Code

Feature Selection: A Data Perspective

Jungdong Li et. al.

abstract

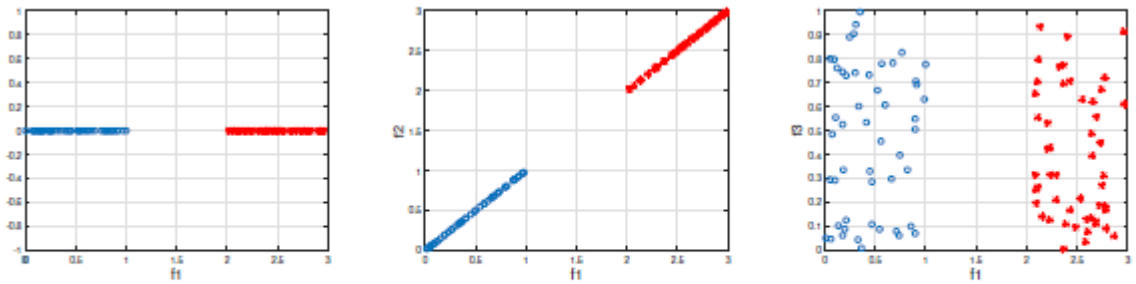
본 논문은 인자선택(Feature Selection. 이하 FS라 한다) 연구의 전반적인 것들을 소개한다. FS는 데이터 전처리 전략의 하나이다. 특히 머신러닝이나 데이터 마이닝에서 사용되는 고차원 데이터를 처리하는 데 있어서 발생하는 문제를 해결하는 방법 중 하나이다.

---

1. Introduction

고차원의 데이터를 머신러닝에 적용하려고 할때의 가장 결정적인 문제는 차원의 저주(Curse of Dimensionality) 이다. 이를 해결하기 위한 방법으로 가장 유용하게 사용되는 방법이 FE(Feature Extraction) 와 FS(Feature Selection)이 있다. 이중 FS는 원래 인자가 가지고 있던 물리적인 의미를 그대로 보존하고 있기 때문에 텍스트 마이닝이나 유전자 분석등의 어플리케이션에서 널리 사용되고 있는 방법이다. 본 논문에서는 FS를 중점적으로 다룰 것이다.

실세계의 데이터셋에는 **중복(redundancy)**되거나 **관련이 없는(not relevant)** 인자 들을 포함하고 있는 경우가 많다.



(a) relevant feature f_1 (b) redundant feature f_2 (c) irrelevant feature f_3

Fig. 1: An illustrative example of relevant, redundant and irrelevant features.

(a)는 f1 인자를 1차원으로 시각화 한 것이고 (b)는 f1 과 f2에 대한 산점도, (c)는 f1과 f2에 대한 산점도 이다. f2는 f1과 그 성질이 중복되고, f3는 class 정보와 관련이 없는 분포를 이루고 있다. 그러므로 f2와 f3는 학습 성능에 영향을 주지 않거나, 오히려 부정적인 영향을 끼친다.

1.1 Traditional Categorization of Feature Selection

FS알고리즘은 예측인자(Class Label)의 유무에 따라 크게 3가지로 나눌수 있다.

1.1.1 Supervision Perspective

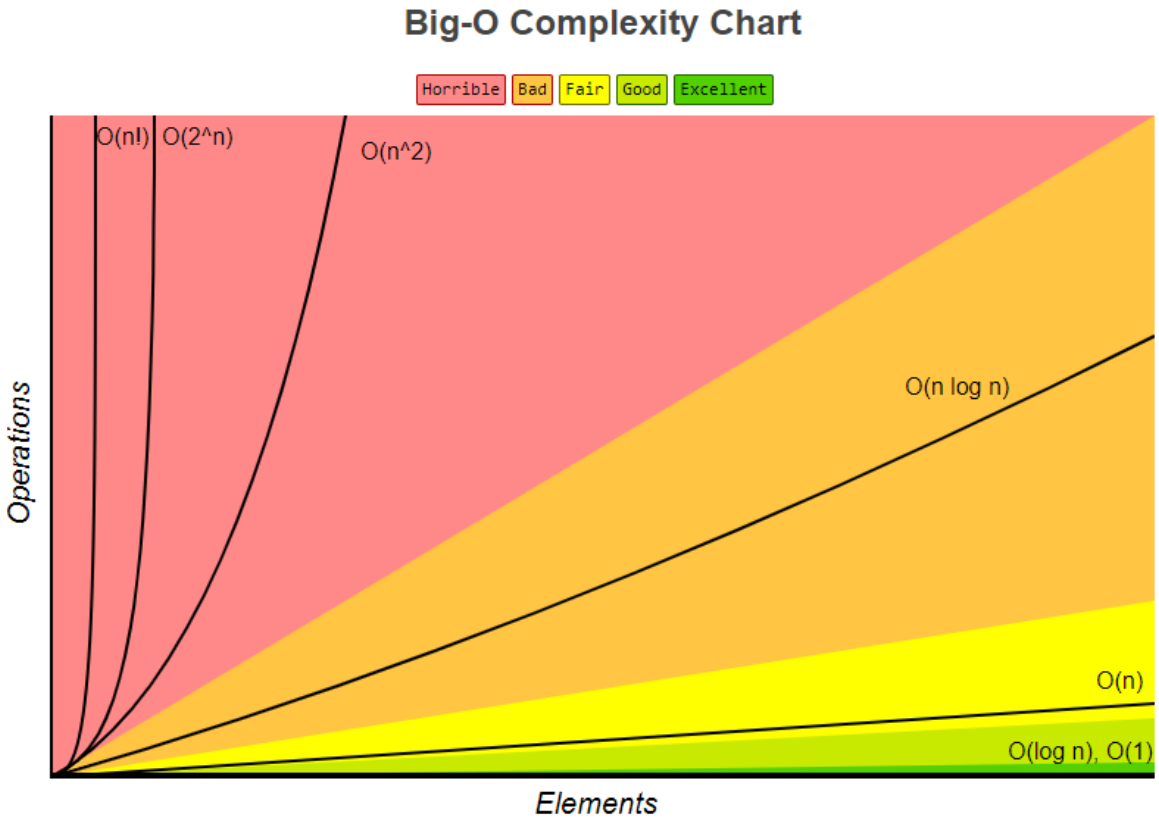
- **Supervised** Feature Selection
  - 분류(Classification)나 회귀(Regression) 문제를 해결하기 위해 고안된 FS 알고리즘
- **Unsupervised** Feature Selection
  - 군집(clustering) 문제를 해결하기 위해 고안된 FS 알고리즘
  - Label 데이터를 수집하는 데에 큰 비용이 들때 주로 사용
- **Semi-Supervised** Feature Selection

- Class Label 정보가 부족할때 이를 보완하기 위한 FS 알고리즘

1.1.2 Selection Strategy Perspective

알고리즘 선택 전략에 따라 크게 3가지로 나눌 수 있다.

- **wrapper** method
  - 부분집합 선택, 정확도 평가를 계속 반복
  - 높은 시간 복잡도 O(2^d)



  - hill-climbing search, best-first search, branch-and-bound search 알고리즘을 이용하여 개선
  - 개선된 알고리즘도 여전히 높은 시간복잡도를 자랑(?) 하기 때문에 현실적으로 잘 쓰이지 않는 편
- **filter** method
  - 평가 척도를 계산하여 순위를 줌
  - TOP-N개를 선택
- **embedded** method
  - filter와 wrapper method의 trade-off 버전
  - 학습알고리즘에 포함된 FS 알고리즘

1.2 Feature Selection Algorithms from a data perspective

FS알고리즘을 적용할 때 데이터의 형태에 따라 선택해야 할 알고리즘이 다르다. Streaming data는 인자간의 연관성(relevance) 와 중복(redundancy)이 실시간으로 변경된다. 따라서 비즈니스 영역에 있는 데이터에 대해 잘 아는 전문가와의 협업이 무엇보다도 중요하다고 할 수 있다.

본 연구에서는 비즈니스 영역을 뺀 데이터 관점으로 FS 알고리즘을 적용하는 방안을 정리하였다.

1.3 Differences with existing surveys

아래 fig.3 그림에 있는 FS 연구에 대한 전반적인 것들을 다룰것이다. Python으로 작성된 **scikit-feature** 오픈소스를 공개하였다. 아래 링크를 참조하기 바란다.

http://featureselection.asu.edu/ (http://featureselection.asu.edu/)

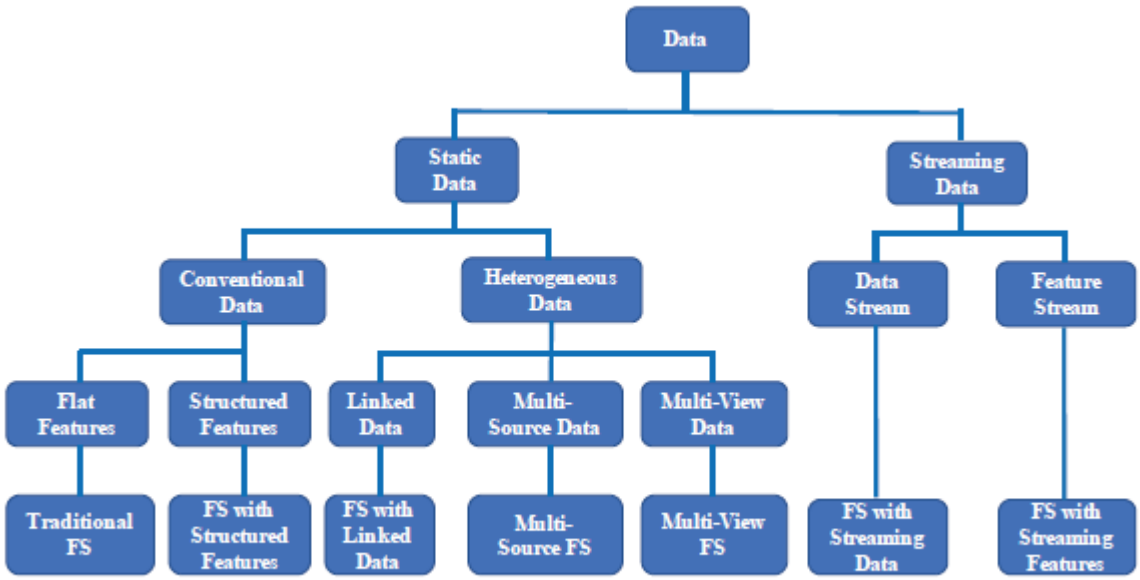


Fig. 2: Feature selection algorithms from the data perspective.

