Introducción al Diseño de Experimentos para el Reconocimiento de Patrones

Capítulo 7: Máquinas de Vectores Soporte

Curso de doctorado impartido por
Dr. Quiliano Isaac Moro
Dra. Aránzazu Simón Hurtado
Enero 2006

Capítulo 7: Máquinas de vectores soporte

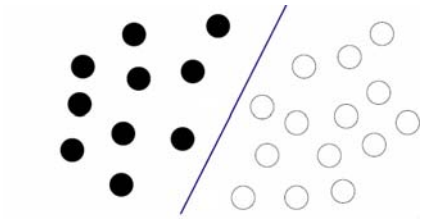
1. Introducción teórica
2. Guía práctica para la experimentación
   1. Preprocesamiento de datos
   2. Selección del modelo
   3. Validación cruzada y búsqueda de rejilla
3. Bibliografía

Introducción teórica

- Modelo de clasificación
  - Presentadas en 1992. Vapnik y Chervonenkis.
  - Técnica para clasificación de datos.
  - Existe aprendizaje.
  - Más fácil de utilizar que las redes neuronales.
  - Trabajan en un "espacio de características" inducido por el kernel.
  - El kernel realiza la separación y traslado de las muestras al espacio de características ≈producto escalar genérico

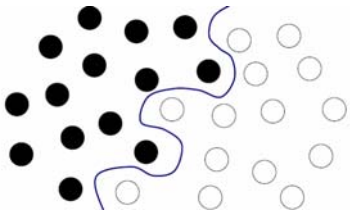
Introducción teórica

Datos linealmente separables en el espacio de las entradas



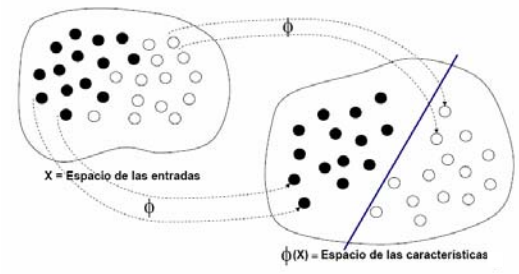
Introducción teórica

Datos no linealmente separables en el espacio de las entradas



Introducción teórica

Datos linealmente separables en el espacio de las características



Introducción teórica

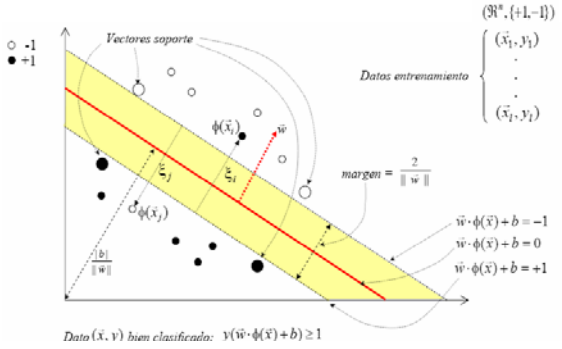
Función kernel genérica: $K(\vec{x}_i,\vec{x}_j) \equiv \langle \phi(\vec{x}_i) \cdot \phi(\vec{x}_j) \rangle$

Funciones kernel utilizadas:

- Lineal: $K(\vec{x}_i,\vec{x}_j) = \vec{x}_i \cdot \vec{x}_j$
- Sigmoide: $K(\vec{x}_i,\vec{x}_j) = \tanh(\gamma \vec{x}_i \cdot \vec{x}_j + coef)$
- Polinómica: $K(\vec{x}_i,\vec{x}_j) = (\gamma \vec{x}_i \cdot \vec{x}_j + coef)^{grado}$
- Gaussiana (RBF): $K(\vec{x}_i,\vec{x}_j) = exp(-\gamma |\vec{x}_i - \vec{x}_j|^2)$

7

Introducción teórica



Introducción teórica

Función a minimizar (sin permitir errores):

$$\frac{1}{2}\vec{w}\cdot\vec{w}$$

sujeta a: $y_i(\vec{w}\cdot\phi(\vec{x}_i)+b) \geq 1 \quad (i=1,...,N)$

Función a minimizar (permitiendo errores):

