MGMT298D
Science and Strategy of AI

Week 1

Course Introduction; Foundations of Machine Learning

Auyon Siddiq
UCLA Anderson School of Management

(Phones and laptops away please!)

When Did "Artificial Intelligence" Begin?

- 1956: Dartmouth Conference — John McCarthy proposes a conference and coins the term

https://spectrum.ieee.org/dartmouth-ai-workshop

When Did "Artificial Intelligence" Begin?

McCarthy's research proposal:

We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.

http://jmc.stanford.edu/articles/dartmouth/dartmouth.pdf

VOL. LIX. No. 236.] [October, 1950

MIND

A QUARTERLY REVIEW

OF

PSYCHOLOGY AND PHILOSOPHY

I.—COMPUTING MACHINERY AND INTELLIGENCE

BY A. M. TURING

1. *The Imitation Game.*

I PROPOSE to consider the question, 'Can machines think?' This should begin with definitions of the meaning of the terms 'machine' and 'think'. The definitions might be framed so as to reflect so far as possible the normal use of the words, but this attitude is dangerous. If the meaning of the words 'machine' and 'think' are to be found by examining how they are commonly used it is difficult to escape the conclusion that the meaning

Confident Predictions!

"Machines will be capable of doing any work a man can do."

— Herbert Simon, 1965

"Within a generation... the problem of creating 'artificial intelligence' will substantially be solved."

— Marvin Minsky, 1967

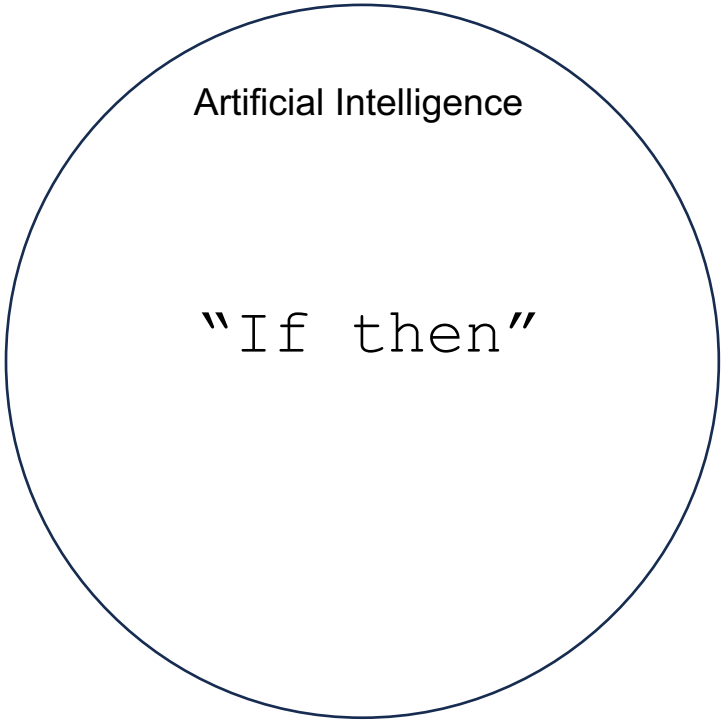
"In 3 to 8 years we will have a machine with the general intelligence of an average human being."

— Marvin Minsky, 1970

Three Major Waves of AI

Traditional
(1960s - 90s)
Logic-based AI

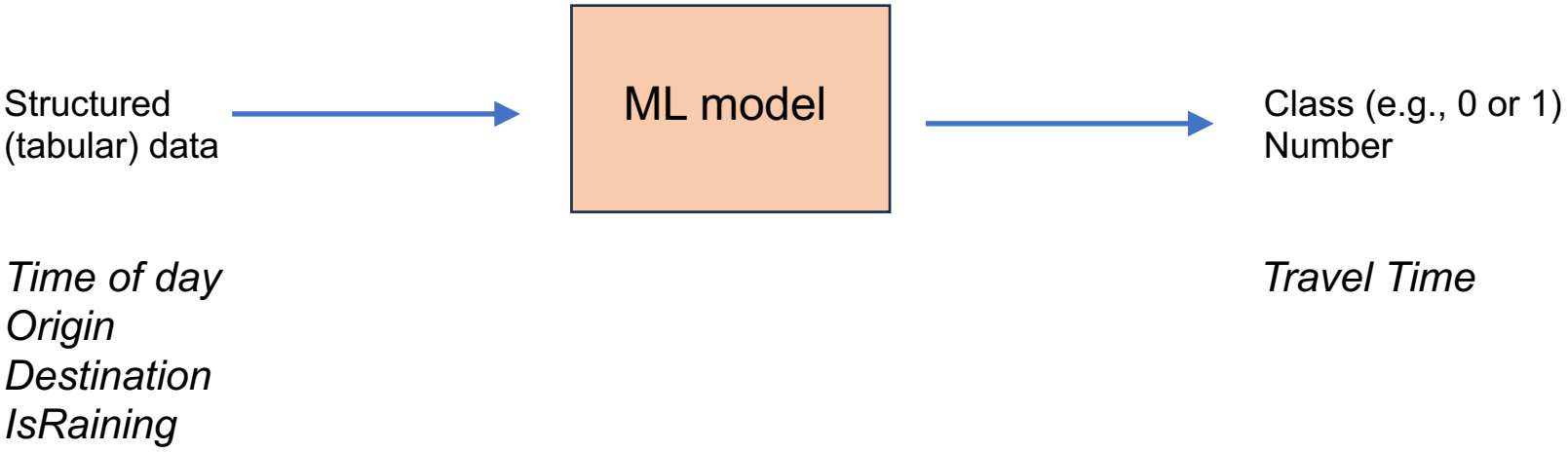
IF *TIME > 3pm* ***AND*** *TIME < 7pm*
THEN *TRAVEL_TIME = 60 minutes*