1.4 Organization of the survey

논문은 아래와 같이 구성되어 있다.

- Traditional Feature Selection for Conventional Data (Section 2)
  - Similarity based Feature Selection Methods
  - Information Theoretical based Feature Selection Methods
  - Sparse Learning based Feature Selection Methods
  - Statistical based Feature Selection Methods
  - Other Methods
- Feature Selection with Structured Features (Section 3)
  - Feature Selection Algorithms with Group Structure Features
  - Feature Selection Algorithms with Tree Structure Features
  - Feature Selection Algorithms with Graph Structure Features
- Feature Selection with Heterogeneous Data (Section 4)
  - Feature Selection Algorithms with Linked Data
  - Multi-Source Feature Selection
  - Multi-View Feature Selection
- Feature Selection with Streaming Data (Section 5)
  - Feature Selection Algorithms with Data Streams
  - Feature Selection Algorithms with Feature Streams
- Performance Evaluation (Section 6)
- Open Problems and Challenges (Section 7)
- Summary of the Survey (Section 8)

1.5 Notations

데이터 셋의 인자에 대한 표기는 아래와 같다.

Notations	Definitions or Descriptions
n	number of instances in the data
d	number of features in the data
k	number of selected features
c	number of classes (if exist)
$\mathcal{F}$	original feature set which contains d features
$\mathcal{S}$	selected feature set which contains k selected features
$\{i_1, i_2, ..., i_k\}$	index of k selected features in $\mathcal{S}$
$f_1, f_2, ..., f_d$	d original features
$f_{i_1}, f_{i_2}, ..., f_{i_k}$	k selected features
$x_1, x_2, ..., x_n$	n data instances
$\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_d$	d feature vectors corresponding to $f_1, f_2, ..., f_d$
$\mathbf{f}_{i_1}, \mathbf{f}_{i_2}, ..., \mathbf{f}_{i_k}$	k feature vectors corresponding to $f_{i_1}, f_{i_2}, ..., f_{i_k}$
$\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$	n data vectors corresponding to $x_1, x_2, ..., x_n$
$y_1, y_2, ..., y_n$	class labels of all n instances (if exist)
$\mathbf{X} \in \mathbb{R}^{n \times d}$	data matrix with n instances and d features
$\mathbf{X}_\mathcal{S} \in \mathbb{R}^{n \times k}$	data matrix on the selected k features
$\mathbf{y} \in \mathbb{R}^n$	class label vector for all n instances (if exist)

Table I: Symbols.

2. Feature Selection on Conventional Data

잘 알려져 있고 널리 사용되는 FS 알고리즘에 대해 소개한다. similarity, information, sparse learning, statstical based 알고리즘에 대해 소개한다.

2.1 Similarity based Methods

알고리즘의 평가 점수에 맞게 인자를 평가하여 골라낸 뒤, 원래의 데이터셋이 가진 특징을 잘 대표하는지를 평가하는 방법이다.

2.1.1 Laplacian Score

같은 클래스의 데이터 끼리 거리가 가까운가를 평가 기준으로 한다. 거리가 가까울 수록, 즉 같은 클래스끼리 분산이 작을 수록 좋은 인자라고 판단한다.

아래는 iris 데이터이다.

Hide

```
# install.packages("Rdimtools")
# ** 설치가 안될 경우
# $ sudo apt-get update
# $ sudo apt-get install libgmp3-dev
# $ sudo apt-get install libmpfr-dev
library(Rdimtools)
library(plotly)
## merge the data and create a label correspondingly
X <- iris[,-5]
y <- iris[,5]
head(iris)
```

	Sepal.Length <dbl>	Sepal.Width <dbl>	Petal.Length <dbl>	Petal.Width <dbl>	Species <fctr>
1	5.1	3.5	1.4	0.2	setosa

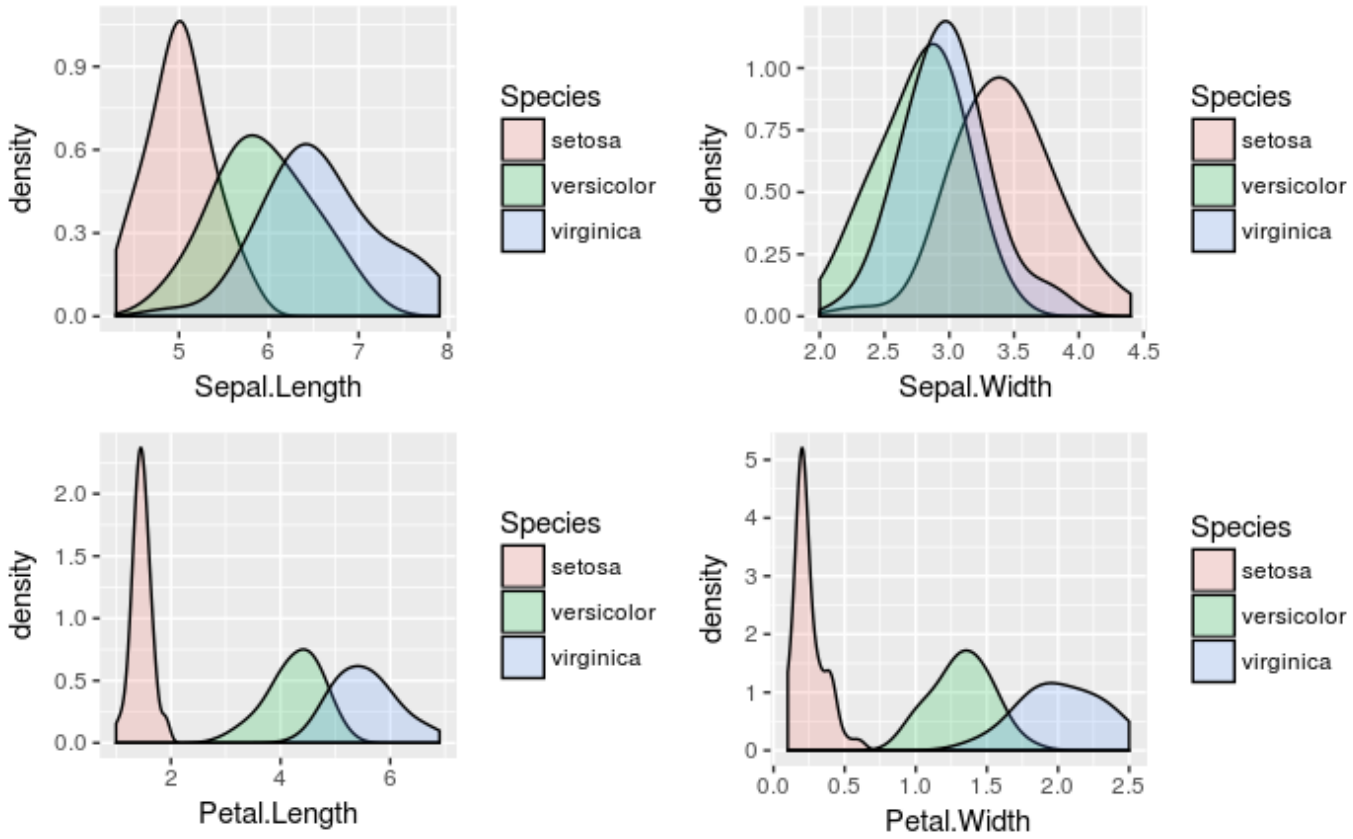
	Sepal.Length <dbl>	Sepal.Width <dbl>	Petal.Length <dbl>	Petal.Width <dbl>	Species <fctr>
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa

6 rows

아래는 iris 데이터의 인자별 데이터의 확률밀도그래프 이다. Class Label 별로 좁은 범위에 분포할수록 좋은 인자라고 판단할 수 있을 것이다. 직관적으로 봐서는 3,4 번째 인자가 좋은 인자처럼 보인다.

Hide

```
#install.packages("gridExtra")
library(gridExtra)
library(ggplot2)
p1 <- ggplot(data=iris, aes(x=Sepal.Length, group=Species, fill=Species)) + geom_dens
ity(adjust=1.5 , alpha=0.2)
p2 <- ggplot(data=iris, aes(x=Sepal.Width, group=Species, fill=Species)) + geom_densi
ty(adjust=1.5 , alpha=0.2)
p3 <- ggplot(data=iris, aes(x=Petal.Length, group=Species, fill=Species)) + geom_dens
ity(adjust=1.5 , alpha=0.2)
p4 <- ggplot(data=iris, aes(x=Petal.Width, group=Species, fill=Species)) + geom_densi
ty(adjust=1.5 , alpha=0.2)
grid.arrange(p1, p2, p3, p4, nrow=2, ncol=2)
```



Laplacian-Score는 아래와같이 계산한다. Laplacian score 점수가 높을수록 안좋은 인자라고 할수 있다. 2번과 1번 인자가 높은 점수를 받아 제거 대상이 되었다.

