

Machine learning versus Algorithms

Classification with generative models I

DSE 210

In both fields, the goal is to develop

procedures that exhibit a desired input-output behavior.

- **Algorithms:** the input-output mapping can be precisely defined.
Input: Graph G .
Output: MST of G .
- **Machine learning:** the mapping cannot easily be made precise.
Input: Picture of an animal.
Output: Name of the animal.

Instead, we simply provide examples of (input,output) pairs and ask the machine to *learn* a suitable mapping itself.

Inputs and outputs

Basic terminology:

- The input space, \mathcal{X} .
E.g. 32×32 RGB images of animals.
- The output space, \mathcal{Y} .
E.g. Names of 100 animals.



x :

y : "bear"

After seeing a bunch of examples (x, y) , pick a mapping

$$f : \mathcal{X} \rightarrow \mathcal{Y}$$

that accurately replicates the input-output pattern of the examples.

Learning problems are often categorized according to the type of *output space*: (1) discrete, (2) continuous, (3) probability values, or (4) more general structures.

Discrete output space: classification

Binary classification:

- **Spam detection**
 $\mathcal{X} = \{\text{email messages}\}$
 $\mathcal{Y} = \{\text{spam, not spam}\}$
- **Credit card fraud detection**
 $\mathcal{X} = \{\text{descriptions of credit card transactions}\}$
 $\mathcal{Y} = \{\text{fraudulent, legitimate}\}$

Multiclass classification:

- **Animal recognition**
 $\mathcal{X} = \{\text{animal pictures}\}$
 $\mathcal{Y} = \{\text{dog, cat, giraffe, ...}\}$
- **News article classification**
 $\mathcal{X} = \{\text{news articles}\}$
 $\mathcal{Y} = \{\text{politics, business, sports, ...}\}$

Continuous output space: regression

- **A parent's concerns**
How cold will it be tomorrow morning?
 $\mathcal{Y} = [-273, \infty)$
- **For the asthmatic**
Predict tomorrow's air quality (max over the whole day)
 $\mathcal{Y} = [0, \infty)$ (< 100 : okay, > 200 : dangerous)
- **Insurance company calculations**
In how many years will this person die?
 $\mathcal{Y} = [0, 200]$

What are suitable predictor variables (\mathcal{X}) in each case?

Conditional probability functions

Here $\mathcal{Y} = [0, 1]$ represents probabilities.

- **Dating service**
What is the probability these two people will go on a date if introduced to each other?
If we modeled this as a classification problem, the binary answer would basically always be "no". The goal is to find matches that are slightly less unlikely than others.
- **Credit card transactions**
What is the probability that this transaction is fraudulent?
The probability is important, because – in combination with the amount of the transaction – it determines the overall risk and thus the right course of action.

Structured output spaces

The output space consists of structured objects, like sequences or trees.

Dating service

Input: description of a person
Output: rank-ordered list of all possible matches

\mathcal{Y} = space of all permutations

Example:

$x = \text{Tom}$

$y = (\text{Nancy}, \text{Mary}, \text{Chloe}, \dots)$

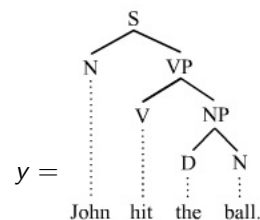
Language processing

Input: English sentence
Output: parse tree showing grammatical structure

\mathcal{Y} = space of all trees

Example:

$x = \text{"John hit the ball"}$



A basic classifier: nearest neighbor

Given a labeled training set $(x^{(1)}, y^{(1)}), \dots, (x^{(n)}, y^{(n)})$.

Example: the MNIST data set of handwritten digits.



To classify a new instance x :

- Find its nearest neighbor amongst the $x^{(i)}$
- Return $y^{(i)}$

The data space

We need to choose a distance function.



Each image is 28×28 grayscale.
One option: Treat images as 784-dimensional vectors, and use Euclidean (ℓ_2) distance:

$$\|x - x'\| = \sqrt{\sum_{i=1}^{784} (x_i - x'_i)^2}.$$

Summary:

- Data space $\mathcal{X} = \mathbb{R}^{784}$ with ℓ_2 distance
- Label space $\mathcal{Y} = \{0, 1, \dots, 9\}$

Performance on MNIST

Training set of 60,000 points.

- What is the error rate on training points? **Zero**.
In general, **training error** is an overly optimistic predictor of future performance.
- A better gauge: separate test set of 10,000 points.
Test error = fraction of test points incorrectly classified.
- What test error would we expect for a random classifier? **90%**.
- Test error of nearest neighbor: **3.09%**.

