Selección de características

Reconocimiento de Patrones – 2013

Selección de características

- Estrategia de búsqueda óptima o sub-optima para la selección de un sub-conjunto de características para el diseño del clasificador.
- Con que propósito?
  - Remover características poco relevantes (puro ruido o “distractores”) para ahorrar tiempo y memoria.
  - Mejorar la performance del clasificador a través de representación más estable, disminuyendo la posibilidad de overfitting.
  - Facilitar visualización y comprensión de los datos

Reducción del número de características

- Difícil establecer relación exacta entre:
  - Error de clasificación
  - *n*= Número de patrones de entrenamiento
  - *d*= Número de características
- En la práctica, puede disminuir el desempeño del clasificador, si el número de patrones de entrenamiento es reducido respecto al número de características (Peaking phenomena)
- Buena práctica: n/d >10

Características relevantes

- Importancia de la selección características relevantes, no solo eficiencia.
- Árboles de decisión y los K-NN son clasificadores sensibles a las características irrelevantes o ruido.
- Bayes Naive es sensible a las características redundantes.

Relevancia vs utilidad

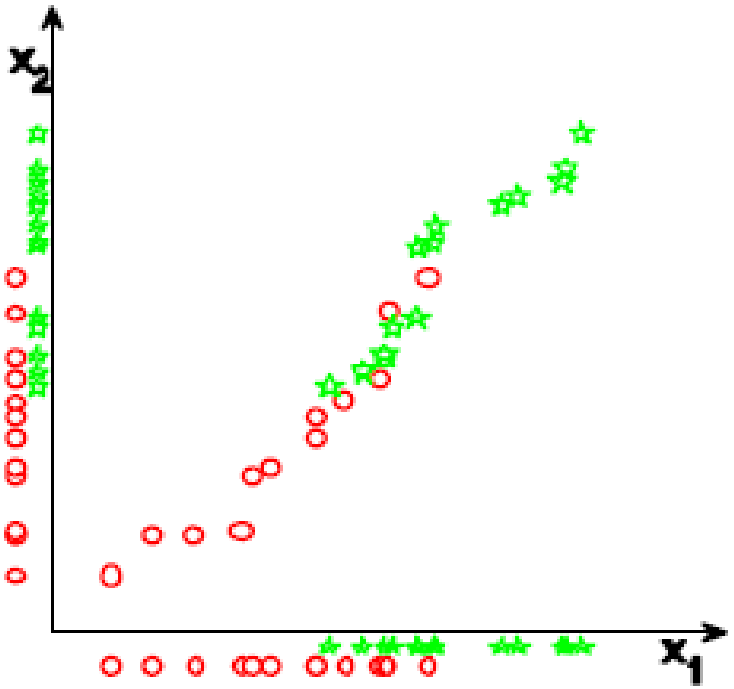


Fig. Guyon

Relevancia: existencia de dependencia entre las características y las etiquetas de clasificación. Ej medida de dependencia: información mutua, correlación entre medida y clases.

Útil: Si al agregarla al subconjunto de otras mejora desempeño.

¿Como impacta la correlación la redundancia?

