Unsupervised Learning

Unsupervised Learning

Features of Unsupervised Learning

- Y에는 no longer interested; just focus on information or structure in $X_1, \ldots, X_p$

Two main methods

- PCA (Principal Components Analysis)
- Clustering

Challenges and Advantages

- Challenges
  - Subjective하고, 잘 됐는지 잘 되지 않았는지에 관한 객관적인 판단이 어렵다.
  - 또한 그 작업 자체의 수행이 어려울 수 있다.
- Advantages
  - Label 자체가 필요하지 않기 때문에 labeling에 들어가는 시간을 줄일 수 있고, 그렇기 때문에 더 다양한 데이터를 사용할 수 있다는 장점이 있다.

PCA (Principal Components Analysis)

- p-dimensional ($p > 2$) 데이터를 보다 lower-dimension으로 줄여서 볼 수 있게 해주는 method
  - 다만 dimension을 줄이면서도 그 안에 들어있는 정보를 maintain하는 것이 optimal
  - 따라서 data의 variance의 크기를 기준으로 principal component를 생성하며, 가장 큰 variance를 보이는 방향을 first principal component, 그리고 방향에 orthogonal한 방향을 second principal component라고 한다.

How to implement PCA

- First PC: 주어진 데이터의 variance를 극대화하는 방향

- How? X들의 linear combination $Z_1 = \phi_{11}X_1 + \phi_{21}X_2 + \cdots + \phi_{p1}X_p$이 최대한의 variance를 유지하도록 $\phi_{11}, \ldots, \phi_{p1}$을 조정
  - 다만 ϕ가 크면 클수록 variance가 높기 때문에 모든 ϕ에 대해서 $\sum_{j=1}^{p} \phi_{j1}^2 = 1$이 되도록 normalize
- 그리고 이러한 ϕ들을 principal component의 loading이라고 하며, ϕ_{ji}에 대해서 j는 feature number $(X_1, \ldots, X_p)$ so $j \in \{1, \ldots, p\}$를, i는 몇 번째 principal component를 가리키는지를 나타냄
  - 이에 따라서 $\phi_1 = [\phi_{11}, \ldots, \phi_{p1}]^\top$를 first principal component에 대한 loading vector라고 한다.
- loading을 normalize하는 이유는 어떤 방향으로 봤을 때 분산이 가장 높은지만을 찾기 위함 → loading 값이 arbitrarily 커지면 전체 first PC의 분산 역시 커지기 때문
  - 여기서 “방향"이란 feature space 상에서의 방향; 즉 전체 데이터 분산을 가장 크게 보존하는

- Detailed
  - 전체 $n \times p$ dataset을 $\boldsymbol{X}$라고 했을 때, first PC의 i번째 observation은 다음과 같이 나타내어진다.

$$z_{i1} = \phi_{11}x_{i1} + \cdots + \phi_{p1}x_{ip}$$

    - where ϕ_{j1} denotes the loading of the first PC for the jth feature and x_{ij} denotes the ith observation of the jth feature
    - 그리고 여기서 모든 $X_1, \ldots, X_p$는 centered to have the mean zero
    - we want the loadings with the largest sample variance subject to the constraint that the squared sum of the loadings equals to 1
  - 이때 각각의 feature들은 모두 zero mean을 가지기 때문에, $z_{ij}\ \text{where}\ j \in \{1, \ldots, j\}$ 역시 zero mean을 가지게 된다.

$$\sum_{i=1}^{n} z_{i1}$$
$$= \phi_{11} \sum_{i=1}^{n} x_{i1} + \cdots + \phi_{p1} \sum_{i=1}^{n} x_{ip}$$
$$= \sum_{i=1}^{n} \sum_{j=1}^{p} \phi_{j1} x_{ij} = 0$$

- 결국 최종 goal은 $z_{11}, \ldots, z_{n1}$의 sample variance를 가장 크게 만드는 것!
  ← 그리고 z_{i1}는 **ith observation의 first PC에 대한 score**라고 부름

$$\max_{\phi_{11}, \ldots, \phi_{p1}} \quad \frac{1}{n} \sum_{i=1}^{n} z_{i1}^2$$
$$\text{subject to} \quad \sum_{j=1}^{p} \phi_{j1}^2 = 1$$
$$\text{where} \quad z_{i1} = \sum_{j=1}^{p} \phi_{j1} x_{ij}$$

