

Support Vector Machines (SVM)

Suponha um conjunto de L pontos, com \bar{x}_i tendo d atributos (a dimensão do espaço das features) e pertencendo a duas possíveis classes $y_i = 1$ ou $y_i = -1$, ou seja, o training set é da forma

$$\{\bar{x}_i, y_i\}_{i=1,2,\dots,L}$$

$$y_i \in \{-1, 1\}$$

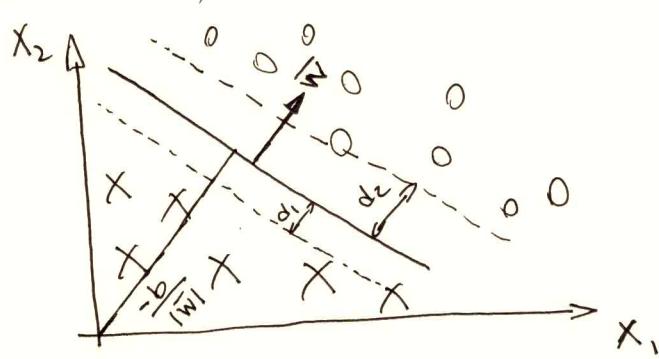
$$\bar{x}_i \in \mathbb{R}^d$$

Assuma, ainda, que esse conjunto é linearmente separável, no sentido que existe um hiperplano capaz de separar os dados nas duas classes. Esse hiperplano pode ser descrito por

$$\bar{w} \cdot \bar{x} + b = 0$$

com \bar{w} sendo um vetor normal ao plano e $\frac{b}{\|\bar{w}\|}$ a distância perpendicular da origem ao hiperplano.

No caso $d=2$, teríamos algo como



$$\begin{aligned} 0 &\rightarrow \{+1\} \\ x &\rightarrow \{-1\} \end{aligned}$$

Note que existem várias retas capazes de separar as duas categorias. A ideia da SVM é encontrar aquela que maximize a distância dos membros mais próximos

de cada classe (os chamados vetores de suporte). Do ponto de vista da equação anterior, implementar a SVM é selecionar \bar{w} e b de modo que

$$\bar{x}_i \cdot \bar{w} + b \geq 1 \quad \text{se } y_i = +1$$

$$\bar{x}_i \cdot \bar{w} + b \leq -1 \quad \text{se } y_i = -1$$

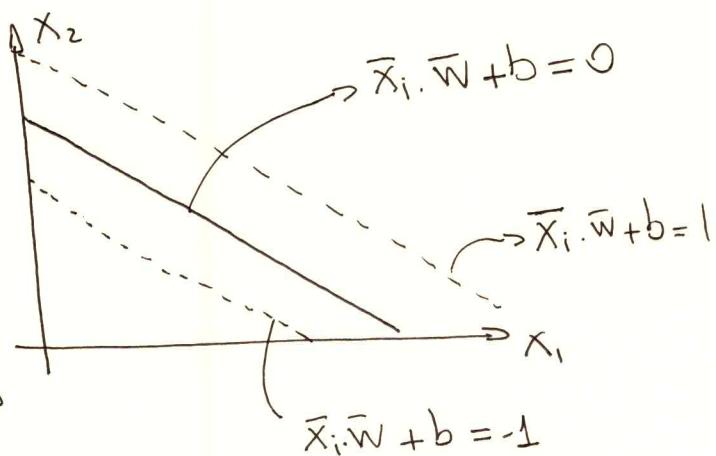
ou

$$y_i (\bar{x}_i \cdot \bar{w} + b) - 1 \geq 0 \quad \forall i$$

Em particular, para os vetores de suporte, vamos supor que

$$\bar{x}_i \cdot \bar{w} + b = 1$$

$$\bar{x}'_i \cdot \bar{w} + b = -1$$



Subtraindo uma da outra, temos

$$(\bar{x}_i - \bar{x}'_i) \cdot \bar{w} = 2$$

$$(\bar{x}_i - \bar{x}'_i) \cdot \frac{\bar{w}}{|\bar{w}|} = \frac{2}{|\bar{w}|}$$

$$(\bar{x}_i - \bar{x}'_i) \cdot \hat{w} = \frac{2}{|\bar{w}|}$$

A quantidade é conhecida como separação e $\frac{1}{|\bar{w}|}$ é a margem SVM. Nossa desejo é maximizar a margem, o que é equivalente a minimizar $|\bar{w}|$. Assim, temos que

$$\min |\bar{w}| \text{ tal que } y_i (\bar{x}_i \cdot \bar{w} + b) - 1 \geq 0 \quad \forall i$$