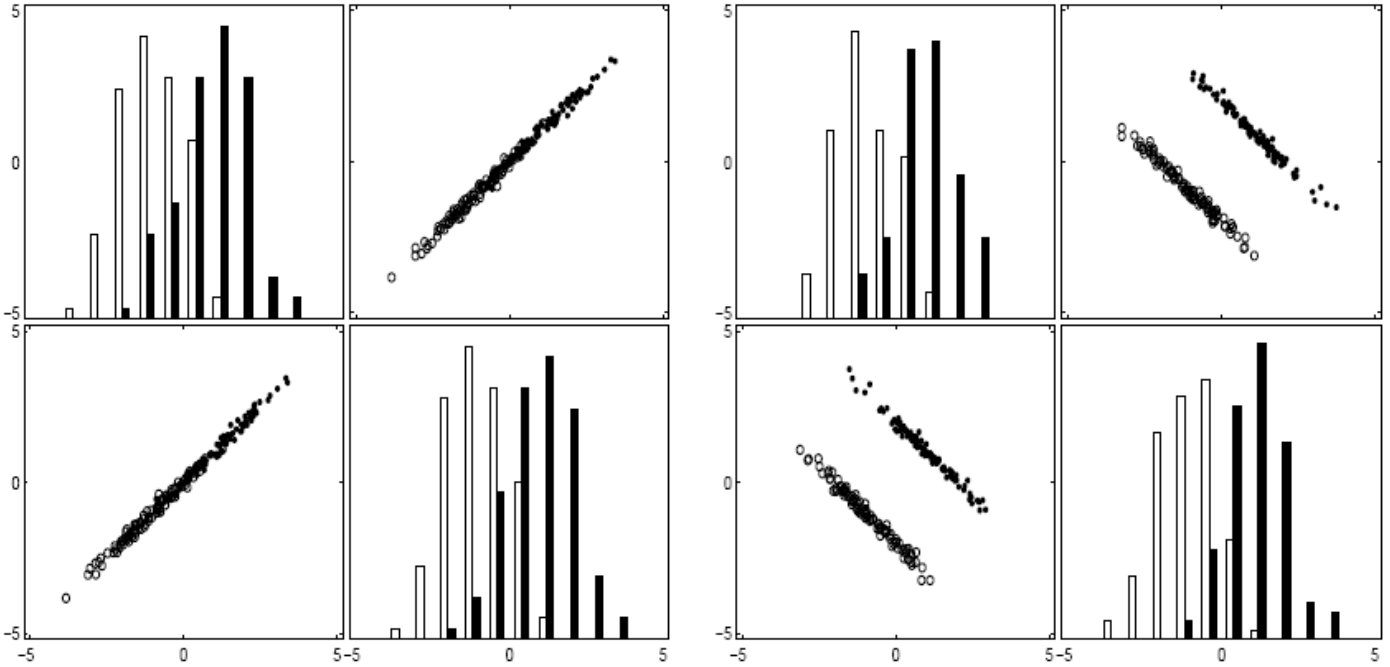


Figure 2: **Intra-class covariance.** In projection on the axes, the distributions of the two variables are the same as in the previous example. (a) The class conditional distributions have a high covariance in the direction of the line of the two class centers. There is no significant gain in separation by using two variables instead of just one. (b) The class conditional distributions have a high covariance in the direction perpendicular to the line of the two class centers. An important separation gain is obtained by using two variables instead of one.

Guyon

¿Como impacta la correlación la redundancia?

- Si dos variables son perfectamente correlacionadas, son verdaderamente redundantes en el sentido que no se agrega información adicional por agregarlas.
- Si dos variables son muy altamente correlacionadas (o anticorrelacionadas) no indican ausencia de complementariedad.

¿Puede una variable que es poco útil por si misma ser útil con otras?