- This is equivalent to the find the direction such that maximizes the variance of the entire data
- This problem can be solved using a singular-value decomposition of the matrix $\boldsymbol{X}$
  - By the singular value theorem, SVD can be applied on any matrix!
- Geometrically, data를 projection했을 때 분산이 가장 크게 나타나는 방향이 **first PC의 loading vector**가 가리키는 방향이며, **data가 projection된 결과 각각이 ith observation의 first PC에 대한 score!**
  - In other words, PCA illustration에 나타난 각각의 point(coordinate)는 score, 그리고 vectors with direction은 loading vector를 나타냄
  - Loading vector의 경우 first PC 방향으로의 길이(또는 기울기)가 큰 벡터일수록 first PC와 관련이 있고, 반대로 second PC 방향으로의 길이(또는 기울기)가 큰 벡터일수록 seoncd PC와 관련이 있다고 할 수 있다.
  - 즉 2차원 PCA illustration에서 각각의 loading vector는 $(\phi_{1j}, \phi_{2j}) \text{ where } j = 1, \ldots, p$를 나타내며, 변수가 p개이면 2차원 벡

터가 총 p개가 그려진다.

- What about other PCs?
  - 똑같은 방법으로 해당 PC에 대한 loading vector와 feature들 사이의 linear combination으로 만들어내며, loading vector에 가해지는 constraint 역시 동일하다.
    - 즉, second PC는 다음과 같이 나타내어진다.

$$z_{i2} = \phi_{12}x_{i1} + \phi_{22}x_{i2} + \cdots + \phi_{p2}x_{ip}$$
$$\text{where} \sum_{j=1}^{p} \phi_{j2}^2 = 1$$

    - 다만 이때 second PC에는 하나의 제약이 추가: Z_2 **should be uncorrelated with** Z_1
    - **<u>이 조건에 따르면 결국 Z_2의 loading vector ϕ_2는 Z_1의 loading vector ϕ_1에 orthogonal한 방향으로 도출됨!</u>**
  - 그런데 전체 data의 dimension이 커질수록 ϕ_1에 직교하는 벡터의 수도 늘어남 → 따라서 그 중에서 가장 variance를 높게 만드는 loading vector를 ϕ_2, 그에 따라서 만들어지는 linear combination을 Z_2라고 한다.
  - 나머지 $\phi_3, \phi_4, \ldots$ 역시 이와 같은 방식으로 만들어질 수 있다.
  - 전체 loading vector와 그에 따른 principal component는 최대 $\min\{n-1, p\}$개까지 만들어질 수 있음

Properties

- First PC의 loading vector는 average squared Euclidean distance를 기준으로 각각의 observation에 가장 가까운 line이 가리키는 방향과 일치
  - This seems to be similar to the linear regression fitted line, but actually is different!
    - Linear regression에서는 Y 방향으로의 sum of square를 가장 줄이고 싶어 함
    - first PC loading vector는 개별 feature를 동등하게 평가하여 그들에 대해 직교되는 방향이 가장 작은 것을 찾고 싶어 함
- PCA를 implement하기 전에는 반드시 **scaling을 수행해야 함**!

- 대체로 standard normalization, 즉 centering & unit standard deviation을 수행하는 것이 필요함

Proportion Variance Explained

- 개별 principal component의 strength를 측정하는 하나의 도구로서, 각각의 PC가 data에 존재하는 variance를 얼마나 설명하는지, 얼마나 represent하는지를 측정
- m번째 principal component의 VE는 다음에 의해서 계산되며

$$Var\left(Z_m\right) = \frac{1}{n}\sum_{i=1}^{n} z_{im}^2$$

- 각 variable의 variance 합(→전체 data variable의 variance)은 다시 다음과 같이 계산된다.

$$\sum_{j=1}^{p} Var\left(X_j\right) = \sum_{j=1}^{p} \frac{1}{n}\sum_{i=1}^{n} x_{ij}^2$$

- 따라서 Z_m의 PVE는 다시 다음과 같이 계산되며

$$\frac{Var\left(Z_m\right)}{\sum_{j=1}^{p} Var\left(X_j\right)} = \frac{\sum_{i=1}^{n} z_{im}^2}{\sum_{j=1}^{p} \frac{1}{n}\sum_{i=1}^{n} x_{ij}^2}$$

  - 개별 PC가 설명하는 variance는 전체 dataset의 variance보다 클 수 없으므로 0에서 1 사이의 값을 가지며, 일반적으로 개별 PC의 PVE를 누적해서 나타내기도 한다.
    - 그리고 전체 PC에 대해서 개별 PVE의 합은 다시 1 (∵ 각각의 PC는 서로에 대해서 uncorrelated이기 때문)
  - 누적해서 나타내는 경우 Z_i의 PVE는 자기 자신의 CPVE에서 Z_{i-1}의 CPVE를 뺀 값

How many PCs should we use?

