Tema 7. Combinación de modelos

- Algoritmo que usa varias instancias do mesmo **modelo base** (clasificador ou regresor): ensemble, comité
- Exemplo: combinación de árbores de clasificación/regresión
- O nome do algoritmo (meta-algoritmo), p.ex. bagging ou boosting, aplícase ao método usado para combinar os modelos
- Exemplo: grupo de clasificadores do mesmo tipo, entrenados de formas distintas, cunha votación para decidir a saída
- Os modelos base son usualmente **débiles**: non funcionan moi ben, pero son simples e de entrenamento rápido
- O obxectivo é que a súa combinación teña máis calidade que a suma dos clasificadores base individuais, dando un modelo **forte**
- Theodoridis: Machine learning: a Bayesian and optimization perspective, Academic Press, Cap. 7

Combinación de modelos (II)

- Os modelos base deben ser **diversos** (distintos entre eles) para que a súa combinación sexa forte (mellor que calquera dos modelos individuais)
- Aprenden distintas facetas do problema de clasificación
- A diversidade pódese deber a diferencias entre modelos base en:

1) Inicialización do entrenamento: p.ex. combinación de redes neuronais MLP con distintas inicializacións aleatorias de pesos
2) Sintonización dos hiper-parámetros: combinación de MLPs con distintos tamanos de capas ocultas
3) Conxunto de entrenamento: distintos conxuntos de entrenamento para os distintos modelos base

- Os algoritmos de combinación pódense aplicar sobre distintos tipos de modelos base: bagging de árbores de decisión ou bagging de clasificadores KNN

Bagging (I)

- **Bootstrap aggregating**: varios modelos base entrénanse en varios conxuntos distintos do mesmo tamano
- Creados usando o método de **bootstrap**: conxunto de patróns seleccionados aleatoriamente con repetición de patróns
- Na mostra seleccionada hai patróns seleccionados varias veces e outros non seleccionados nunca
- O uso de bootstrap diversifica os modelos base mostrándolles aos modelos distintos conxuntos de entrenamento
- A saída decídese por votación (clasificación) ou promedio/ mediana (regresión) entre os modelos base
- Os modelos base son normalmente árbores de decisión, aínda que poden ser outros. En Weka podes cambiar o modelo (opción -W)

Bagging (II)

- Bootstrap mellora os modelos base porque reduce a súa varianza e o sobreaprendizaxe sen aumentar o sesgo
- As árbores tenden a sobreaprender o conxunto de entrenamento
- Bagging fai que ás árbores sexan máis diversas porque entrenan sobre distintos conxuntos
- A votación/promedio non é tan sensíbel ao ruido nos datos e compensa esta sobreaprendixaze das árbores individuais
- Hiper-parámetro B (bag size): nº de árbores. Sintonizábel, calidade pouco sensíbel a B se o nº de árbores é suficientemente elevado
- Función **bagging** en paquete **ipred** de R: B=25 por defecto. Normalmente, B~100-200, dependendo do tamano dos datos

Bagging (III)

- Alternativamente a grid-search, pódese determinar B observando o “erro fora de bolso” (out-of-bag error, OOB):

  Erro promedio, excluíndo para cada árbore os patróns do seu conxunto de entrenamento (creado mediante bootstrap)

- O erro OOB estabilízase cando o nº B de árbores é dabondo
- Saída do bagging: se $z_b(\mathbf{x})$ é a saída do b-ésimo modelo base:

Clasificación: $$z(\boldsymbol{x}) = \underset{c=1\ldots C}{arg\,max} \left\{ \sum_{b=1}^{B} \delta[z_b(\boldsymbol{x}), c] \right\} \quad \delta(x, y) = \begin{cases} 1 & x = y \\ 0 & x \neq y \end{cases}$$

Votación entre os B clasificadores

Regresión: $$z(\boldsymbol{x}) = \frac{1}{B} \sum_{b=1}^{B} z_b(\boldsymbol{x})$$

Promedio das saídas dos B regresores

Boosting

- Empuxa (boosts) cara arriba a calidade dos modelos base
- Como en bagging, os modelos base diferéncianse no conxunto de entrenamento: os seus patróns pondéranse de forma distinta para cada modelo base
- Modelo 1: tódolos patróns teñen pesos iguais
- Modelos posteriores: cada patrón pondérase en base ao erro cometido polos modelos anteriores nese patrón, comparado cos outros patróns (pesos w_i, con i=1..N)
- Pondéranse máis os patróns nos que houbo fallos para que sexan correxidos polos seguintes modelos base
- A saída do comité z(**x**) é un promedio ponderado das saídas dos modelos base, con pesos a_b, con b=1..B
- O máis popular: **adaboost** (adaptive boosting), clasificación

Adaboost (I)

$z_b(\mathbf{x},\theta_b)\in\{\pm1\}$: saída do clasificador b (coñecida)