Hide

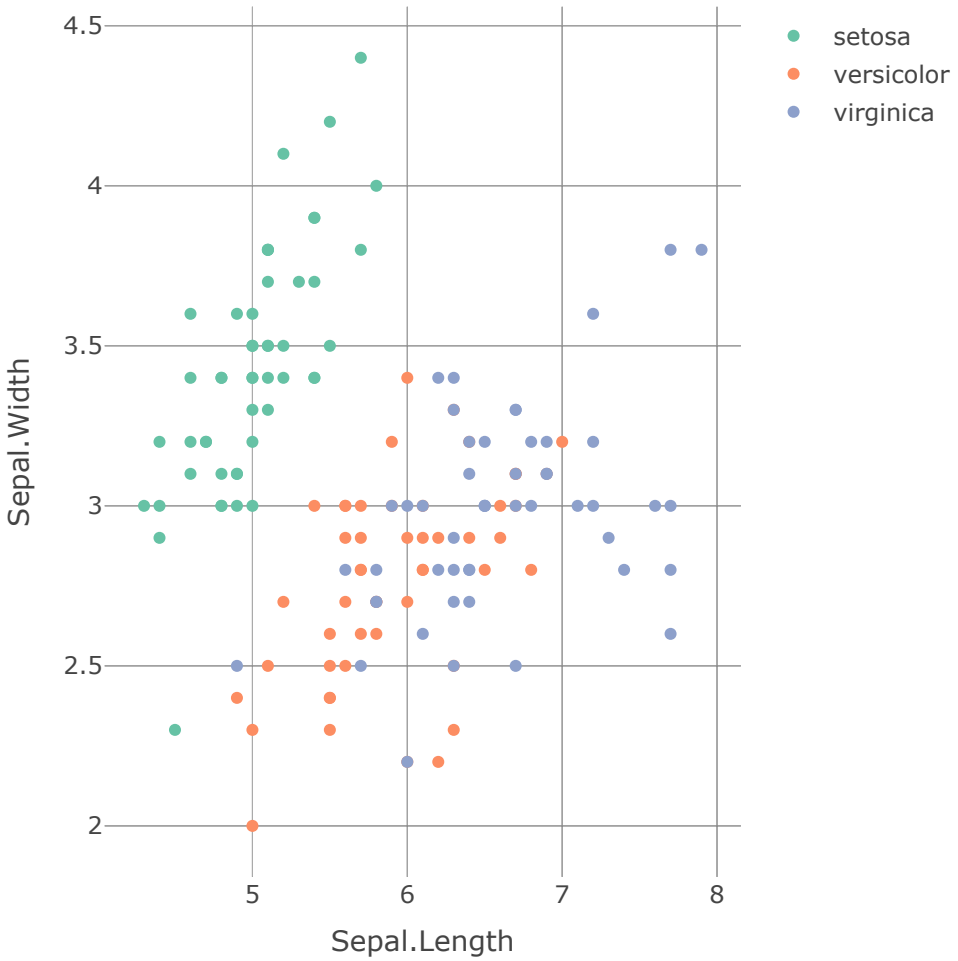
```
## try different kernel bandwidth
out = do.lscore( as.matrix(X) , t=10) # laplacian score 실행
out$featidx # 제거할 인자 번호
```

```
[1] 2 1
```

결론적으로 3,4번 인자가 FS 알고리즘에 의해 선택되었다. 선택되지 않은 인자와 선택된 인자만을 가지고 산점도를 그려 직관적으로 비교해 볼때도 아래 3,4번째 인자가 class 정보를 더 잘 설명해주는 인자임을 확인할 수 있다.

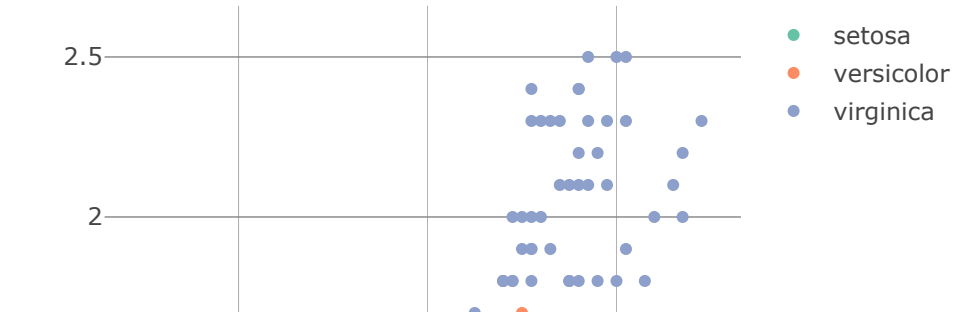
Hide

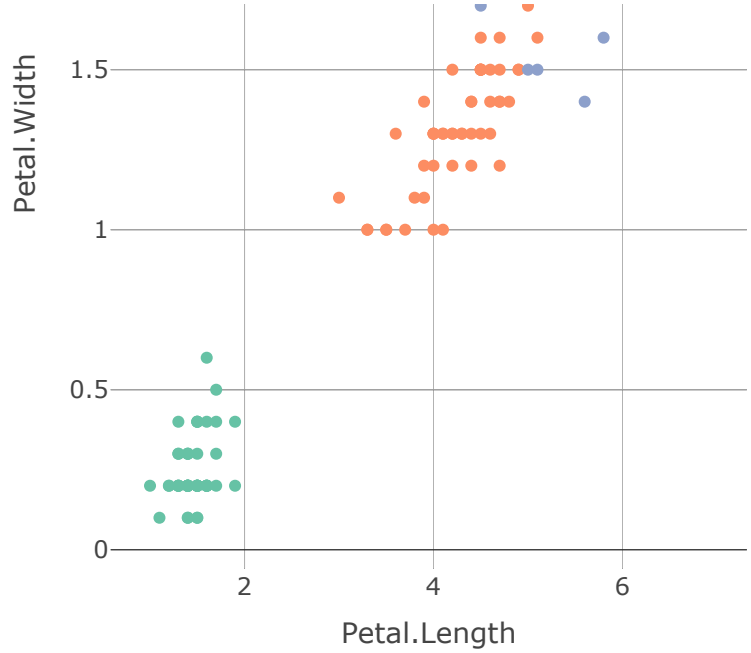
```
plot_ly(data = iris, x = ~Sepal.Length, y = ~Sepal.Width, color = ~Species)
```



Hide

```
plot_ly(data = iris, x = ~Petal.Length, y = ~Petal.Width, color = ~Species)
```





2.1.2 SPEC

Laplacian Score 알고리즘의 supervised 확장 버전이다.(추가필요)

2.1.3 Fisher Score

서로 다른 CLASS를 가진 데이터의 평균 간 거리가 멀수록, 같은 CLASS를 가진 데이터간의 분산이 작을수록 높은 점수를 가진다. 평균간의 거리가 멀다는 것은 클래스 간의 경계가 멀다는 의미이고, 분산이 작다는 것은 데이터가 모여있다는 의미이다.

$$fisher\_score(f_i) = \frac{\sum_{j=1}^{c} n_j(\mu_{ij} - \mu_i)^2}{\sum_{j=1}^{c} n_j \sigma_{ij}^2},$$

iris 데이터의 1~4번째 인자에 대해 fisher score를 구한다. 4,3,1,2 순서로 인자가 선택되었다.

Hide

```
#install.packages("PredPsych")
library(PredPsych)
fscore(cbind( as.numeric( iris[,5] ) , iris[, -5] ),classCol = 1,featureCol = c(2:5))
```

```
Performing Feature selection f-score analysis

Sepal.Length  Sepal.Width Petal.Length  Petal.Width
         2.2          1.8         31.0         23.0
```

2.1.4 Trace Ratio Criterion

알고리즘의 아이디어를 이해하지 못함. (추가필요) Trace Ratio Criterion 알고리즘을 소개한 논문이니 참고하기 바람.
Feiping Nie, Shiming Xiang, Yangqing Jia, Changshui Zhang, and Shuicheng Yan. Trace ratio criterion for feature selection. In AAAI, volume 2, pages 671–676, 2008.

> http://www.aaai.org/Papers/AAAI/2008/AAAI08-107.pdf
> (http://www.aaai.org/Papers/AAAI/2008/AAAI08-107.pdf)

2.1.5 ReliefF

좋은 인자일수록 같은 class끼리의 데이터는 서로 모여있을 것이라는 가정에서 출발한 알고리즘이다. 데이터 하나를 xi 라고 하고, xi 와 최근접 동일 class 데이터를 nearHIT, 최근접 다른 class 데이터를 nearMISS 라고 하자. 인자가 class를 잘 분류한다면 nearHIT 간의 거리는 가까울수록, nearMISS 간의 거리는 멀어진다. 인자는 0~1 사이로 표준화(standardization) 한다. W는 ReliefF 값을 구하기 위해 각 인자들이 가지는 가중치 값이다. W값을 바탕으로 ReliefF 평가값을 계산한다.

$$W_i = W_i - (x_i - \text{nearHit}_i)^2 + (x_i - \text{nearMiss}_i)^2$$

2.1.6 Discussion

유사도 기반 FS 알고리즘은 지도/비지도 학습 문제에 뛰어난 성능을 보인다. 이 알고리즘은 직관적이면서도 간단하다. 또한 머신러닝이나 데이터 마이닝 알고리즘에 상관 없이 독립적으로 선택된 인자를 선택하는 알고리즘이기 때문에 널리 사용된다. 하지만 이 방법의 단점은 데이터셋의 feature redundancy(인자 중복)를 극복하지 못한다. 다시말해 높은 상관관계를 가진 인자들을 고려해서 인자 부분집합을 고르지 못한다.

2.2 Information Theoretical based Methods

이 장에서는, 정보량 기반의 알고리즘을 소개한다. FS알고리즘의 또 다른 부류 중 하나는 정보량 기반 방법(information theoretical method) 이다. 대부분 정보량 기반 방법은 연관성을 최대화 하고 중복성을 최소화 하는 방법을 제안하고 있다. 연관성은 Class Label 과의 상관성으로 측정되기 때문에, 알고리즘의 대부분이 지도학습(supervised) 방식으로 돌아간다. FS 알고리즘은 이산형 데이터에 대해 적용되는 경우가 많다.

