CS M146:
Introduction to Machine Learning

Einar Balan

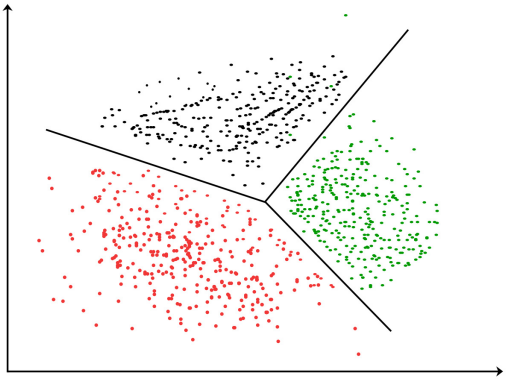
Contents

1 | Introduction

Machine learning is the study of algorithms that improve performance when executing a task based on experience. For example, an algorithm that recognizes hand written digits. The task is the recognition, the performance is measured by the accuracy of recognition, and the experience is the database of human labeled images. Some more applications include reinforcement learning with playing games, language generation, and image generation (stable diffusion).

There are several major types of learning protocols

1. Supervised Learning
    - feed in **labeled** data in order to generalize to unencountered, unlabeled data
    - given target function $f : x \to y$ build a model g that mimics the behaviour of f in a generalized way
    - we build the model based on the training set and test it on the test set (80:20 split)
    - the test set should be labeled in order to measure accuracy of the model (but the model should never see these labels)
2. Unsupervised Learning
    - given **unlabeled** inputs, cluster them together based on traits in common



3. Reinforcement Learning
    - give sequences of states & actions w/ rewards
    - learn actions that maximize reward

Challenges in ML

- structured inference: classifcation may change based on context

- robustness ambiguities may arise that make classifcation difficult
- adversarial attack: adding a small amount of noise in a systematic way may change classification
- common sense: humans can reason through ambiguity based on context & common sense; this is more difficult for machines
- fairness & inclusion: models may treat different races differently and inadvertently excluse certain groups; stereotypes may also be reinforced

Defining a Supervised Learning Problem

Consider the Badges Game in which name badges at a conference were labeled with either a "+" or "-". Given a labeled training set, can we determine what general rules produced the labels?

Several important definitions:

- instance space - what features are we using to produce labels
- label space - what is our learning task i.e. what labels
- hypothesis space - what kind of model are we using
- loss function - how do we evaluate the performance of our model; what makes a good prediction

Instance Space

- consider $\vec{x} \in X$, where x is a feature vector in X our instance space, which is a vector space
- typically $\vec{x} \in \{0,1\}^n$ or $\mathbb{R}^n$
- each dimension of $\vec{x}$ represents a feature
- Examples of features: length of first name, does the name contain the letter X, how many vowels, is the nth letter a vowel, etc.
- Good features are **essential**; we cannot generalize without them

Example

X = [first-char-vowel, first-char-A, first-char-N]

Naoki Abe = [0, 0, 1]

Label Space

How should we classify based on $\vec{x}$? There are a couple options.

- Binary $y \in \{-1, 1\}$
- Multiclass $y \in \{1, 2, 3, ..., k\}$
- Regression $y \in \mathbb{R}$
- Structured Output $y \in \{1, 2, 3, ..., k\}^N$

Example Animal recognition $\vec{x}$: Image Bitmap y:

- Binary: Is it a lion?
- Multiclass: Is it a lion a cat or a dog?

- Multilabel: Is it a lion, mammal, cat, or dog?

Hypothesis Space

This is the set of all possible models. We need to find the best one for our use case. Consider an unknown boolean function. A potential hypothesis space could be every possible function.

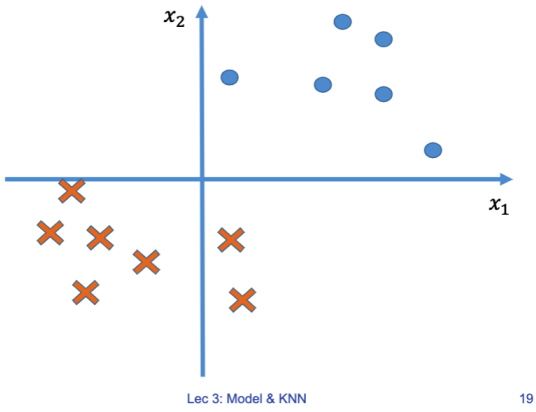
In general there are $|Y|^{|X|}$ possible functions from the instance space X to the label space Y. The hypothesis space is typically a subset of these. We can add rules to limit the hypothesis space, but we need to take care that these rules are not too restrictive, as it is possible that **no** simple rule can explain the data.

To **learn** is to remove remaining uncertainty and find the best function/model in our hypothsis space. In general it is a good idea to start the hypothesis space as restrictive, and get more general.

General Problem Flow

1. Develop a flexible hypothesis space i.e. Decision Tree, Neural Network, Nested Collections, etc
2. Develop algorithm for finding the best hypothesis
3. Hope that it generalizes beyond the data

Example



How to define hypothesis space?

- Option 1: Lines separating the two groups
- Option 2: Proximity to existing data represented by circles

Option 1 will likely generalize better. Option 2 will suffer due to **overfitting**. Underfitting, or not fitting the data well enough, is another concern.

How can we prevent overfitting? Some guidelines:

- use a simpler model e.g. linear
- add regularization
- add noise
- halt optimization earlier

How can we learn? Brute force or optimization with calculus.

Aside: Bias vs Variance

- bias: data is shifted a consistent amount
- variance: data is spread out around a point

2 | K-Nearest Neighbors

KNN is a type of supervised learning classifier which uses proximity to make classifications. It can be useful to determine a category for something i.e. spam or not spam, or type of flower. KNN is quite good at this.

Basic Algorithm

- Learning: just store training samples
- Prediction: Find k existing examples closest to input and group them; construct new labels based on k Neighbors
- Issues:
  - Need to define distance based on the domain i.e. Euclidean, Manhattan, L-p norm, Hamming (# diff bits), etc.
  - What is the best value of k?