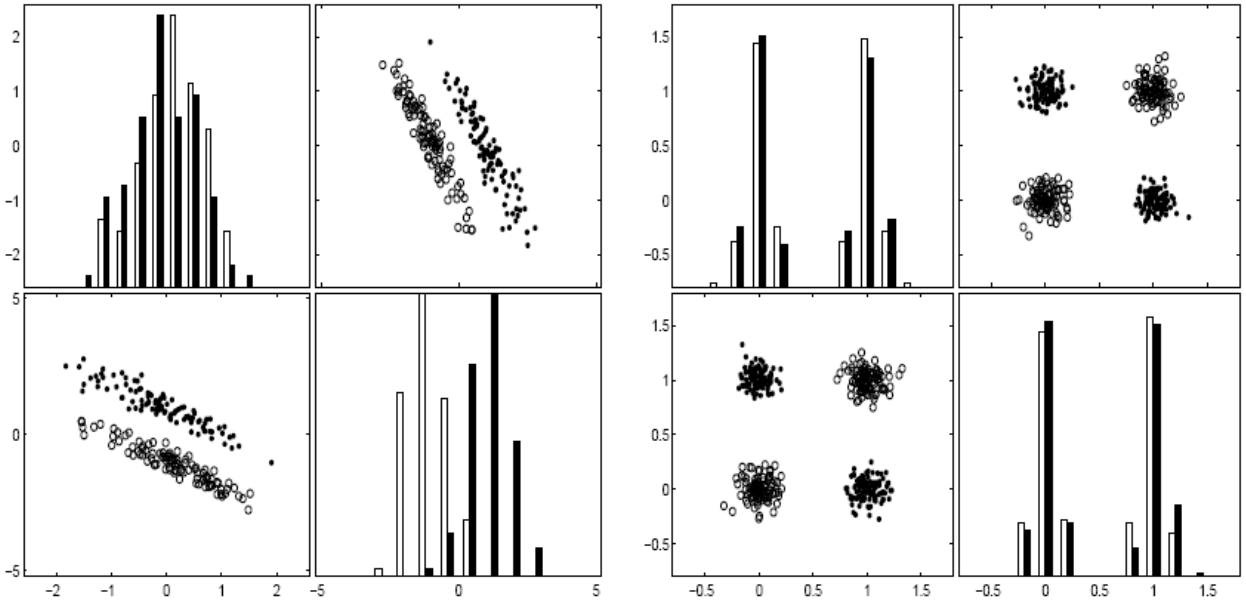


Figure 3: **A variable useless by itself can be useful together with others.** (a) One variable has completely overlapping class conditional densities. Still, using it jointly with the other variable improves class separability compared to using the other variable alone. (b) XOR-like or chessboard-like problems. The classes consist of disjoint clumps such that in projection on the axes the class conditional densities overlap perfectly. Therefore, individual variables have no separation power. Still, taken together, the variables provide good class separability .

Guyon

¿Puede una variable que es poco útil por si misma ser útil con otras?

- Una variable que es completamente inútil por si misma puede generar una mejora en el desempeño tomada con otras.
- Dos variables que son poco útiles ambas pueden ser útiles juntas.

¿Cómo seleccionar?

- ¿Que criterios?
- ¿relevancia o utilidad?

- ¿Cómo ?
- ¿características individuales o sub-conjuntos de características?

Reducción de la dimensionalidad

- Selección de características (espacio muestral)
  - Descartar las que no contribuyen separabilidad
  - Función de mérito *J*: criterio de discriminación
    (ej: distancia, disimilitud, ganancia de información, etc.)

Dado un conjunto $(x_1, x_2 \ldots . x_p)$,

χ_d : conjunto de todos los subconjuntos de tamaño d

$$J(\tilde{X}_d) = \max_{X \in \chi_d} J(X)$$

Optimizo sobre todos los conjuntos posibles con d caracteristicas tomadas de las p.

Reducción de la dimensionalidad

- Extracción de características (espacio transformado)
- Encontrar una transformación que lleva el espacio de medidas p en un espacio de características de dimensión menor.
  - Combinación lineal o no lineal
  - Supervisada o no supervisada
  - ***A***: conjunto de transformaciones posibles
  - $$J(A^*) = \max_{A \in \boldsymbol{A}} J(A(x))$$
  - Optimizo sobre las transformaciones posibles
  - Nuevas características $y = A^*(x)$

Enfoques de selección

- Filtrado (filtering): Selecciona las características en forma independiente del clasificador, usando un criterio de "relevancia" .
- Encapsulado (wrapping): Selecciona los subconjuntos de características en función del desempeño de un clasificador. Costoso computacionalmente. Necesita estrategia de búsqueda para explorar en forma eficiente el espacio de subconjuntos.
- Intrínseco (embedding): Realizan la selección en el proceso de aprendizaje devuelve un subconjunto de características y el clasificador entrenado. n entrenamientos, evalúo costo de agregar o quitar característica pero no reentreno.

Embedded con SBS

- Comienzo con todas las características
- Entreno el clasificador con el subconjunto actual minimizando la función de costo J(f).
- Para cada característica (remanente) x_i estimo sin reentrenar f. el cambio en J(f) al remover x_i.
- Remuevo la característica que resulta en la menor degradación de J.

Evaluación de características:

- Individual: bondad de cada característica. Económico, puede haber redundancia. Puede usarse como primer paso y luego uno que tenga en cuenta redundancia.

$J(x_1) \geq J(x_2) \geq \ldots J(x_p)$

$x_1, x_2 \ldots . x_d \ldots . x_p$

selección

- Subconjuntos: Recorro el espacio de características formando subconjuntos y valoro los subconjuntos. Más costoso pero mejores resultados.

→ Estrategias de búsqueda y criterios de evaluación

Algoritmos de búsqueda - Selección

- Búsqueda exhaustiva: Evaluar todas las posibles combinaciones de d características tomadas de un conjunto de *p*, se selecciona la combinación para la que J(X) es óptimo:

$$n_d = \frac{p!}{(p-d)!d!}$$

$d=10, p=25\ n_d = 3.268.760$

  - Impracticable incluso con *d* y *p* moderados.
- Búsquedas óptimas:
  - Exhaustiva, solo problemas pequeños.
  - Branch and Bound, Simulated anneling, algoritmos genéticos.

