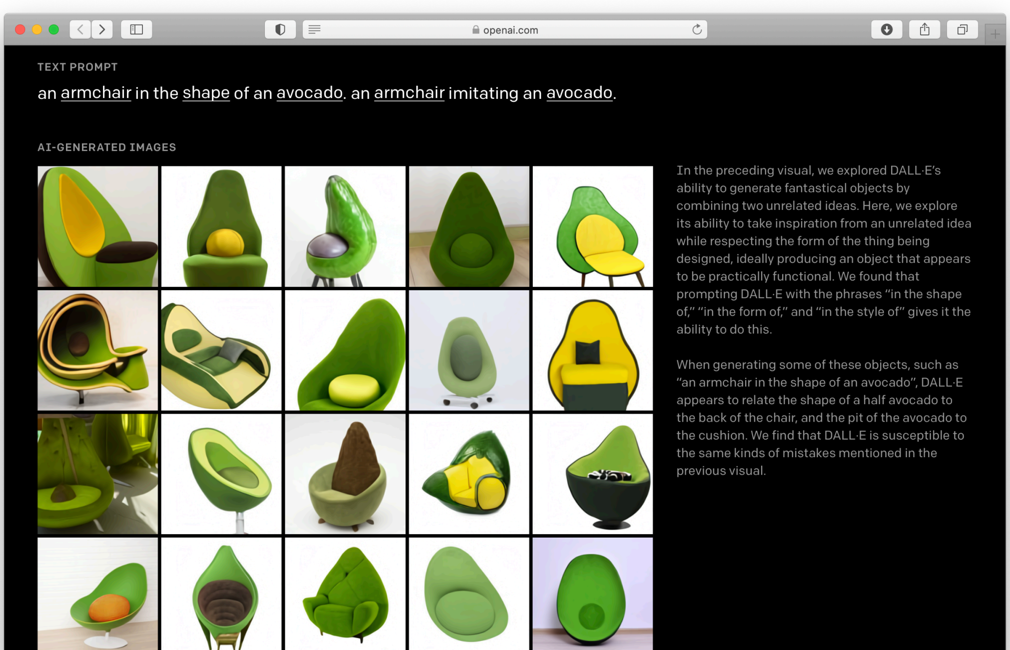
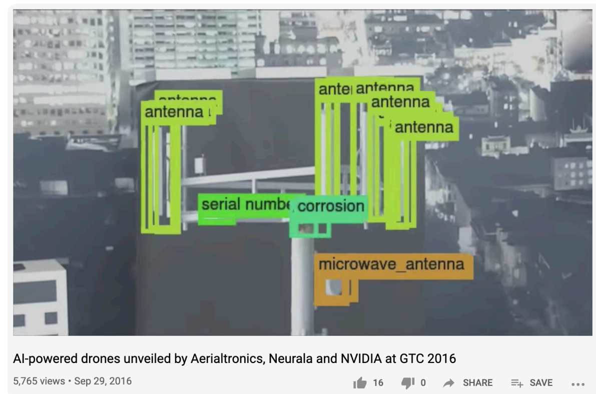


Figure 4.15 Style transfer in action, applying zebra patterns to horses (and humans). Images were generated using code from "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks" by Jun-Yan Zhu et al., https://arxiv.org/abs/1703.10593.

Example: Image generation from a text description - DALL E



Example: Object Detection



Machine learning versus traditional programming

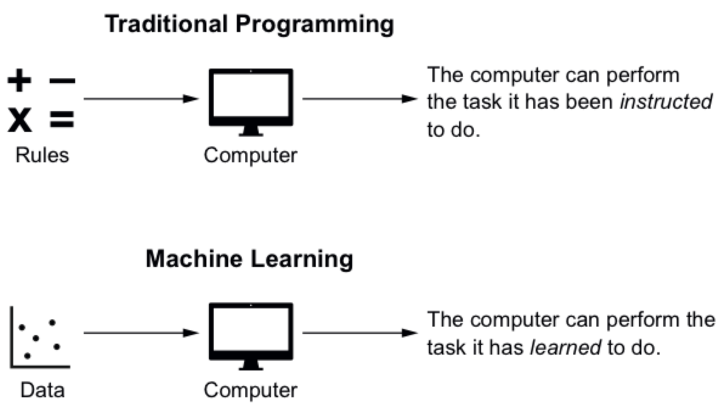
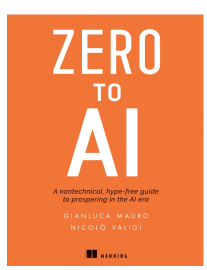


Figure 1.1 The difference between the traditional programming approach and machine learning: the first relies on precise rules and instructions, the latter on data and learning.