Three Major Waves of AI

Machine Learning

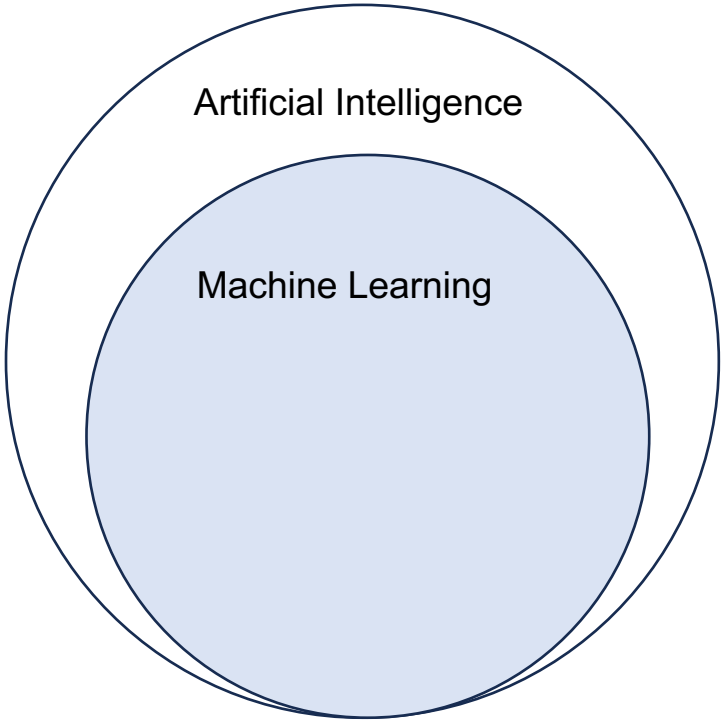
Training examples



Three Major Waves of AI

Traditional

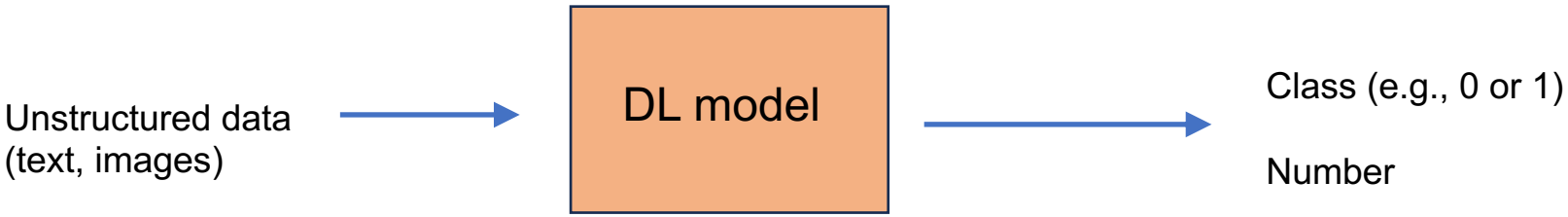
Machine Learning



Three Major Waves of AI

Deep Learning

Representations



Traffic camera images
Live accident reports

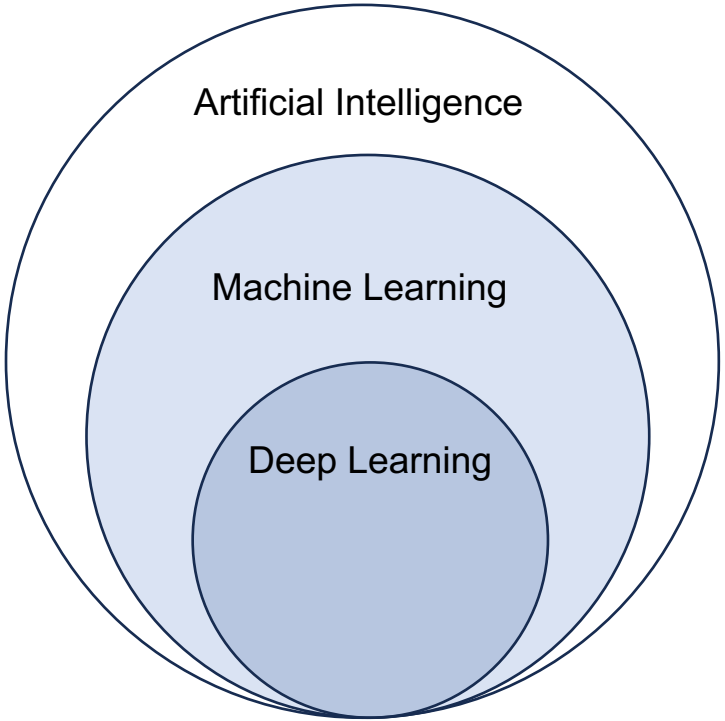
Travel Time

Three Major Waves of AI

Traditional

Machine Learning

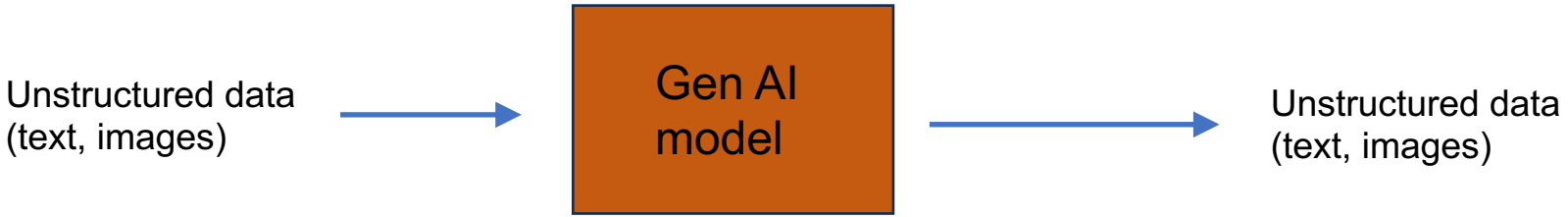
Deep Learning



Three Major Waves of AI

Generative AI

Output unstructured data



Traffic camera images
Live accident reports
Time of day
Origin
Destination
IsRaining

Voice navigation

“There's a 20-minute delay on I-10 due to an accident near exit 4A.”

