[PPCI: an R package for Cluster Identification using Projection Pursuit 요약]

2022.05.18. 예지혜

초록

PPCI R 패키지는 최근 제안된 '클러스터링을 위한 Projection Pursuit' 방법론 3가지를 구현한 것이다. 이 방법론들은 클러스터를 분리하기 위한 최적의 hyperplane을 정의한다. 또한, divisive hierarchical clustering을 통해 반복적으로 클러스터링을 진행한다.

Introduction

노이즈가 많거나 차원이 큰 데이터에선 공간적인 구조를 보는 것이 별로 도움이 되지 않는다. 이런 경우 클러스터를 분리하기 좋은 subspace를 찾아야 하고, 하나의 subspace로는 모든 클러스터를 분리해내긴 어려워 여러 개를 찾아야 한다. 클러스터링을 위한 좋은 subspace를 찾는 방법으로 projection pursuit을 이용할 수 있다.

클러스터링과 PP를 결합시킨 패키지로는 ProjectionBasedClustering이 있다. 이는 ICA, t-SNE 등의 차원축소 방법을 제공하지만, 이 방법들은 클러스터링 기준을 차원축소 공식에 직접적으로 적용하지는 않는다. 그 결과 저차원의 임베딩이 클러스터 구조를 띤다는 보장이 없다. 또한, 방법에 따라 매우 다른 투영을 만들어 어떤 결과가 좋은 결과인지 이해하기 어렵다는 문제가 있다. subspace라는 패키지도 있으나, 최적의 subspace를 찾아내지는 않는다.

이 논문은 PPCI 패키지를 소개하며, 클러스터링을 위한 세 가지 PP 방법을 제공한다. 이 투영 인덱스들은 k-means와 같은 density clustering과 clustering by normalised graph cuts를 기반으로 한다. 전반적인 클러스터링 모델은 divisive hierarchical 구조를 띠며, 반복적으로 하나 이상의 군집을 분리해낸다.

Projection pursuit, hyperplanes and divisive hierarchical clustering

이 파트에서는 데이터를 둘로 나누는 하이퍼플레인을 찾고, 거기서 투영 지수를 도출해내는 것을 설명한다. 또한, 각 하이퍼플레인을 divisive 계층적 클러스터링을 위해 어떻게 결합하는지 설명한다.

Finding optimal hyperplanes via projection pursuit

$\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$: d차원의 n개 데이터

$\mathbf{v} \in \mathbb{B}^d = \left\{\mathbf{x} \in \mathbb{R}^d \,|\, \|\mathbf{x}\|_2 = 1\right\}$: unit-length vector

$H(\mathbf{v}, b) = \left\{\mathbf{x} \in \mathbb{R}^d \,|\, \mathbf{v}^\top \mathbf{x} = b\right\}$: d차원에서의 하이퍼플레인 (벡터 v에 투영했을 때 b에 해당하는 데이터들의 집합)

즉, 하이퍼플레인 H에 의해 데이터 X는 두 그룹으로 나뉘어진다.

$$\mathcal{X} = \mathcal{X}^+_{\mathbf{v},b} \cup \mathcal{X}^-_{\mathbf{v},b},$$
$$\mathcal{X}^+_{\mathbf{v},b} := \{\mathbf{x} \in \mathcal{X} | \mathbf{v}^\top \mathbf{x} \geqslant b\},$$
$$\mathcal{X}^-_{\mathbf{v},b} := \{\mathbf{x} \in \mathcal{X} | \mathbf{v}^\top \mathbf{x} < b\}.$$

이렇게 하이퍼플레인을 사용하게 되면, 데이터를 벡터로 투영하면서 매우 큰 차원 축소의 효과를 얻을 수 있고, 1차원에서의 클러스터링 또한 다루기 쉬운 문제가 된다.

Q는 이진 분류의 quality measure라 정의한다. 이때 quality란, 높은 clusterability, 또는 두 그룹 간 낮은 connectedness로 정의할 수 있다.

