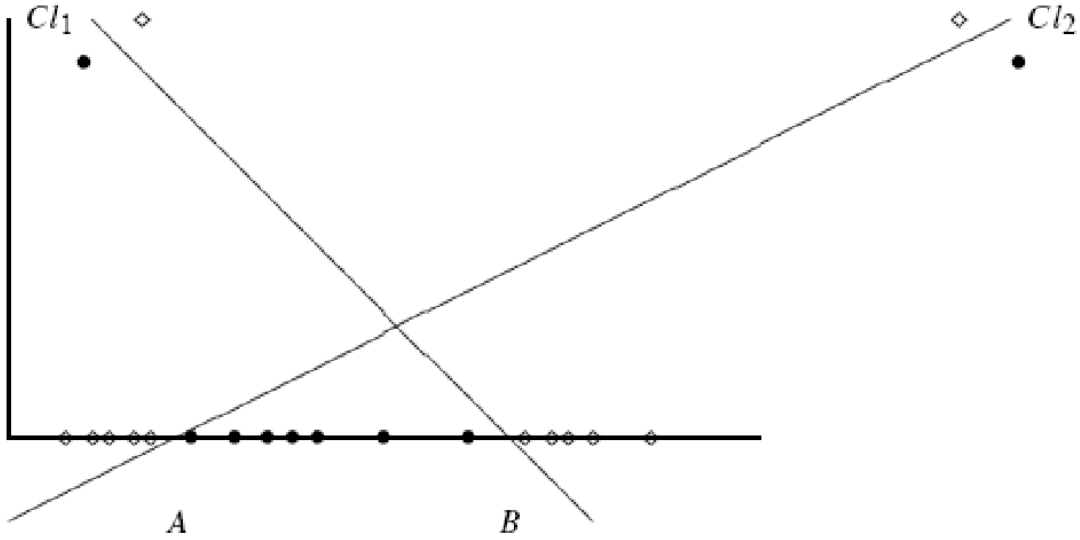
METACLASIFICADORES

- Teorema *No free lunch* (Wolpert y MacReady, 1996):
  - No hay un algoritmo de aprendizaje que en cualquier dominio siempre induzca el clasificador más preciso
- Optar por una combinación de opiniones de expertos (modelos) antes de tomar una decisión final
- Cada paradigma tiene asociado una región de decisión de un determinado tipo
- Al combinar paradigmas tratar de obtener la región de decisión idónea para el problema

METACLASIFICADORES

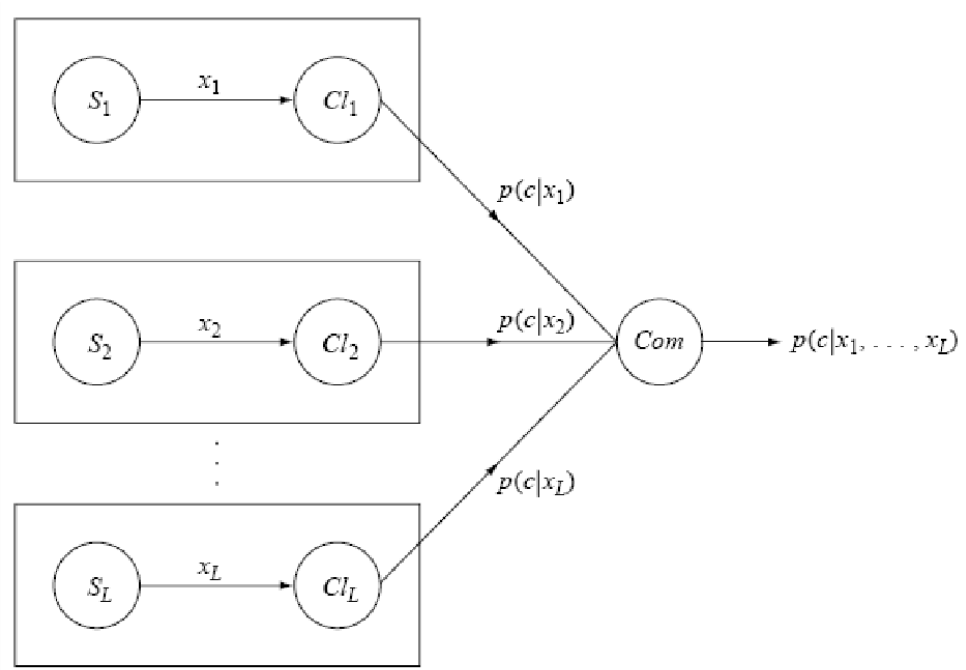
- Ejemplo:



  - Datos con fronteras complejas de decisión
  - Ambos clasificadores (Cl_1 y Cl_2) cometen fallos en la clasificación
    - Cl_1:"•" a la izquierda; "o" a la derecha
    - Cl_2: "•" abajo; "o" arriba
  - Las muestras mal clasificadas son distintas en cada uno
    - Dan información complementaria: al combinarlos son útiles
  - Se pueden combinar con la regla:
    - "•" si tanto Cl_1 como Cl_2 predicen "•"; "o" en otro caso

METACLASIFICADORES

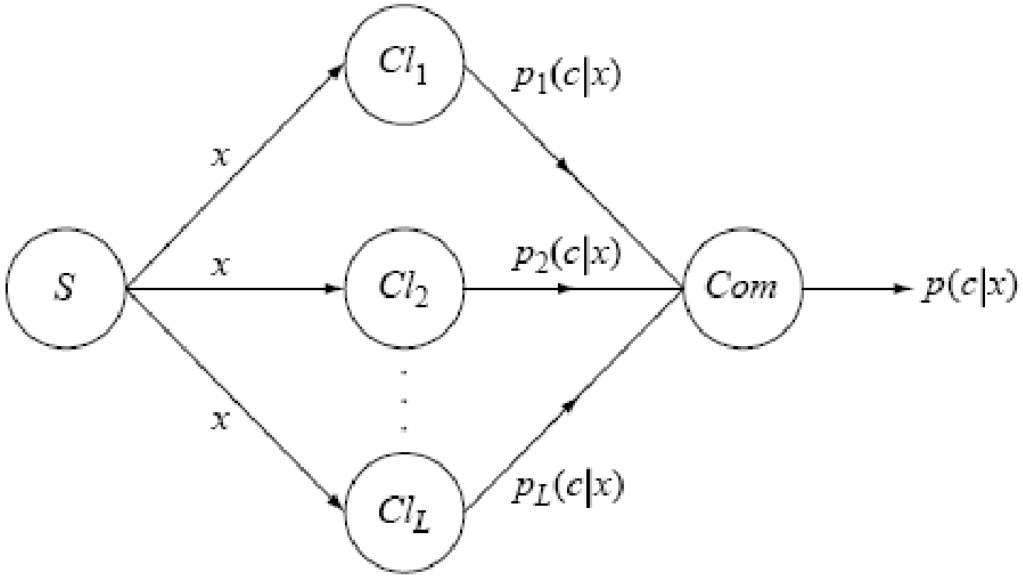
- Espacios de diferentes características:



  - Com es la regla de combinación
    - Es, a su vez, un clasificador que combina las probabilidades para estimar $p(c|x_1,\ldots,x_L)$
  - ¿Cuál es la mejor regla de combinación?

METACLASIFICADORES

- Espacios de iguales características:



  - Clasificadores pueden ser de distinto tipo
  - O del mismo tipo pero entrenados con distintas muestras: aleatorias (*bagging*), etc.
  - Del mismo tipo pero con parámetros distintos

METACLASIFICADORES

- Se pueden combinar:
  - Clasificadores distintos
  - Del mismo tipo, pero
    - Con distintos parámetros
    - Entrenados con distintas muestras
    - O entrenados distintas veces (si no determinísticos)
- Importante:
  - Precisión de cada uno:
    - Tasa de error de cada uno menor que el azar
  - Diversidad de cada uno:
    - Cometen errores distintos al predecir una clase
    - Si todos producen salidas idénticas, no se gana al combinarlos

METACLASIFICADORES

- Que cada clasificador sea razonablemente preciso
  - Pero no muy preciso individualmente (no hay que optimizarlo por separado para que optimice su comportamiento)
- Se escogen por su simplicidad
- Cada uno sea preciso en instancias diferentes, especializándose en subdominios del problema
- Al combinarlos disminuye el error esperado porque disminuye su componente de la varianza (debida al conjunto de entrenamiento particular con que se construyó el modelo, que puede no ser totalmente representativa de la población)