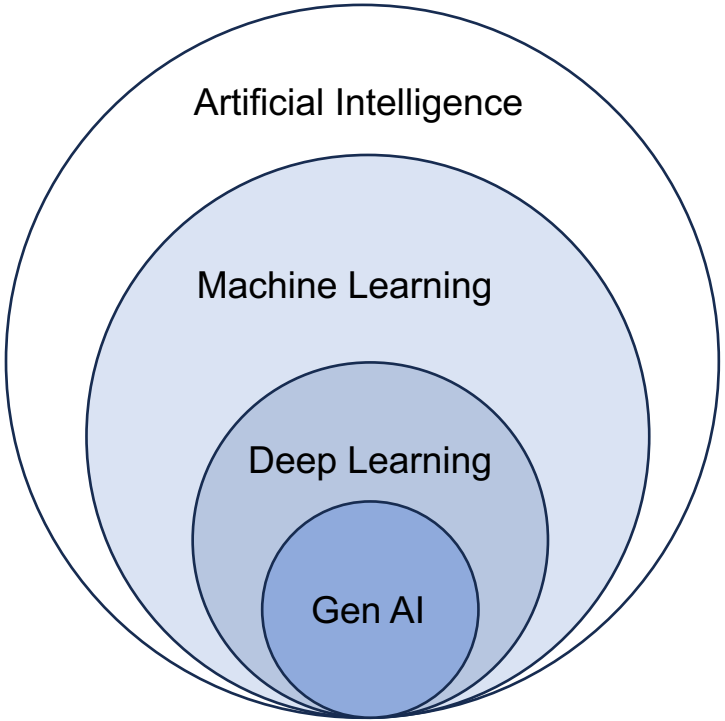
Three Major Waves of AI

Traditional

Machine Learning

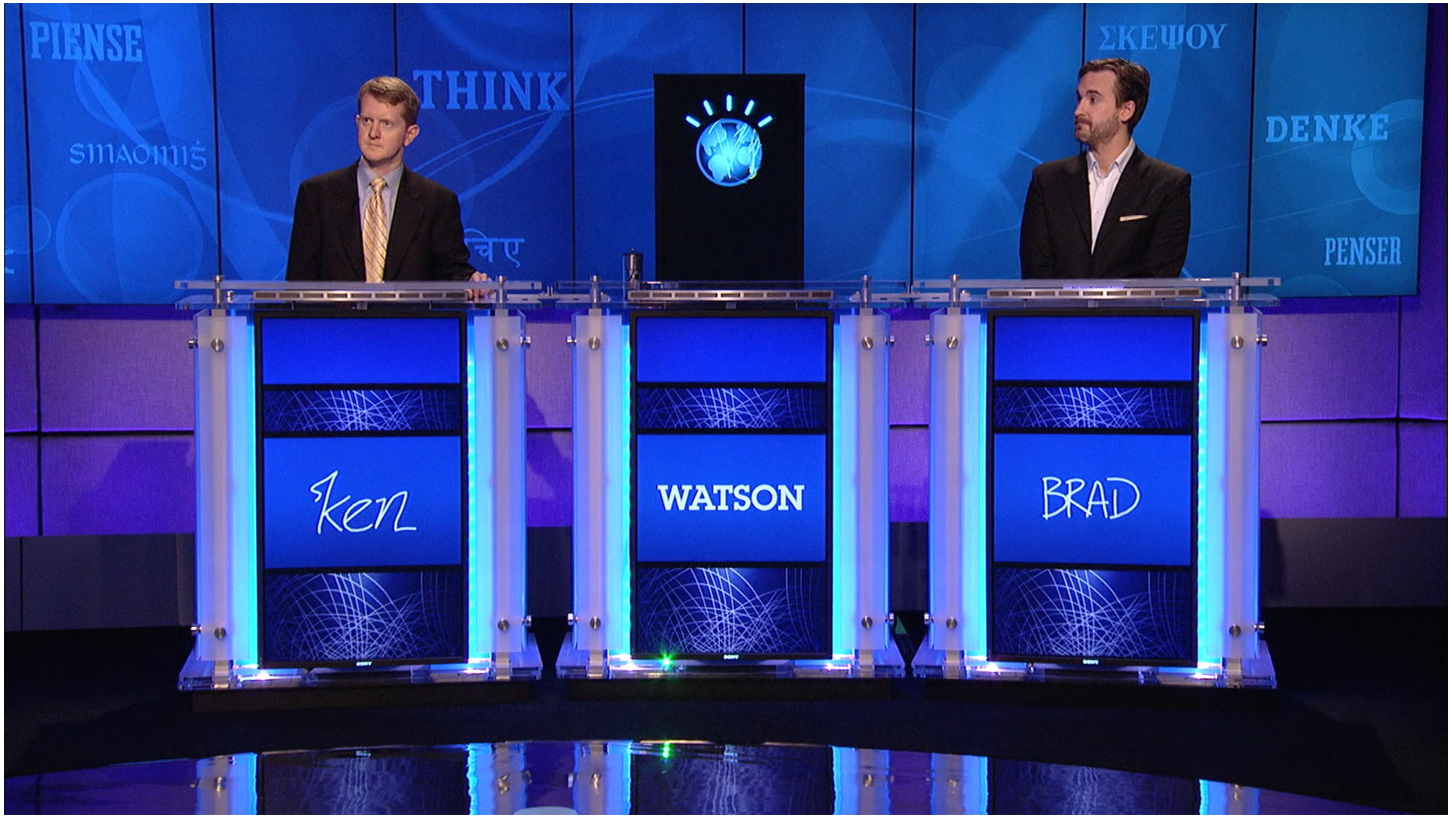
Deep Learning

Generative AI

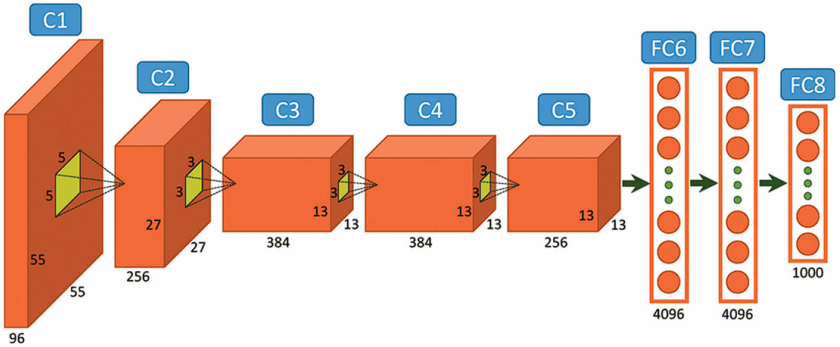


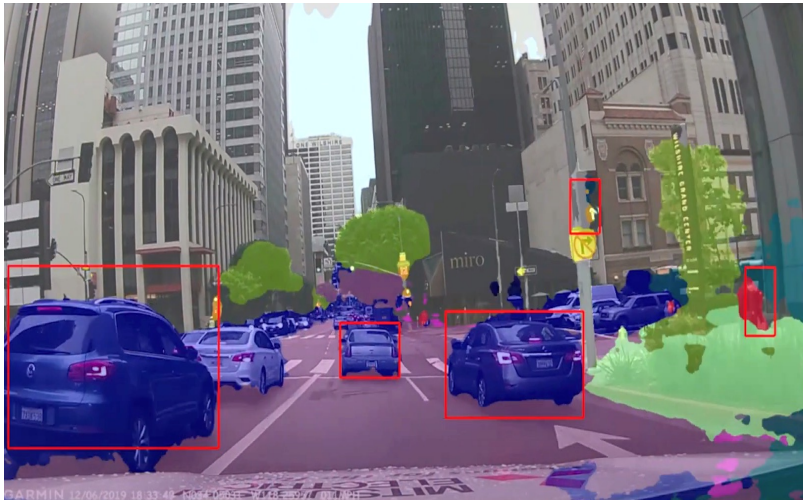
Deep Blue (1997)

IBM Watson (2011)

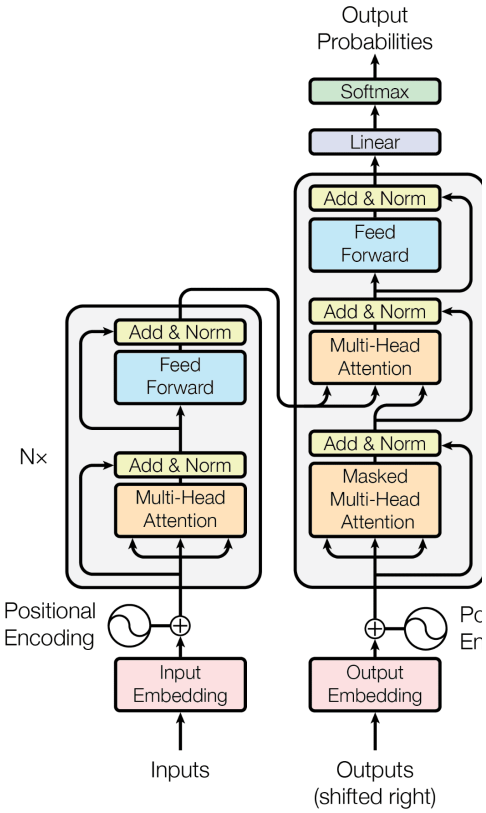


AlexNet (2012)





Transformers (2017)



Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.0 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature.

ChatGPT (2022)

So what is this course about?

A *wide lens* survey core machine learning and AI models

Tech-forward: understand how the technology *works* and leverages data; explore some implementations in Python and in simulations

Gentle on math, heavy on intuition

We will get technical to build deep understanding – goal is to **massively expand** your vocabulary around ML and AI

Who is this course for?

Non-technical leaders (consultants, PMs, analysts) who are curious about how ML/AI models work to extract value from data

“Do I need to invest in technical knowledge?”

1. Precise understanding is important for building true confidence in AI topics: When to deploy, what it can/can't do, etc
2. Credibility with technical staff, facilitates leading and collaboration
3. Differentiation in the job market
4. Complementary to other Anderson courses
5. It is fun

Course Roadmap

WEEKS 1-3

Machine Learning

Goal: *Understand how to deploy ML models to make predictions/decisions based on rich data*

WEEKS 4-7

Deep Learning

Goal: *Build intuition on how complex AI models extract "meaning" from numeric, text, and image data*

WEEKS 8-10

Product Sprint

Goal: *Prototype, publish, and pitch an AI-powered web app*

Course Roadmap

WEEKS 1–3

Machine Learning

***Goal:** Understand how to deploy ML models to make predictions/decisions based on rich data*

Overview of the ML “pipeline” – from data to predictions

State-of-the-art prediction models used in industry and best practices for deployment

