UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escola Tècnica Superior d'Enginyeria Informàtica

etsinf

Tema 7. Combinación de clasificadores

Percepción (PER)

Curso 2019/2020

Departamento de Sistemas Informáticos y Computación

Índice

Índice

Introducción

- Las fuentes de error de un clasificador son:
  - ***Bias*** (sesgo): asunciones erróneas, error en la selección del tipo de clasificador. Relacionado con la capacidad de ajuste del clasificador elegido a los datos.
  - ***Variance*** (varianza): dependencia de los datos de entrenamiento. Relacionado con la bondad del aprendizaje del clasificador en función de la cantidad de datos disponibles.
  - ***Noise*** (ruido): ruido inherente en los datos
- Compromiso entre *bias* y *variance* para el diseño de un buen clasificador
- Caracterización de *bias* y *variance* de los distintos clasificadores

Caracterización del error

Clasificador G como regresor (aprendido en entrenamiento): $G(x) : E \to \mathbb{R}$

Valor verdadero y: $y = F(x) + \epsilon$

- $F(x)$: función verdadera
- ϵ: ruido inherente de los datos

Representación del error como el ***valor esperado del error cuadrático***:

$$\mathbb{E}[(y - G(x))^2]$$

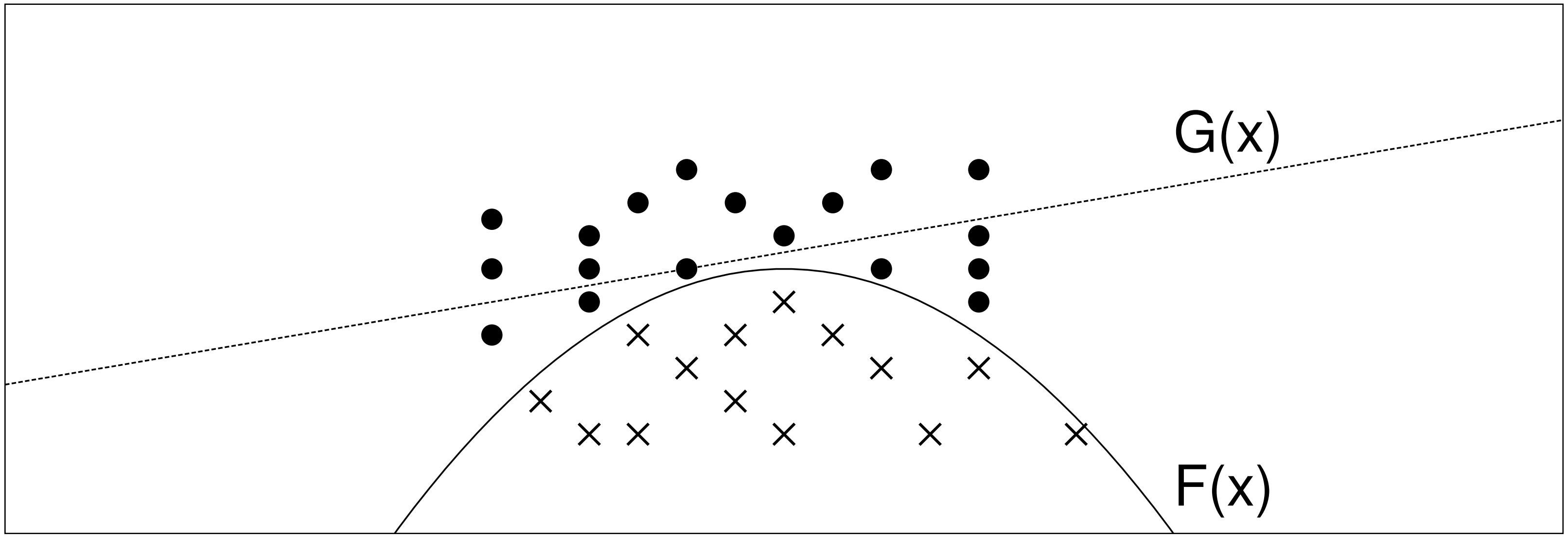
Definiendo $\overline{G(x)} = \mathbb{E}\left[G(x)\right]$, finalmente se tiene:

$$\mathbb{E}\left[(y - G(x))^2\right] = \underbrace{\mathbb{E}\left[\left(G(x) - \overline{G(x)}\right)^2\right]}_{\text{Variance}} + \underbrace{\left(\overline{G(x)} - F(x)\right)^2}_{\text{Bias}} + \underbrace{\mathbb{E}\left[(y - F(x))^2\right]}_{\text{Noise}}$$

Detalles de los calculos en documento en PoliformaT

Caracterización del error

- ***Variance***: variación de $G(x)$ según datos de entrenamiento
- ***Bias***: error del clasificador promedio, capacidad de adaptarse al entrenamiento
- ***Noise***: ruido presente en los datos



Tipos de clasificadores

- Clasificadores con *bias* alto y *variance* bajo: (p.ej., clasificador lineal)
  - Poco flexibles
  - Pocos parámetros
  - Bajo requerimiento de datos de entrenamiento
  - Clasificadores débiles (*weak learners*): apenas mejores que el aleatorio
- Clasificadores con *bias* bajo y *variance* alto: (p.ej., k-NN)
  - Muy flexibles (aprenden cualquier frontera de decisión)
  - Muchos parámetros
  - Alto requerimiento de datos de entrenamiento
  - Clasificadores fuertes (*strong learners*): *arbitrariamente* precisos
- ***Ensemble learning***: combinación de clasificadores
  - ***Bagging***:
    combinación de clasificadores fuertes modificando el entrenamiento
  - ***Boosting***:
    construcción de clasificadores fuertes a partir de clasificadores débiles