METACLASIFICADORES

- Métodos básicos
  - Fusión de etiquetas
  - Fusión de salidas continuas
  - *Stacking*
  - Cascada
- Métodos avanzados:
  - *Bagging*
  - Aleatorización
  - *Boosting*
  - Híbridos
- Métodos más comunes:
  - Random Forest
  - XGBoost

METACLASIFICADORES

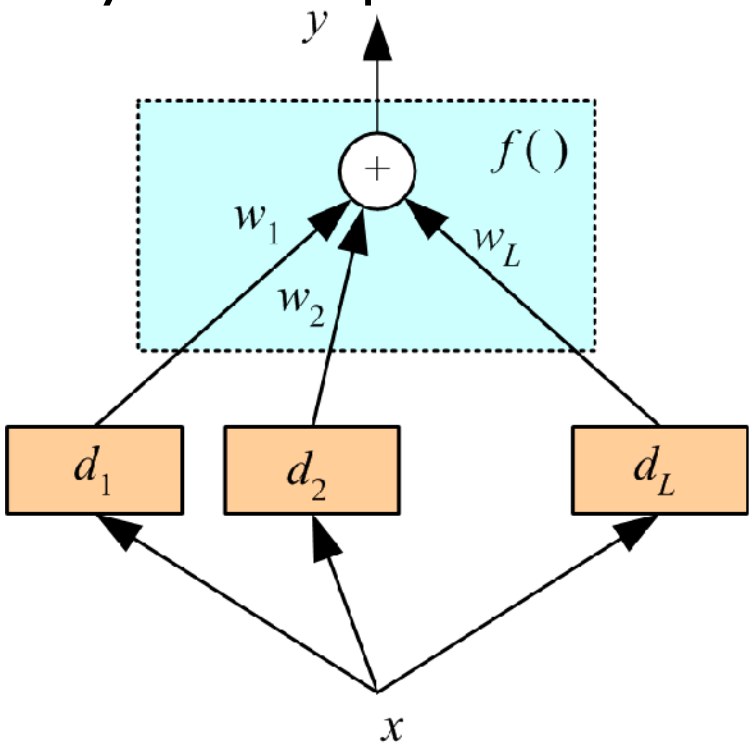
- **Métodos básicos**
  - **Fusión de etiquetas**
    - Voto por mayoría
    - Mayoría absoluta
    - Voto por mayoría con umbral
    - Voto por mayoría con peso
  - Fusión de salidas continuas
  - *Stacking*
  - Cascada
- Métodos avanzados:
  - *Bagging*
  - Aleatorización
  - *Boosting*
  - Híbridos
- Métodos más comunes:
  - Random Forest
  - XGBoost

METACLASIFICADORES

- Métodos básicos: Fusión de etiquetas:
  - Dados varios clasificadores, que clasifican la instancia x en distintas clases...
    - Voto por mayoría
      - Se asigna la clase más con más votos
    - Mayoría absoluta
      - La clase debe conseguir al menos la mitad de los votos más uno
    - Voto por mayoría con umbral
      - La clase debe de conseguir un mínimo de votos
    - Voto por mayoría con peso
      - Se pondera el voto de cada clasificador
        - Generalmente, según la bondad
        - Se suelen normalizar estas ponderaciones

METACLASIFICADORES

- Métodos básicos: Fusión de etiquetas:
  - Dados varios clasificadores, que clasifican la instancia x en distintas clases...
    - Voto por mayoría con peso



METACLASIFICADORES

- Métodos básicos
  - Fusión de etiquetas
  - **Fusión de salidas continuas**
  - *Stacking*
  - Cascada
- Métodos avanzados:
  - *Bagging*
  - Aleatorización
  - *Boosting*
  - Híbridos
- Métodos más comunes:
  - Random Forest
  - XGBoost

METACLASIFICADORES

- Métodos básicos: Fusión de salidas continuas:
  - Dados varios clasificadores, que asignan la pertenencia de la instancia x en una determinada clase (de varias clases posibles) de forma numérica (como una RNA sin establecer umbral)
    - Ejemplo: la clasificación de la instancia x dada por 5 clasificadores distintos en cada una de las 3 posibles clases es:

	Clase 1	Clase 2	Clase 3
Clasificador 1	0.1	0.5	0.4
Clasificador 2	0	0	1
Clasificador 3	0.4	0.3	0.4
Clasificador 4	0.2	0.7	0.1
Clasificador 5	0.1	0.8	0.1

    - Estos valores pueden ser combinados para producir una única salida de distintas maneras, como por ejemplo:

METACLASIFICADORES

- Métodos básicos: Fusión de salidas continuas:

	Clase 1	Clase 2	Clase 3
Media aritmética	0.16	0.46	0.4
Mínimo	0	0	0.1
Máximo	0.4	0.8	1
Mediana	0.1	0.5	0.4
Trimmed media	0.1	0.8	0.1
Producto	0	0	0.0016

- *Trimmed* media:
  - Realizar la media quitando % extremo
    - En este ejemplo, se han eliminado los dos extremos

METACLASIFICADORES

- Métodos básicos
  - Fusión de etiquetas
  - Fusión de salidas continuas
  - ***Stacking***
  - Cascada
- Métodos avanzados:
  - *Bagging*
  - Aleatorización
  - *Boosting*
  - Híbridos
- Métodos más comunes:
  - Random Forest
  - XGBoost

METACLASIFICADORES

- Métodos básicos: *Stacking* o apilado (Wolpert, 1992):
  - Votando no deja claro en qué clasificador confiamos
    - *Stacking* intenta aprender qué clasificadores son los fiables descubriendo cómo combinar mejor sus salidas
  - *Stacking* construye otro clasificador con las salidas de los clasificadores base, en vez de votar
  - Menos usado que *baggaging* y *boosting*
  - Generalmente utilizado para combinar **clasificadores de distinto tipo** para que se complementen

METACLASIFICADORES

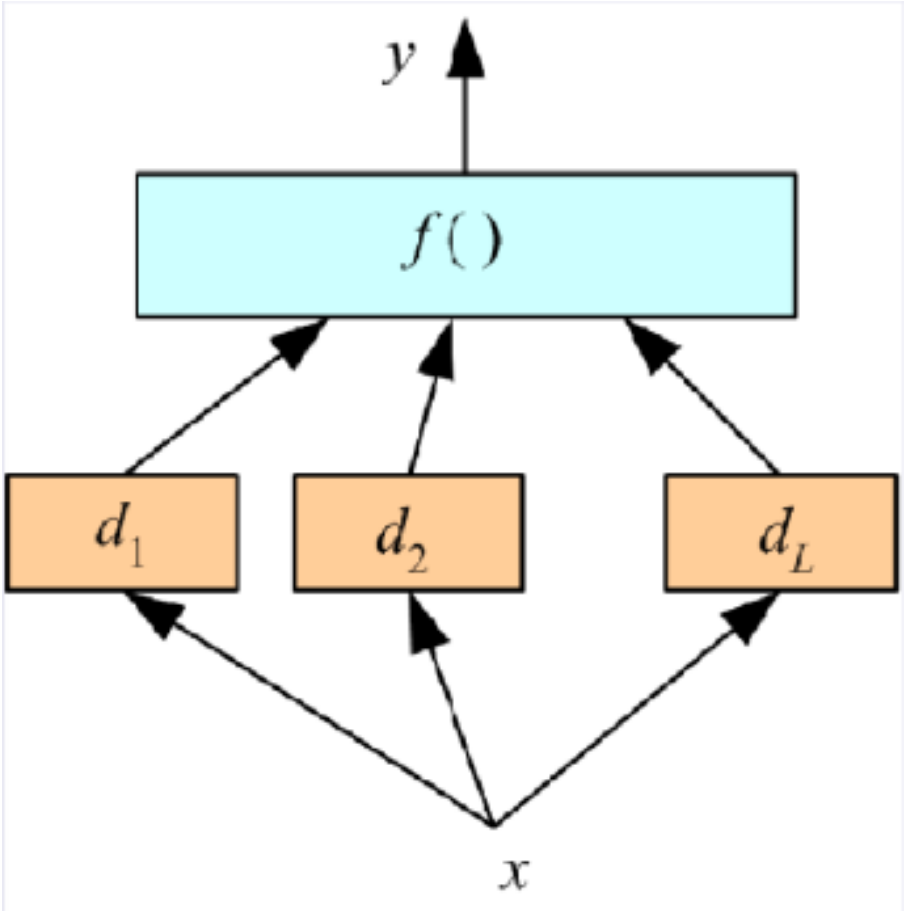
- Métodos básicos: *Stacking* (Wolpert, 1992):
  - Idea general:
    - Diversas capas (o niveles) de clasificadores
      - Cada capa utiliza los resultados obtenidos por la capa anterior
    - La última capa está formada por un único clasificador que es el que toma la decisión final
    - Los clasificadores utilizados constituyen el nivel 0
    - Las salidas (predicciones) de estos modelos son la entrada al metamodelo: nivel 1.
      - Una instancia de nivel 1 tiene tantas características como clasificadores base haya