Algorithms for data-driven decision making in dynamic environments

Course Roadmap

WEEKS 4–7

Deep Learning

***Goal:** Build intuition on how complex AI models extract "meaning" from numeric, text, and image data*

Overview of *neural network* architecture and training algorithms

How massive networks transform language and image data to extract meaning

Best practices for AI deployment

Economics of building and using AI

Course Roadmap

WEEKS 8–10

Product Sprint

***Goal:** Prototype, publish, and pitch an AI-powered web app*

Work in teams to prototype a simple AI-powered app and publish it online

Use AI as software developer – you focus on the idea and design

Top teams will pitch in Week 10

Showcase your skills and keep a souvenir from the course

Important AI topics we won't cover as much

How to best use LLMs and agents

- Covered in AI Agents for Managers (Schubert)
- But we will still use LLMs during product sprint

Industry-by-industry survey of AI adoption and strategy / leadership and management of AI

- Easton courses (Khormae, Null, Holloway)
- We will still touch on deployment

Goal is to understand the science better so we can make strategic decisions

Assessments

1. **Seven assignments (Individual)**
   - Usually last ~1 hour of each class to work on these
   - Completely non-code based – we'll use custom dashboards
   - Explore models and answer 3-5 questions in a document and submit on BruinLearn by the end of the day (5pm)

2. **AI product prototype (In groups of 1, 2 or 3)**
   - Based in Python and HTML but vibe-coded with AI, details to come in Week 8

Google Colaboratory

We will often look at Python code together to see how ML/AI models are actually implemented

The goal is not to fixate on coding but to understand the general AI pipeline

Google's **Colaboratory** is an amazing free resource for implementing AI models in Python → entirely browser-based and no need for local installations on your laptop