θ_b: parámetros entrenábeis do clasificador b

- Clasificación binaria: $y_i, z(\mathbf{x}_i)\in\{\pm1\}$: $z(\boldsymbol{x})=sign[u(\boldsymbol{x})], u=\sum_{b=1}^{B} a_b z_b(\boldsymbol{x},\boldsymbol{\theta}_b)$
- Combinación de B clasificadores $\{z_b(\mathbf{x},\theta_b)\}_{b=1}^{B}$ con $z_b(\mathbf{x},\theta_b)\in\{\pm1\}$.
- Función de custe (a minimizar): $J[t,z(\mathbf{x})]=e^{-tz(\mathbf{x})}$
- Sexa $u_b(\mathbf{x})$ a suma parcial dos *b* primeiros clasificadores: $u_b(\boldsymbol{x})=\sum_{k=1}^{b} a_k z_k(\boldsymbol{x},\boldsymbol{\theta}_k), b=1,\ldots,B$
- Os $u_b(\mathbf{x})$ calcúlanse iterativamente:

$$u_b(\boldsymbol{x})=u_{b-1}(\boldsymbol{x})+a_b z_b(\boldsymbol{x},\boldsymbol{\theta}_b), b=2\ldots B$$

- Na iteración *b* (para o clasificador base *b*) calcúlanse a_b e θ_b:

$$(a_b,\boldsymbol{\theta}_b)=\underset{a,\boldsymbol{\theta}}{argmin}\left\{\sum_{i=1}^{N} w_i^b \exp[-y_i u_b(\boldsymbol{x}_i)]\right\}$$

- w_i^b é o peso de $\mathbf{x}_i$ na iteración b: $w_i^1=1/N; w_i^b=\exp[-y_i u_{b-1}(\mathbf{x}_i)]$, b>1

Adaboost (II)

- Como $u_b(\mathbf{x}_i)=u_{b-1}(\mathbf{x}_i)+a_b z_b(\mathbf{x}_i,\theta_b)$:

$$(a_b,\boldsymbol{\theta}_b)=\underset{a,\boldsymbol{\theta}}{argmin}\left\{\sum_{i=1}^{N} w_i^b \exp\{-y_i[u_{b-1}(\boldsymbol{x}_i)+a\, z_b(\boldsymbol{x}_i,\boldsymbol{\theta})]\}\right\}$$

- Iteración b: a mantense constante e calcúlase θ_b (entrenamento):

$$\boldsymbol{\theta}_b=\underset{\boldsymbol{\theta}}{argmin}\left\{\sum_{i=1}^{N} w_i^b \exp[-y_i a\, z_b(\boldsymbol{x}_i,\boldsymbol{\theta}_b)]\right\}$$

- O entrenamento persigue minimizar a suma de pesos dos patróns clasificados incorrectamente

$$\boldsymbol{\theta}_b=\underset{\boldsymbol{\theta}}{argmin}\{P_b(\boldsymbol{\theta})\} \qquad P_b(\boldsymbol{\theta})=\sum_{y_i z_b(\boldsymbol{x}_i,\boldsymbol{\theta})<0} w_i^b$$

  $P_b(\theta)$ é a suma dos pesos dos patróns $\mathbf{x}_i$ nos que $y_i z_b(\mathbf{x}_i,\theta_b)<0$: erros de clasificación do ensemble cos *b*-1 primeiros clasificadores base

- Coñecido θ_b , calcúlase a_b: $a_b=\underset{a}{argmin}\{P_b e^{a}+(1-P_b)e^{-a}\}$

Adaboost (III)

- Derivando a expresión entre chaves e igualando a 0 obtense:

$$a_b = \frac{1}{2}\ln\left(\frac{1-P_b}{P_b}\right)$$

- Coñecidos θ_b e a_b, os pesos w_i^{b+1} para i=1,...,N, calcúlanse como:

$$w_i^{b+1} = \frac{w_i^b \exp\left[-y_i a_b z_b\left(\boldsymbol{x}_i, \boldsymbol{\theta}_b\right)\right]}{Z_b}$$

  onde Z_b e o factor de normalización: $Z_b = \sum_{i=1}^{N} w_i^b \exp\left[-y_i a_b z_b\left(\boldsymbol{x}_i, \boldsymbol{\theta}_b\right)\right]$

- O proceso continúa para o seguinte clasificador b+1 ata b=B

- Finalmente, coñecidos $\{a_b, \theta_b\}_{b=1}^{B}$, a saída da combinación adaboost ven dada por:

$$z(\boldsymbol{x}) = sign\left[\sum_{b=1}^{B} a_b z_b\left(\boldsymbol{x}, \boldsymbol{\theta}_b\right)\right]$$

$$z_b\left(\boldsymbol{x}, \boldsymbol{\theta}_b\right) = \sum_{i=1}^{R_b} y_{bi} I_{bi}\left(\boldsymbol{x}\right)$$

(Classification tree)

$\theta_b = \{R_b, \{y_{bi}, I_{bi}\}_{i=1}^{Rb}\}$

Adaboost (IV)