METACLASIFICADORES

- Métodos básicos: *Stacking* (Wolpert, 1992):
  - Funcionamiento:
    - Llega una instancia
    - Cada modelo de nivel 0 predice la clase
    - Las predicciones entran al clasificador de nivel 1, que las combina y genera la predicción final

METACLASIFICADORES

- Métodos básicos: *Stacking* (Wolpert, 1992):



METACLASIFICADORES

- Métodos básicos: *Stacking* (Wolpert, 1992):
  - ¿Cómo entrenar ese clasificador de nivel 1?
    - No deben usarse los mismos datos con que se entrenaron los del nivel 0, porque precisamente se intenta corregir sus sesgos, aprender cómo cometen errores, y pueden tener memoria del conjunto de entrenamiento
    - Puede utilizarse *holdout* o un *k-fold cross-validation*

METACLASIFICADORES

- Métodos básicos: *Stacking* (Wolpert, 1992):
  - Por ejemplo: *4-fold cross-validation* (1/2)
    - Se divide el conjunto de patrones en D_1, D_2, D_3, D_4

Entrenamiento	$\{D_1, D_2, D_3\}$	$\{D_2, D_3, D_4\}$	$\{D_3, D_4, D_1\}$	$\{D_4, D_1, D_2\}$
Test	D_4	D_1	D_2	D_3

    - Se pasa este esquema a todos los clasificadores y se obtiene 4 versiones de cada uno:
      - Entrenando con $\{D_1, D_2, D_3\}$, y haciendo test con D_4
      - Entrenando con $\{D_2, D_3, D_4\}$, y haciendo test con D_1
      - Entrenando con $\{D_3, D_4, D_1\}$, y haciendo test con D_2
      - Entrenando con $\{D_4, D_1, D_2\}$, y haciendo test con D_3

METACLASIFICADORES

- Métodos básicos: *Stacking* (Wolpert, 1992):
  - Por ejemplo: *4-fold cross-validation* (2/2)
    - Se entrena el metaclasificador con todas las instancias.
      - Para cada instancia,
        - Por ejemplo, si la instancia está en D_1, se toman las salidas de los clasificadores entrenados en $\{D_2, D_3, D_4\}$
      - Para no usar datos vistos en el entrenamiento del nivel 0, y que el nivel 1 refleje el rendimiento de los del nivel 0
  - Válido para salidas categóricas (etiquetas de clase) o probabilidades (a cada clase)

