Applied Machine Learning

Saul Pierotti

March 27, 2020

Introduction

- Professor is a physicist in high-energy physics
- We will not go so much into theory
- The exam will be a ML project
- ML is the capacity of a computer to do a task without being explicitly programmed
- AI contains ML, which contains DL (deep learning)
  - ML started in 1980, DL in 2010
- Strong AI is really far
- ML can learn faster and with lower latency than humans
- It is useful for tasks that humans cannot or don't want to do
- Why today? Data avalilable and Cloud computing
- ML can be supervised, unsupervised and reinforcement learning
- Supervised: I know some real solutions
  - It is a regression or classification problem
  - Regression: continuous
  - Classification: discrete
- Unsupervised: no label on the data
  - I use clustering algos
  - I want to find some structure in the data
  - I can get groups, but I don't know the meaning of these groups

Univariate linear regression

- I can define a cost function that measures the average distance of the real outcomes from my regression
- I want to choose the parameters θs that minimize the cost function $J(\theta_1, \theta_2, ..., \theta_n)$
  - In a linear regression the cost function has 2 parameters (!)
    - Intercept and angular coefficient
- To minimise a function I can use a gradient descent algo
  - It is an iterative process
  - For now, only local minima, no global
  - It uses an aggressivness factor α , which is how big every step is
    - If too small it is too slow
    - If it is too large I can miss a minimum
    - α is referred to as an hyperparameter
      - It refers to the learning, not to the problem
  - When updating θs, all of them must be updated simultaneously
- The minimization algo can be analytical or iterative
  - An analytical solution to univariate linear regression exists
  - In ML the analytical version does not scale well
  - GD is the iterative approach

- The iterative update of θ is done by subtracting to its previous value α times the partial derivative of the cost function with respect to θ
    - If the derivative is positive θ decreases, if negative increases, if 0 doesn't change
    - The magnitude of the change is proportional to the derivative at that point (!)
- In a linear regression the cost function is always a convex quadratic: the only minimum is the global minimum (!)
- Batch GD: start from any point and apply GD until I get to a minimum
    - It is batch since at every iteration I evaluate the cost function for the whole batch of datapoints

Multivariate linear regression

- The real world is multivariate (!)
    - Nonetheless, unuvariate is useful for understanding concepts
- I have one θ for each x, plus θ_0
    - θ_0 is a bit unconfortable, since it is different from the others (no x associated!)
    - To make things easier, I introduce $x_0 = 1$ that multiplies θ_0
    - This means that I have n+1 dimentional vectors if n is the number of independent variables
    - In this way, I have a vector of xs and a vector of θs
    - I can represent the whole multivariate function as a product of the x vector with the traspose of the θ vector
    - $h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta 2_2 x_2 + ... + \theta_n x_n = \boldsymbol{\theta}^T \boldsymbol{x}$
- The different variables can have different magnitudes, and I want to account for this
    - To correct, I will do feature scaling
    - I divide the data for the highest value for that variable
    - My data becomes all in the range 0-1
    - Outliers can skew my features: I remove them
    - More generally I want to be in the -1/+1 range since x_0 is already 1
    - I need to rescale also features which are really small
- A different way can be to do mean normalisation
    - I subtract the mean and divide for the range (max-min) or stdev

Learning rate

- The selection of α is important for determining if the GD converges, and if it does how much does it take
- How do I determine if the GD has converged?
    - I can decide a threshold decrese, i.e. if J decreses of less than 10^{-3} in one iteration I stop
- If I see a strange behaviour (divergence, bouncing around) the first thing to try is to decrease α
- But what values for α?
    - First try in factor 10 steps: 0.0001, 0.001, 0.01, 1, 10, . . .
    - Then go to a factor 3

Polynomial regression

- It is the simplest non-linear model but it can fit really complicated behaviours
- I can create features: instead of using x, why not e^x?
    - I can make linear dependencies which are not linear
- I can reduce any polynomial regression to a linear by adding new features (!)
    - I can use x and x^2 instead of only x