Figures in today's lecture are from:



Example of challenges of traditional programming approaches:

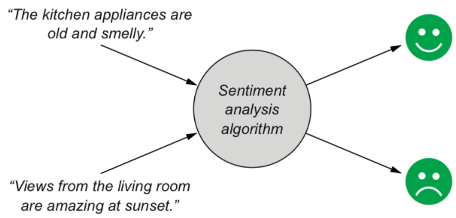


Figure 5.6 The basic concept of sentiment analysis

Let's suppose that we don't know machine learning, and we need to explain to a computer how to rate these sentences by developing an algorithm. A good rule of thumb would be to look at certain telltale words, such as "terrible" and "amazing." In fact, this is exactly how the earliest algorithms for sentiment analysis worked: researchers painstakingly built a dictionary of important words, and labeled each as positive, negative, or neutral. For example, such a word-sentiment dictionary might look like this:

- *Delighted*—Positive
- *Killed*—Negative
- *Shout*—Negative
- *Desk*—Neutral

Once you have an *emotional glossary* like that, you can classify each sentence by counting the number of positive and negative words and get a final score for the sentence.

This simplistic approach has a bunch of problems. Language is extremely complex, and we use it in very different ways from person to person; the same word can be used in different ways to communicate completely opposite messages. Let's say you listed "nice" as a positive-sentiment word, and one of the reviewers writes this:

It would be nice if they completely removed that tapestry.

Even if the sentence has an overall negative opinion of the house, our naive system would consider it positive, because of the positive connotation of "nice." Maybe you could try improving this system by adding more-complex rules, something like this:

'It would be [POSITIVE WORD] if ..' => negative

Although this rule would work on the preceding snippet, it's still easy to fool. For example, the following sentence is actually extremely positive, but would be ranked as a negative opinion:

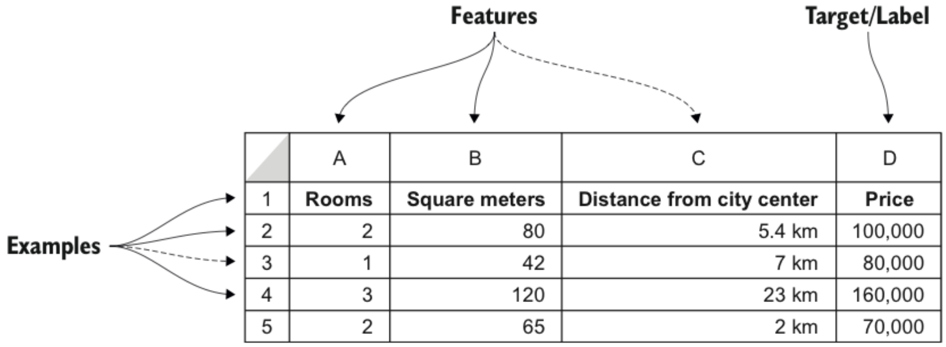
It would be nice if I could move into this place right away!

Should we keep adding hardcoded rules? The game is already becoming complicated (and boring), yet it's still easy to fool our system. Notice also that we're still playing with just a few words; we haven't even started mapping the vast ocean of the English vocabulary. Even if we did get to the bottom of this, we would almost need to start all over again for other languages.