Branch and Bound

- Procedimiento óptimo que no implica búsqueda exhaustiva.
- Top-Down : comienzo con p características y construyo árbol borrando en forma sucesiva características
- Se basa en usar funciones J monótonas: si $X \subset Y \rightarrow J(X) < J(Y)$

Ejemplo: $(x_1,x_2....x_5)$ p=5 d=3

- Árbol cuyos nodos representan todos los subconjuntos de cardinalidad 3, 4 y 5.
- Evitar repetidos: se quitan variables en orden creciente.
- Ejemplo libro Webb: “Statistical Pattern Recognition” pag 313.

Branch and Bound

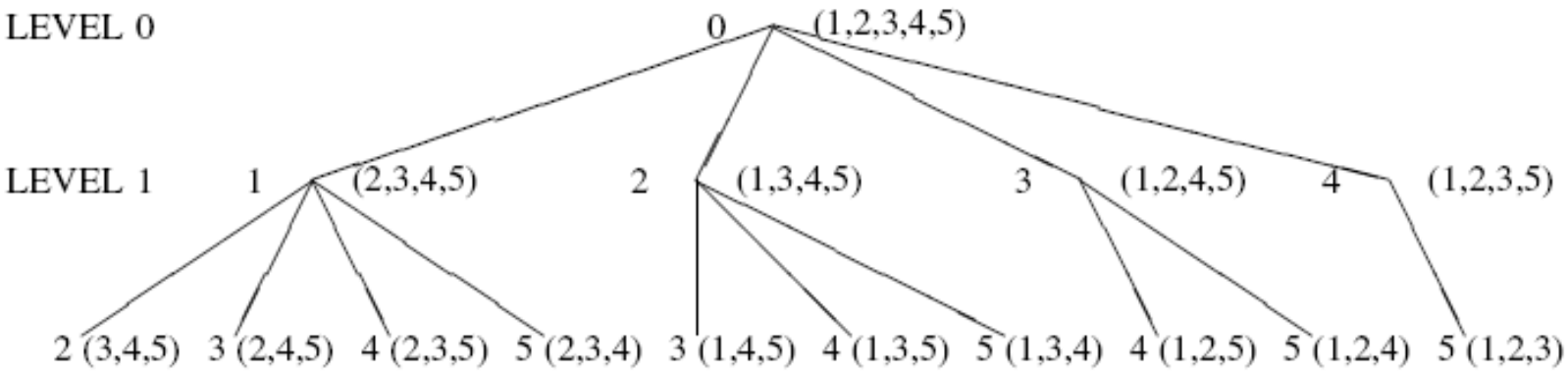
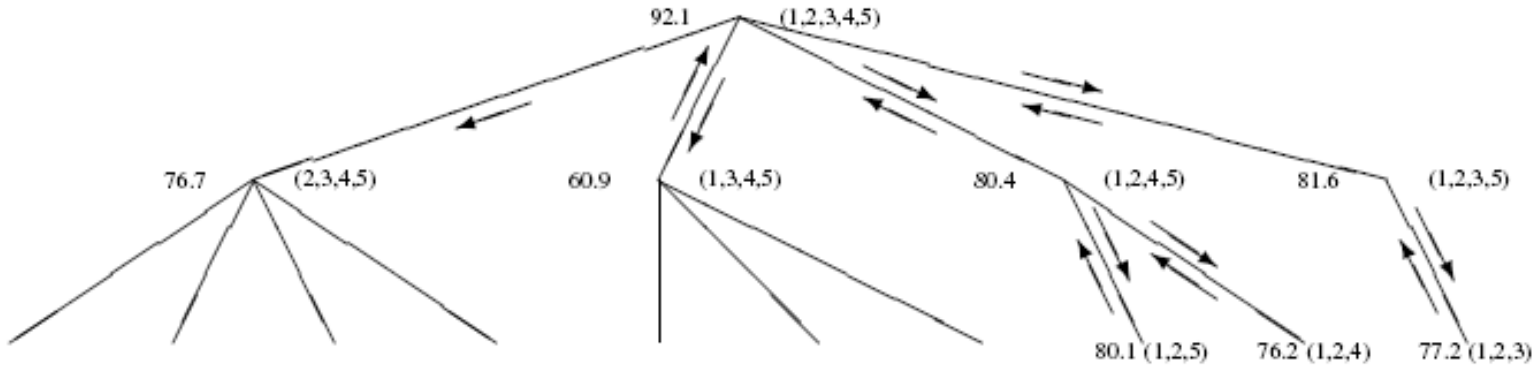