2.2.1 Entropy measures

먼저 정보량 기반이란 무엇인지 그 개념에 대해 설명한다.

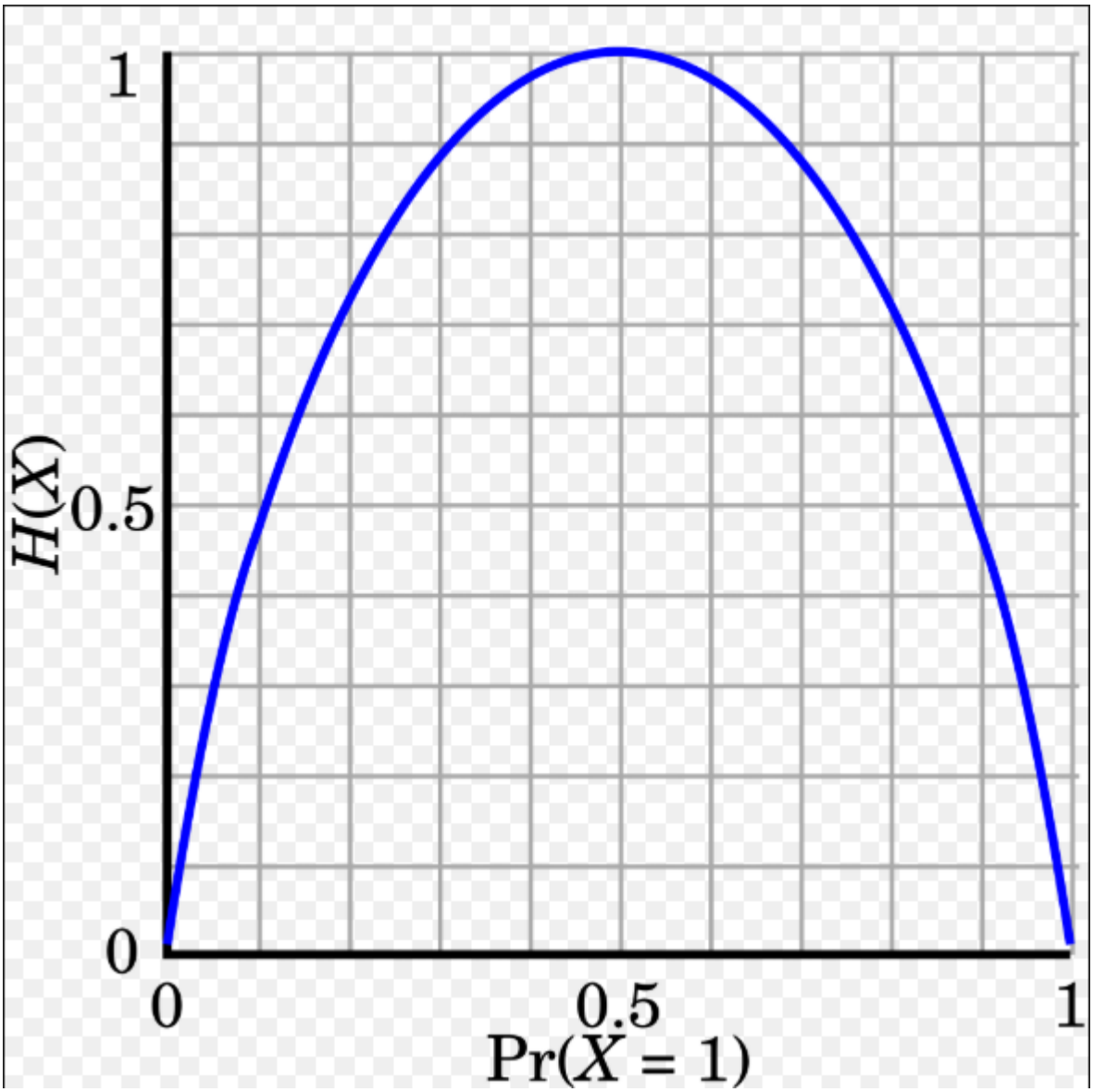
집단 내의 데이터의 순도(purity)를 계산하기 위해 Entropy 를 계산한다. Shanon Entropy라고도 한다. Entropy 측정법은 이산 랜덤변수의 불확실성에 기반을 두는 방법이다. Entropy의 기본 정의는 아래와 같다.

$$H(X) = -\sum_{x_i \in X} P(x_i) log(P(x_i)),$$

X에 대한 y의 조건부 Entropy의 정의는 다음과 같다.

$$H(X|Y) = -\sum_{y_j \in Y} P(y_j) \sum_{x_i \in X} P(x_i|y_j) log(P(x_i|y_j)),$$

즉, 서로 다른 Class Label 의 확률에 따라 Entropy는 아래 그림과 같은 값을 가지게 된다.



정보획득량(information gain)이란, 인자를 특정 조건으로 나누었을 때 감소되는 Entropy를 측정하는 척도이다. 데이터를 나누었는데 분리한 각각의 데이터셋 복잡도가 감소한다면, 데이터를 잘 분류하는 인자라고 할 수 있다. X와 Y에 대한 정보획득량 (또는 상호정보량(mutual information)) 의 총 합은 아래와 같이 계산된다.

$$I(X;Y) = H(X) - H(X|Y) = \sum_{x_i \in X} \sum_{y_j \in Y} P(x_i, y_j) log \frac{P(x_i, y_j)}{P(x_i)P(y_j)},$$

마지막으로 X, Y와 이산 변수 Z 에 대한 조건부 정보 획득량(conditional information gain)은 아래와 같이 정의한다.

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z) = \sum_{z_k \in Z} P(z_k) \sum_{x_i \in X} \sum_{y_j \in Y} P(x_i, y_j|z_k) log \frac{P(x_i, y_j|z_k)}{P(x_i|z_k)P(y_j|z_k)}.$$

정보량을 기반으로 최적화된 인자를 찾는 문제는 NP-hard 문제이다. 그래서 대부분의 알고리즘은 휴리스틱 방식으로 인자를 하나씩 추가해 가면서 구현하는 것이 일반적이다. 대부분의 정보량 기반 알고리즘은 선택되지 않은 인자들을 하나씩 추가하는 방식으로 인자의 부분집합을 단계별로 평가하여 점수가 높아지면 채택하고 아니면 버리는 방식으로 구현한다.

2.2.2 Mutual Information Maximization(MIM)

인자의 중요도를 class label과의 상관성을 가지고 측정하는 방식이다. 인자가 class label 과 밀접한 관련이 있다면, 그 인자를 선택하는 것이 좋은 분류 성능을 낼것이라는 가정에서 출발한다.

$$J_{\text{MIM}}(X_k) = I(X_k;Y)$$

평가값은 각 인자별로 별도로 매겨지게 된다. 그러므로 이 방식은 중복성(redundancy) 를 고려하지 못한다. 모든 인자들이 평가값을 얻게 되면, 높은 평가점수를 받은 인자를 선택하게 된다. 원하는 인자의 집합 개수를 얻을 때까지 알고리즘을 반복하게 된다.

Hide

```
#install.packages("mRMRe")
library(mRMRe)
data(cgps)
data.annot <- data.frame(cgps.annot)
data.cgps <- data.frame(cgps.ic50, cgps.ge)
dd <- mRMR.data(data = data.cgps)
dd <- subsetData(dd, 1:5, 1:5)
# Uses Spearman as correlation estimator
spearman_mim <- mim(dd, continuous_estimator = "pearson")
print(spearman_mim)
```

```
            cgps.ic50 geneid_3310 geneid_2978 geneid_6352 geneid_2621
cgps.ic50         Inf  0.01793127  0.25391518  0.09483033  0.16774973
geneid_3310 0.01793127         Inf  0.06015939  0.06014209  0.38063381
geneid_2978 0.25391518  0.06015939         Inf  0.22869783  0.06358484
geneid_6352 0.09483033  0.06014209  0.22869783         Inf  0.27176659
geneid_2621 0.16774973  0.38063381  0.06358484  0.27176659         Inf
```

2.2.3 Minimum Redundancy Maximum Relevance

class label과 상관성이 높은 인자는 포함하고(Maximum Relevance), 중복성이 높은 설명변수는 제거 (Minimum Redundancy) 하는 방식이다. 선택된 인자의 수의 역수를 이용하여 계산한다. 즉, 선택된 인자가 많으면 평가값이 낮아진다. 중복성이 낮은 인자가 선택되었을 경우, 이후의 단계에서 새로 추가될 인자는 포함되지 않을 가능성이 높아지게 된다.

$$J_{\mathrm{MRMR}}(X_k) = I(X_k;Y) - \frac{1}{|\mathcal{S}|}\sum_{X_j\in\mathcal{S}} I(X_k;X_j)$$

2.2.4 Conditional Infomax Feature Extraction

인자의 중복성을 최소화 하는 것과는 대조적으로, 선택되지 않은 인자간의 조건부 중복성과 이미 선택된 class label이 있는 인자간의 중복성도 동시에 커지게 된다. 다시말해 class label이 주어진 인자 중복성이 커지는 만큼, 인자 선택의 효율이 점점 떨어지게 된다. 이러한 것을 반영한 것이 CIFE 이다.

> http://ww.dahua.me/publications/dhl06_cil.pdf
> (http://ww.dahua.me/publications/dhl06_cil.pdf)

$$J_{\mathrm{CIFE}}(X_k) = I(X_k;Y) - \sum_{X_j\in\mathcal{S}} I(X_j;X_k) + \sum_{X_j\in\mathcal{S}} I(X_j;X_k|Y).$$

MIFS와 비교하여 조건부 중복성을 최대화 하기 위한 수식이 마지막에 추가되었다.

