DSC 190

Machine Learning: Representations

Lecture 1 | Part 1

Introduction

Welcome to DSC 190

Introduction to Machine Learning: Representations

History

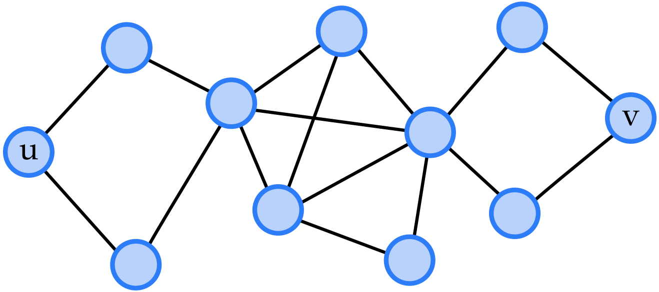
You've had two classes in ML already...

- DSC 40A (theory)
- DSC 80 (practice)

What is Machine Learning?

- Computers can do things very quickly.
- But must be given really specific instructions.
- **Problem**: Not all tasks are easy to dictate.

Example (Easy)



Problem: Find a shortest path between *u* and *v*.

Example (Not so easy)

Problem: On a scale from 1-10, how happy is this person?

The Trick: Use Data

8

3

5

4

7

6

10

?

What is Machine Learning?

- Before: Computer is **told** how to do a task.
- Instead: **learn** how to do a task using data.

What is Machine Learning?

- Before: Computer is **told** how to do a task.
- Instead: **learn** how to do a task using data.
- We still have to **tell** the computer how to learn.

An **ML algorithm** is a set of precise instructions telling the computer **how to learn** from data.

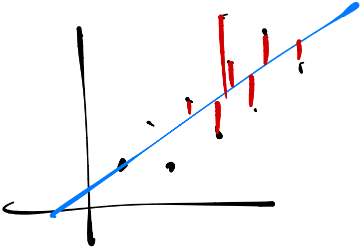
An **ML algorithm** is a set of precise instructions telling the computer **how to learn** from data.

Spoiler: the algorithms are usually pretty simple. It’s the **data** that does the real work.

An **ML algorithm** is a set of precise instructions telling the computer **how to learn** from data.

Spoiler: the algorithms are usually pretty simple. It's the **data** that does the real work.

This is because real world data has "**structure**".

Problem: On a scale from 1-10, how happy is this person?

Exercise

What kind of learning task is this (e.g., classification)? What learning algorithm(s) have you heard of for this kind of task?

Recall: Least Squares Regression

- Example: predict the price of a laptop.

- Choose some **features**:
  - CPU speed, amount of RAM, weight (kg).

- Prediction function (weighted "vote"):

$$(\text{price}) = w_0 + w_1 \times (\text{cpu}) + w_2 \times (\text{ram}) + w_3 \times (\text{weight})$$

- Learn w_i by minimizing **squared error**.

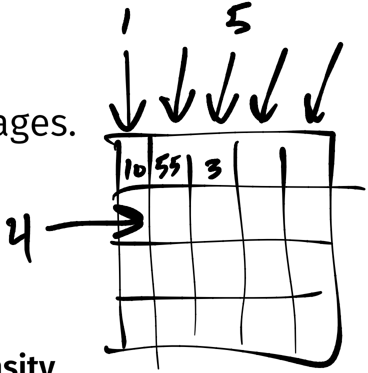
Representations

- Computers don't understand the concept of a laptop.
- We had to **represent** a laptop as a set of features.
  - CPU speed, amount of RAM, weight (kg).
- Clearly, choosing right **feature representation** is important.

Now: Predict Happiness

- Given an image, predict happiness on a 1-10 scale.
- This is a **regression** problem.
- Can we use least squares regression?