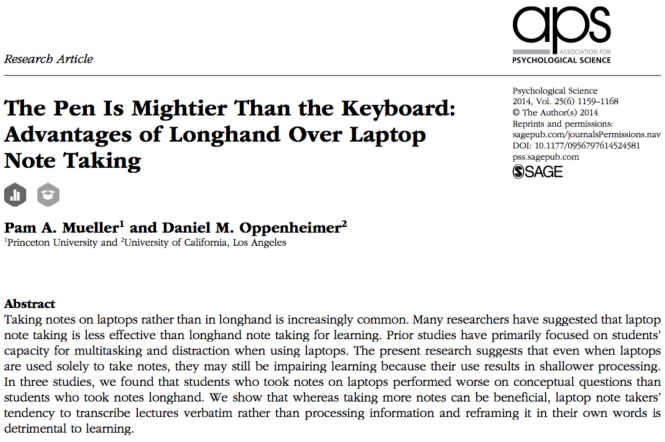
You will need any Google/Gmail account to access

Tech policy

Laptops in *back row only*, except for during Colab or in-class assignment components

No phones

Mueller and Oppenheimer (2014)
handwriting notes improves retention



Research Article

aps ASSOCIATION FOR PSYCHOLOGICAL SCIENCE

The Pen Is Mightier Than the Keyboard: Advantages of Longhand Over Laptop Note Taking

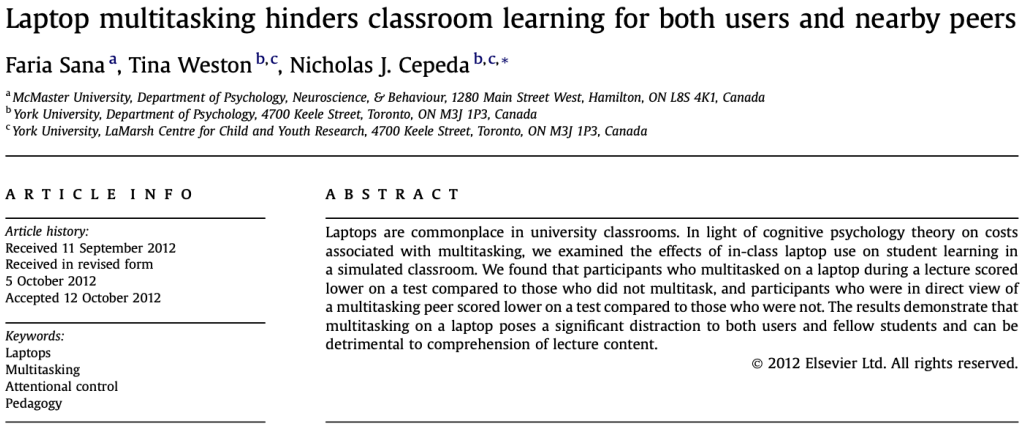
Psychological Science
2014, Vol. 25(6) 1159–1168

DOI: 10.1177/0956797614524581
pss.sagepub.com
SAGE

Pam A. Mueller[1] and Daniel M. Oppenheimer[2]
[1]Princeton University and [2]University of California, Los Angeles

Abstract
Taking notes on laptops rather than in longhand is increasingly common. Many researchers have suggested that laptop note taking is less effective than longhand note taking for learning. Prior studies have primarily focused on students' capacity for multitasking and distraction when using laptops. The present research suggests that even when laptops are used solely to take notes, they may still be impairing learning because their use results in shallower processing. In three studies, we found that students who took notes on laptops performed worse on conceptual questions than students who took notes longhand. We show that whereas taking more notes can be beneficial, laptop note takers' tendency to transcribe lectures verbatim rather than processing information and reframing it in their own words is detrimental to learning.

Sana et al (2012)
"second-hand distraction"



Laptop multitasking hinders classroom learning for both users and nearby peers

Faria Sana [a], Tina Weston [b,c], Nicholas J. Cepeda [b,c,*]

[a] *McMaster University, Department of Psychology, Neuroscience, & Behaviour, 1280 Main Street West, Hamilton, ON L8S 4K1, Canada*
[b] *York University, Department of Psychology, 4700 Keele Street, Toronto, ON M3J 1P3, Canada*
[c] *York University, LaMarsh Centre for Child and Youth Research, 4700 Keele Street, Toronto, ON M3J 1P3, Canada*

ARTICLE INFO

Article history:
Received 11 September 2012
Received in revised form
5 October 2012
Accepted 12 October 2012

Keywords:
Laptops
Multitasking
Attentional control
Pedagogy

ABSTRACT

Laptops are commonplace in university classrooms. In light of cognitive psychology theory on costs associated with multitasking, we examined the effects of in-class laptop use on student learning in a simulated classroom. We found that participants who multitasked on a laptop during a lecture scored lower on a test compared to those who did not multitask, and participants who were in direct view of a multitasking peer scored lower on a test compared to those who were not. The results demonstrate that multitasking on a laptop poses a significant distraction to both users and fellow students and can be detrimental to comprehension of lecture content.

(My 3-week old enjoying some crayons)

Course Roadmap

WEEKS 1-3

Machine Learning

Goal: *Understand how to deploy ML models to make predictions/decisions based on rich data*

WEEKS 4-7

Deep Learning

Goal: *Build intuition on how complex AI models extract "meaning" from numeric, text, and image data*

WEEKS 8-10

Product Sprint

Goal: *Prototype, publish, and pitch an AI-powered web app*

Demand Forecasting at H&M

Case Handout

H&M, a Fashion Giant, Has a Problem: $4.3 Billion in Unsold Clothes

The fashion retailer H&M reported poor earnings on Tuesday, and is sitting on a huge pile of unsold clothes. Horacio Villalobos/Corbis, via Getty Images

By Elizabeth Paton

March 27, 2018

Analysts have been pressing Karl-Johan Persson, the company's chief executive, over the issue. Inventory levels were up, Mr. Persson said, because H&M was opening 220 new stores and expanding its e-commerce operations, and so needed to fill the racks.

Critics, however, blamed poor inventory management and underwhelming product offerings, prompting once-loyal shoppers to take their wallets elsewhere.

BUSINESS

H&M Pivots to Big Data to Spot Next Big Fast-Fashion Trends

Instead of cookie-cutter stores, the H&M chain is using granular data to customize the offerings in each one of its 4,200 locations

By *Saabira Chaudhuri* Follow

May 7, 2018 8:00 am ET

STOCKHOLM—The world's largest clothing brand is turning to artificial intelligence to win back shoppers, as it works to reverse one of the worst sales slumps in its history.

Discussion

Machine Learning

What is Machine Learning?