Entretanto, é mais fácil minimizar $\frac{1}{2} \|\bar{w}\|^2$, de modo que ficamos com

$$\min \frac{1}{2} \|\bar{w}\|^2 \text{ tal que } y_i(\bar{x}_i \cdot \bar{w} + b) - 1 \geq 0 \quad \forall i$$

Para isso, usamos multiplicadores de Lagrange (α_i)

$$\begin{aligned} L_p &= \frac{1}{2} \|\bar{w}\|^2 - \sum_{i=1}^L \alpha_i [y_i(\bar{x}_i \cdot \bar{w} + b) - 1] \\ &= \frac{1}{2} \|\bar{w}\|^2 - \sum_{i=1}^L \alpha_i y_i (\bar{x}_i \cdot \bar{w} + b) + \sum_{i=1}^L \alpha_i \end{aligned}$$

Queremos encontrar \bar{w} e b assim como $\alpha_i \geq 0 \quad \forall i$.

Derivando, temos

$$\frac{\partial L_p}{\partial \bar{w}} = 0 \Rightarrow \bar{w} = \sum_{i=1}^L \alpha_i y_i \bar{x}_i$$

$$\frac{\partial L_p}{\partial b} = 0 \Rightarrow \sum_{i=1}^L \alpha_i y_i = 0$$

Substituindo esses resultados em L_p , temos

$$L_D = \sum_{i=1}^L \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \bar{x}_i \cdot \bar{x}_j$$

tal que $\alpha_i \geq 0 \quad \forall i$ e $\sum_{i=1}^L \alpha_i y_i = 0$

Definindo $H_{ij} = y_i y_j \bar{x}_i \cdot \bar{x}_j$, temos

$$L_D = \sum_{i=1}^L \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i H_{ij} \alpha_j$$

$$= \sum_{i=1}^L \alpha_i - \frac{1}{2} \bar{\alpha}^T \bar{H} \bar{\alpha}$$

Nesse caso, vamos procurar por

$$\max_{\alpha} \left[\sum_{i=1}^L \alpha_i - \frac{1}{2} \bar{\alpha}^T \bar{H} \bar{\alpha} \right] \quad \alpha_i \geq 0 \quad \forall i \quad \sum_{i=1}^L \alpha_i y_i = 0$$

Esse é um problema de otimização quadrática, de onde podemos encontrar os valores de α_i ($\bar{\alpha}$). Sendo assim, encontramos \bar{w} via

$$\bar{w} = \sum_{i=1}^L \alpha_i y_i \bar{x}_i$$

e b fica dada por

$$b = \frac{1}{N_s} \sum_{s \in S} \left[y_s - \sum_{m \in S} \alpha_m y_m \bar{x}_m \cdot \bar{x}_s \right]$$

sendo que S representa o índice dos vetores de suporte, sendo que s representa o índice dos vetores de suporte.

N_s o número deles.

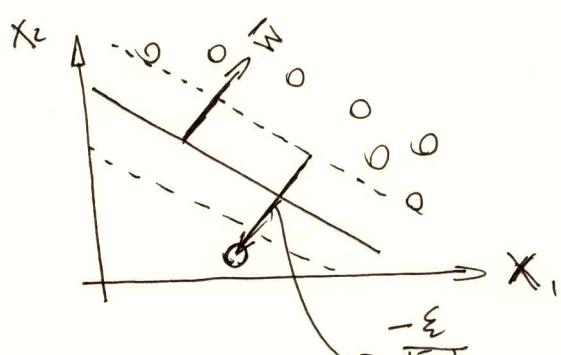
Usualmente ocorre de não ser possível separar os dois conjuntos por um hiperplano de maneira completa, ou seja, vamos existir pontos não classificados corretamente. Para isso, vamos introduzir variáveis ξ_i , tal que,

$$\begin{aligned} \bar{x}_i \cdot \bar{w} + b &\geq +1 - \xi_i & \forall y_i = +1 \\ \bar{x}_i \cdot \bar{w} + b &\leq -1 + \xi_i & \forall y_i = -1 \end{aligned}$$

ou

$$y_i (\bar{x}_i \cdot \bar{w} + b) - 1 + \xi_i \geq 0$$

com $\xi_i \geq 0 \quad \forall i$



Nesse caso, além de minimizar $\frac{1}{2} \|\bar{w}\|^2$ (maximizar a margem) vamos buscar minimizar o número de classificações erradas. Para isso, vamos considerar

