Lecture 3 Report

EnHsien Chou
zex18@mails.tsinghua.edu.cn

Abstract

Reading Notes for Lecture 4, "Clustering and Dimension Reduction"

1 Introduction

Last two lectures, we mainly focused on SVM and logistic regression, which belongs to algorithms of "supervised learning". This lecture we then focused on Clustering, which is an algorithm of unsupervised learning (no label of input data set). Then, we talked about the idea of dimension reduction.

2 K-means algorithm

We want to divide N items $(x_1, x_2, ..., x_N)$ into k groups $(M_1, M_2, ..., M_k)$. Therefore, we can enumerate every situation : let r_{ij} be the index of whether x_i belongs to cluster M_j

$$\forall 1 \leq i \leq N, \forall 1 \leq j \leq k \quad r_{ij} = \begin{cases} 1, & if \quad x_i \in M_j \\ 0, & if \quad x_i \notin M_j \end{cases}$$

The average of every element in the cluster is its center:

$$\mu_j = \frac{1}{|M_j|} \sum_{i=1}^{N} r_{ij} \cdot x_i$$

The question is to minimize :

$$\sum_{i=1}^{N} \sum_{j=1}^{k} r_{ij} \cdot |x_i - \mu_j|^2$$

Since μ_j and r_{ij} are not independent, there is no explicit solution to the optimization goal. However, we can fix one variant each time and optimize the other until the solution coverges, that is:

Loop until convergence:

- Keep μ fixed. $r_{ij} = 1, \quad$ if $|x_i - \mu_j| < |x_i - \mu_k|, \quad \forall j \neq k$
- Keep r fixed. Find $\mu_j, \quad \forall 1 \leq j \leq k$

The convergence of K-means algorithm is guaranteed (Note : How to prove it?).

2.1 Generalization

Actually, we can define inner-product to depict the "similarity" of two elements. In this case, it is hard to compute the center of each cluster. We instead choose μ from elements to symbolize the center of each cluster. (Note: I think it is similar to the method of kernel function. The map is not important as long as we can define a suitable inner product).

3 Expectation-Maximization algorithm

There are some problems of K-means algorithm: First, elements are "discrete points" on a vector space, but in real cases, it should follow some kind of distribution law on the space. Second, in unsupervised learning, we don't actually know what cluster the element really belongs into. Each element is of some "probability" to be divided into each cluster.

First, we use the mixture of k gaussian distribution. And x can be generate by the mixture of the k models:

$$p(x) = \sum_{i=1}^{k} \pi_k N(\mathbf{x}|\mu_k, \Sigma_k), \quad \sum_{i=1}^{k} \pi_k = 1$$

Then, we define "latent variable" z meaning how much percentage each model generates x.

By MLE, we can find a distribution of the latent variable z:

$$Q_i(z^{(i)}) := P(z^{(i)}|x^{(i)}\theta))$$

$$\sum_z Q_i(z) = 1, \quad 0 \leq Q_i(z) \leq 1$$

After some math (Jenson inequality), our problems can be transformed into:

$$arg\max_{\theta} \sum_{i=1}^{n} \sum_{z^{(i)}} Q_i(z^{(i)}) logP(x^{(i)} z^{(i)}|\theta)$$

Similary, Q and θ are not independent. We fix one and optimize the other until convergence:

- E steps—Fix z, compute $Q_i(z^{(i)})$
- M steps—Fix Q, use MLE find θ

4 Dimension Reduction

4.1 Principle Component Analysis

PCA is a method of dimension reduction by projection vectors to a space from higher dimension D to lower dimension M. A better projection is the one that the variance of the projection is better, or that the distance between the data and the projection is smaller.

Our goal is to maximize sample covariance:

$$S = \frac{1}{N} \sum_i (x_i - \bar{x})(x_i - \bar{x})^T$$

This can be done by choosing the largest M eigenvalues.

Simarly, we can also minimize error formulation. This can be done by choosing the smallest D-M eigenvalues, which indicates the two optimizations are equivalent.

4.2 Probabilistic Principle Component Analysis

We define the latent variable z to represent the probability in PPCA.

In what case do we use PPCA instead of PCA? Since we know the complexity of finding eigenvector is $O(n^3)$, PPCA is better when facing high dimension. Also, when data of some dimension is missing, PPCA can represent the "uncertainty" of the data. That is, PPCA has a better ability to "guess" than PCA.

4.3 Other PCA

(Note: This part is not yet fully understand and should review again later)

There are many other kinds of PCA:

- Maximum likelihood PCA
- Bayesian PCA
- Kernel PCA

And we can also apply EM algorithm to PCA.

Acknowledgments

Thanks for the speecher, Yin Haoxuan, and every one in our reading group. Additionally, thanks for all the learning materials provided from our leading teacher, Pro. Su Hang.

References

[1] Murphy, K. P. (2012). Machine learning: a probabilistic perspective. MIT press.