2.2.5 Joint Mutual Information

MIFS와 MRMR 은 인자의 중복성을 제거한다. JMI는 다른 평가기준으로 결합 상호정보량을 제안한다. 선택되지 않은 인자와 선택된 인자간의 평가값을 비교해 가면서 인자를 선택한다.

http://www.jmlr.org/papers/volume13/brown12a/brown12a.pdf (http://www.jmlr.org/papers/volume13/brown12a/brown12a.pdf)

$$J_{\text{JMI}}(X_k) = \sum_{X_j \in \mathcal{S}} I(X_k, X_j; Y).$$

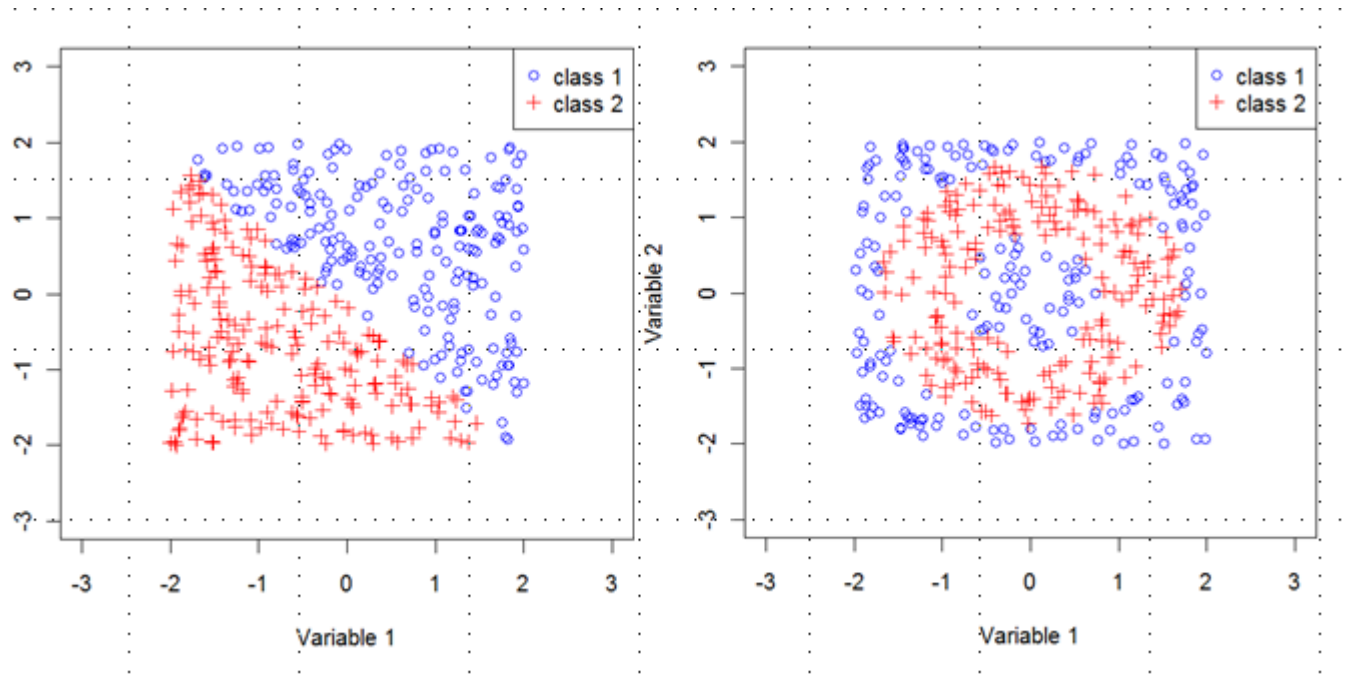
Hide

```
#install.packages("praznik")
library(praznik)
data(MadelonD)
JMI(MadelonD$X,MadelonD$Y,20)
```

```
$selection
 Rel7 Rel11 Rel12  Rel4 Rel18 Rel19 Rel13  Rel3  Rel5 Rel14 Rel16 Rel10  Rel2  Rel8 R
el20  Rel9  Rel6  Rel1 Rel15 Rel17
  242   339   379   106   473   476   434    65   129   443   454   337    49   282
494   319   154    29   452   456

$score
      Rel7      Rel11      Rel12      Rel4      Rel18      Rel19      Rel13       Re
l3       Rel5      Rel14      Rel16      Rel10
0.03229728 0.10366153 0.17323337 0.22608508 0.30501767 0.37577003 0.43573912 0.487056
03 0.57245848 0.62775062 0.66346880 0.72297606
      Rel2       Rel8      Rel20       Rel9       Rel6       Rel1      Rel15      Rel
17
0.76368695 0.78870899 0.82602547 0.85446328 0.87668512 0.90102204 0.90848744 0.789136
37
```

2.2.6 Conditional Mutual Information Maximization(CMIM)



Non-Linear 조합을 가진 인자에 대한 중복성을 제거하기 위한 알고리즘이다. 다른 알고리즘은 선형성을 가지고 측정 지표를 선택한 반면, CMIM은 비선형성 인자간의 중복성을 제거하기 위한 지표로서 예측력을 사용한다.

$$J_{\mathrm{CMIM}}(X_k) = \min_{X_j \in \mathcal{S}} [I(X_k; Y|X_j)].$$

Hide

```
data(MadelonD)
CMIM(MadelonD$X,MadelonD$Y,20)
```

```
$selection
  Rel7   Rel4  Rel11  Rel10  Rel18  Rel16  Rel12   Rel5  Rel20  Rel13   Rel3 Irr296
Rel2 Irr275  Irr11  Rel19 Irr432  Irr32 Irr230 Irr321
   242    106    339    337    473    454    379    129    494    434     65    304
   49    283     11    476    446     33    236    330

$score
       Rel7        Rel4       Rel11       Rel10       Rel18       Rel16       Rel12
      Rel5       Rel20       Rel13        Rel3
0.032297277 0.028152132 0.026347876 0.021784040 0.014947856 0.009002574 0.008501701
0.007338058 0.007126401 0.006311766 0.006284111
     Irr296        Rel2      Irr275       Irr11       Rel19      Irr432       Irr32
    Irr230      Irr321
0.006171071 0.005964899 0.005675982 0.005270937 0.005270280 0.005246876 0.005175817
0.005084441 0.004998719
```

2.2.7 Informative Fragments (IF)

IF 라고 하는 평가지표를 제안한다. IF는 아래와 같이 정의한다. (설명 추가필요)

$$J_{\mathrm{IF}}(X_k) = \min_{X_j \in \mathcal{S}} [I(X_j X_k; Y) - I(X_j; Y)].$$

2.2.8 Interaction Capping

CMIM과 동일하게 Non-Linear 방식에 대한 중복성을 제거하기 위한 알고리즘이다. CMIM과 유사한 평가방식을 사용한다. 다만 CMIM에서 $I(X_j; X_k) - I(X_j; X_k|Y)$ 부분을 양수로 세팅해서 가져간다.

$$J_{\mathrm{CMIM}}(X_k) = I(X_k; Y) - \sum_{X_j \in \mathcal{S}} \max[0, I(X_j; X_k) - I(X_j; X_k|Y)].$$

2.2.9 Double Input Symmetrical Relevance

정보량 기반의 또다른 평가방식이 바로 DISR이다. 정보량을 normalization(정규화) 하는 방식을 활용한다. 서로다른 정보량을 정규화 하여 선형과 비선형 알고리즘의 성능비교를 가능하게 한다.

$$J_{\mathrm{DISR}}(X_k) = \sum_{X_j \in \mathcal{S}} \frac{I(X_j X_k; Y)}{H(X_j X_k Y)}.$$

2.2.10 Fast Correlation Based Filter

클래스 인자와 상관관계가 높을수록, 다른 X인자와 상관관계가 낮을수록 높은 값을 가진다. SU는 두 명목변수간의 연관성을 측정하는 지표이다. SU(Symmetrical uncertainty)를 계산하고, 계산된 값이 임계치(threshold) 보다 낮은 인자들을 제거한다. filter 방식으로 동작한다. 연속형 변수인 경우 이산화 하여 계산하게 된다. SU는 아래와 같이 정의된다.

$$SU(X_{\mathcal{S}}, Y) = 2\frac{I(X_{\mathcal{S}}; Y)}{H(X_{\mathcal{S}}) + H(Y)}.$$

알고리즘은 아래와 같다.