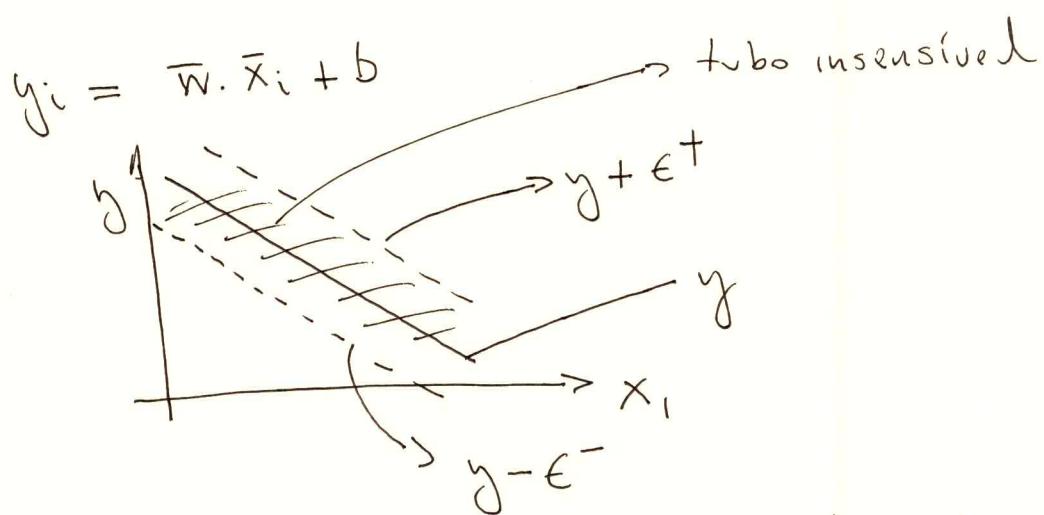
$$\min \left\{ \frac{1}{2} \|\bar{w}\|^2 + C \sum_{i=1}^L \xi_i \right\} \text{ tal que } y_i(\bar{x}_i \cdot \bar{w} + b) - 1 + \xi_i \geq 0 \quad \forall i$$

O parâmetro C controla o peso da penalização em comparação com o tamanho da margem. Por um procedimento similar, podemos encontrar \bar{w} , b e \bar{x} , sendo que $0 < \bar{x}_i \leq C \xi_i$.

SVM para regressão

No lugar de classificar um novo valor de \bar{x}_i nas classes $y_i = \pm 1$, vamos procurar prever um valor real y_i . Nesse caso, o training set fica

$$\{\bar{x}_i, y_i\}_{i=1,2,\dots,L} \quad y_i \in \mathbb{R} \quad \bar{x}_i \in \mathbb{R}^d$$



No caso em que o ponto esteja dentro do tubo insensível ϵ , não existe penalidade, caso contrário, consideramos uma penalização ξ^+ ou ξ^- , dependendo

se o ponto está acima de $y + \epsilon$ ou abaixo de $y - \epsilon$.

SVM não linear

Para resolver o problema da SVM linearmente separável, definimos a matriz \bar{H} como

$$H_{ij} = y_i y_j K(\bar{x}_i, \bar{x}_j)$$

onde $K(\bar{x}_i, \bar{x}_j) = \bar{x}_i \cdot \bar{x}_j$. Essa função é chamada Kernel linear e ela mapeia o espaço d-dimensional dos vetores \bar{x}_i nos recursos. A ideia aqui é imaginar que existe uma outra função $K(\bar{x}_i, \bar{x}_j)$ que torna problemas não separáveis linearmente em problemas separáveis nesse outro espaço mapeando por $K(\bar{x}_i, \bar{x}_j)$.