Entrenamento do clasificador base

$w_i^1=1/N$, i=1..N

for *b=1:B-1*

$P_b(\boldsymbol{\theta})=\sum_{y_i z_b(\boldsymbol{x}_i,\boldsymbol{\theta})<0}^{N} w_i^b$; $\theta_b=\text{argmin}_\theta \{P_b(\theta)\}$; $a_b=\frac{1}{2}\log\left(\frac{1-P_b^m}{P_b^m}\right)$

$P_b^m=P_b(\theta_\theta)$; $Z_b=0$

for i=1:N

$w_i^{b+1}=w_i^b \exp[-y_i a_b z_b(\mathbf{x}_i,\theta_b)]$; $Z_b=Z_b+w_i^{b+1}$

endfor

for i=1:N

$w_i^{b+1}=w_i^{b+1}/Z_b$

endfor

endfor

Saída $z(\mathbf{x})$ para un patrón de teste $\mathbf{x}$: $z(\boldsymbol{x})=sign\left[\sum_{b=1}^{B} a_b z_b(\boldsymbol{x},\boldsymbol{\theta}_b)\right]$

Random forest (I)

- Bosque aleatorio: combinación de árboles de clasificación ou regresión de modo que se corrixa a súa sobreaprendizaxe
- Usa bagging e selección aleatoria de características para o desdobramento, deixando parte dos patróns fóra do entrenamento
- Nunha árbore de decisión clásica, en cada nodo divídese a característica que máis reduce a entropía (clasificación) ou o erro cadrático medio (regresión)
- No bagging, se algunhas características son moi importantes, serán seleccionadas por case tódalas árbores: pouca diversidade
- No random forest, auméntase a diversidade usando, en cada árbore, un grupo de q<n características escollidas aleatoriamente e distintas das seleccionadas polas outras árbores
- En cada nodo selecciónase a mellor característica neste grupo
- O entrenamento é máis eficiente porque usa menos características

Random forest (II)

- Usualmente, o número q de entradas seleccionadas é $q=\sqrt{n}$ para clasificación e $q=n/2$ para regresión
- Se as saídas dos B modelos son variábeis aleatorias $\{z_b\}_{b=1}^{B}$, todos con varianza σ e correlación ρ, demóstrase que a varianza do RF é:

$$var\left(\frac{1}{B}\sum_{b=1}^{B} z_b\right)=\left(\frac{1-\rho}{B}+\rho\right)\sigma^2$$

- A selección aleatoria de características do RF:

1) Incrementa lixeiramente o sesgo (inconvinte)
2) Incrementa a varianza (σ^2) de cada árbore (inconvinte)
3) Reduce a correlación (ρ) entre as árbores (vantaxe: +diversidade)

- A reducción na correlación (3) é o termo máis importante: aumenta a calidade a respeito das árbores individuais

Random forest (III)

RF=$\emptyset$; K=nº máx. de nodos na árbore

Árbore actual

Nº de nodos

for b=1:B

S=mostra de bootstrap de $\{\mathbf{x}_i,y_i\}_{i=1}^{N}$; A=$\emptyset$; r=0

repeat

$V_j=\{x_{ij}\}_{i=1}^{N}$

F=$\{i_1..i_q\}\subset\{1..n\}$ con q<n, selección aleatoria

Selecciona característica j∈F e umbral u∈V_j de modo que:

clasificación: $(j,u)=\underset{i=1..n,k\in V_i}{arg\,max}\left\{\Delta E_{ik}\right\}$ ← E=entropía

regresión: $(j,u)=\underset{i=1..n;k\in V_i}{argmin}\left\{\sum_{m\in A_1}(y_m-z_1)^2+\sum_{m\in A_2}(y_m-z_2)^2\right\}$

r=r+1

Crea nodo n_r (x_j<u e x_j≥u) con j e u

A=A∪{n_r}

$z_p(i,k)=\frac{1}{N_p}\sum_{q\in A_p}y_q, p=1,2$

$A_1(i,k)=\{m: x_{mi}<k\}$

$A_2(i,k)=\{m: x_{mi}\geq k\}$

until r>K

RF=RF∪A

endfor

Saída: clasificación: $z(\boldsymbol{x})=\underset{c=1\ldots C}{arg\,max}\left\{\sum_{b=1}^{B}\delta[z_b(\boldsymbol{x}),c]\right\}$; regresión: $z(\boldsymbol{x})=\frac{1}{B}\sum_{b=1}^{B}z_b(\boldsymbol{x})$

$\delta(x,y)=\begin{cases}1 & x=y\\ 0 & x\neq y\end{cases}$

Votación

Promedio

Random forest (IV)

- RF proporciona unha medida da importancia de cada característica
- Poucos hiperparámetros e pouco importantes (fácil de usar):

1) Nº de árbores B no bosque

2) Nº de características q a desdobrar en cada nodo

3) Nº mínimo de patróns para desdobrar un nodo interno

- Aumentando B non se incrementa o sobreaprendizaxe
- Moi paralelizábel
- Require pouco preprocesamento dos datos
- O uso de $q<n$ características ou de $N'<N$ patróns para o bootstrap é eficiente con datos grandes
- Moi bos resultados: método state-of-the-art