Hide

```
## FCBF feature selection.
## 20170210 dhkang
#symmetrical uncertainty
SU <- function( x, y, nx=5, ny=5 ){

  if(is.factor( x )){
    xgrp <- x
  }else{
    xgrp <- cut( x, nx )
  }

  if( is.factor( y ) ){
    ygrp <- y
  }else{
    ygrp <- cut( y, ny )
  }
  px <- table(xgrp) / length(xgrp)
  py <- table(ygrp) / length(ygrp)
  pxy <- apply( table(xgrp,ygrp), 2, function(x) { x/sum(x) } )
  lpx <- ifelse(px==0,0,log(px,base=2))
  lpy <- ifelse(py==0,0,log(py,base=2))
  lpxy <- ifelse(pxy==0,0,log(pxy,base=2))
  Hx <- -sum(px*lpx)
  Hy <- -sum(py*lpy)
  Hxy <- -sum(py*colSums(pxy*lpxy))
  return(2*(Hx-Hxy)/(Hx+Hy))
}
## fcbf function
fcbf <- function(train,group,delta,lambda=1,nx=5,ny=5){
  Slist <- numeric(0)
  vals <- apply(train,2,function(x)SU(x,group,nx=nx,ny=ny))
  Slist <- order(vals,decreasing=TRUE)
  Slist <- Slist[vals[Slist]>=delta]
  N <- length(Slist)
  for(p in 1:N){
    if(!is.na(Slist[p])){
      indxUse <- na.omit(Slist[-(1:p)])
      trainUse <- train[,indxUse]
      if(class(trainUse)=="numeric"){
        valp <- SU(trainUse,train[,Slist[p]])
        Slist[indxUse[valp >= lambda*vals[na.omit(Slist[-(1:p)])]]] <- NA
      }else if(dim(trainUse)[2]>0){
        valp <- apply(train[,na.omit(Slist[-(1:p)])],2,function(x)SU(x,train[,Slist
[p]]))
        Slist[Slist%in%indxUse[valp >= lambda*vals[na.omit(Slist[-(1:p)])]]] <- NA
      }
    }
  }
  return(as.vector(na.omit(Slist)))
}
ds <- iris[,-5]
cl <- iris[,5]
## run
fcbf(ds, cl, delta = 0.5)
```

```
[1] 4 3
```

2.2.11 Discussion

정보량 기반 FS 알고리즘은 연관성과 중복성을 통계적인 방식으로 계산하여 인자를 평가한다. 유사도 기반 feature selection 과 마찬가지로, 어떠한 데이터마이닝이나 머신러닝 알고리즘에 전처리로 적용이 가능하다는 장점이 있다. 하지만 정보량을 평가하기 위해서는 class label이 반드시 필요하며, 이산형 데이터와 연속형 데이터에만 적용될수 있고, 연속형 데이터는 이산화 과정이 필요하다.

2.3 Sparse Learning based Methods

세번째 방법으로 fiiting error를 최소화 하는 방법인 정규화(regularizer) 를 소개하려고 한다. 정규화는 인자의 계수를 0으로 만들어서, 필요 없는 인자의 영향력을 제거하는 방법이다. 성능이 좋고 해석력이 높아 최근 주목받고 있는 방법이다. 이번 장에서는 지도/비지도 관점에서의 sparse learning 기반의 FS 알고리즘에 대해 살펴보도록 한다.

2.3.1 Feature Selection with Lp-norm Regularizer

대표적으로 LASSO Regression, Ridge Regression에 이용된다. 잔차제곱 합에 가중치의 절대값의 합을 최소화하는 것을 추가적인 제약 조건으로 가져간다. MSE가 최소가 되게 하는 가중치와 편향을 찾는데 동시에 가중치들의 절대값들의 합, 즉 가중치의 절대값들이 최소(기울기가 작아지도록)가 되게 해야한다는 것이다. 다시 말해서 가중치의 모든 원소가 0이 되거나 0에 가깝게 되게 해야한다. 따라서 어떤 인자들은 모델을 만들때 사용되지 않을 수도 있다. 어떤 벡터의 요소들의 절대값들의 합은 L1-norm이므로 라쏘는 간단히 말해서 L1-norm 패널티를 가진 선형 회귀 방법이다.

$$\min_{\mathbf{w}} loss(\mathbf{w}; \mathbf{X}, \mathbf{y}) + \alpha \|\mathbf{w}\|_p,$$

경제 지표를 나타내는 데이터이다.

Hide

```
data(longley) # Predict number of people employed from economic variables
head(longley)
```

	GNP.deflator <dbl>	GNP <dbl>	Unemployed <dbl>	Armed.Forces <dbl>	Population <dbl>	Year <int>	Employed <dbl>
1947	83.0	234.289	235.6	159.0	107.608	1947	60.323
1948	88.5	259.426	232.5	145.6	108.632	1948	61.122
1949	88.2	258.054	368.2	161.6	109.773	1949	60.171
1950	89.5	284.599	335.1	165.0	110.929	1950	61.187
1951	96.2	328.975	209.9	309.9	112.075	1951	63.221
1952	98.1	346.999	193.2	359.4	113.270	1952	63.639

6 rows

Hide

```
ds <- longley[,-7]
cl <- longley[,7]
```

인자간의 상관관계를 보여준다.

Hide

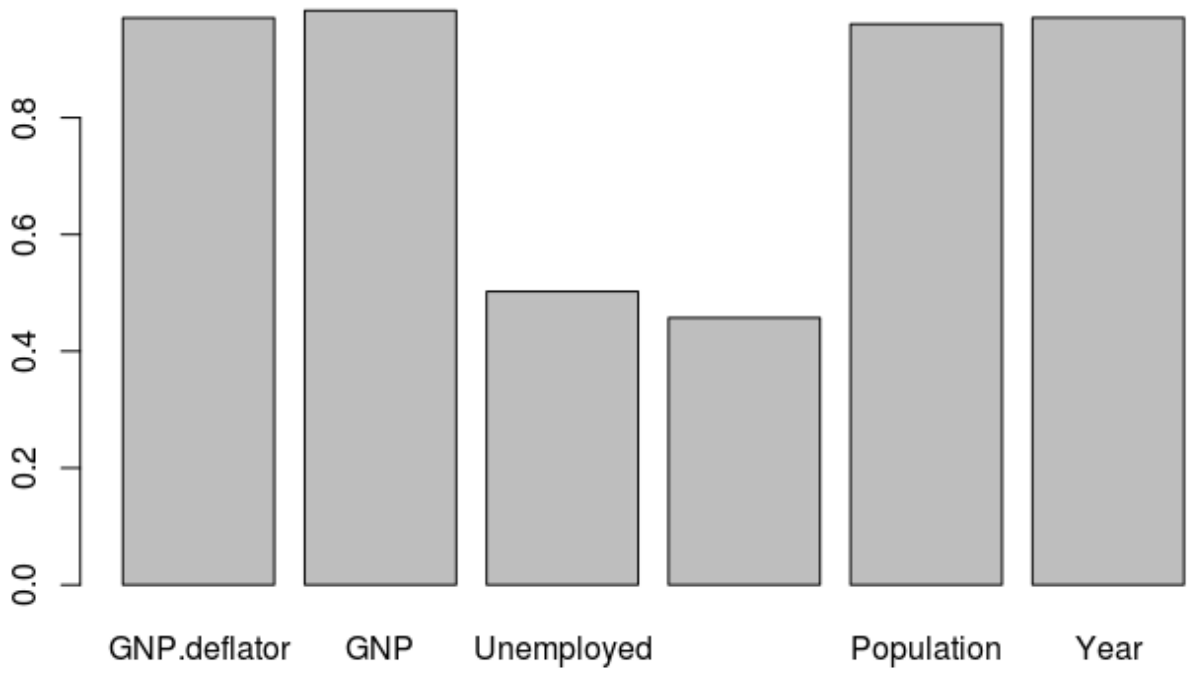
```
cor(longley)
```

```
             GNP.deflator       GNP Unemployed Armed.Forces Population      Year  Emp
loyed
GNP.deflator    1.0000000 0.9915892  0.6206334    0.4647442  0.9791634 0.9911492 0.97
08985
GNP             0.9915892 1.0000000  0.6042609    0.4464368  0.9910901 0.9952735 0.98
35516
Unemployed      0.6206334 0.6042609  1.0000000   -0.1774206  0.6865515 0.6682566 0.50
24981
Armed.Forces    0.4647442 0.4464368 -0.1774206    1.0000000  0.3644163 0.4172451 0.45
73074
Population      0.9791634 0.9910901  0.6865515    0.3644163  1.0000000 0.9939528 0.96
03906
Year            0.9911492 0.9952735  0.6682566    0.4172451  0.9939528 1.0000000 0.97
13295
Employed        0.9708985 0.9835516  0.5024981    0.4573074  0.9603906 0.9713295 1.00
00000
```

설명인자(X)와 목표인자(y) 간의 상관관계는 다음과 같다.

Hide

```
barplot(
      as.vector(
        cor(ds, cl)
      ),
      names.arg = factor(names(ds))
  )
```



alpha = 1일때 Lasso regression 계산 결과이다.

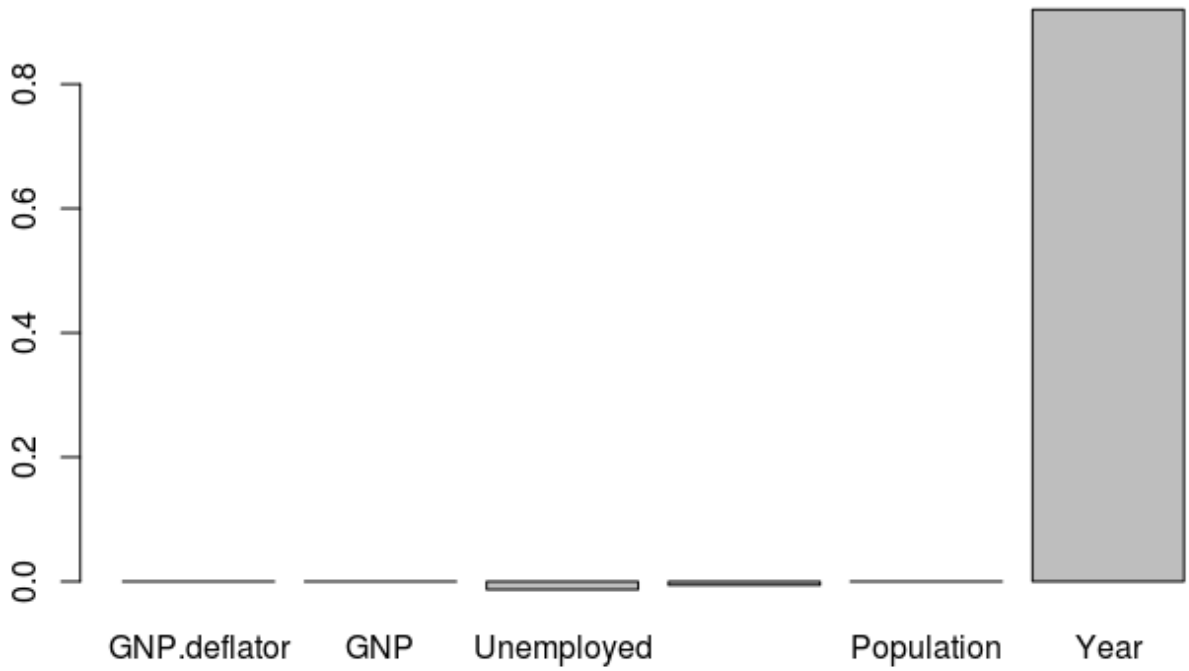
Hide

```
#install.packages("glmnet")
library(glmnet)
cv.lasso <- cv.glmnet(as.matrix(ds), as.numeric(cl), alpha=1, grouped = FALSE)
coef(cv.lasso)
```

```
7 x 1 sparse Matrix of class "dgCMatrix"
                         1
(Intercept)  -1.727355e+03
GNP.deflator  .
GNP           .
Unemployed   -1.302924e-02
Armed.Forces -6.015634e-03
Population    .
Year          9.201336e-01
```

Hide

```
barplot(
      as.vector(
        coef(cv.lasso)[-1]
      ),
      names.arg = factor(names(ds))
  )
```



alpha = 0.1일때 Lasso regression 계산 결과이다.