Figure 9.2 Tree figure for branch and bound method



Búsqueda Óptima

- Se evalúa *J* desde la parte menos densa a la más y se retiene el máximo
- Con el valor de *J* para 4 características tenemos una cota para los conjuntos de 3 características, no es necesario visitarlos todos. La solución es óptima por la monotonía
- Eficiencia : sensible al orden. Se remueven en orden creciente para no repetir cálculos. Árbol no simétrico.
- Puede no ser viable computacionalmente o no cumplirse monotonía.

Búsquedas Sub- óptimas

- Heurísticas y no-determinísticas
- Mejores características individuales: Estimo el poder de discriminación para cada característica individual del conjunto original.

$J(x_1) \geq J(x_2) \geq \ldots J(x_p)$ y selecciono las d mejores $\{x_i / i \leq d\}$

- Si las características originales están muy correlacionadas da un conjunto subóptimo. Aún con independencia no garantiza optimalidad.

Selección Secuencial:

- SFS :Forward, conjunto creciente – agrego 1 en c/paso
- SBS: Backward, conjunto decreciente –quito1 en c/paso.

Búsquedas Sub- óptimas

$J_j = J(X_{k-1} + x_j)$: se elige x_j que maximiza J →

$X_k = X_{k-1} + x_{max}$

- Inicio: Conjunto vacio (SFS) o completo (SBS)
- Parada: Si no se logra mejorar el conjunto anterior con ninguna característica o k=d.
- Denominaciones : SFS - “hill climbing” y “greedy step wise”

Comparación SFS y SBS

- SFS cerca optimalidad en conjuntos pequeños. SBS en grandes.
- Desventajas: evalúan una sola característica a la vez y no hay mecanismos para quitar características. Debido a la correlación pueden construir conjuntos poco óptimos
- Se puede incorporar un sesgo hacia conjuntos más pequeños, por ejemplo un umbral de incremento mínimo para continuar SFS.
- Ejemplo : $J(x_j + x_k) > J(x_i + x_k)$