Ospções mais comuns incluem

$$\rightarrow K(\bar{x}_i, \bar{x}_j) = \exp \left\{ \frac{|\bar{x}_i - \bar{x}_j|^2}{2\sigma^2} \right\} \quad (\text{Gaussian radial basis kernel})$$

$$\rightarrow K(\bar{x}_i, \bar{x}_j) = (\bar{x}_i \cdot \bar{x}_j + d)^b \quad (\text{Polynomial kernel})$$

$$\rightarrow K(\bar{x}_i, \bar{x}_j) = \tanh(a \bar{x}_i \cdot \bar{x}_j - b) \quad (\text{Sigmoidal kernel})$$

Reduções de dimensionalidade

Em muitas situações podemos lidar com um conjunto muito grande de features, ou seja, a dimensão de \bar{x}_i pode ser muito grande. Nesses casos, pode ocorrer de apenas um conjunto menor de features ser importante para descrever a resposta y_i . Porém, pode ser difícil conhecer de partida quais são as features mais importantes. Nesses casos, podemos procurar por alguma transformação que forneça uma redução do espaço das features, ~~$\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n \in \mathbb{R}^m$~~ ,

ou seja,

$$\bar{X} = \begin{pmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,n} \\ x_{2,1} & x_{2,2} & \dots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,2} & \dots & x_{m,n} \end{pmatrix} \quad \bar{X} \in \mathbb{R}^{m \times n}$$

$$\bar{X} = \begin{pmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_m \end{pmatrix} \quad \bar{x}_i \in \mathbb{R}^n$$

Aqui n é o número de amostras e m o número de features. Assim, cada vetor \bar{x}_i contém informações de n amostras da feature i .

Usando \bar{X}_i , vamos de finir a matriz de covariância

$$\bar{C}_x = \frac{1}{n-1} \bar{X} \bar{X}^T = \frac{1}{n-1} \begin{pmatrix} \bar{X}_1 \bar{X}_1^T & \bar{X}_1 \bar{X}_2^T \dots \bar{X}_1 \bar{X}_m^T \\ \bar{X}_2 \bar{X}_1^T & \bar{X}_2 \bar{X}_2^T \dots \bar{X}_2 \bar{X}_m^T \\ \vdots & \vdots \quad \ddots \quad \vdots \\ \bar{X}_m \bar{X}_1^T & \bar{X}_m \bar{X}_2^T \dots \bar{X}_m \bar{X}_m^T \end{pmatrix}$$

com $\bar{C}_x \in \mathbb{R}^{m \times m}$. Note que os elementos da diagonal são a variância da features assumindo que

$$\langle \bar{X}_i \rangle = \frac{1}{n} \sum_{j=1}^n X_{ij} = 0$$

Ou seja,

$$\begin{aligned} \bar{\sigma}_{ii}^2 &= \frac{1}{n-1} \bar{X}_i \bar{X}_i^T \\ &= \frac{1}{n-1} \sum_{j=1}^n X_{ij} X_{ij} = \frac{1}{n-1} \sum_{j=1}^n \bar{X}_{ij}^2 \end{aligned}$$

é a variância de \bar{X}_i e

$$\bar{\sigma}_{ij} = \frac{1}{n-1} \cancel{\bar{X}_i} \bar{X}_i \bar{X}_j^T$$

$$\bar{\sigma}_{ij} = \frac{1}{n-1} \sum_{k=1}^n X_{ik} X_{jk}$$

é a covariância entre \bar{X}_i e \bar{X}_j . Essa matriz pode ser pensada como um medida sobre como as variáveis \bar{X}_i estão correlacionadas.

O método PCA (principal component analysis) consiste em procurar uma transformação \bar{Y} do tipo

$$\bar{Y} = \bar{P} \bar{X}$$

de modo que a matriz de covariância de \bar{Y} , \bar{C}_Y , seja diagonal, ou seja, um espaço no qual as features sejam não correlacionadas (ortogonais). Desse modo, podemos escolher um certo número de features que mais descreve a variável resposta nesse novo espaço \bar{Y} .