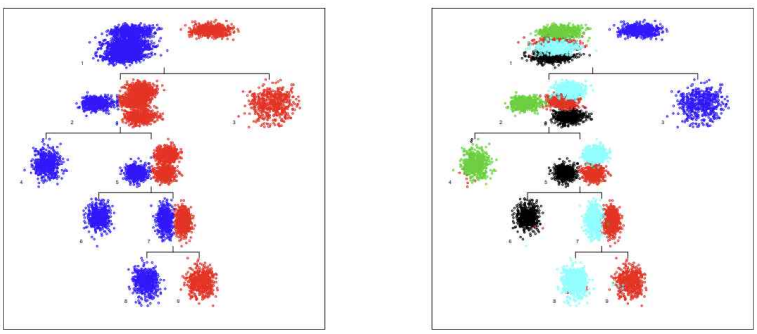
$$\phi(\mathbf{v}, b|\mathcal{X}) = Q(\mathbf{v}^\top \mathcal{X}^-_{\mathbf{v},b}, \mathbf{v}^\top \mathcal{X}^+_{\mathbf{v},b})$$

이 경우, v와 b 두 파라미터에 대해 최적화해야 하기 때문에, v가 고정된 상태에서 최적의 b만 유지하는 방식으로 문제를 완화시킨다. 최적화 방법으로는 BFGS를 사용하였다.

$$\mathbf{v}_{\text{opt}} = \arg\max_{\mathbf{v} \in \mathbb{B}^d} \Phi(\mathbf{v}|\mathcal{X}),$$
$$\Phi(\mathbf{v}|\mathcal{X}) = \max_{b \in \mathbb{R}} \phi(\mathbf{v}, b|\mathcal{X}).$$

Divisive hierarchical clustering using hyperplanes

2개의 군집으로 분리하는 하이퍼플레인을 반복적으로 찾아 이진 트리 구조의 군집화를 진행한다. 군집의 개수만 결정된다면, cluster quality measure에 해당하는 Q에 따라 군집화를 진행하기만 하면 된다. 여러 노드 중 가장 큰 군집을 군집화하기도 하고, 가장 clusterability가 큰 군집을 군집화하기도 한다.



(a) Without cluster labels　　(b) With true cluster labels

군집화 결과 예시를 보자. 각 노드의 산점도에서 가로축은 최적의 투영 벡터, 세로축은 v에 직교하면서 가장 분산이 큰 방향이다. 정답에 해당하는 라벨이 있다면 인풋으로 넣어 색깔로 결과를 확인할 수도 있다.

Clustering and Projection Pursuit in PPCI

PPCI에서는 (mdh, mddc), (mch, mcdc), (ncuth, ncutdc) 세 쌍의 방법을 제공한다. (PP, clustering) 조합으로, PP가 최적의 투영을 찾으면 clustering이 divisive 계층적 클러스터링을 진행한다.

Minimum Density Hyperplanes (mdh and mddc)

밀도 클러스터링에서는 데이터를 unknown 분포들의 샘플로 가정한다. probability density가 낮은 곳이 곧 각 클러스터를 분리하는 경계로 이해할 수 있고 이를 low density separation 가정이라 한다.

MDH는 추정된 분포 $\hat{p}$을 기반으로, 분포가 가장 작은 하이퍼플레인을 추정한다. 하이퍼플레인 $H(v,b)$에 대해 추정된 분포 $I(\widehat{v,b})$를 surface integral로 정의하여, 이를 최소화하는 H를 찾는 것이다. 다만, b가 $\pm\infty$로 가면 $I(\widehat{v,b})$는 0으로 수렴하기 때문에 penalty term을 사용해야 한다.

$$\hat{p}(\mathbf{x}) = \frac{1}{n(2\pi h^2)^{d/2}} \sum_{i=1}^{n} e^{-\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{2h^2}} \Rightarrow \hat{I}(\mathbf{v},b) := \int_{H(\mathbf{v},b)} \hat{p}(\mathbf{x}) d\mathbf{x} = \frac{1}{n\sqrt{2\pi h^2}} \sum_{i=1}^{n} e^{-\frac{(b-\mathbf{v}^\top \mathbf{x}_i)^2}{2h^2}}$$

$$\min_{\mathbf{v}} \Phi(\mathbf{v}|\mathcal{X}) = \min_{b\in\mathbb{R}} \phi(\mathbf{v},b|\mathcal{X}),$$

$$\phi(\mathbf{v},b|\mathcal{X}) = \hat{I}(\mathbf{v},b) + C\max\{0, -\alpha\sigma_{\mathbf{v}} - b, b - \alpha\sigma_{\mathbf{v}}\}^{1+\epsilon}$$

Maximum clusterability clustering (mch and mcdc)