Other challenges of traditional programming approaches:

Less adaptable
Ambiguities in language
Large amount of computing (e.g. deep search tree)
Limited to expertise (won't beat world expert, perhaps)
May be too complex for us to understand/prescribe
Requires an understanding of the context

Example: Prediction of Home Prices



	A	B	C	D
1	Rooms	Square meters	Distance from city center	Price
2	2	80	5.4 km	100,000
3	1	42	7 km	80,000
4	3	120	23 km	160,000
5	2	65	2 km	70,000

Figure 2.2 An Excel sheet with features and labels for several examples

Other Examples:

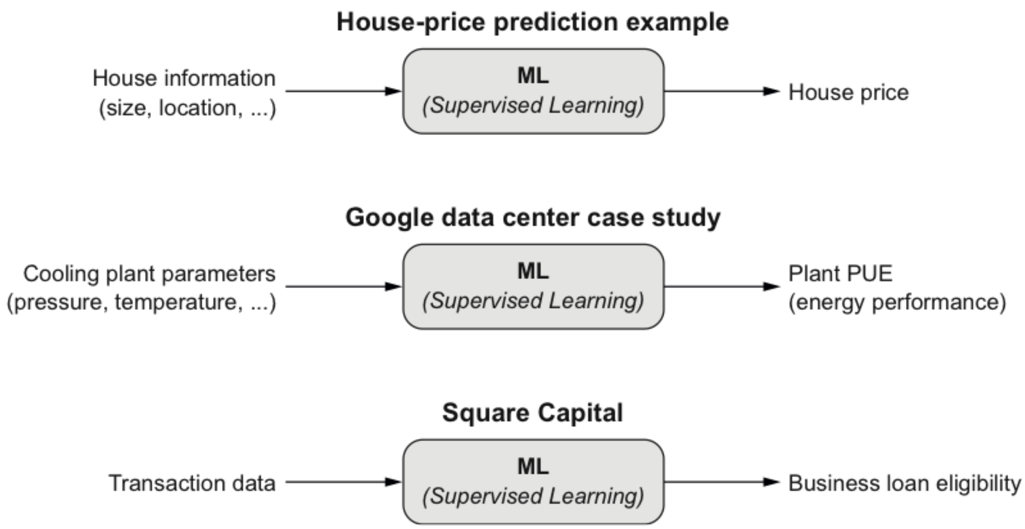


Figure 2.7 How the house-prediction example and the Square and Google case studies used supervised learning