Assim, vamos procurar por uma matriz \bar{P} de modo que \bar{C}_Y seja diagonal. Para isso, escrevemos

$$\bar{C}_Y = \frac{1}{n-1} \bar{Y} \bar{Y}^T = \frac{1}{n-1} (\bar{P} \bar{X}) (\bar{P} \bar{X})^T$$

$$= \frac{1}{n-1} \bar{P} \bar{X} \bar{X}^T \bar{P}^T$$

$$= \frac{1}{n-1} \bar{P} \bar{S} \bar{P}^T \quad \text{com} \quad \bar{S} = \bar{X} \bar{X}^T$$

Vamos ver que \bar{S} é simétrica e avançada, sabemos que

$$\bar{S} = \bar{E} \bar{D} \bar{E}^T$$

com $\bar{E} \in \mathbb{R}^{m \times m}$ cujas colunas são os autovetores de \bar{S} e $\bar{D} \in \mathbb{R}^{m \times m}$ é uma matriz diagonal cujos

elementos (da diagonal) são os autovalores de \bar{S} . Vamos assumir que a ordem dos autovetores em \bar{E} é definida de maneira decrescente de seus autovalores correspondentes. O mesmo vale para \bar{D} , ou seja,

$$\bar{D} = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ 0 & 0 & \cdots & \lambda_m \end{pmatrix} \text{ com } \lambda_1 < \lambda_2 < \cdots < \lambda_m$$

Usando esse resultado, temos

$$\bar{C}_y = \frac{1}{n-1} \bar{P} (\bar{E} \bar{D} \bar{E}^T) \bar{P}^T$$

de onde podemos ver que se $\bar{P} = \bar{E}^T$, \bar{C}_y se torna diagonal, isto é,

$$\bar{C}_y = \frac{1}{n-1} \bar{E}^T \bar{E} \bar{D} \bar{E}^T \bar{E}$$

pois \bar{E} é uma matriz ortogonal $\bar{E}^T \bar{E} = \bar{I}$. Logo,

$$\bar{C}_y = \frac{1}{n-1} \bar{D}$$

Note que devido ao ordenamento, a primeira linha de \bar{D} terá a maior variação, a

segunda terá a segunda maior, e assim por diante. Essa decomposição é o que chamamos de PCA e as linhas de \bar{Y} são as componentes principais. Assim, a \bar{P} que procuramos para transformar X via

$$\bar{Y} = \bar{P} \bar{X}$$

é composta pelos autovalores da matriz $\bar{S} = \bar{X} \bar{X}^T$, a qual é proporcional à matriz de covariância de \bar{X} , \bar{C}_x .

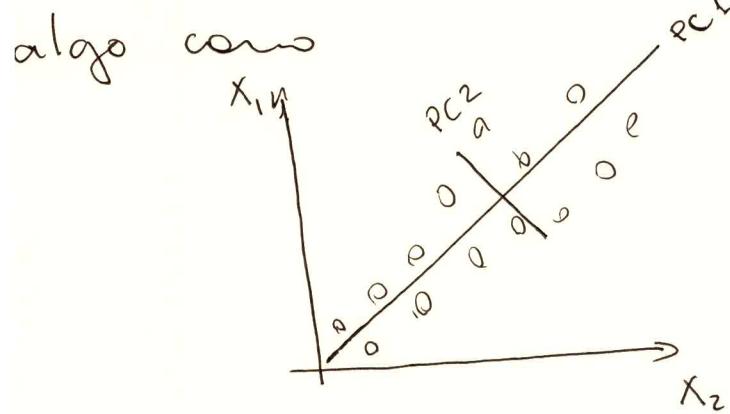
O procedimento PCA tem relação direta com a chamada Singular Value Decomposition (SVD). Para ver essa conexão, vamos usar o resultado de que dada uma matriz $\bar{A} \in \mathbb{R}^{n \times m}$, então