Examples of errors:

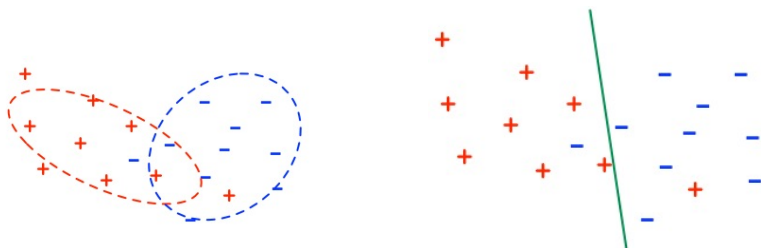


Properties of NN: (1) Can model arbitrarily complex functions
(2) Unbounded in size

Classification with parametrized models

Classifiers with a fixed number of parameters can represent a limited set of functions. Learning a model is about picking a good approximation.

Typically the x 's are points in p -dimensional Euclidean space, \mathbb{R}^p .



Two ways to classify:

- **Generative**: model the individual classes.
- **Discriminative**: model the decision boundary between the classes.

Quick review of conditional probability

Formula for conditional probability: for any events A, B ,

$$\Pr(A|B) = \frac{\Pr(A \cap B)}{\Pr(B)}.$$

Applied twice, this yields Bayes' rule:

$$\Pr(H|E) = \frac{\Pr(E|H)}{\Pr(E)} \Pr(H).$$

Summation rule: Suppose events A_1, \dots, A_k are disjoint events, one of which must occur. Then for any other event E ,

$$\begin{aligned} \Pr(E) &= \Pr(E, A_1) + \Pr(E, A_2) + \dots + \Pr(E, A_k) \\ &= \Pr(E|A_1)\Pr(A_1) + \Pr(E|A_2)\Pr(A_2) + \dots + \Pr(E|A_k)\Pr(A_k) \end{aligned}$$

Generative models

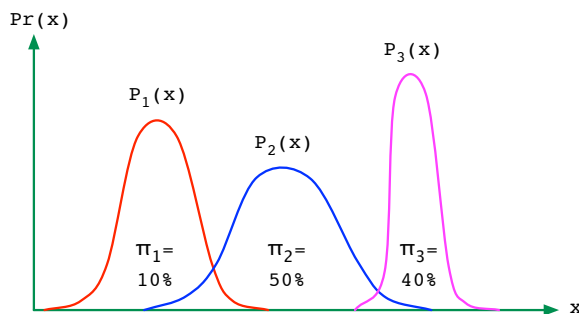
Generating a point (x, y) in two steps:

- 1 First choose y
- 2 Then choose x given y

Example:

$$\mathcal{X} = \mathbb{R}$$

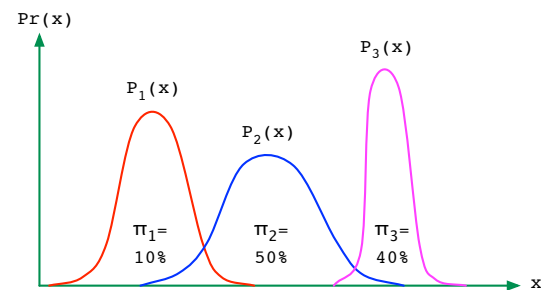
$$\mathcal{Y} = \{1, 2, 3\}$$



The overall density is a mixture of the individual densities,

$$\Pr(x) = \pi_1 P_1(x) + \dots + \pi_k P_k(x).$$

The Bayes-optimal prediction



Labels $\mathcal{Y} = \{1, 2, \dots, k\}$, density $\Pr(x) = \pi_1 P_1(x) + \dots + \pi_k P_k(x)$.

For any $x \in \mathcal{X}$ and any label j ,

$$\Pr(y = j|x) = \frac{\Pr(y = j)\Pr(x|y = j)}{\Pr(x)} = \frac{\pi_j P_j(x)}{\sum_{i=1}^k \pi_i P_i(x)}$$

Bayes-optimal (minimum-error) prediction: $h^*(x) = \arg \max_j \pi_j P_j(x)$.

Estimating the π_j is easy. Estimating the P_j is hard.

Estimating class-conditional distributions

Estimating an arbitrary distribution in \mathbb{R}^p :

- Can be done, e.g. with kernel density estimation.
- But number of samples needed is exponential in p .

Instead: approximate each P_j with a simple, parametric distribution.

Some options:

- Product distributions.
Assume coordinates are independent: naive Bayes.
- Multivariate Gaussians.
Linear and quadratic discriminant analysis.
- More general graphical models.

