Notes on Pattern Recognition

Andres Ponce

March 10, 2020

Pattern Recognition is a subset of Machine Learning and Aritficial Intelligence in general. It's concerned with analyzing data, extracting particular features or patterns from that data. We will discuss severl methods for feature extraction, all the way to neural networks and just touch on deep learning!

Introduction

Here, I am using the `tufte-latex` package for LaTeX. I thought the design looked quite nice and wanted to give it a shot!

Curve Fitting

There exist two main types of problems in the field of pattern recognition. First, there is ***regression***, and there is ***categorization***. [1]

The terms *Artificial Intelligence, Machine Learning*, and *Pattern Recognition* all share common properties. PR $\wedge$ ML $\subset$ AI. ML atempts to make computers take in empirical data and make decisions. AI, more broadly, tries to make computers perform actions that were usually thought to be exclusive to humans.

Some different types of PR problems involve **supervised** and **unsupervised** learning. [2] There might be a couple of problems that better suit un-supervised learning, such as:

- **Clustering**: Clustering involves finding patterns in the data which more often resemble themselves rather than other data, i.e. finding similar patterns within the data.
- **Density Estimation**: Here, we try and find the probability density function given a set of points.[3]
- **Dimensionality Reduction**: When multiple variables are involved in a problem, the space in which their solution exists rises exponentially for each new variable.[4]

Although we can imagine supervised scenarios, IRL our model would probably have to find the solutions on its own.

We can imagine having a network actually *generate* new data based on the patterns its seen before. For example, given many human faces, we could create a model that generates human faces which do

[1] **regression** problems deal with the mapping of an input vector to a continuous space, whereas **categorization** takes an input and places it in a finite and discrete set of different categories.

[2] Their difference is whether their target data is available when the problem begins, i.e. whether we know the correct answer.

[3] Remember the PDF only represents a probability density over a continuous interval.

[4] i.e. solving for three variables is exp. harder than solving for two.

not exist in reality. This would be the idea of a **Generative Adversarial Network**(GAN)[5]

[5] A GAN could have *two* networks, one that generates the data and the other one that checks its validity.

Finallly, curve fitting! So, we have a polynomial function, whose generic form is

$$y(x,w) = w_0 + w_1 x + w_2 x^2 + ... + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

M gives us the order of the polynomial, and if I remember my algebra correctly, it should have $M-1$ curves. If M is too large, we *overfit* and our function will have too many curves and not match the actual function all that well.

w gives us the value of the constant mulitplier, which can be found by minimizing the **expectation** $E(w)$.[6] This expectation is fiven by

[6] The expectation is like the average, the value we should expect to see from a random variable.

$$E(w) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n,w) - t_n\}^2$$

Basically we want the difference between the value that our function gave us and the acutal value of the function.

Because we want to avoid overfitting, we introduce a weight term to our cost function. In Figure 1, our model tries to overcompensate because it tries to fit to every single data point, with all its discrepancies included. Therefore, if we add a specific value to our cost function, it would encourage the model to choose a curve that works within the defined parameters and is not too complex. Our new curve might be something like:

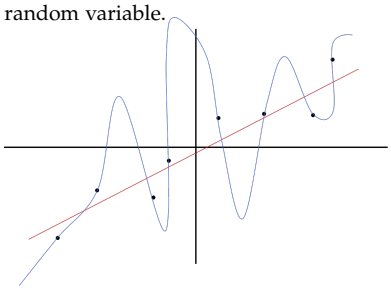

Figure 1: Overfitting leads to overly complex models. The blue line is our model.

$$\tilde{E}(w) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n,w) - t_n\}^2 + \frac{\lambda}{2}\|w\|^2$$

where $\|w\|^2 = w^T w = \sum_{n=0}^{M} \omega_n^2$. We could also include a linear term for w, but with its quadratic version it would be a **ridge regression**. We will have to choose our w such that $\tilde{E}(w)$ is minimized, thus if we add a bigger term at the end(quadratic), the value rises quicker and we'll choose a smaller w.

Probability

We need to review the fundamentals of probability before moving.

The **expectation** of a probability can be thought of as the average value of a probability distribution, or the value that we should expect should we pick a random value from the distribution; the value that the distribution revolves around. The main idea is that we multiply all the possible values of $f(x)$ by the probability that it is such a way. Thus, for a discrete distribution, the expectation is

$$\mathbb{E}[f] = \sum_x p(x) f(x)$$

We replace the sum with an integral for a continuous distribution.

The **variance** is the expected value that a random variable deviates from the expected value. [7]

$$var[f] = \mathbb{E}[(f(x) - \mathbb{E}[f(x)])^2]$$

[7] **Variance** is the expected amount that the expected amount differs from $f(x)$. The value is squared so the expectation removes positive.

Covariance refers to the amount that two random variables vary together. The **Gaussian Distribution** refers to a very complicated formula that describes a probability distribution. There can be multivariate distributions that utilize a Gaussian distribution, which has a **covariance matrix** for all the variables and how they relate to each other.

Bayes's Theorem

This theorem relates the inverse conditional probability $p(w|D)$ with $p(D|w)$.

$$p(w|D) = \frac{p(D|w)p(w)}{p(D)}$$

Given the set of observations D, we can formulate how likely it was for a certain coefficient w to arise both before observing the data and after. [8]

[8] The **prior probability** $p(w)$ of the parameter w is its likelihood before observing D. $p(D|w)$, the **likelihood function**, relates how likely the observations are given the parameter. It's gotten after observing D.

Probability Distributions

Suppose we have an unfair coin that we are throwing in the air n times. We would like to measure the probability of getting x heads. The probability distribution would be: $p(x) = (p(x))^x(1 - p(x))^{1-n}$. This is known as the **Bernoulli Distribution**

Similar to the Bernoulli distribution, we might consider the **Binomial Distribution**, where factor in the *different* possible ways to draw M heads. The formula then becomes:

$$\text{Bin}(m|N, \mu) = \binom{N}{M}\mu^m(1 - \mu)^{N-m}$$

[9]

[9] This formula relates the probability fo the *combinations*, or possible orderings, in which M events happen.