  $J(x_i) > J(x_j) > J(x_k)$

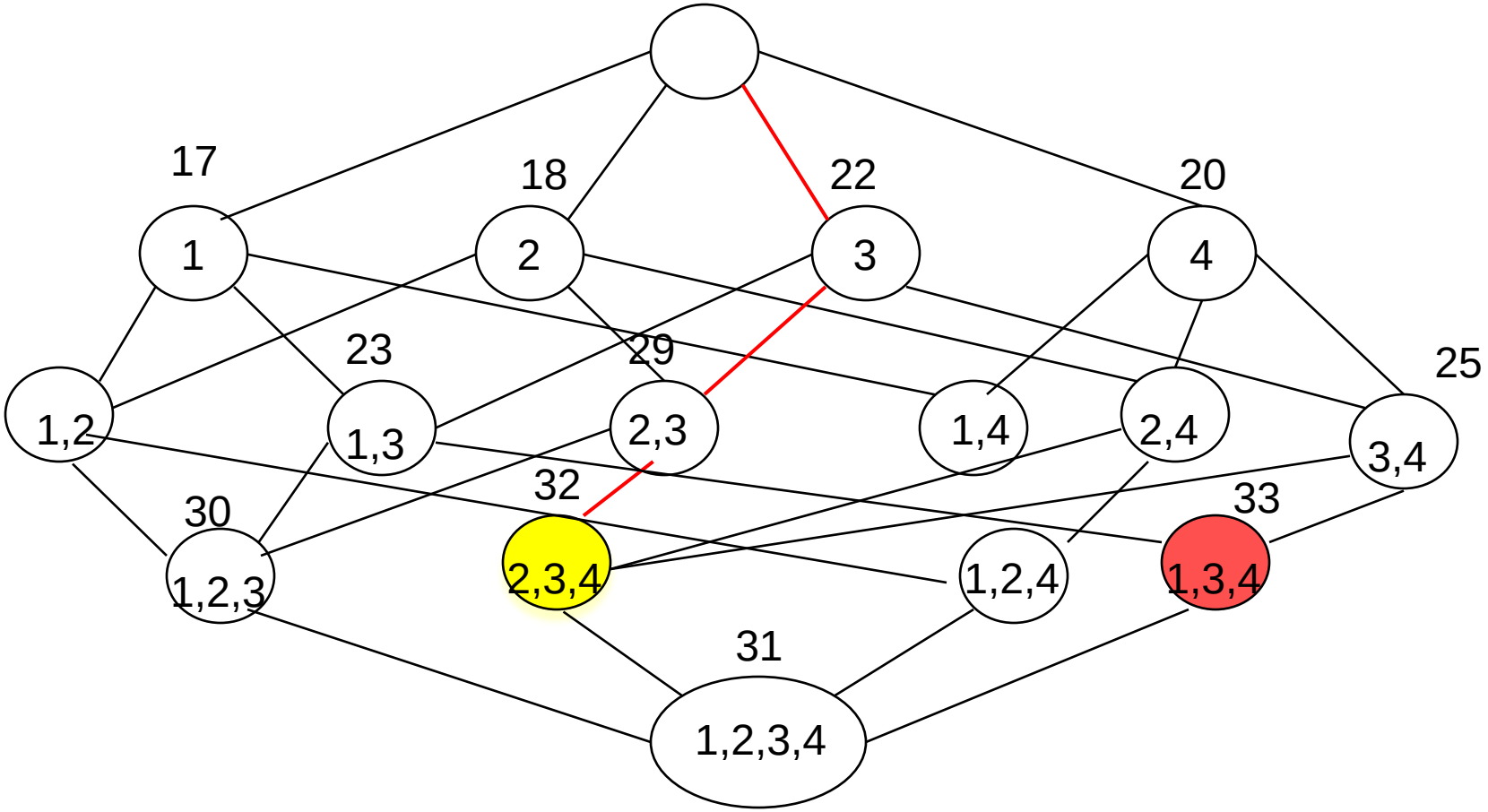
  $J(x_i + x_k) > J(x_i + x_j)$

Variantes de la selección secuencial

- Selección secuencial generalizada: Se agregan o quitan r características a la vez. Toma en cuenta la correlación en cierta forma.
- *+l –r*: Se agregan l características con SFS y se quitan r con SBS. Permite marcha atrás. Variantes en las que l y r varían a lo largo de la búsqueda (floating SBFS, SFFS). SFFS desempeño cercano Branch and Bound.

- Best First (Weka): No termina al dejar de mejorar desempeño sino que mantiene registro de los mejores subconjuntos y regresa a considerar conjuntos previos (backtrack).
- Se incrementa el costo computacional entre 2 y 10 veces.

Ejemplo SFS vs Best First



Ejemplo SFS vs Best First

- SFS : se detiene al evaluar 1,2,3,4 porque no mejora respecto 2,3,4 (el mejor hasta el momento), cae en un máximo local.
- Best First: mantiene lista de nodos
  - Best: mejor nodo hasta el momento
  - Open: nodos con hijos no explorados (en orden)
  - Closed: nodos sin hijos no explorados.
- No se detiene cuando decrece el desempeño, sino que retoma el mejor de los nodos que se mantiene abierto. Menos propenso a caer un óptimo local. Explora todo el espacio sino se uso criterio de parada.

Criterios de evaluación

- Evaluación individual:
  - Ganancia de información
  - Razón de ganancia de información
- Evaluación de subconjuntos
  - Error de clasificación usando determinado clasificador
  - Distancia de Mahalanobis entre clases
  - Basada en correlación

Ganancia de información

- Se utiliza como función de mérito la Ganancia de información (decremento de entropía). Idem construcción de árboles selecciono la característica que produce el mayor decremento de impureza.

$$J(x_i) = G_{info}(x_i) = H(c) - H(c/x_i)$$

$$J(x_1) \geq J(x_2) \geq \ldots\ldots J(x_d) \geq \ldots\ldots J(x_p)$$

- Se usa MDL (Minimum Description Length) para detener el proceso: Contabilizar la información necesaria para describir el modelo $\log_2(N-1)$ bits, más la información necesaria para describir los datos dado el modelo.