Example: Clustering Users

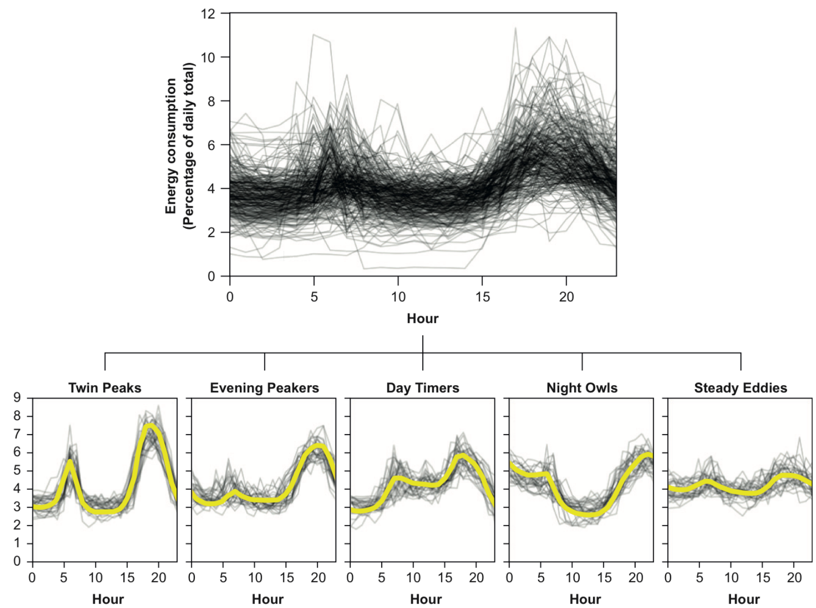


Figure 3.12 Turning a “hairball” of 1,000 users into five clusters

Example: Predicting Customer Churn

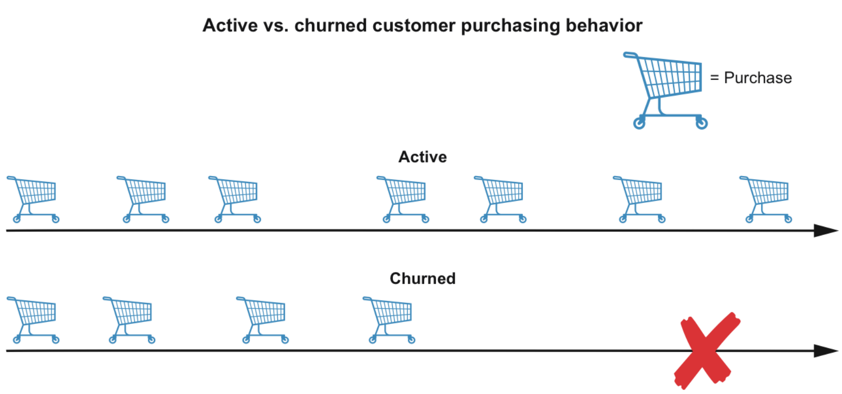


Figure 3.3 A graphical representation of the buying pattern behavior of churned and active customers

Supervised Learning vs. Unsupervised Learning

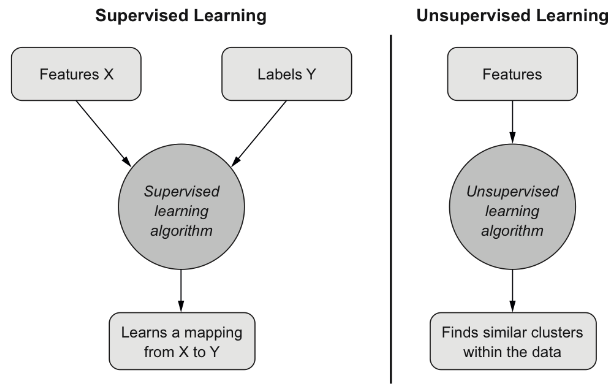


Figure 3.6 The differences in input and output of supervised and unsupervised algorithms

Supervised Learning Pipeline:

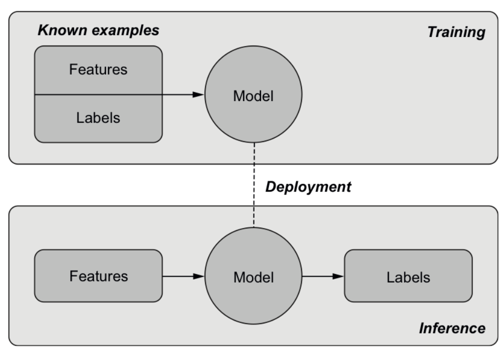
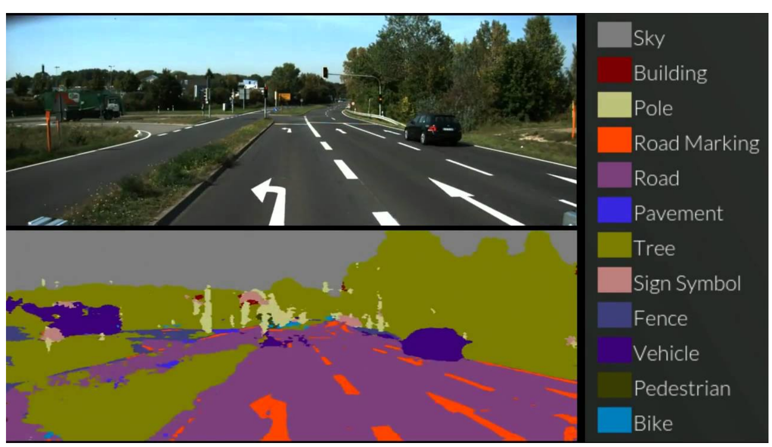


