CS229: Machine Learning

Junchuan Zhao

May 17, 2022

Abstract

This document includes my notes of CS229: Machine Learning.

1 Lecture 2

1.1 Linear Regression

Hypothesis: $h(x) = \sum_{j=0}^{n} \theta_j x_j$ (n: number of features)
Linear Regression: the hypothesis is the linear combination of the training dataset.
Loss function (goal): $\min_{\theta} \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)^2$
Parameters: θ, m, n, x, y

1.2 Gradient Descent

1.2.1 Batch Gradient Descent

Basic Ideas: start with some θ, keep changing θ to reduce $J(\theta)$, repeat until convergent
$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$ (α: learning rate)
$\frac{\partial}{\partial \theta_j} J(\theta) = (h_\theta(x) - y) \cdot x_j$
$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x)^i - y^i) \cdot x_j^i$ (α: learning rate)
Batch Gradient Descent: all the training data as a batch when optimizing the loss function. (-): not good for large scale dataset, very expensive.

1.2.2 Stochastic Gradient Descent

Algorithm 1 Stochastic Gradient Descent

Require: the parameter of j^{th} feature
Ensure: the updated parameter of j^{th} feature
1: **for** $i = 1$ to m **do**
2: $\quad \theta_j := \theta_j - \alpha \cdot (h_\theta(x)^i - y^i) \cdot x_j^i$;
3: **end for**

stochastic gradient descent heads over to the global optimum.

1.2.3 Total Equations

A Total Equation for Stochastic Gradient Descent: $\nabla_\theta J(\theta) = \mathbf{0}$
If A is square ($A \in \mathbb{R}_{n\times n}$)
Trace of A: $tr(A) = \sum_i A_{ii}$
Features of Trace:

- $tr(A) = tr(A^T)$
- $f(A) = tr(AB)$, $\nabla_A f(A) = B^T$
- $tr(AB) = tr(BA)$
- $tr(ABC) = tr(CAB)$
- $\nabla_A tr(AA^TC) = CA + C^TA$

Loss Function: $\nabla_\theta J(\theta) = \frac{1}{2}(X\theta - y)^T(X\theta - y)$
$= \frac{1}{2}(\theta^T X^T - y^T)(X\theta - y)$
$= \frac{1}{2}(\theta^T X^T X\theta - \theta^T X^T y - y^T X\theta)$
$= X^T X\theta - X^T y$
Loss Function: $X^T X\theta - X^T y = \mathbf{0} \iff X^T X\theta = X^T y$
$\theta = (X^T X)^{-1} X^T y$

2 Lecture 3

2.1 Locally Weighted Regression

"Parametric" learning algorithm: fit fixed set of parameters (θ_i) to data.
"Non-parametric learning algorithm": amount of data/parameters you need to keep grows (linearly) with the size of data.

Linear Regression: to evaluate h at certain x
Fit θ to minimize
$\frac{1}{2}\sum_i (y^i - \theta^T x^i)^2$,
return $\theta^T x$

Linear Regression: to evaluate h at local region of x
Fit θ to minimize
$\sum_i w^i (y^i - \theta^T x^i)^2$, where w^i is a "weight function".
common choice for w^i is $w^i = e^{(-\frac{(x^i - x)^2}{2\tau^2})}$
If $|x^i - x|$ is small, then $w^i \approx 1$.
If $|x^i - x|$ is large, then $w^i \approx 0$.
τ: bandwidth, control a larger or narrower window.

2.2 Why Square Error?

Assume $y^i = \theta^T x^i + \epsilon^i$, where $\epsilon^i \sim N(0, \sigma^2)$ models effects of random noise
$p(y^i|x^i;\theta) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y^i - \theta^T x^i)^2}{2\sigma^2}}$
$\iff y^i|x^i;\theta \sim N(\theta^T x^i, \sigma^2)$

Difference between Likelihood (L) and Probability (P): L variess parameters, P varies datapoints.
Assume the datapoints are IID,
The "likelihood" of θ: $L(\theta) = p(\boldsymbol{y}|\boldsymbol{x};\theta) = \prod\limits_{i=1}^{m} p(y^i|x^i;\theta)$
$l(\theta) = log \prod\limits_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} e^{(\dots)} = \sum\limits_{i=1}^{m} [log \frac{1}{\sqrt{2\pi}\sigma} + log e^{(\dots)}]$
$= mlog \frac{1}{\sqrt{2\pi}\sigma} - \sum\limits_{i=1}^{m} \frac{(y^i - \theta^T x^i)^2}{2\sigma^2}$

MLE: maximum likelihood estimation. Choose θ to maximize $L(\theta)$
i.e. choose θ to minimize $\frac{1}{2} \sum\limits_{i=1}^{m} (y^i - \theta^T x^i)^2$, which is actually $J(\theta)$