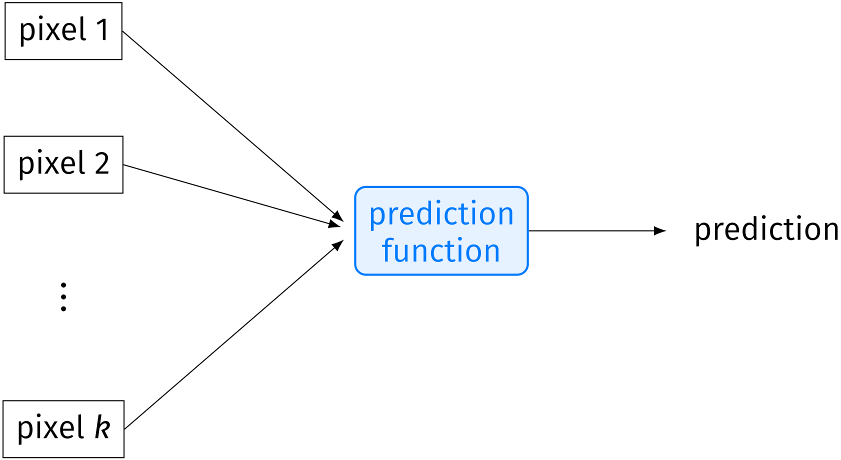
Problem

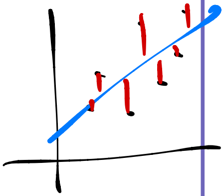


- Computers don't understand images.
- How do we **represent** them?
- Easy approach: a bag of pixels.
  - **Each** pixel has an numerical **intensity**.
  - Each pixel is a feature.
  - In this way, an image is represented as a **vector** in some **high dimensional space**.

Least Squares for Happiness

$$
\begin{aligned}
\text{(happiness)} = {} & w_0 \\
& + w_1 \times \text{(pixel 1)} \\
& + w_2 \times \text{(pixel 2)} \\
& + \ldots \\
& + w_k \times \text{(pixel k)}
\end{aligned}
$$


pixel 1
pixel 2
pixel k
prediction function
prediction



pixel 1

↕

pixel 3000

Exercise

Say we train a least squares regression model on a set of images to predict happiness. We achieve a mean squared error of M_1.

Now we scramble every image's pixels *in exactly the same way* (same transformation of each image). We retrain, and achieve MSE of M_2.

Which is true:

- $M_1 < M_2$
- $M_1 = M_2$
- $M_1 > M_2$

Answer

- The regression model will work just as well if the images are all scrambled in exactly the same way.
- This is because the model doesn't use the **proximity** of pixels.
- The **representation** (each pixel is a feature) does not capture this.

Exercise

Say we train a least squares regression model on a set of images to predict happiness. We achieve a mean squared error of M_1.

Now we scramble every image's pixels *independently*. We retrain, and achieve MSE of M_2.

Which is true:

- $M_1 < M_2$
- $M_1 = M_2$
- $M_1 > M_2$

Happiness: it's in the Pixels

- The information is contained in the image... but not in individual pixels.

- In **patterns** of pixels:
  - The shape of the eyebrows.
  - Angle of the corners of the mouth.
  - Are teeth visible?

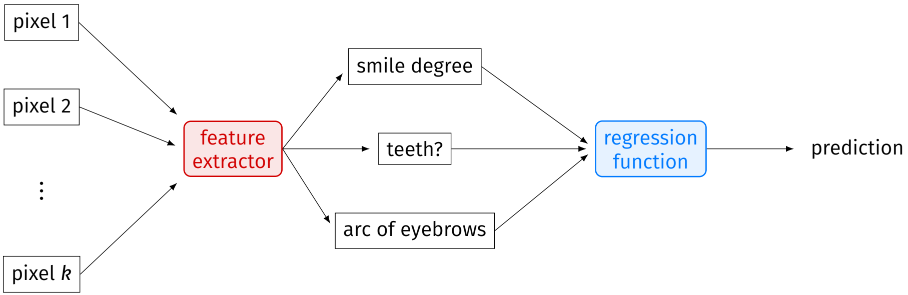
- The representation is **too simple** – probably won't work well[1].

[1] On this example! Works OK on, e.g., MNIST.

Handcrafted Representations

- Idea: build a **feature extractor** to detect:
  - The shape of the eyebrows.
  - Angle of the corners of the mouth.
  - Are teeth visible?