Figure 2.3 The two phases of machine learning: training and inference

Example: Image Segmentation



Feature Engineering vs. Feature Learning

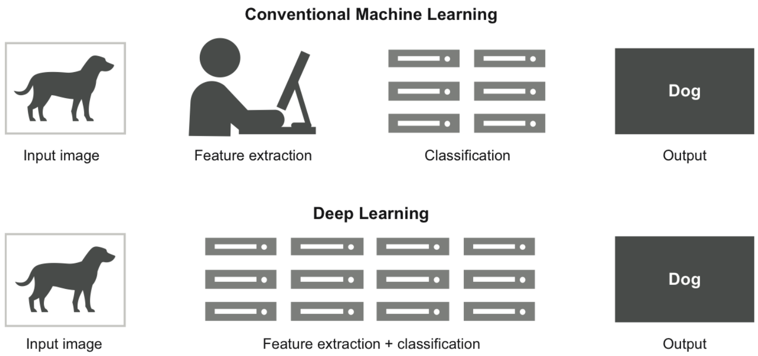
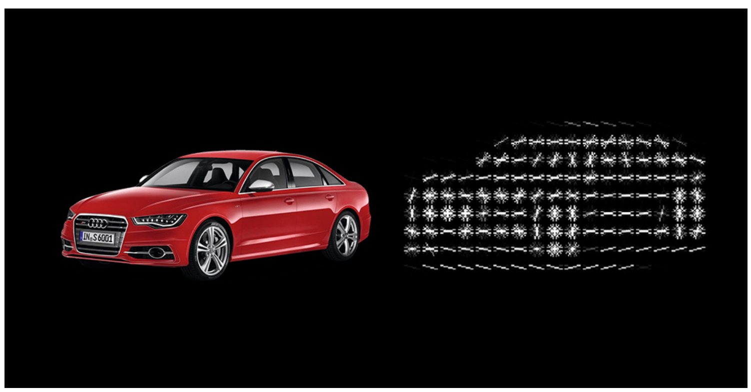


Figure 4.5 In traditional ML, engineers have to develop algorithms to extract features that can be fed to the model. A DL model doesn't need this complex preliminary step.

Example of feature engineering: HOG Features



Example of feature engineering: SIFT Features

Feature Learning (aka Representation Learning):

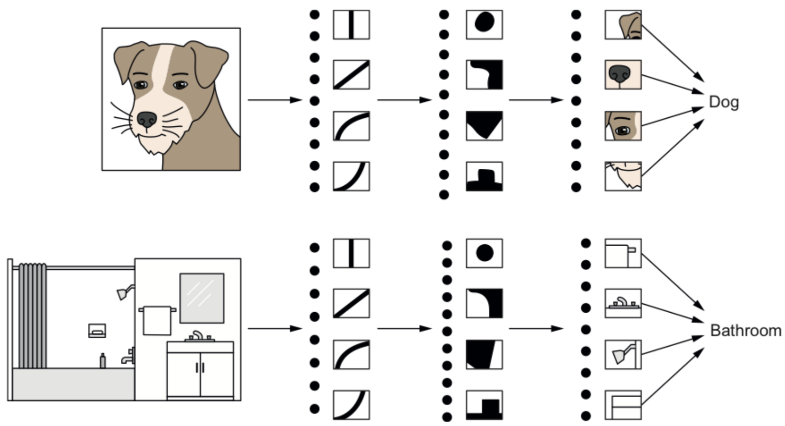
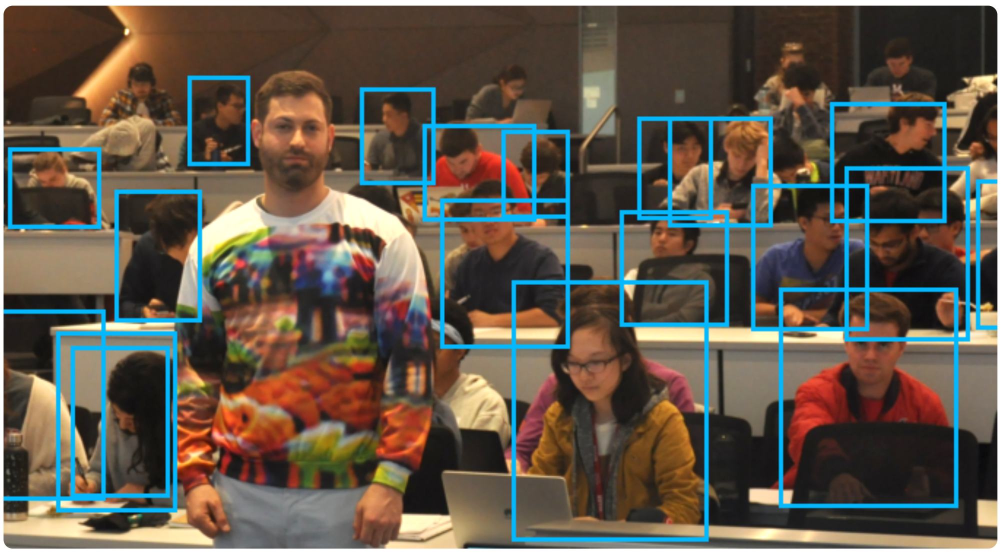


Figure 4.10 Similarities between DL-based classifiers for rooms and animals. The first layers learn to identify the same general geometric patterns.