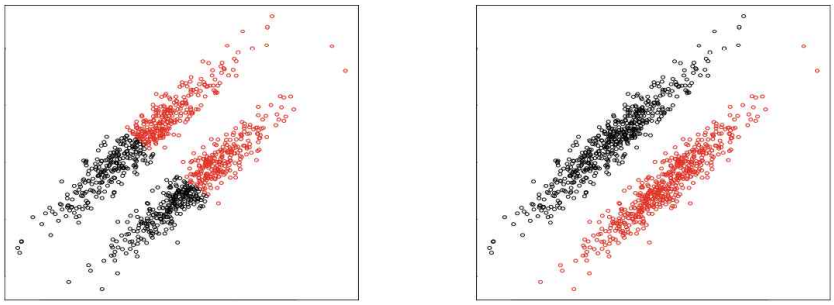
mch는 clusterability가 최대화되는 클러스터를 찾는 방법으로, clusterability로 variance ratio를 정의한다. 이는 클러스터 내의 분산 대비 클러스터 간 분산의 비율로 LDA 인덱스와 유사하다. 또한, kmeans 방법론과 유사하다. 이는 NP-hard로 알려져있지만 1차원 세팅에서는 다루기 쉽다.

$$\max_{\substack{\mathcal{X}=\mathcal{C}_1\cup\cdots\cup\mathcal{C}_k \\ \mathcal{C}_i\cap\mathcal{C}_j=\emptyset, i\neq j}} \frac{\sum_{i=1}^{k} |\mathcal{C}_i| \, \|\boldsymbol{\mu}_{\mathcal{C}_i} - \boldsymbol{\mu}\|^2}{\sum_{i=1}^{k} \sum_{j:\mathbf{x}_j\in\mathcal{C}_i} \|\mathbf{x}_j - \boldsymbol{\mu}_{\mathcal{C}_i}\|^2},$$

$$\Phi(\mathbf{v}|\mathcal{X}) = \max_{b} \phi(\mathbf{v},b|\mathcal{X}),$$

$$\phi(\mathbf{v},b|\mathcal{X}) := \frac{\left|\mathcal{X}^+_{\mathbf{v},b}\right| \left(\mu_{\mathbf{v}^\top\mathcal{X}^+_{\mathbf{v},b}} - \mu_{\mathbf{v}^\top\mathcal{X}}\right)^2 + \left|\mathcal{X}^-_{\mathbf{v},b}\right| \left(\mu_{\mathbf{v}^\top\mathcal{X}^-_{\mathbf{v},b}} - \mu_{\mathbf{v}^\top\mathcal{X}}\right)^2}{\frac{n}{n-1}\sum_{i=1}^{n} \left(\mathbf{v}^\top\mathbf{x}_i - \mu_{\mathbf{v}^\top\mathcal{X}}\right)^2 + \sum_{\mathbf{x}_i\in\mathcal{X}^+_{\mathbf{v},b}} \left(\mathbf{v}^\top\mathbf{x}_i - \mu_{\mathbf{v}^\top\mathcal{X}^+_{\mathbf{v},b}}\right)^2 + \sum_{\mathbf{x}_j\in\mathcal{X}^-_{\mathbf{v},b}} \left(\mathbf{v}^\top\mathbf{x}_j - \mu_{\mathbf{v}^\top\mathcal{X}^-_{\mathbf{v},b}}\right)^2}.$$

다만, 데이터의 분산이 너무 큰 경우 이 방법으로 클러스터가 적절히 나뉘지 않을 수 있어, MCDC 알고리즘은 1차원으로 투영된 데이터의 variance ratio를 최대화하도록 클러스터링을 반복적으로 시도한다.



(a) k-means solution (b) MCDC solution

이처럼 within 분산이 between 분산에 비해 너무 클 때 k-means는 클러스터를 잘 잡아내지 못하나, MCDC는 잘 잡아낸다. 주로 가늘고 긴 데이터에서 이러한 현상이 나타난다.

Minimum normalised cut hyperplanes (ncuth and ncutdc)

graph partitioning 알고리즘이 뜨면서 normalised cut objective가 클러스터링에 빈번히 사용되기 시작했다. 서로 다른 클러스터 간 유사도에 해당하는 Cut은 최소화하고, 같은 클러스터 간 유사도에 해당하는 Volume은 최소화하는 방식이다.