Branch of artificial intelligence focused on prediction

Mostly uses structured, tabular data

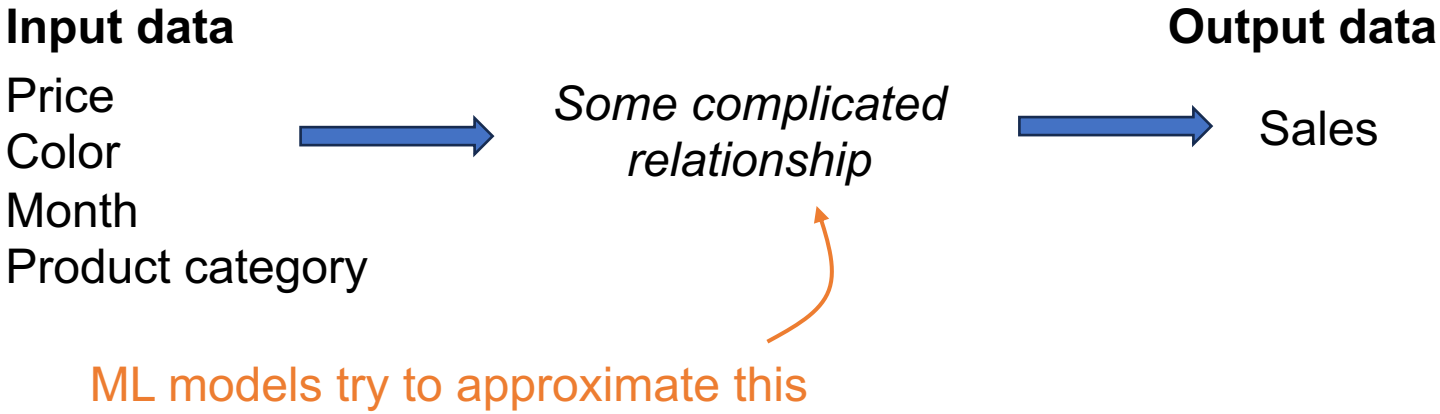
Extension of statistics, but we care much less about parameter estimates, confidence intervals, p-values – just want to make good predictions!

What is a Model?

Most standard setup: Develop using *training examples* (input and output data pairs), and produce a prediction

Two types: **Regression** Predict a number *"Sales next month?"*

Classification Predict a category *"Will this shopper make a purchase?"*



Data

H&M sales data: Each row = one product-month

product id	price	month	category	sales
1	$42.50	January	T-Shirt	110
1	$38.00	February	T-Shirt	88
2	$55.00	January	T-Shirt	185
2	$55.00	February	T-Shirt	65
...	...	...		...

First ML model? Linear regression!

Linear Regression

Linear Regression

(General linear regression model)

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_d x_d$$

(An example model)

$$\text{sales} = \beta_0 + \beta_1 * \text{price} + \beta_2 * \text{onSale}$$

x: features
y: label
β: model "weights"

How do we find the best weights?

Model Training: Finding "Best" Weights

Find β values that minimize a **loss function:**

$$\text{Loss}(\boldsymbol{\beta}) = \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} \ldots \beta_d x_{di}))^2$$

actual sales

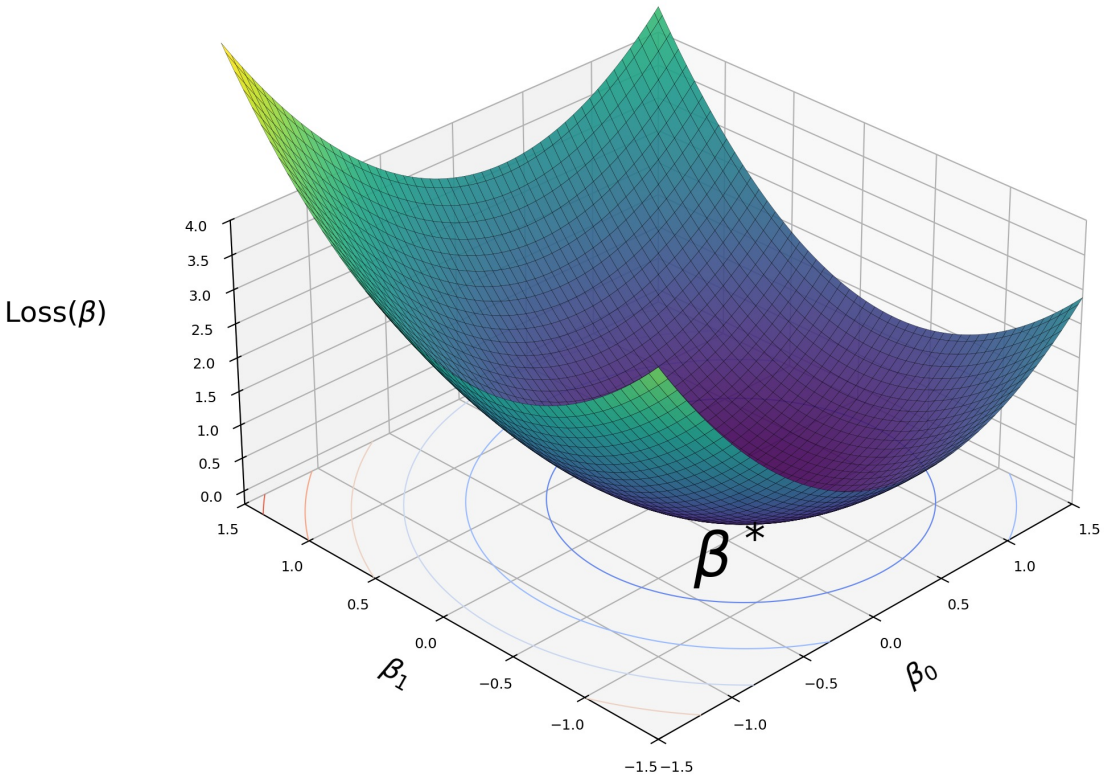
predicted sales

Measures the mismatch between model predictions under **β** and the true labels

Finding the best **β** is an optimization problem

Whenever you hear "model training" think "finding optimal or good enough weights"

Loss Surface



Optimal weights **β*** found by minimizing the loss function

Lots of Python packages etc. to handle this -- we will skip details for now

Making Predictions

With trained coefficients, prediction is simple arithmetic:

Trained Model:

$$sales = 850 - 15 * Price + 50 * onSale$$

Example Calculation:

Given: price = $38, onSale = 1:

$$\begin{aligned} sales &= 850 + (-15)(38) + (50)(1) \\ &= 850 - 570 + 50 \\ &= 330 \text{ units} \end{aligned}$$

Evaluating Prediction Error

RMSE: “Root Mean Squared Error”: average (actual - predicted)², then square-root it

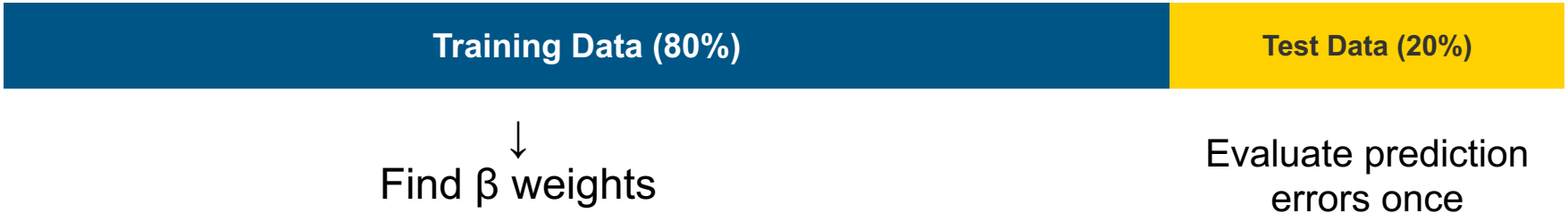
MAE: “Mean Absolute Error”: average of |actual - predicted| across all observations. Easier to interpret: "on average, we're off by X units"

Best Practices: Train/Test Split

Test set represents future, unseen data

Detects *overfitting* (memorization vs learning)

Test set is locked until final evaluation



15-min Break

Feature Engineering

Feature Engineering

What to do when features are limited?

Feature engineering: Use existing features to create new features!

Example: We observe prices for each product-month. We can create new variable

$$\%PriceChange_t = (Price_t - Price_{t-1}) / Price_{t-1}$$

What other new features would make sense for H&M sales prediction?

Feature Engineering: Proceed with Caution

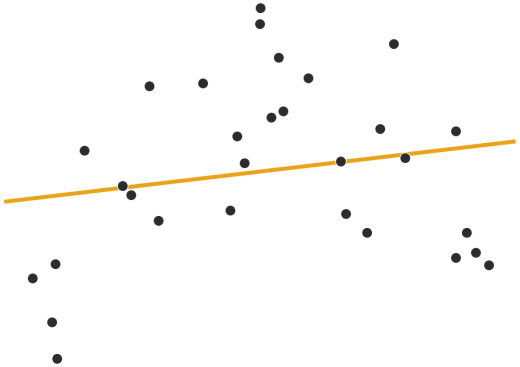
Feature engineering allows for richer models by increasing predictors

But can risk overfitting depending on how many features we have vs. data

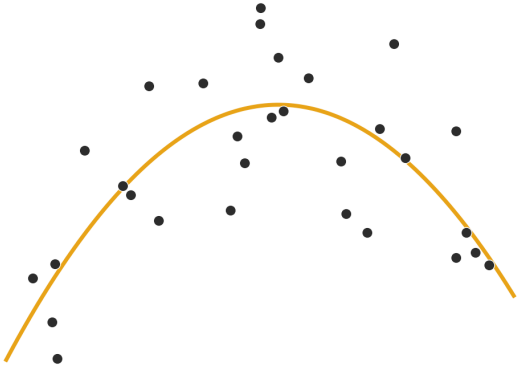
Overfitting: Model is too complex for available data, fits to random noise instead of extracting real signal

Warning signs: test error much higher than training error

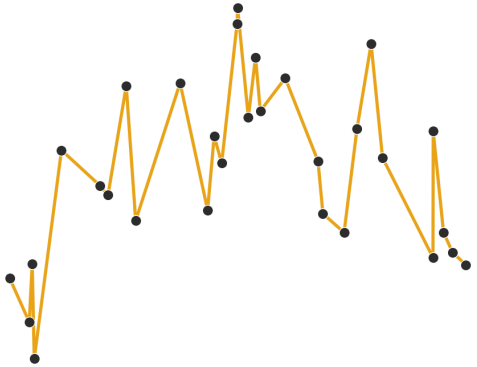
Underfit



Good Fit



Overfit



Regularization: Preventing Overfitting

Regularization

Regularization refers to any method that aims to limit model complexity

In linear regression, regularization means **shrinking large model weights**

Baseline: Standard Linear Regression

Regularization refers to any method that aims to limit model complexity

In linear regression, regularization means **shrinking large model weights**

$$\text{SSE} = \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} \dots \beta_d x_{di}))^2$$

LASSO Regression

$$\text{Loss} = \text{SSE} + \lambda \sum_{j=1}^{d} |\beta_j|$$

Feature	λ=0	λ=1	λ=10	λ=100
price	-2.5	-2.3	-1.8	-0.9
January	+3.2	+2.9	+1.5	**0**
February	+45	+42	+30	+12
black	+0.3	+0.1	**0**	**0**
striped	-0.8	-0.2	**0**	**0**

Most predictive features survive

Best Practices: Feature Standardization

Problem: Features have different scales

Feature	Range	Units
price	$10 – $150	dollars
productAge	1-10	months

Rescaling formula:

$$z = (x - \text{mean}) / \text{std\_dev}$$

Example:

Feature	Before (raw)	After (scaled)
price	42.50	0.15
productAge	5	0.4

All features now have mean=0, std=1

Makes LASSO penalty apply “more evenly” across variables

Also allows for direct comparison of coefficient weights

Ridge Regression

$$\text{Loss} = \text{SSE} + \lambda \sum_{j=1}^{d} \beta_j^2$$

Ridge vs LASSO (λ=10):

Feature	No Reg	LASSO	Ridge
Price	-2.5	-1.8	-2.1
January	+3.2	+1.5	+2.6
February	+45	+30	+39
black	+0.3	**0 (removed)**	**+0.2 (kept)**
striped	-0.8	**0 (removed)**	**-0.6 (kept)**