Razón de ganancia de información (info gain ratio)

$$J(x_i) = \frac{G_{info}(x_i)}{H(x_i)} = \frac{H(c) - H(c/x_i)}{H(x_i)}$$

- Ganancia de información tiene sesgo hacia características que toman más valores?(Quinlann 1982)

$A: A_1, A_2, A_3 \ldots\ldots A_i \ldots\ldots A_B$ $\qquad$ $A': A_1, A_2, A_3 \ldots\ldots A'_i\, A'_{ii} \ldots\ldots A_B$

$H(A_i) \leq H(A'_i) + H(A'_{ii})$ $\qquad$ $H(A) \leq H(A')$

- Razón de ganancia de información puede sobrecompensar y preferir características solo porque tiene información intrínseca mucho menor que las otras

Evaluación de subgrupos

- Error de clasificación

  $J=(1-Pe)$ *Pe:* error de clasificación

- La selección depende del clasificador, pero también del tamaño de la muestra de entrenamiento y test
- Distancia de Mahalonobis entre clases: cumple la propiedad de monotonía

$$c = 2 \qquad \mu_1, \mu_2 \qquad \Sigma_1, \Sigma_2$$

$$J = (\mu_1 - \mu_2)^T \frac{(\Sigma_1 + \Sigma_2)^{-1}}{2} (\mu_1 - \mu_2)$$

- Si c>2 se pueden considerar las sumas sobre todos los pares de clases.

Basados en correlación

- Alta correlación con la clase y poca correlación entre sí.
- Evaluamos la información mutua entre características y las clases.
- Características nominales (o se discretizan usando Ginfo y MLD) Fayyad 1992

$$X = \{A, B, ...F\} \quad Y = \{1,2,3\}$$

A	B	C	D	E	F
1	1	1	2	3	3
1	2	1	2	3	3
1			2	2	3

$$G_{info}(Y, X) = H(Y) - H(Y / X)$$

$$H(Y) = -\sum_{y \in R_y} p(y) \log_2 p(y)$$

$$H(Y / X) = -\sum_{x \in R_x} p(x) \sum_{y \in R_y} p(y / x) \log_2 p(y / x)$$

Evaluación

Se normaliza para evitar sesgo de G_{info}

$$U(X,Y) = \frac{2G_{info}(X,Y)}{H(X)+H(Y)} = \frac{2(H(Y)-H(Y/X))}{H(X)+H(Y)}$$

$$U(X,Y) = \frac{2(H(X)-H(X/Y))}{H(X)+H(Y)}$$

Coeficiente simétrico de incertidumbre (normalizado entre [0,1]

Evaluación

$$J_{CBFS}(S) = \frac{\sum_j U(x_j, C)}{\sqrt{\sum_j \sum_i U(x_j, x_i)}} \quad \forall i, j \in S$$

- Numerador: correlación media entre las características de S y la clase (capacidad de predecir clase)
- Denominador: intercorrelación media entre las características de S (redundancia)
- Características irrelevantes: pobre predicción
- Características redundantes: correladas con una o varias características de S

Filters

- Criterio: Medida de "relevancia" de características.
- Búsqueda: Usualmente ranking de características individuales o de subconjuntos anidados de características.
- Evaluación: Usa test estadísticos
- Resultados:
  - Son relativamente robustos frente a overfitting
  - Pueden equivocarse al seleccionar las características más útiles. Criterio de separabilidad puede llevar estimaciones pobres de discriminabilidad.
  - Ranking: Implementación sencilla, escalable, resultados razonables.

Wrappers

- Criterio: Medida de utilidad de subconjunto de características.
- Búsqueda: Sobre todo el espacios de subconjuntos
- Evaluación: Usa validación cruzada.
- Resultados:
  - Puede en principio encontrar el subconjunto más “útil”.
  - Tienden a generar overfitting

Embedded

- Criterio: Medida de utilidad de subconjunto de características.
- Búsqueda: Guiada por el proceso de aprendizaje
- Evaluación: Usa validación cruzada.
- Resultados:
  - Similares a Wrappers pero menos costos y con menor tendencia a generar overfitting.

Comparación de complejidad

Método	Número de subconjuntos evaluados	Complejidad
Búsqueda exhaustiva wrapper	2^d	d
Búsqueda sub-óptima (SFS- SBF) wrapper	d (d+1)/2	log d
Ranking de características o métodos embedded	d	log d