$$\mathrm{NCut}(\mathcal{C}_1,\ldots,\mathcal{C}_k) = \sum_{m=1}^{k} \frac{\mathrm{Cut}(\mathcal{C}_m, \mathcal{X}\setminus\mathcal{C}_m)}{\mathrm{volume}(\mathcal{C}_m)},$$

$$\mathrm{Cut}(\mathcal{C}, \mathcal{X}\setminus\mathcal{C}) := \sum_{\substack{i,j:\mathbf{x}_i\in\mathcal{C},\\ \mathbf{x}_j\notin\mathcal{C}}} \mathrm{similarity}(\mathbf{x}_i,\mathbf{x}_j), \quad \mathrm{volume}(\mathcal{C}) := \sum_{\substack{i,j:\mathbf{x}_i\in\mathcal{C}\\ \mathbf{x}_j\in\mathcal{X}}} \mathrm{similarity}(\mathbf{x}_i,\mathbf{x}_j).$$

$$\Phi(\mathbf{v}|\mathcal{X}) = \min_b \phi(\mathbf{v}, b|\mathcal{X})$$

$$\phi(\mathbf{v}, b|\mathcal{X}) := \mathrm{NCut}(\mathbf{v}^\top\mathcal{X}^+_{\mathbf{v},b}, \mathbf{v}^\top\mathcal{X}^-_{\mathbf{v},b}).$$

이는 NP-hard이고, 로컬 최적화로는 좋은 퀄리티를 보장할 수 없어 spectral clustering을 사용한다. 다만 이 방법은 매우 큰 데이터에서는 잘 작동하지 않는다는 단점이 있다. 이 방법은 pairwise similarities를 어떻게 정의할 것인가가 중요한데, 라플라스 커널을 예시로 들 수 있다.

Modifying and validating a clustering solution

클러스터링 모델의 수정은 노드를 합치는 pruning, 노드를 더 분리하는 extending이 있다. 트리의 중간에 위치한 노드도 수정할 수 있다. 최종 목표는 동일 군집이 여러 개로 분리되지 않고, 다른 군집이 하나로 합쳐지지 않은 상태로 클러스터링하는 것이다.

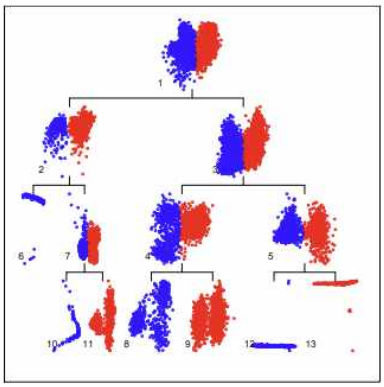
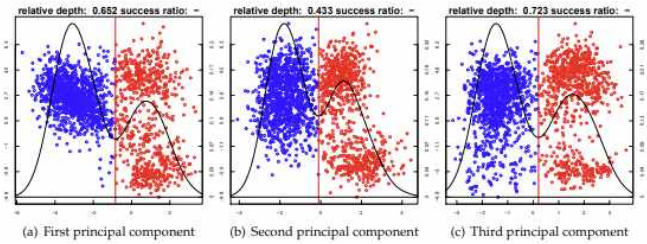


Figure 5: Initial clustering of optidigits data set through MDDC with 7 clusters.



(a) First principal component (b) Second principal component (c) Third principal component

이 예시의 경우 4번 노드가 잘 분리되지 않은 것으로 보여, 이 노드만 첫 번째 PC가 아닌 세 번째 PC로 분리하도록 옵션을 바꿀 수 있다. 세 번재 PC를 사용하도록 함수를 정의해 4번 노드를 클러스터링하는 인풋으로 넣어주면 된다.

Extensions

Maximum margin hyperplanes for clustering

Maximum margin clustering, 즉 MMC는 SVM처럼 마진을 최대화하는 방법론인데, 비지도 학습이라 정수 최적화 문제가 있고, 매우 작은 데이터에서만 가능하다. 그러나 MDH, MNCH의 bandwidth 파라미터를 0으로 설정하면 결국 MMC와 같아지기 때문에 두 방법을 통해 MMC를 사용할 수 있다. 예시에서는 bandwidth에 계속 0.9를 곱하면서 mdh를 반복적으로 적용하였다.

Non-linear separators using Kernel PCA

일반적으로 하이퍼플레인을 사용하면 클러스터가 잘 분리되지만, 저차원의 비선형 형태의 데이터는 분리하기 어렵다. 이 경우, KPCA를 데이터에 적용해 분리가 잘 되는 고차원 space로 피쳐를 임베딩하고, 다시 이 논문에서 제안하는 방법들을 사용하여 하이퍼플레인으로 클러스터링할 수 있다.