Índice

Bagging

Bagging: **B**ootstrap **Ag**regat**ing**

Clasificadores G_i a partir de variación de los datos de entrenamiento X

- Obtener X_i por *bootstrapping* desde X
- *Bootstrapping*: muestreo aleatorio con reemplazamiento
- Entrenar G_i con X_i

Combinación de clasificadores G_i por suma no ponderada

Bagging

Algoritmo Bagging:

- Entrenamiento:

  For $i = 1 \cdots M$

  Obtener X_i a partir de X

  Entrenar G_i con X_i

  End

- Clasificación:

$$G(x) = \frac{1}{M} \sum_{i=1}^{M} G_i(x)$$

Bagging se emplea en clasificadores binarios, con $\hat{c}(x) = \text{sgn}(G(x))$

Propiedades de Bagging

- ***Variance***:

$$\mathbb{E}\left[\left(G(x) - \overline{G(x)}\right)^2\right] \qquad G(x) = \frac{1}{M}\sum_{i=1}^{M} G_i(x),$$ *variance* se reduce

- ***Bias***:

$$\left(\overline{G(x)} - F(x)\right)^2 \qquad \overline{G(x)}$$ no cambia, y *bias* no cambia

- El error del clasificador generado mediante Bagging se reduce
- Bagging adecuado para combinar clasificadores fuertes (flexibles, *bias* bajo)

Índice

Boosting

- Combinación de clasificadores débiles ponderando los datos de entrenamiento
- Se dispone de un conjunto de L clasificadores débiles: $\mathcal{G} = \{G_1, \ldots, G_L\}$
- Se asumen clasificadores débiles binarios: $G_l(x) \in \{-1, 1\}$
- Conjunto de entrenamiento: $\mathcal{X} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ con $y_n \in \{-1, 1\}$
- En cada iteración, toma $C_i \in \mathcal{G}$ de menor error sobre $\mathcal{X}$ ponderado por $w^{(i)}$
- $G(x)$ es la combinación lineal de los seleccionados hasta iteración m:

$$G(x) = G^{(m)}(x) = \sum_{i=1}^{m} \alpha_i \, C_i(x) \;\; \text{donde} \;\; C_i \in \mathcal{G}$$

Boosting

En la iteración m seleccionamos un clasificador C_m junto con su peso α_m

$$G^{(m)}(x) = G^{(m-1)}(x) + \alpha_m \, C_m(x)$$

El criterio de error E a minimizar es la pérdida exponencial en cada dato

$$E = \sum_{i=1}^{N} \exp(-y_i \, G^{(m)}(x_i)) = \sum_{i=1}^{N} \exp(-y_i \, G^{(m-1)}(x_i) - y_i \, \alpha_m \, C_m(x_i))$$

Definiendo el peso de x_i para la iteración m ($w_i^{(m)}$) como su pérdida exponencial:

$$w_i^{(m)} = \exp(-y_i \, G^{(m-1)}(x_i)) \;\longrightarrow\; E = \sum_{i=1}^{N} w_i^{(m)} \exp(-y_i \, \alpha_m \, C_m(x_i))$$

Se buscan C_m y α_m que minimicen E

Boosting

- Minimización respecto a C_m
  - $E \approx \sum_{y_i \neq C_m(x_i)} w_i^{(m)}$
  - Por tanto, ***se selecciona el clasificador $C_m \in \mathcal{G}$ que minimice el error de clasificación sobre los datos ponderados***

- Minimización respecto a α_m: por derivación e igualación a cero
  - Error en iteración m:

$$\epsilon_m = \frac{\sum_{y_i \neq C_m(x_i)} w_i^{(m)}}{\sum_{i=1}^{N} w_i^{(m)}}$$

  - Valor de α_m:

$$\alpha_m = \frac{1}{2} \ln \left(\frac{1 - \epsilon_m}{\epsilon_m} \right)$$

Detalles de los cálculos en documento en PoliformaT

Algoritmo AdaBoost

Entrada:

- Conjunto de entrenamiento $\mathcal{X} = \{(x_1, y_1) \dots (x_N, y_N)\}$
- Conjunto clasificadores débiles (binarios) $\mathcal{G} = \{G_1, \dots, G_L\}$

Proceso:

1. $w_i^{(1)} = \frac{1}{N} \quad i = 1, \dots, N$

2. Para $m = 1 \dots M$

2.1. $C_m = \text{argmin}_{g \in \mathcal{G}} \sum_{y_i \neq g(x_i)} w_i^{(m)}$

2.2. $\epsilon_m = \min_{g \in \mathcal{G}} \sum_{y_i \neq g(x_i)} w_i^{(m)}$

2.3. Si $\epsilon_m > 0.5$ fin

2.3. $\alpha_m = \frac{1}{2} \ln \left(\frac{1 - \epsilon_m}{\epsilon_m} \right)$

2.4. $w_i^{(m+1)} = \dfrac{w_i^{(m)} \exp(-y_i \, \alpha_m \, C_m(x_i))}{\sum_{i'=1}^{N} w_{i'}^{(m)} \exp(-y_{i'} \, \alpha_m \, C_m(x_{i'}))}$

Salida: $G(x) = \sum_{m=1}^{M} \alpha_m \, C_m(x)$

Propiedades de AdaBoost

Boosting:

- Aprovecha el bajo *variance* de los clasificadores (débiles) combinados
- Reduce el *bias*
- Es más sensible a datos ruidosos
- En comparación con Bagging, puede comportarse peor según los datos