$$\bar{A} = \bar{U} \Sigma \bar{V}^T$$

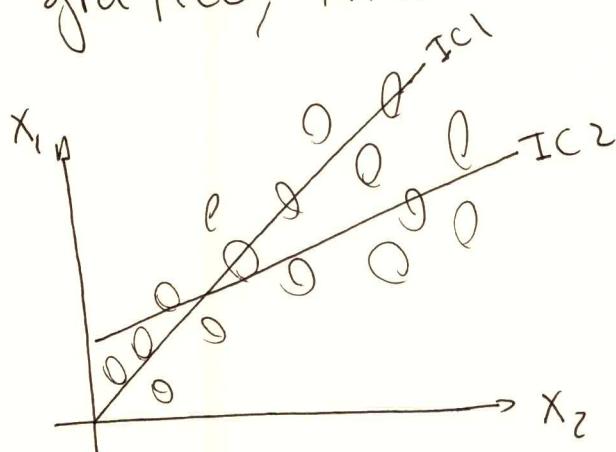
sendo $\bar{U} \in \mathbb{R}^{n \times n}$ e $\bar{V} \in \mathbb{R}^{m \times m}$ matrizes ortogonais e $\Sigma \in \mathbb{R}^{n \times m}$ uma matriz diagonal, cujos elementos são chamados valores singulares de \bar{A} e estão ordenados em ordem crescente. Nesse caso, é possível mostrar que $\bar{V} = \bar{P}^T$. (a menos da ordem inversa).

Independent component Analysis (ICA)

Similarmente ao PCA, ICA é usado para extrair a importância das features e para reduzir a dimensionalidade do espaço das features. Porém, enquanto PCA gera componentes ~~independentes~~, na medida que ICA gera componentes o mais independentes possível. Num gráfico, temos



PCA escolhe as direções de maior variância



ICA escolhe as direções que maximizam a independência dos dados.

Suponha que temos uma matriz \bar{X} cujas linhas sejam as amostras e as colunas representem diferentes sinais. ICA consiste em escrever

$$\bar{X} = \bar{S} \bar{A}^T$$

ou seja, \bar{X} é decomposta como uma mistura \bar{A} de fontes independentes \bar{S}

$$\bar{S} = [\bar{s}_1, \bar{s}_2, \dots, \bar{s}_n]$$

De modo que

$$\bar{x}_j = a_{j1}\bar{s}_1 + a_{j2}\bar{s}_2 + \dots a_{jn}\bar{s}_n$$

O problema do ICA é encontrar a matriz \bar{A} de modo que os \bar{s}_i sejam independentes. Porém, uma vez conhecidos \bar{A} , podemos encontrar sua inversa \bar{W} e obter \bar{s} via

$$\bar{s} = \bar{W} \bar{x}$$

Clustering

Clustering representa uma classe de métodos que não requer dados preclassificados para aprender padrões. Por conta disso, esse tipo de método é denominado não supervisionado. De modo geral, métodos de clustering funcionam minimizando a função

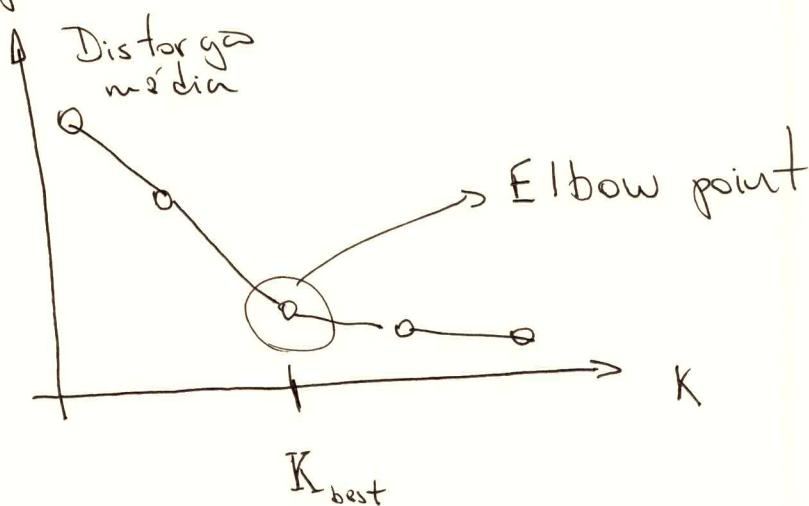
$$J = \sum_k \sum_{i \in S_k} |\bar{x}_i - \bar{\mu}_k|^2$$

na qual $\bar{\mu}_k$ representa a média dos elementos \bar{x}_i que pertencem ao grupo k (S_k). A quantidade

