Big Data Dimension Reduction using PCA

Xuelong Wang

April 23, 2018

PCA on Big Data

Big Data Issue

- The sample size is too large that it cannot be loaded into memory or even hard disk
- The number of predictors is much larger than the number of observation ($p >> n$)

Solution

For large sample size

- Tonglin Zhang (2016) introduced a method which only read data row-by-row and records the sufficient statistics vector $\mathcal{C}_0$. After scanning the data, the PCA result can be recovered by the sufficient statistics.
- The paper also used parallel computation to reduce the computation time
- HALKO (2011) also proposed a algorithm to approximate the PCA result on streaming data. The goal is to construct a low-rank approximation $U\Sigma V^T$ to any given real matrix A. More details can be found at the paper in the reference page.

Solution

For large sample p

- For the p>>n problem, Yang (2010) introduced the Stratified Population, e.g. $S_1, \ldots, S_k$ subpopulation with probabilities $\pi_1, \ldots, \pi_k$ to be selected. The author also introduced Dual eigen-analysis and models which is based on the Stratified Population.

Challenge of Big Data

1. Memory Barrier
   - The size of data is too large to load into memory
   - More specifically, n is way more large than p
2. The computation time
   - It could be very time consuming if only use single core or cluster

Solution

1. Sufficient statistics

- PCA regression only uses sufficient statistics
- Sufficient statistics can be calculated by scanning the data row-by-row

2. Parallel computation

- Multiple threads
- Map-Reduced structure

Basic idea of PCA

Singular Value Decomposition

$$X_s = UDV^T \quad \text{,where} \quad x_{ij,s} = \frac{x_{ij} - \bar{x}}{s_j}$$

$U = (u_1, \ldots, u_r)$ is a n by r orthogonal matrix

$D = diag(d_1, \ldots, d_r)$ is a r by r diagonal matrix

$V = (v_1, \ldots, v_r)$ is a p by r orthogonal matrix

Basic idea of PCA

Principle Component and Loading

$$X_s = \underbrace{\begin{bmatrix} d_1 u_1 \dots d_r u_r \end{bmatrix}}_{\text{PCs}} \underbrace{\begin{bmatrix} {v_1}^T \\ \vdots \\ {v_r}^T \end{bmatrix}}_{\text{Loading}}$$

- $PC_j = d_j \boldsymbol{u_j} = X \boldsymbol{v_j}$ is the jth principle component
- The sample variance of PC_j is d_j^2/n

Basic idea of PCA

Reduced matrix $X_{s,k}$

$$X_{s,k} = \sum_{j=1}^{k} d_j \boldsymbol{u_j v_k} = U_k D_k V_k^T, \quad \text{Its Variation} \quad \sum_{j=1}^{k} d_j^2/n.$$

Its proportion of the total variation is

$$\lambda_k = \frac{\sum_{j=1}^{k} d_j^2}{\sum_{j=1}^{r} d_j^2}$$

- If a small k such that $\lambda_k \approx 1$, we can use $U_k D_k$ in the follow up analysis

Follow-up analysis

The PCA approach is applied to a linear regression

Model

$$y = \mathbb{1}_n \alpha_s + U_k D_K \beta_{s,k} + \epsilon_{s,k},$$

Where $\epsilon_{s,k} \sim N(0, \sigma^2_{s,k} \boldsymbol{I}_n)$

LSE and their variance

$$\hat{\alpha}_s = \bar{Y}, \quad \hat{\beta}_{s,k} = D_k^{-1} U_k^T y \quad \textcolor{red}{(\textit{PCs are Orthogonal})}$$

$$\mathbb{V}(\hat{\sigma}^2_{s,k}) = [y^T \underbrace{(\mathbb{I}_n - \mathbb{J}_n/n - U_k U_k^T)}_{\mathbb{I}_n - P_k} y]/(n-k)$$

Sufficient Statistics

Factorization Theorem

$$f(x_1, x_2, ..., x_n; \theta) = \phi\left[u(x_1, ..., x_n); \theta\right] h(x_1, ..., x_n)$$

- $u(x_1, ..., x_n)$ is the sufficient statistics for θ
- If θ is a vector, then $u(x_1, ..., x_n)$, the Joint Sufficient Statistics, will be also a vector.