METACLASIFICADORES

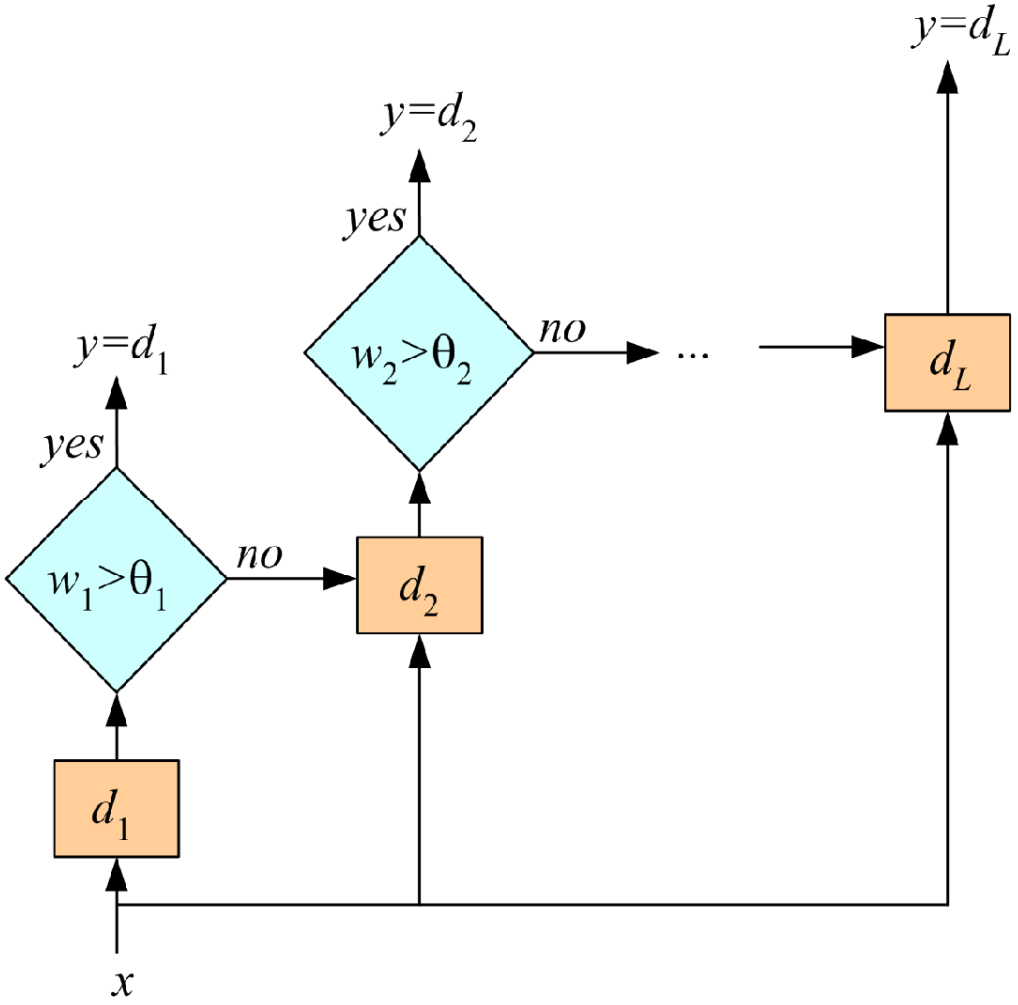
- Métodos básicos
  - Fusión de etiquetas
  - Fusión de salidas continuas
  - *Stacking*
  - **Cascada**
- Métodos avanzados:
  - *Bagging*
  - Aleatorización
  - *Boosting*
  - Híbridos
- Métodos más comunes:
  - Random Forest
  - XGBoost

METACLASIFICADORES

- Métodos básicos: Combinación en cascada:
  - Se tiene una secuencia de clasificadores ordenados
    - **Distintos modelos**
    - Se usa el clasificador j para la instancia x si los clasificadores previos no han proporcionado la suficiente confianza (probabilidad superior a cierto umbral) al clasificarla
  - Proceso:
    - Entrenar el clasificador d_1 y validarlo con otro conjunto
    - Las instancias en las que d_1 no es fiable (tanto las que no pasan el umbral como las mal clasificadas) entrenan d_2, etc.

METACLASIFICADORES

- Métodos básicos: Combinación en cascada:



METACLASIFICADORES

- Métodos básicos: Combinación en cascada:
  - Se utilizan sucesivamente clasificadores más complejos porque uno simple clasificará bien la mayoría de instancias y uno complejo sólo se usará para un pequeño porcentaje
  - Para no incrementar mucho el número de clasificadores, las pocas instancias que no pasen ninguna prueba se almacenan como están y son tratadas por un clasificador no paramétrico como el k-NN
  - Las clases pueden explicarse por un pequeño conjunto de reglas de complejidad creciente, con un pequeño conjunto adicional de excepciones que estas reglas no cubren (mejor manejadas por métodos no paramétricos)

METACLASIFICADORES

- Métodos básicos
  - Fusión de etiquetas
  - Fusión de salidas continuas
  - *Stacking*
  - Cascada
- **Métodos avanzados:**
  - ***Bagging***
  - Aleatorización
  - *Boosting*
  - Híbridos
- Métodos más comunes:
  - Random Forest
  - XGBoost