Does not fully eliminate variables from model

ElasticNet: Best of Both Worlds

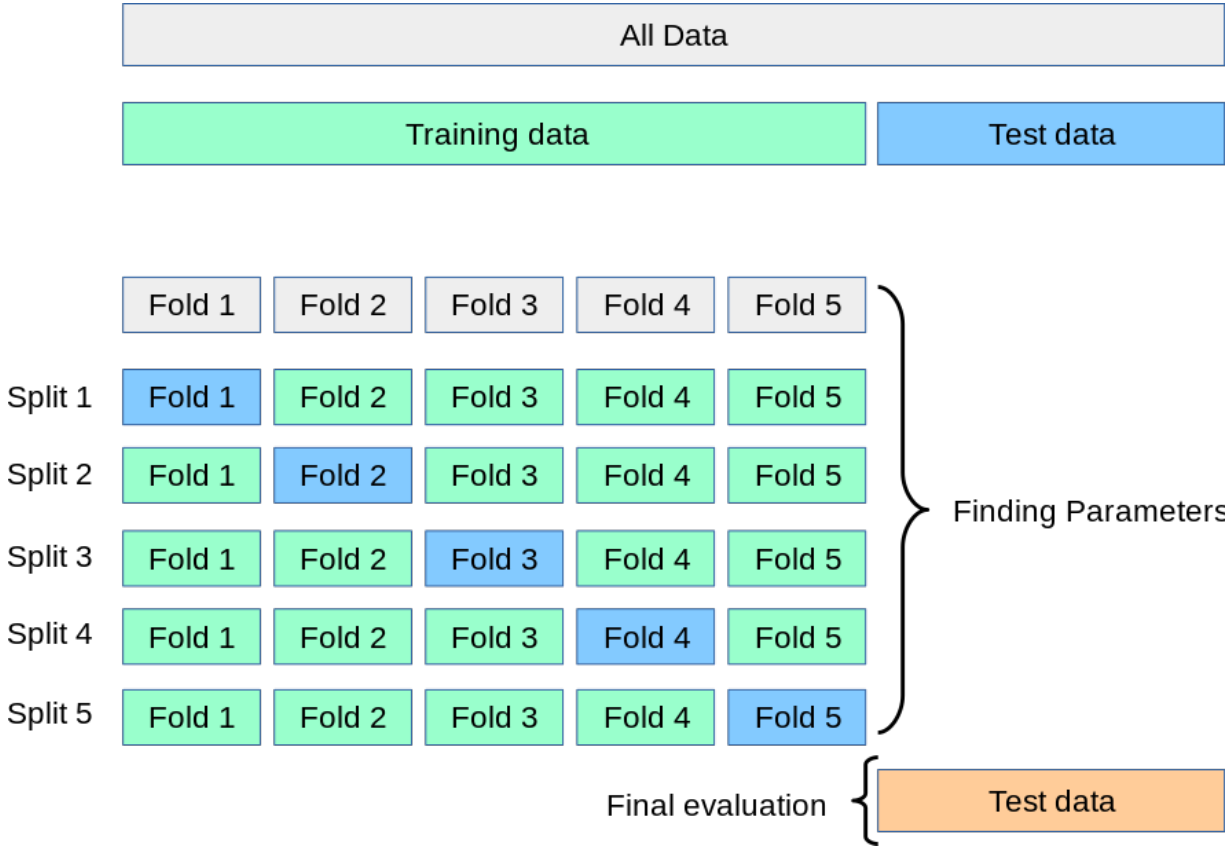
$$\text{Loss} = \text{SSE} + \lambda \left[\underbrace{(1-\alpha) \sum_{j=1}^{d} \beta_j^2}_{\text{Ridge}} + \underbrace{\alpha \sum_{j=1}^{d} |\beta|}_{\text{LASSO}} \right]$$

λ - how strongly to regularize (a positive number)

α - how strongly the regularization leans toward LASSO vs. Ridge (between 0 and 1)

Cross-Validation

We choose best penalty parameters through **cross-validation**



Regularization Summary

LASSO

Feature Selection
✓ Yes (zeros)

Correlated Features
Struggles

Best For
Feature selection

Ridge

Feature Selection
✗ No

Correlated Features
✓ Excellent

Best For
Correlated features

ElasticNet

Feature Selection
✓ Partial

Correlated Features
✓ Good

Best For
General purpose

A Standard ML Pipeline for Linear Regression

1. Organize and clean data, choose features (x) and label (y)
2. Engineer new features if needed based on context
3. Standardize features, create training/test split
4. Train model on training data; use cross validation for regularization parameters if using LASSO/Ridge/ElasticNet
5. Report errors


colab

In-Class Assignments

How this will work:

Each week's in-class assignment has a web frontend on top of a Python backend
→ avoids the need to deal with code so you can focus on concepts

Questions available on BruinLearn as a PDF, add your responses and submit

Due 5pm the day of class, with two missed assignments allowed

Individual, mostly graded "on completion"

Assignment 1

Glossary (1/2)

Machine learning	A branch of AI where models learn patterns from data instead of being explicitly programmed with rules.
Features (x)	The input variables a model uses to make predictions, such as price or discount rate.
Label (y)	The output variable the model is trying to predict, such as units sold.
Weights	Learned numerical values that determine how much each feature influences the prediction.
Training	The process of finding the best model weights by minimizing prediction error on known data.
Loss function	A formula that measures how wrong the model's predictions are; training tries to make this as small as possible.
Overfitting	When a model memorizes the training data (including noise) and performs poorly on new data.

Glossary (2/2)

Train/test split	Dividing data into a training set (to learn from) and a held-out test set (to evaluate on), so we can evaluate performance and detect overfitting.
Feature engineering	Creating new input variables from existing ones to help the model capture more complex patterns.
Regularization	Any technique that penalizes model complexity to prevent overfitting, such as LASSO or Ridge.
LASSO (L1)	A regularization method that adds a penalty based on the absolute value of coefficients, which can zero out weak features entirely.
Ridge (L2)	A regularization method that adds a penalty based on the squared value of coefficients, shrinking all of them but keeping them nonzero.
Cross-validation	A technique for evaluating model performance by rotating which portion of the data is used for testing, used to choose hyperparameters like the regularization strength.