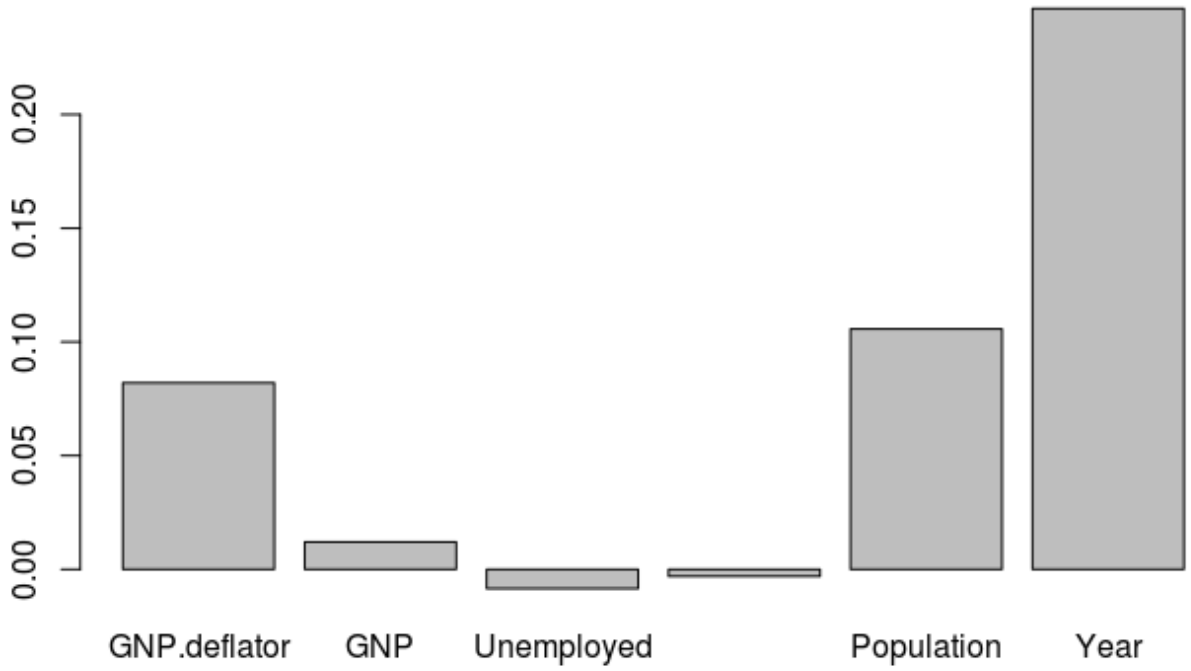
Hide

```
cv.lasso <- cv.glmnet(as.matrix(ds), as.numeric(cl), alpha=0.1, grouped = FALSE)
coef(cv.lasso)
```

```
7 x 1 sparse Matrix of class "dgCMatrix"
                          1
(Intercept)  -4.385807e+02
GNP.deflator  8.204091e-02
GNP           1.204013e-02
Unemployed   -8.478631e-03
Armed.Forces -3.101011e-03
Population    1.057129e-01
Year          2.466054e-01
```

Hide

```
barplot(
      as.vector(
        coef(cv.lasso)[-1]
      ),
      names.arg = factor(names(ds))
  )
```



Loss function에 가중치의 합을 추가적인 조건으로 하여 계산하기 때문에, 가중치의 합도 최소가 되야 한다. 따라서 회귀모델을 설명하는데 필요없는 인자들에 대한 가중치를 0(또는 0에 가깝게) 으로 만든다.

2.3.2 Feature Selection with Lpq-norm Regularizer

(추가바람)

2.3.3 Efficient and Robust Feature Selection

(추가바람)

2.3.4 Multi-Cluster Feature Selection

(추가바람)

2.4 Statistical based Methods

편차를 이용한 통계에 기반한 방법이다. 대부분이 filter method 방식을 사용한다. 따라서 중복성(redundancy) 평가는 고려하지 않는다.

2.4.1 Low Variance

임계치보다 낮은 분산을 가진 인자를 제거하는 방법이다. 예를들어 분산이 0일 경우, 이 데이터는 분류를 하는데 있어서 아무런 도움이 되지 않는다. 따라서 이러한 변수를 우선적으로 제거해 준다.

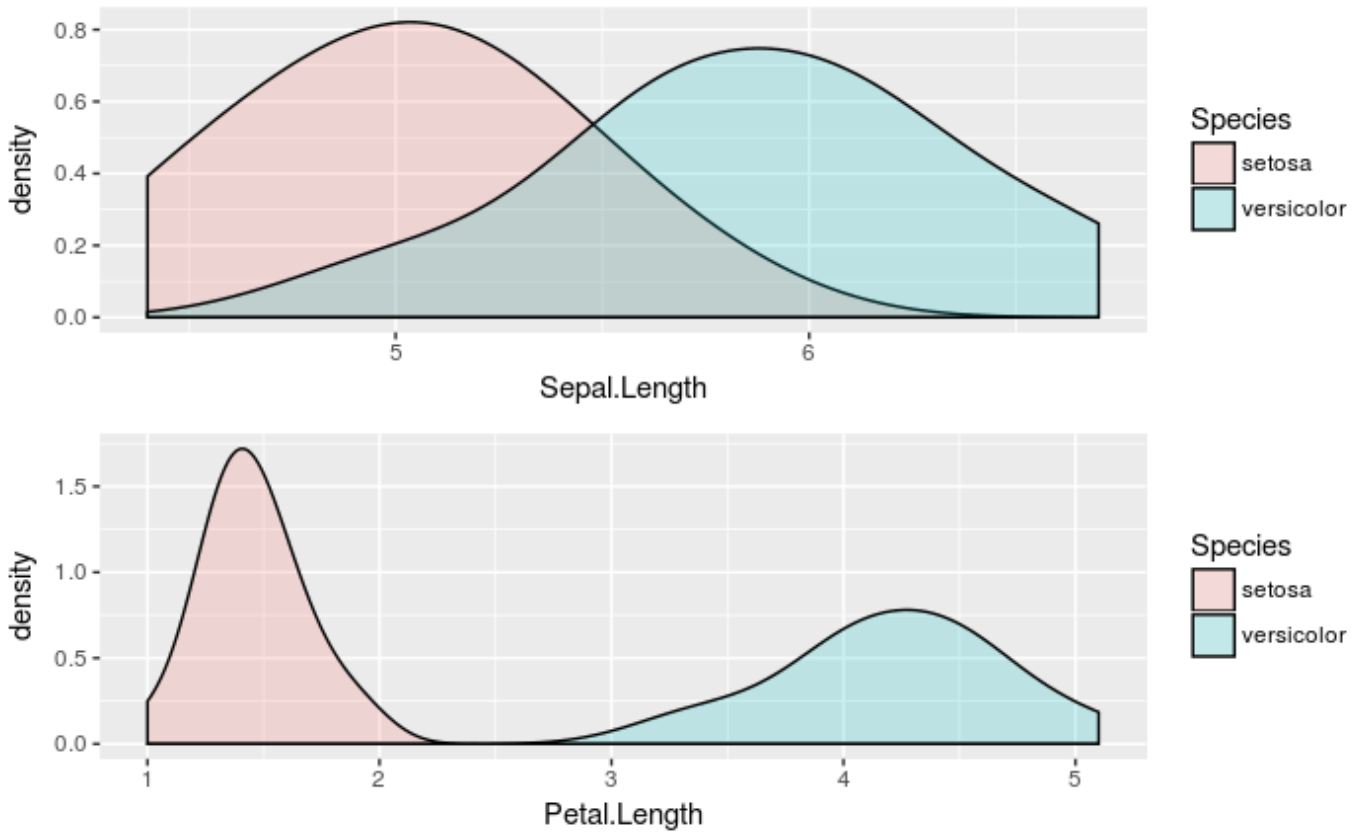
2.4.2 T-score

대응표본 T-TEST를 이용한다. 2개의 집단 차이를 측정하는 지표로 T-test를 사용한다. 각 인자의 평균과 표준편차를 이용하여 두 인자 통계적 차이가 유의미한 차이를 가지지 못할 경우 제거하는 방식이다. T-score 가 높을수록 두 집단을 나누는데 중요한 인자로 선택한다. T-score 는 아래와 같이 계산된다.

$$t\_score(f_i) = |\mu_1 - \mu_2| / \sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}.$$

Hide

```
#library(gridExtra)
library(ggplot2)
p5 <- ggplot(data=iris[iris$Species==c("setosa", "versicolor"),], aes(x=Sepal.Length,
 group=Species, fill=Species)) + geom_density(adjust=1.5 , alpha=0.2)
p6 <- ggplot(data=iris[iris$Species==c("setosa", "versicolor"),], aes(x=Petal.Length,
 group=Species, fill=Species)) + geom_density(adjust=1.5 , alpha=0.2)
grid.arrange(p5,p6,nrow=2, ncol=1)
```



2.4.3 Chi-Square Score

카이제곱검정을 이용하여 인자를 평가한다. 명목형 X, y값을 이용하며, 카이제곱값이 상대적으로 높을 수록 영향력 있는 중요한 인자라는 것을 나타낸다. 카이제곱검정의 정의는 아래와 같다.

$$Chi\_square\_score(f_i) = \sum_{j=1}^{r}\sum_{s=1}^{c}\frac{(n_{js}-\mu_{js})^2}{\mu_{js}},$$

iris data의 기대값이 0.33, 0.33, 0.34 로 균일할 때 p-value가 0.98로 유의성이 없다.