Naive Bayes

Labels $\mathcal{Y} = \{1, 2, \dots, k\}$, density $\Pr(x) = \pi_1 P_1(x) + \dots + \pi_k P_k(x)$.



Binarized MNIST:

- $k = 10$ classes
- $\mathcal{X} = \{0, 1\}^{784}$

Assume that **within each class**, the individual pixel values are independent:

$$P_j(x) = P_{j1}(x_1) \cdot P_{j2}(x_2) \cdots P_{j,784}(x_{784}).$$

Each P_{ji} is a coin flip: trivial to estimate!

Smoothed estimate of coin bias

Pick a class j and a pixel i . We need to estimate

$$p_{ji} = \Pr(x_i = 1 | y = j).$$

Out of a training set of size n ,

$n_j = \#$ of instances of class j

$n_{ji} = \#$ of instances of class j with $x_i = 1$

Then the maximum-likelihood estimate of p_{ji} is

$$\hat{p}_{ji} = n_{ji} / n_j.$$

This causes problems if $n_{ji} = 0$. Instead, use “Laplace smoothing”:

$$\hat{p}_{ji} = \frac{n_{ji} + 1}{n_j + 2}.$$

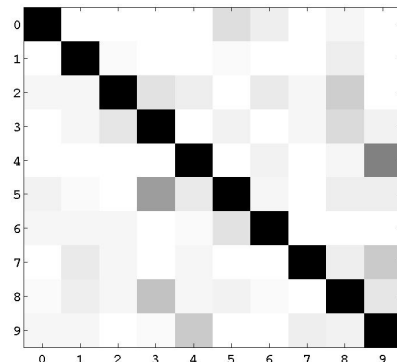
Example: MNIST

Result of training: mean vectors for each class.



Test error rate: 15.54%.

“Confusion matrix” →



Form of the classifier

Data space $\mathcal{X} = \{0, 1\}^p$, label space $\mathcal{Y} = \{1, \dots, k\}$. Estimate:

- $\{\pi_j : 1 \leq j \leq k\}$
- $\{p_{ji} : 1 \leq j \leq k, 1 \leq i \leq p\}$

Then classify point x as

$$\arg \max_j \pi_j \prod_{i=1}^p p_{ji}^{x_i} (1 - p_{ji})^{1-x_i}.$$

To avoid underflow: take the log:

$$\arg \max_j \underbrace{\log \pi_j + \sum_{i=1}^p (x_i \log p_{ji} + (1 - x_i) \log(1 - p_{ji}))}_{\text{of the form } w \cdot x + b}$$

A linear classifier!

Other types of data

How would you handle data:

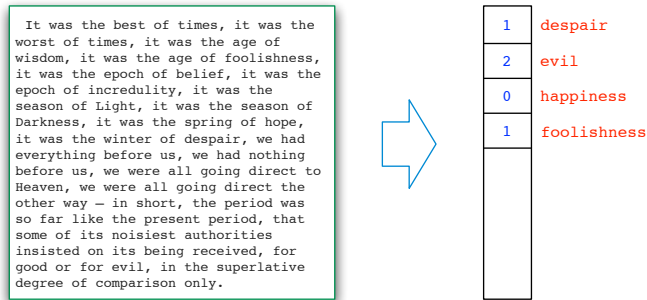
- Whose features take on more than two discrete values (such as ten possible colors)?
- Whose features are real-valued?
- Whose features are positive integers?
- Whose features are mixed: some real, some Boolean, etc?

How would you handle “missing data”: situations in which data points occasionally (or regularly) have missing entries?

- At train time: ???
- At test time: ???

Multinomial naive Bayes

Bag-of-words: vectorial representation of text documents.



- Fix V = some vocabulary.
- Treat the words in a document as independent draws from a **multinomial** distribution over V :

$$p = (p_1, \dots, p_{|V|}), \text{ such that } p_i \geq 0 \text{ and } \sum_i p_i = 1$$

Multinomial naive Bayes: each class is modeled by a multinomial.

Improving performance of multinomial naive Bayes

A variety of heuristics that are standard in text retrieval, such as:

① **Compensating for burstiness.**

Problem: Once a word has appeared in a document, it has a much higher chance of appearing again.

Solution: Instead of the number of occurrences f of a word, use $\log(1 + f)$.

② **Downweighting common words.**

Problem: Common words can have a unduly large influence on classification.

Solution: Weight each word w by **inverse document frequency**:

$$\log \frac{1}{\#(\text{docs containing } w)}$$