大数据分析

Scalable Machine Learning

least square regression

刘盛华

What is machine learning

- **Study of algorithms that**
  - **improve their performance**
  - **at some task**
  - **with experience**

Data → Learning algorithm → Understanding

(experience) **(task)** **(performance)**

Barnabás Póczos, CMU

Warnings about the Class

"There is nothing more practical than a good theory"

Lewin (1952)

Linear Regression

Sketching

Massive data sets

- **Examples**
  - **Internet traffic logs**
  - **Financial data**
  - **etc.**

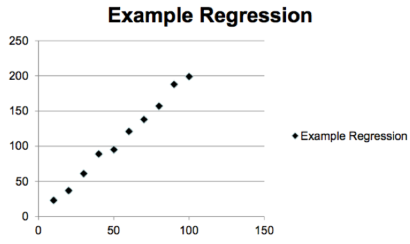
- **Algorithms**
  - **Want nearly linear time or less**
  - **Usually at the cost of a randomized approximation**

Regression analysis

- **Regression analysis**
  - **Statistical method to study dependencies between variables in the presence of noise.**

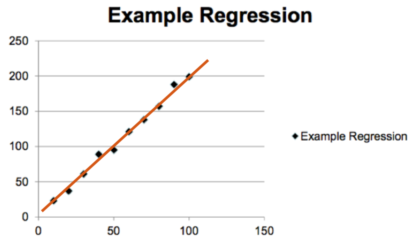
Regression analysis

- Linear Regression
  - Statistical method to study linear dependencies between variables in the presence of noise.
- Example
  - Ohm's law V = R · I



Regression analysis

- Linear Regression
  - Statistical method to study linear dependencies between variables in the presence of noise.
- Example
  - Ohm's law V = R · I
  - Find linear function that best fits the data



Regression analysis

- **Linear Regression**
  - **Statistical method to study linear dependencies between variables in the presence of noise.**

- **Standard Setting**
  - One measured variable b
  - A set of predictor variables $a_1, \ldots, a_d$
  - Assumption:

$$b = x_0 + a_1 x_1 + \ldots + a_d x_d + \varepsilon$$

  - ε is assumed to be noise and the x_i are model parameters we want to learn
  - Can assume $x_0 = 0$
  - Now consider n observations of b

Regression analysis

- **Matrix form**

Input: n×d-matrix A and a vector $b=(b_1,\ldots, b_n)$
n is the number of observations; d is the number of predictor variables

Output: x^* so that Ax^* and b are close

- Consider the over-constrained case, when $n \gg d$
- Can assume that A has full column rank

Regression analysis

- **Least Squares Method**
  - Find x* that minimizes $|Ax-b|_2^2 = \Sigma (b_i - \langle A_{i*}, x\rangle)^2$
  - A_{i*} is i-th row of A
  - Certain desirable statistical properties

Regression analysis

- **Geometry of regression**
  - We want to find an x that minimizes $|Ax-b|_2$
  - The product Ax can be written as

$$A_{*1}x_1 + A_{*2}x_2 + \ldots + A_{*d}x_d$$

where A_{*i} is the i-th column of A

  - This is a linear d-dimensional subspace
  - The problem is equivalent to computing the point of the column space of A nearest to b in l_2-norm

Time Complexity

- **Solving least squares regression via the normal equations**
  - **Need to compute x = A^{-}b**
    - **Moore-Penrose Pseudoinverse $A^{-} = V\Sigma^{-1}U^T$**
  - **Naively this takes nd^2 time**
  - **Can do $nd^{1.376}$ using fast matrix multiplication**
  - **But we want much better running time!**

Sketching to solve least squares regression

- How to find an approximate solution x to $\min_x |Ax-b|_2$?
- Goal: output x' for which $|Ax'-b|_2 \leq (1+\varepsilon) \min_x |Ax-b|_2$ with high probability
- Draw S from a k x n random family of matrices, for a value k << n
- Compute S*A and S*b
- Output the solution x' to $\min_{x'} |(SA)x-(Sb)|_2$
  - $x' = (SA)^{-}Sb$

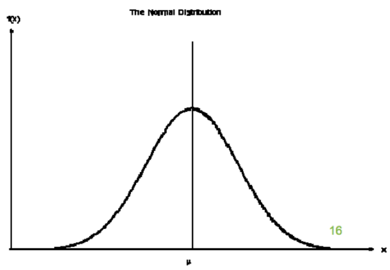
How to choose the right sketching matrix S?

- Recall: output the solution x' to $\min_{x'} |(SA)x-(Sb)|_2$
- Lots of matrices work
- S is d/ε^2 x n matrix of i.i.d. Normal random variables

- **S is a subspace embedding**

  For all x, $|SAx|_2 = (1\pm\varepsilon)|Ax|_2$

  * poof skipped



ref: David P. Woodruff, Sketching as a Tool for Numerical Linear Algebra, Foundations and Trends in Theoretical Computer Science, vol 10, issue 1-2, pp. 1-157 (ref to 10-40)

Subspace Embeddings for Regression

- Want x so that $|Ax-b|_2 \leq (1+\varepsilon) \min_y |Ay-b|_2$
- Consider subspace L spanned by columns of A together with b
- Then for all y in L, $|Sy|_2 = (1\pm \varepsilon) |y|_2$
- Hence, $|S(Ax-b)|_2 = (1\pm \varepsilon) |Ax-b|_2$ for all x
- Solve $\mathrm{argmin}_y |(SA)y - (Sb)|_2$
- Given SA, Sb, can solve in poly(d/ε) time

Only problem is computing SA takes $O(nd^2)$ *time*

Faster Subspace Embeddings S

- **CountSketch matrix**
- **Define k x n matrix S, for** $k = O(d^2/\varepsilon^2)$
- **S is really sparse: single randomly chosen non-zero entry per column**

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Can compute $S \cdot A$ in nnz(A) time! $\ll nd < nd^2$

nnz(A) is number of non-zero entries of A

High Probability and Complexity

- **Theorem 2.5.** ([27]) For $\mathbf{S}$ a sparse embedding matrix with $r = O(d^2/\varepsilon^2 \mathrm{poly}(\log(d/\varepsilon)))$ rows, for any fixed $n \times d$ matrix $\mathbf{A}$, with probability .99, $\mathbf{S}$ is a $(1 \pm \varepsilon)$ ℓ_2-subspace embedding for $\mathbf{A}$. Further, $\mathbf{S} \cdot \mathbf{A}$ can be computed in $O(\mathrm{nnz}(\mathbf{A}))$ time.
- **Theorem 2.14.** The ℓ_2-Regression Problem can be solved with probability .99 in $O(\mathrm{nnz}(A)) + \mathrm{poly}(d/\varepsilon)$ time.