METACLASIFICADORES

- Métodos avanzados: *Bagging* (Breiman, 1996):
  - Bagging = Bootstrap AGGregatING
  - Se combinan clasificadores obtenidos de réplicas *bootstrap* del conjunto de patrones
    - A partir del conjunto de patrones, se crean varios conjuntos bootstrap, que se utilizan para entrenar distintos clasificadores
      - El mismo tipo de clasificadores
  - Para generar la salida se selecciona la clase más frecuentemente predicha
  - Si el tamaño muestral es muy grande, los clasificadores salen parecidos y no suele merecer la pena hacer *bagging*

METACLASIFICADORES

- Métodos avanzados: *Bagging* (Breiman, 1996):
  - Algoritmo:
    - Fase de entrenamiento:
      - Sea N el número de ejemplos en el conjunto de entrenamiento
      - Para k = 1, …, L (L clasificadores):
        - Tomar una muestra *bootstrap* S_k del conjunto de entrenamiento
          - (muestrear N instancias con reemplazamiento)
        - Construir un clasificador d_k usando dicha muestra
        - Añadirlo al metaclasificador
    - Fase de clasificación:
      - Para cada clasificador, predecir la clase de la instancia usando el modelo
      - Devolver la clase que ha sido predicha más veces (más votada)

METACLASIFICADORES

- Métodos avanzados: *Bagging* (Breiman, 1996):
  - La probabilidad de que una instancia del conjunto de entrenamiento sea seleccionada para formar parte de la muestra *bootstrap* es $1-(1-1/N)^N$, que, para N grande, es aproximadamente 0.63
    - Esto quiere decir que cada muestra *bootstrap* se espera que contenga sólo un 63% de instancias diferentes del conjunto de entrenamiento
    - Esto hace que los clasificadores construidos sean distintos (se entrenan sobre conjuntos diferentes)
  - Si los clasificadores son inestables (ej: RNA), el metaclasificador debería ser mejor
  - En cambio, si son estables (ej: kNN), no van a ser muy distintas las predicciones
    - Poca mejora con el *bagging*

METACLASIFICADORES

- Métodos avanzados: *Bagging* (Breiman, 1996):
  - Tipos de salidas a las que se aplica:
    - Etiquetas de clase
    - Probabilidades a posteriori: 2 opciones:
      - Usar la clase con mayor probabilidad. Votar después.
      - Promediar las probabilidades sobre las muestras *bootstrap* obteniendo unas nuevas probabilidades de cada clase y tomar la clase de la más grande.
      - Ambas producen tasas de error similares
    - Predicciones numéricas: tomar la mediana

METACLASIFICADORES

- Métodos básicos
  - Fusión de etiquetas
  - Fusión de salidas continuas
  - *Stacking*
  - Cascada
- Métodos avanzados:
  - *Bagging*
  - **Aleatorización**
  - *Boosting*
  - Híbridos
- Métodos más comunes:
  - Random Forest
  - XGBoost

METACLASIFICADORES

- Métodos avanzados: Aleatorización:
  - *Bagging* introduce aleatorización en la entrada del algoritmo, con excelentes resultados
  - Pero hay muchas otras formas de crear diversidad introduciendo aleatoriedad
    - Cualquier algoritmo es susceptible de permitir algún tipo de aleatorización
      - Por ejemplo, un árbol de decisión que, al abrir las ramas en cada nodo, en lugar de escoger la variable ganadora, escoge una al azar entre las mejores
      - Precio a pagar: más diversidad pero menor uso de los datos, quizá disminuyendo el rendimiento de cada modelo
    - Aleatorización es más costosa que *bagging* porque hay que modificar el algoritmo
      - Sin embargo, puede aplicarse a clasificadores estables
      - El truco es aleatorizar para que haya diversidad sin sacrificar demasiado el rendimiento
    - Aleatorización y *bagging* pueden combinarse introduciendo aleatoriedad de forma complementaria y diferente.