Hide

```
species.type <- table(iris$Species)
species.type.prob <- c(0.33, 0.33, 0.34) # not important
chisq.test(x=species.type, p=species.type.prob)
```

```
    Chi-squared test for given probabilities

data:  species.type
X-squared = 0.029709, df = 2, p-value = 0.9853
```

iris data의 기대값이 0.4, 0.4, 0.2 로 다를 때 p-value가 0.0002로 유의성이 있다.

Hide

```
species.type.prob <- c(0.4, 0.4, 0.2) # important
chisq.test(x=species.type, p=species.type.prob)
```

```
    Chi-squared test for given probabilities

data:  species.type
X-squared = 16.667, df = 2, p-value = 0.0002404
```

2.4.4 Gini Score

통계적으로 차이를 계산할 때 널리 쓰이는 지표이다. 경제불평등 지수 등을 산출할 때 널리 사용된다. class가 0.5/0.5 비율로 존재하고, 인자가 class를 반으로 양분할 경우 0.5의 값을 가지게 될때 좋은 인자라고 할 수 있다. 예를들어 담배를 피운 년수 대한 정보를 x, 기관지 질병에 대한 정보를 y라고 할 때, 담배를 피운 년수의 중위값 미만과 이상을 나누는 지점이 기관지 질병에 대한 정보와 일치할 때 지니지수는 0.5의 값을 가진다. 즉, 인자가 factor를 반으로 나눈다는 것은 해당 인자가 class 정보를 잘 설명한다는 것을 의미 한다. 지니지수는 아래와 같이 계산한다.

$$gini\_index\_score(f_i) = \min_{\mathcal{W}}\left(p(\mathcal{W})(1-\sum_{s=1}^{c}p(C_s|\mathcal{W})^2) + p(\overline{\mathcal{W}})(1-\sum_{s=1}^{c}p(C_s|\overline{\mathcal{W}})^2)\right)$$

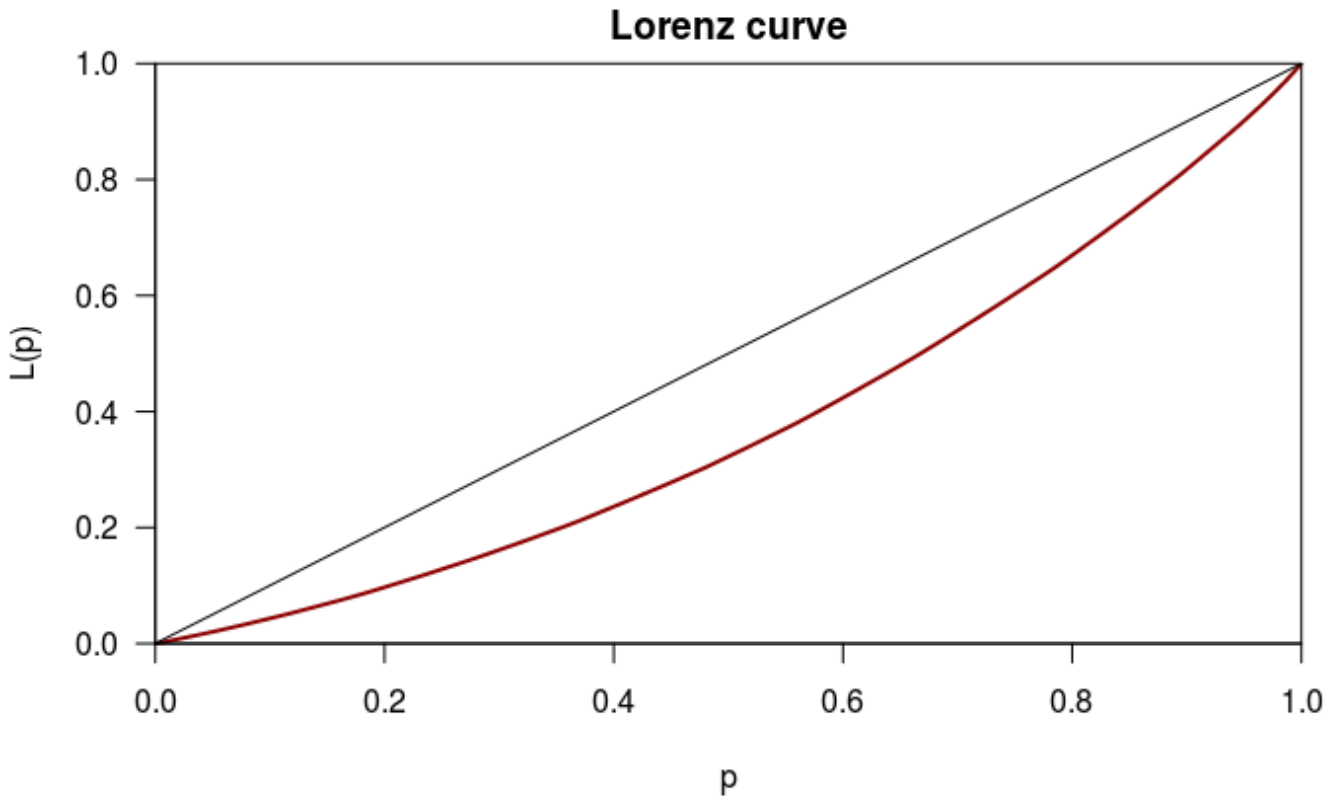
Hide

```
#install.packages("ineq")
library(ineq)
data(AirPassengers)
ineq(AirPassengers,type="Gini")
```

```
[1] 0.2407563
```

Hide

```
plot(Lc(AirPassengers),col="darkred",lwd=2)
```



2.4.5 Correlation-based Feature Selection(CFS)

휴리스틱 방법론으로, 인자 - 인자 간 상관관계가 작을수록, 인자 - class label간의 상관관계가 클수록 좋은 인자라는 가정 하에 인자의 부분집합을 평가한다. 기본적인 CFS 알고리즘은 wrapper 방식 알고리즘에 사용되기도 한다. 아래와 같이 정의된다.

$$CFS\_score(\mathcal{S}) = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}},$$

아래 데이터는 경제지표에 따른 고용 지수를 나타낸 데이터이다.

Hide

```
data(longley) # Predict number of people employed from economic variables
head(longley)
```

	GNP.deflator <dbl>	GNP <dbl>	Unemployed <dbl>	Armed.Forces <dbl>	Population <dbl>	Year <int>	Employed <dbl>
1947	83.0	234.289	235.6	159.0	107.608	1947	60.323
1948	88.5	259.426	232.5	145.6	108.632	1948	61.122
1949	88.2	258.054	368.2	161.6	109.773	1949	60.171
1950	89.5	284.599	335.1	165.0	110.929	1950	61.187

	GNP.deflator <dbl>	GNP <dbl>	Unemployed <dbl>	Armed.Forces <dbl>	Population <dbl>	Year <int>	Employed <dbl>
1951	96.2	328.975	209.9	309.9	112.075	1951	63.221
1952	98.1	346.999	193.2	359.4	113.270	1952	63.639

6 rows

GNP.deflator, GNP, Population, Year 인자가 높은 상관관계를 보였고, Unemployed, Armed.Forces 인자는 낮은 상관관계를 보였다.

Hide

```
ds <- longley[,-7]
cl <- longley[,7]
cor(ds, cl) # 1,2,5,6 인자가 상관관계가 높음
```

```
                 [,1]
GNP.deflator 0.9708985
GNP          0.9835516
Unemployed   0.5024981
Armed.Forces 0.4573074
Population   0.9603906
Year         0.9713295
```

CFS함수를 아래와 같이 구현하였다.

Hide

```
# INPUT / S : subset
# OUTPUT/
CFS_score <- function(ds, cl){
  k <- ncol(ds)
  mean.rcf <- mean(apply(ds, 2, function(x) { cor(x, cl) }))
  mean.rff <- (sum(cor(ds))-k)/2

  return ((k*mean.rcf) / sqrt(k+k*(k-1)*mean.rff))
}
```

GNP.deflator, GNP 인자 부분집합과 Unemployed, Armed.Forces 인자의 부분집합을 비교하였다. 결과는 아래와 같다.

Hide

```
ds1 <- ds[,1:2]
ds2 <- ds[,3:4]
CFS_score(ds1, cl)
```

```
[1] 0.9792864
```

Hide

```
CFS_score(ds2, cl)
```

```
[1] 0.7483062
```

GNP.deflator, GNP 인자 부분집합은 0.74, Unemployed, Armed.Forces 는 0.46 으로 계산되어, 상관관계가 높은 인자들의 부분집합이 더 높은 점수를 얻는것을 확인하였다.

Other Methods

이 외에도 다른 방법들이 있으니 관심이 있으면 찾아보기 바란다.

Hybrid Feature Selection

Deep Feature Selection(DFS)

Convex Principal Feature Selection(CPFS)