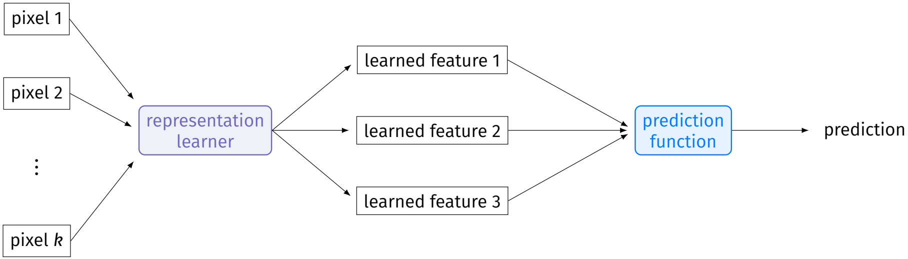
- Use these as high-level features instead.


pixel 1
pixel 2
pixel k
feature extractor
smile degree
teeth?
arc of eyebrows
regression function
prediction

Problem

- Extractors (may) make good **representations**.
- But building a feature extractor is **hard**[2].
- Can we **learn** a good representation?

[2] It took evolution a while to come up with the visual hierarchy.


pixel 1
pixel 2
⋮
pixel k
representation learner
learned feature 1
learned feature 2
learned feature 3
prediction function
prediction

DSC 190

- We'll see how to **learn good representations**.

- Good representations help us when:
  1. making predictions;
  2. doing EDA (better visualizations).

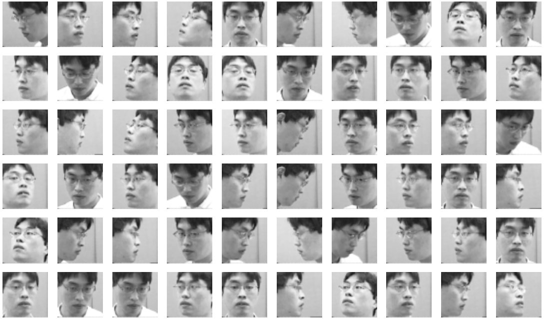
Claim

- Many of the famous recent advancements in AI/ML are due to **representation learning**.

Representations and Structure

- Real world data has structure.
- But "seeing" the structure requires the right representation.

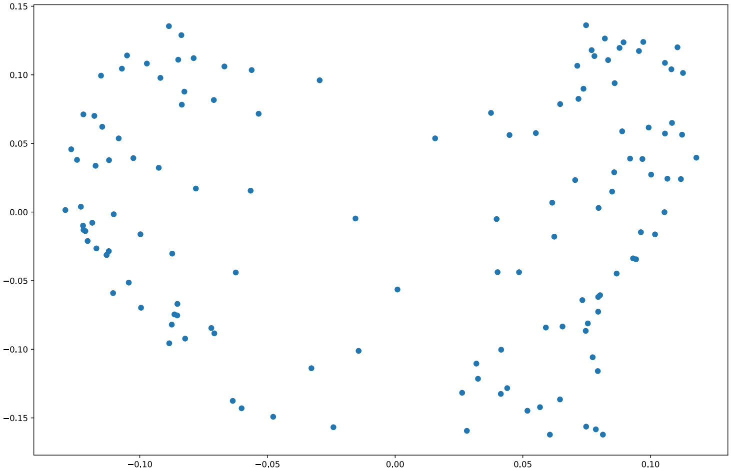
Example: Pose Estimation

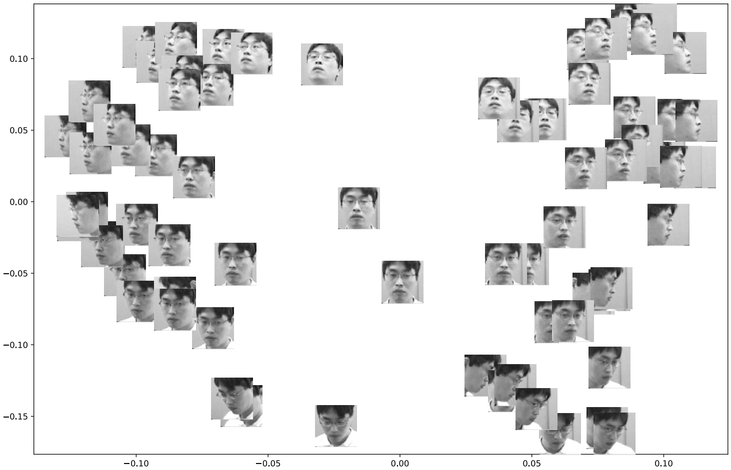


Problem: Classify, is person looking left, right, up, down, netural?