METACLASIFICADORES

- Métodos básicos
  - Fusión de etiquetas
  - Fusión de salidas continuas
  - *Stacking*
  - Cascada
- Métodos avanzados:
  - *Bagging*
  - Aleatorización
  - ***Boosting***
  - Híbridos
- Métodos más comunes:
  - Random Forest
  - XGBoost

METACLASIFICADORES

- Métodos avanzados: *Boosting* (Freund y Shapire, 1996):
  - El metaclasificador se desarrolla de manera incremental, añadiendo uno en cada iteración
  - El clasificador en el paso t se entrena sobre un conjunto de entrenamiento muestreado de manera selectiva
    - Los objetos mal clasificados en la iteración anterior ven aumentada su probabilidad de ser seleccionados
      - Algoritmo AdaBoost
  - Se combinan modelos del mismo tipo
    - Esta técnica se basa en buscar explícita y activamente modelos que se complementen, en vez de dejarlo en manos del azar y de la inestabilidad del algoritmo (*bagging*)
  - Para combinar la salida, se utiliza un sistema de votación con pesado

METACLASIFICADORES

- Comparación *boosting* / *bagging*:
  - Similitudes:
    - Combinar modelos del **mismo tipo**
    - Usar **voto por mayoría** para combinar las salidas
    - Los clasificadores base se generan usando la **misma muestra** de entrenamiento
  - Diferencias:
    - En *bagging*, **las muestras de entrenamiento** se generan aleatoriamente y puede generar los clasificadores en paralelo; *boosting* es determinístico y la generación es secuencial basada en los resultados previos
    - En *bagging* cada modelo individual se construye de manera separada; en boosting cada **nuevo modelo está influenciado** por el comportamiento de los anteriormente construidos
      - *Boosting*: los nuevos modelos tratan de ser expertos para instancias mal clasificadas por los anteriores modelos
    - En *boosting* la **contribución final de cada modelo** se basa en su bondad (en vez de dar igual peso a todos)

METACLASIFICADORES

- Métodos básicos
  - Fusión de etiquetas
  - Fusión de salidas continuas
  - *Stacking*
  - Cascada
- Métodos avanzados:
  - *Bagging*
  - Aleatorización
  - *Boosting*
  - **Híbridos**
- Métodos más comunes:
  - Random Forest
  - XGBoost

METACLASIFICADORES

- Métodos avanzados: Hibridaciones:
  - El clasificador se induce teniendo en cuenta dos o más paradigmas, por ejemplo:
    - *Lazy Bayesian Rules* (Zheng y Webb, 2000):
      - Para cada ejemplo a clasificar obtiene un naive Bayes con aquellos casos más cercanos
    - *Naive Bayes Tree* (Kohavi, 1996):
      - Induce árboles cuyas hojas son clasificadores naive Bayes, los cuales se usarán en los ejemplos que alcancen dichas hojas
    - *Logistic Model Trees* (Landwehr, Hall y Frank, 2005):
      - Árboles en cuyas hojas se lleva a cabo la clasificación, de aquellos ejemplos que los alcancen, por medio de una regresión logística

METACLASIFICADORES

- Métodos básicos
  - Fusión de etiquetas
  - Fusión de salidas continuas
  - *Stacking*
  - Cascada
- Métodos avanzados:
  - *Bagging*
  - Aleatorización
  - *Boosting*
  - Híbridos
- **Métodos más comunes:**
  - Random Forest
  - XGBoost

METACLASIFICADORES

- Métodos más comunes:
  - Random forest
    - Se genera un conjunto de árboles de decisión mediante *bagging* y selección de un subconjunto aleatorio de atributos
      - *Feature bagging*
      - Este subconjunto aleatorio asegura una baja correlación entre los árboles de decisión
    - Salida final: votación de todos los árboles
  - XGBoost

METACLASIFICADORES

- Métodos más comunes:
  - Random forest
  - XGBoost
    - GBDT (*Gradient-Boosted Decision Tree*)
      - Similar a *Random Forest*, pero con varias diferencias importantes:
        - En lugar de *bagging*, GBDT usa *gradient boosting*:
          - El proceso de generación de nuevos modelos («débiles») se formaliza como un algoritmo de descenso de gradiente sobre una función objetivo
          - Cada modelo se basa en el anterior intentando minimizar su error
            - Se usa el gradiente del error con respecto a la predicción realizada por el modelo anterior para ajustar el siguiente modelo
            - Si el nuevo modelo mejora el anterior, se toma como base para el siguiente modelo
            - Si no lo mejora, se vuelve al anterior para modificarlo de forma distinta
            - Esto se repite hasta que la diferencia entre modelos consecutivos es muy pequeña
              - O cuando se ha llegado al número máximo de iteraciones
        - La predicción final es una suma ponderada de las predicciones de todos los árboles
    - XGBoost (*eXtreme Gradient Boosting*) es una librería que implementa GBDT