Sufficient Statistics

model

$$y = \mathbb{1}_n \alpha + X\beta + \epsilon, \;\; \epsilon \sim N(0, \sigma^2 \boldsymbol{I}_n)$$

Log-Likelihood function

$$\begin{aligned} \log\{f(y|x, \alpha, \beta, \sigma)\} &= \frac{n}{2}\log 2\pi - \frac{n}{2}\log \sigma^2 \\ &- \frac{1}{2\sigma^2}(c_{yy} - 2\alpha c_y + 2c_{xy}\beta + n\alpha^2 + 2\alpha c_x^T \beta + \beta^T C_{xx}\beta) \end{aligned}$$

- Define: $\mathcal{C}(y, X) = (c_0, c_{yy}, c_y, \boldsymbol{c}_{xy}, \boldsymbol{c}_x, C_{xx})$
- Note that $\ell(\alpha, \beta, \sigma)$ only depends on $\mathcal{C}(y, X)$

More about $\mathcal{C}(y, X)$

Elements of $\mathcal{C}(y, X)$

$$c_0 = n, \; c_{yy} = \sum_{i=1}^{n} Y_i^2, \; c_y = \sum_{i=1}^{n} Y_i,$$
$$c_{xy} = \sum_{i=1}^{n} Y_i x_i, \; c_x = \sum_{i=1}^{n} x_i, \; C_{xx} = \sum_{i=1}^{n} x_i x_i^T$$

- Notice that $\mathcal{C}(y, X)$ is the Joint sufficient statistics for (α, β, σ)
- All terms are in the summation format, so it can be calculated by reading the data row-by-row

Computation of $\mathcal{C}(y, X)$

Algorithm 1 Computation of $\mathcal{C}(\mathbf{y}, \mathbf{X})$ Based on A Single Processor

---

Input: row-by-row of the data
Output: $\mathcal{C}(\mathbf{y}, \mathbf{X})$

1: **procedure** ALGORITHM FOR $(c_0, c_{yy}, c_y, \mathbf{c}_{xy}, \mathbf{c}_x, \mathbf{C}_{xx})$
2: Let c_0, c_{yy}, c_y, $\mathbf{c}_{xy}$, $\mathbf{c}_x$, and $\mathbf{C}_{xx}$ be values, vectors, and matrix, respectively, all equal to zero
3: **for** the ith row of the data **do** update $c_0 = c_0 + 1$, $c_{yy} = s_{yy} + Y_i^2$, $c_y = c_y + Y_i$, $\mathbf{c}_{xy} = \mathbf{c}_{xy} + Y_i\mathbf{x}_i$, $\mathbf{c}_x = \mathbf{c}_x + \mathbf{x}_i$, and $\mathbf{C}_{xx} = \mathbf{C}_{xx} + \mathbf{x}_i\mathbf{x}_i^T$ until the last row is scanned
4: **end for**
5: Output
6: **end procedure**

- $\mathcal{O}((p+1)^2)$ memory size
- $\mathcal{O}(n(p+1)^2)$ floating operations

Sufficient Statistics based on X_s

Statistics affected by standardization

$$C_{s,xx} = X_s^T X_s, \quad c_{s,xy} = X_s^T y, \quad c_{s,x} = \mathbb{1}_n^T X_s / n$$

- It can be proved that, those statistics can be computed directly from $\mathcal{C}(y, X)$

Key step

$$c_{s,k,xy} = V_k^T c_{s,xy} = V_k^T V D U^T y = D_k U_k^T y \;\Rightarrow\; U_k^T y = D_k^{-1} c_{s,k,xy}$$

Back to PCA regression

$$X_s = UDV^T$$

Since n is large, it's not available to calculate U. However, we can only use V and D to get the estimated coefficients and variance.

PCA regression based on sufficient statistics

1. D, V can be calculated by $C_{s,xx} = X_s^T X_s = VD^2V^T$
2. $U_k^T y = D_k^{-1} c_{s,k,xy}$, so $\hat{\beta}_{s,k} = D_k^{-2} c_{s,k,xy}$
3. $\mathbb{V}(\hat{\sigma}^2_{s,k}) = [c_{yy} - c_{yy}/n - c^T_{s,k,xy} D_k^{-2} c_{s,k,xy}]/(n-k)$

Parallel Computation with Distributed Systems

Algorithm 2 Computation of $C(\mathbf{y}, \mathbf{X})$ in MapReduce

Input: row-by-row of individual sub-data sets
Output: $C(\mathbf{y}, \mathbf{X})$

1: **procedure** PARALLEL COMPUTATION BASED ON (17)
2: **Map tasks:** compute $C(\mathbf{y}_i, \mathbf{X}_i)$ using Step 3 of Algorithm 1 for $i = 1, \cdots, K$ individually
3: **Reduce task:** let $C(\mathbf{y}, \mathbf{X}) = \sum_{i=1}^{K} C(\mathbf{y}_i, \mathbf{X}_i)$
4: Output
5: **end procedure**

Thank you

Reference

HALKO, NATHAN. 2011. "AN Algorithm for the Principal Component Analysis of Large Data Sets."

Tonglin Zhang, Baijian Yang. 2016. "Big Data Dimension Reduction Using Pca."

Yang, Fan. 2010. "Principal Component Analysis (Pca) for High Dimensional."