- PCA 자체에 cross-validation을 사용하는 것은 적절하지 않음
- 다만 PVE를 기준으로 CPVE plot을 역으로 그려볼 수 있음(scree plot)
  - Elbow(가장 크게 감소하는 부분)를 기준으로 끊을 수 있음

Clustering

- Refers to a very broad set of techniques for finding subgroups, or clusters
- PCA vs Clustering
  - PCA: low-dimensional representation of the observations that explains a good fraction of the variance
  - Clustering: homogeneous subgroups among the observations
- Methods
  - K-means clustering: 미리 정해둔 K개만큼의 cluster로 나누는 방법
  - Hierarchical clustering

K-means Clustering

- Pre-defined된 K에 의해서 구분되는 cluster를 각각 $C_1, C_2, \dots, C_K$라고 하면, 이들은 다음의 properties를 갖는다.
  - $\bigcup_{i=1}^{n} C_i = \boldsymbol{X}$
  - $C_i \cap C_j = \emptyset \ \forall i \neq j$
  - In other words, the clusters are partition of the given dataset.

How to implement K-means clustering

- Within-cluster Variation(WCV)라는 값을 as small as possible하게 만드는 것
- WCV는 개별 cluster C_i마다 측정되는 것이며, **개별 cluster의 center로부터 cluster에 속한 data point들이 평균적으로 떨어진 거리**를 의미함
- 따라서 다음의 optimization 문제를 푸는 것과 같음

$$\min_{C_1,\dots,C_K} \sum_{i=1}^{K} WCV\left(C_i\right)$$

  - 이때 $\sum_{i=1}^{K} WCV\left(C_i\right)$를 가장 작게 만드는 $\arg\min_K$는 $K = n$, 즉 전체 dataset의 크기임
  - where WCV is defined as $\frac{1}{|C_i|} \sum_{k,k' \in C_i} \sum_{j=1}^{p} \left(x_{kj} - x_{k'j}\right)^2$
- K-means clustering algorithm

1. Randomly assign a number from 1 to K to each of the observations
2. Iterate until the cluster assignments stop changing:
    a. 각각의 cluster에 대한 centroid를 구함
        - centroid: 해당 cluster에 속한 data point에 대한 feature mean
    b. centroid를 기준으로 가장 가까이에 있는 데이터들을 re-assign

Properties

- 이 방법을 단순히 반복하기만 하더라도 guaranteed that WCV decreases
    - since the given objective function, WCV, is equivalent to **the distance between each data point and the centroid**
- 하지만 단순히 이러한 방법만으로 global minimum에 도달할 수 있다는 보장은 없음
    - since this objective is not convex
    - 따라서 initial value를 어떻게 설정하는가에 따라서 그 결과가 제각각으로 달라질 수 있으며, 보통 practice에서는 몇 번을 반복한다.

Hierarchical Clustering

- K-means와 달리 몇 개의 cluster로 나눌 것인지 미리 결정하지 않아도 됨
- Agglomerative approach → bottom-up으로 보면서 어디에서 cluster를 나눌지를 결정함
- 서로 비슷한(가까이에 있는) data point끼리 합치는 작업을 반복 until every cluster is merged to one
- cut-off depth에 따라서 나눠지는 cluster의 수가 달라진다.

Algorithm

1. n개의 observation으로부터 시작해 모든 pairwise에 대한 dissimilarity를 측정한다. 다만 initial step에서는 각각의 observation을 개별 cluster로 regard한다.
    - 이때 사용하는 metric으로는 각각의 거리 등을 사용할 수 있다.
2. For $i = n, n-1, \dots, 2$:
    - 모든 pairwise inter-cluster dissimilarities를 측정한 뒤 가장 가까운 pair끼리 뭉치며, 이렇게 뭉쳐진 pair는 cumulative하게 하나의 집단으로 간주됨
    - 이러한 집단을 기준으로 다시 pairwise inter-cluster dissimilarities를 계산

- 다만 inter-cluster dissimilarity를 계산할 때, new point에 대해서 cluster에 포함된 data 가운데 가장 먼 거리를 기준으로 판단할지, 가장 가까운 거리를 기준으로 판단할지, 또는 centroid를 기준으로 판단할지는 pre-define할 수 있는 문제
    - 이러한 판단 기준을 linkage라고 하며, 총 3가지의 방법이 있음
        - single linkage: new point와 가장 가까이에 있는 within point 사이의 거리
        - complete linkage: new point와 가장 멀리 있는 within point 사이의 거리
        - average linkage: within points들의 mean과의 거리

Practical issues for Clustering

- 모든 방법에 대해서 Scaling이 굉장히 중요함
- 특히 hierarchical의 경우 dissimlarity measure를 어떤 것을 쓸 것인지도 중요함
- How many clusters to choose?
    - 한편, hierarchcial clustering의 경우 branch의 length가 error reduction degree를 나타내는데, 이 값이 가장 크게 감소하는 쪽에서 자르는 것이 reasonable
- 특히 Unsupervised learning의 경우 한 번의 결과만으로 모든 것을 판단할 수는 없음!