Machine Learning Talk I

Why does stochastic gradient descent work so well?

Axel G. R. Turnquist

NJIT Department of Mathematical Sciences

September 25, 2020

Machine Learning

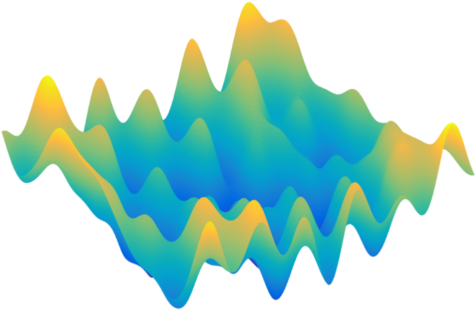
- **Unsupervised learning**: given data x and labels y, does there exists a smooth $f(x) = y$ (regression)? Quantitative or qualitative data (classification).
- **Uniform function approximator** Guarantees convergence?
- Regression is done by minimizing a **loss function** $L(\mu) := \frac{1}{n}\sum_n ||g(x_n, \mu) - y_n||$, via adjusting parameters μ:

$$\bar{\mu} = \text{argmin}_\mu L(\mu) \tag{1}$$

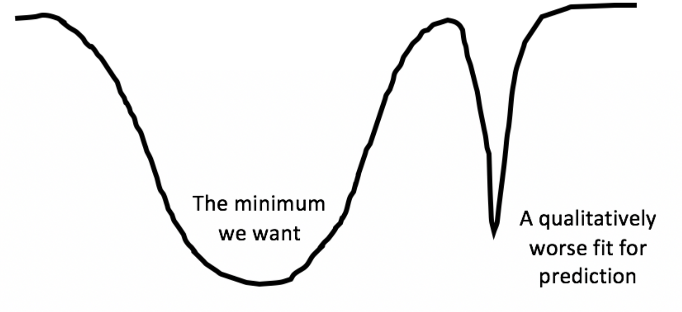
- Done via **backpropagation**. Could try to use **Newton's method** to move downhill, but matrix inversion too expensive.
- Not smooth enough, so use **gradient descent** (still expensive for large n)
- Use **stochastic gradient descent**

Observations Made in the Literature

Stochastic gradient descent works better than it should!

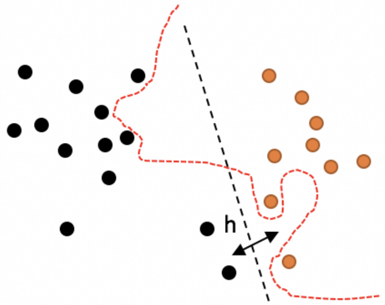


Why does stochastic gradient descent work so well? In high-dimensions:

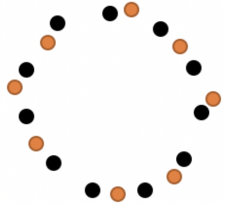


Separability & Classification

- For the classification problem, the hyperplane is the solution found by machine learning, volume of minimum given by h.
- Other fit is too sensitive and has less flexibility (smaller volume in parameter space)



Hard Classification Problem



We do not typically see these kinds of problems. High-dimensional "natural" data is "easily" separable.

Concentration on n-Sphere

Fact: The uniform measure clusters about any equator.

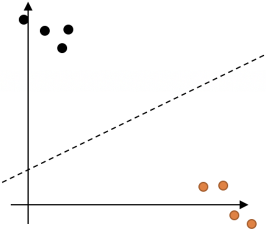
- Uniform measure on n-sphere σ_n.
- Define spherical cap A, where $\sigma_n(A) = 1/2$. This is extremal set of the **isoperimetric inequality**, which means that it is a hemisphere of the n-sphere.
-
$$A_r := \{x \in \mathbb{S}^n : d(x, A) < r\} \tag{2}$$
where $d(x, \cdot)$ is the Riemannian distance on the n-sphere.
- Then,
$$1 - \sigma_n(A_r) \leq e^{-(n-1)r^2/2} \tag{3}$$
Mass collects around *any* equator, "equators are large".

Approximate Orthogonality

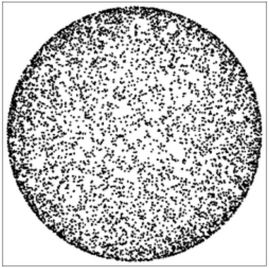
Fact: Randomly sampled vectors on the n-sphere are approximately orthogonal for large n.

- Take a randomly sampled point $\mathbf{x} \in \mathbb{S}^n$, and define an axis in this direction. I.e., define an orthonormal basis, where in this basis $\mathbf{x} = (1, 0, 0) \in \mathbb{R}^{n+1}$, so $\mathbf{x}$ is at the north pole.
- Now, sample $\mathbf{y}$ randomly from $\mathbb{S}^n$. With high probability, it will be located within a distance $1/\sqrt{n}$ of the equator (at zero latitude). Thus, with high probability it will be approximately orthogonal to $\mathbf{x}$ at the north pole.

Upshot: Approximately orthogonal data points are easy to separate (classify) in finite vector spaces.



Q: Why do we care about points randomly chosen on the sphere?

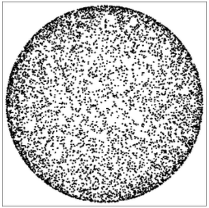


A: It is a good model for randomness in high dimensions.

Gaussians in High-Dimensions

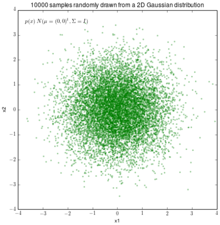
The normalized *n*-**dimensional Gaussian**:

$$p(|\mathbf{x}|) = \frac{1}{(2\pi)^{\frac{n}{2}}} e^{\frac{-|\mathbf{x}|^2}{2}} \tag{4}$$



Gaussian Annulus Theorem:
For a n-dimensional unit variance spherical Gaussian, for any positive real number $\beta \leq \sqrt{n}$, all but at most $3e^{-c\beta^2}$ of the mass lies within the annulus $\sqrt{n} - \beta \leq r \leq \sqrt{n} + \beta$, where c is a fixed positive constant.

$$P(r-\epsilon \leq |\mathbf{x}| \leq r+\epsilon) = \frac{n\pi^{\frac{n}{2}}}{\Gamma\left(\frac{n}{2}+1\right)} \int_{|\rho-r|<\epsilon} p(\rho)\rho^{n-1} d\rho \tag{5}$$



Noisy data is easily separable in high dimensions!

This is perhaps why backpropragation via stochastic gradient descent works well on “natural” data.

Questions?

Resources & Future Topics

- “Pattern Recognition and Machine Learning” Christopher M. Bishop
- “The Concentration of Measure Phenomenon” Michel Ledoux
- Machine learning talks given by applied mathematicians

Future Topics:

1. Machine learning as function regression, conditional expectation (*Binan* ?)
2. Adversarial attacks
3. GAN, WGAN, etc.
4. Data Augmentation
5. Further Information Geometry, High-Dim. Information (*Axel*)