Figure 2. Visualization of features in a fully trained model. For layers 2-5 we show the top 9 activations in a random subset of feature maps across the validation data, projected down to pixel space using our deconvolutional network approach. Our reconstructions are *not* samples from the model: they are reconstructed patterns from the validation set that cause high activations in a given feature map. For each feature map we also show the corresponding image patches. Note: (i) the the strong grouping within each feature map, (ii) greater invariance at higher layers and (iii) exaggeration of discriminative parts of the image, e.g. eyes and noses of dogs (layer 4, row 1, cols 1). Best viewed in electronic form.

Zeiler and Fergus, 2013

Drawbacks of current AI techniques



Wu et al. 2019

Eykholt et al. 2018

Examples to think about:

1) User has a house they want to sell. They upload many photos of the house to a website. You want an automatic way of determining whether each photo is of a bedroom/bathroom/kitchen/etc.

Would you approach this problem as a supervised or unsupervised learning problem?

Would you do feature engineering or feature learning?

2) For a rare type of disease, some patients have good outcomes and some patients have poor outcomes. Doctors don't know why and don't know which category any particular patient falls into. They decide to measure gene expression levels among multiple patients in order to learn more about this disease.

Would you approach this problem as a supervised or unsupervised learning problem?

What are the features you would use in this problem?

3). You are running a self-driving car company. You notice that your car's vision system based on supervised learning can not correctly identify highway paint markings when it's hailing. Your system was trained on a large collection of images, but you then realize none of those images involved hail. What could you do?

4) You are trying to sell a product and you are devising the marketing strategy. You know that you have multiple groups of customers who respond to different types of marketing. You are trying to decide things like:

Should you group 20-25 year olds together or should you group 20-30 year olds together?

Should college students in large cities get different marketing than college students in small cities?

Should you have different marketing approaches for men and women?

Would you approach this problem as a supervised learning problem or as an unsupervised learning problem?

5) You are making the next generation of MRI machines. These machines aim to take one eighth as many measurements of the person in side, and they aim to output the same quality of MRI image.

Would you approach this problem as a supervised learning problem or as an unsupervised learning problem?

What is machine learning?

> This deluge of data calls for automated methods of data analysis, which is what **machine learning** provides. In particular, we define machine learning as a set of methods that can automatically detect patterns in data, and then use the uncovered patterns to predict future data, or to perform other kinds of decision making under uncertainty (such as planning how to collect more data!).

— Kevin Murphy

> The first definition of *machine learning* dates back to 1959, from American AI pioneer Arthur Samuel:
>
> *Machine learning is the field of study that gives computers the ability to learn without being explicitly programmed.*

— Mauro and Valigi

Relationship of ML to AI and Deep Learning

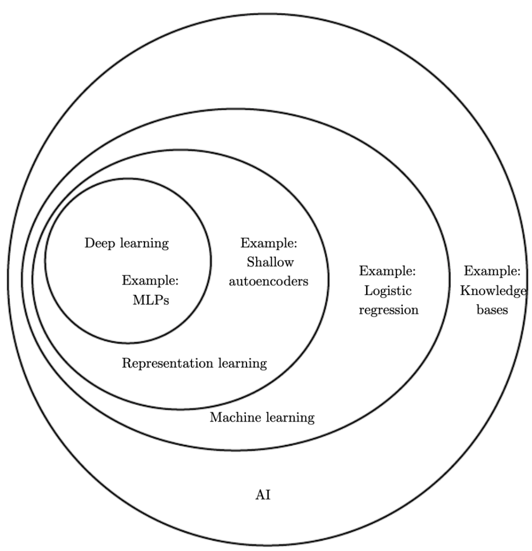


Figure 1.4: A Venn diagram showing how deep learning is a kind of representation learning, which is in turn a kind of machine learning, which is used for many but not all approaches to AI. Each section of the Venn diagram includes an example of an AI technology.