Example: Pose Estimation

- As a “bag of pixels” each image is a vector in $\mathbb{R}^{10,000}$.
- Later: we’ll see how to reduce dimensionality while preserving “closeness”.


0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.10
−0.05
0.00
0.05
0.10


0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.10
−0.05
0.00
0.05
0.10

Main Idea

By learning a better representation, the original classification problem becomes easy, almost trivial.

Example: `word2vec`

- How do we represent a word?
- Google's `word2vec` learned a representation of words as points in 300 dimensional space.
- Two points close $\iff$ words have similar meanings.

Example: `word2vec`

- Fun fact: we can now add and subtract words.
  - They're represented as vectors.
- Surprising results:

$$\vec{v}_{\text{Paris}} - \vec{v}_{\text{France}} + \vec{v}_{\text{China}} \approx \vec{v}_{\text{Beijing}}$$

Example: `word2vec` [3]

Table 8: *Examples of the word pair relationships, using the best word vectors from Table 4 (Skip-gram model trained on 783M words with 300 dimensionality).*

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

[3] "Efficient Estimation of Word Representations in Vector Space" by Mikolov, et al.

Example: Neural Networks

- `word2vec` is an example of a neural network model.
- Deep neural networks have been very successful on certain tasks.
- They **learn** a good representation.








Main Idea

Building a good model requires picking a good **feature representation**.

We can pick features by hand.

Or we can **learn** a good feature representation from data.

That is what this class is about.

Roadmap

- Review of DSC 40A:
  - Minimizing loss
  - Linear models for regression and classification
- Clustering as feature learning
  - k-means clustering
  - RBF networks
- Dimensionality reduction
  - Review of linear algebra
  - Eigenvalues/Eigenvectors
  - PCA

Roadmap

- Manifold learning
- Neural Networks
- Autoencoders
- Deep Learning

Practice vs. Theory

- Goal of this class: understand the fundamentals of representation learning.
- Both practical and theoretical.
- Think: more DSC 40A than DSC 80, but a bit of both.

Tools of the Trade

- We'll see some of the popular Python tools for feature learning.
  - `numpy`
  - `keras`
  - `sklearn`
  - `...`

DSC 190

Machine Learning: Representations

Lecture 1 | Part 2

Syllabus

Miscellaneous

- Campuswire > Email
- No discussion tomorrow.

DSC 190

Machine Learning: Representations

Lecture 1 | Part 3

Is DSC 190 for You?

Is DSC 190 for you?

- DSC 190 will eventually become DSC 140B.
- DSC 140A/140B are **targeted** to DSC majors.
  - Compared to other ML classes, Assume some ML background (40A, 80).

Is DSC 190 for you?

- Unfortunately, it's a little confusing.
- DSC 190 and CSE 151A are equivalent in credit.
- Not equivalent in topics.
- Consequence of creating our own ML in DSC.

Bottom Line

- If you are a DSC major, haven't taken an ML class:
  - Take this class and DSC 140A (in either order).

- If you are a DSC major, have taken an ML class:
  - Talk to an advisor.

"This course substitutes the CORE CSE 151A Requirement. Students cannot receive major or minor credit for both CSE 151A and DSC 190 A00- SP22, as only one course can fulfill this major core requirement."

Bottom Line

- If you’re not a DSC major, looking for an ML elective:
  - This course *might* be a good option if you already have some ML background.
  - But it is targeted to data scientists.
  - CSE 151A, DSC 80, DSC 148, CSE 158, etc. may be better options.

Next Time

- Review of DSC 40A topics.
- Learning as optimizing loss.
- Linear models for regression and classification.