$$\sum_{i \in S_k} |\bar{x}_i - \bar{\mu}_k|^2$$

é chamada distância do cluster k . Um dos algoritmos mais populares para essa tarefa é o K-means, o qual busca maximizar J por meio da escolha

dos centros dos grupos ($\bar{m}_1, \bar{m}_2, \dots, \bar{m}_K$) e das partigões (S_1, S_2, \dots, S_K). A questão do método reside em como escolher o valor de K , isto é, o melhor número de partigões. Uma possibilidade é calcular a distorção média em função de K , e escolher o melhor K como sendo aquele a partir do qual a distorção não diminui mais de modo apreciável.

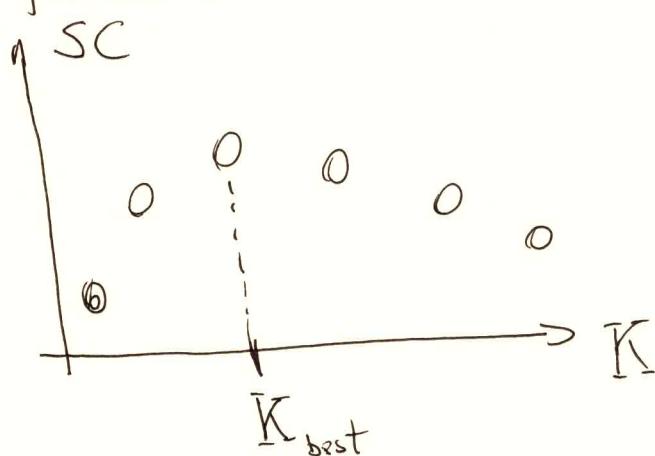


Outra métrica popular é o coeficiente de silhueta, definido como a média de

$$SC_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

sendo a_i a distância média de \bar{x}_i para os elementos de seu grupo (intra-cluster distance) e b_i a distância média de \bar{x}_i para os elementos do grupo mais próximo do seu. Note que quanto mais próximo de seu grupo for \bar{x}_i

$(a_i \approx 0)$, mais próximo de 1 sera SC_i . Por outro lado, se $a_i \gg b_i$, teremos $SC_i \approx -1$. Desse modo, valores do coeficiente de silhueta próximos de 1 indicam um bom agrupamento, enquanto valores negativos indicam um agrupamento ruim. O melhor valor de K pode ser obtido maximizando esse coeficiente.



Métodos baseados em Ensemble

Existem duas abordagens principais para combinar métodos de machine learning (classificação e regressão) e produzir novos métodos: Bagging e boosting.

Como já vimos no caso do random forest, bagging refere-se a bootstrap aggregating. Nesse caso, o dado é reamostrado e cada subamostra é usada em um classificador os quais são agregados usando o voto da maioria ou a média ponderada.

Esse procedimento é bastante efetivo quando os classificadores são muito dependentes de um único elemento, de modo que bagging ajuda a diminuir a alta variância dos classificadores individuais.

Por outro lado, boosting é muito eficiente para reduzir o alto viés de classificadores. Boosting também usa a regra da maioria ou médias ponderadas para agrupar os resultados. No entanto, esse método é iterativo e a cada passo ele se concentra mais nos dados que são classificados de maneira errada. Uma das principais implementações é o AdaBoost - Adaptive Boosting. Nesse caso, no primeiro passo, os n elementos do training set recebem o mesmo peso $D_0(i) = 1/n$. Usando esse peso, uma amostra é extraída e apresentada ao algoritmo. Da qual estimamos o erro ϵ_0 . Para atualizar os pesos $D_k(i)$, calculamos

$$\alpha_k = \frac{1}{2} \log \frac{1 - \epsilon_k}{\epsilon_k}$$

$$z_k = 2 \sqrt{\epsilon_k(1 - \epsilon_k)}$$

sendo o novo peso definido por

$$D_{k+1}(i) = \frac{1}{z_k} D_k(i) \exp \left\{ -\alpha_k y_i h_k(x_i) \right\}$$

sendo $y_i \in \{-1, +1\}$. Por fim, o classificador resultante
é dado por

$$g = \operatorname{sgn} \left\{ \sum \alpha_k h_k \right\}$$