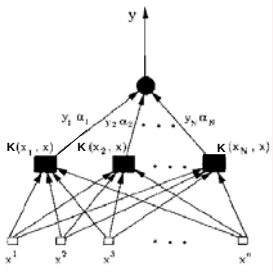
$$\frac{1}{2}\vec{w}\cdot\vec{w} + C\sum_{i=1}^{N}\xi_i$$

sujeta a: $y_i(\vec{w}\cdot\phi(\vec{x}_i)+b) \geq 1-\xi_i \quad (\xi_i \geq 0, i=1,...,N)$

9

Introducción teórica

Una vez entrenada la máquina de vectores soporte



Función de decisión

$$y = signo\left(\sum_{i=1}^{N} y_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) - b\right)$$

Pesos $y_1\alpha_1,...,y_N\alpha_N$

Transformación no lineal basada en vectores soporte $\mathbf{x}_1,...,\mathbf{x}_N$

Vector de entrada $\mathbf{x} = (x^1,...,x^n)$

10

Introducción teórica

Tipos de SVM:

Clasificación:

C-SVM (ó SVM tipo 1)

v-SVM (ó SVM Tipo 2)

El parámetro v ϵ (0, 1] y permite un control sobre el número de vectores soporte y los errores de entrenamiento

Regresión:

Ε-SVR (ó SVR Tipo 1)

v-SVR (ó SVR Tipo 2)

11

Introducción teórica

- Clasificación Multiclase (Muestra con k clases)
  - Uno-contra-resto
    - Se entrenan k clasificadores (una clase es la positiva y el resto la negativa)
    - Se predice la clase para todos los clasificadores
    - La clase asignada es aquélla con la que se consiguió mayor margen (en el caso en que se clasifique como positiva en más de un clasificador)
  - Uno-contra-uno
    - Se construyen k (k-1) / 2 clasificadores cada uno entrena datos de dos clases diferentes
    - Se usa la estrategia de votación para clasificar: cada clasificador binario se considera como un voto y se toma la clase con mayor número de votos.
- Estimación probabilística
  - Extensión de SVM para estimar la probabilidad de que pertenezca a una clase u otra

12

Guía práctica para la experimentación

- Preprocesamiento de datos
  - Cada ejemplo: vector de números reales
    - Si hay atributos categóricos -> convertirlos a datos numéricos

      Ejemplo: un atributo con tres categorías posibles {rojo, verde, azul} se puede representar como (0,0,1), (0,1,0) y (1,0,0).

      Si el número de categorías del atributo no es muy grande, esta codificación puede ser más estable que usando un único número para cada una.
  - Escalado de los datos antes de aplicar SVM
    - Ventajas:
      - Evitar que los atributos que tengan rangos grandes dominen sobre los que tengan rangos más pequeños
      - Evitar dificultades numéricas durante el cálculo
    - Escalar linealmente cada atributo al rango [-1,+1] o [0,1]
  - Usar el mismo método de escalado para los datos de entrenamiento y los de prueba

13

Guía práctica para la experimentación

- Selección del modelo
  - Decidir en qué orden probamos los kernels
  - RBF es una primera elección razonable:
    - Este kernel hace corresponder de un modo no lineal ejemplos en un espacio de dimensión mayor
    - Tiene menos hiperparámetros (C y γ) que el polinomial -> menor complejidad de modelo
    - Menos problemas numéricos

      $0 < K_{ij} \leq 1$ (los polinomiales están entre 0 e infinito y el sigmoideo no es válido bajo algunos parámetros)

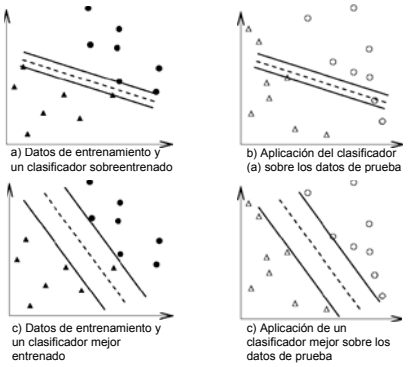
14

Guía práctica para la experimentación

- Validación cruzada y búsqueda de rejilla
  - Buscar los mejores C y γ para clasificar con precisión datos desconocidos (datos de prueba)
  - Conviene usar validación cruzada sobre los datos de entrenamiento
  - Así se evita el sobreentrenamiento

15

Guía práctica para la experimentación



a) Datos de entrenamiento y un clasificador sobreentrenado

b) Aplicación del clasificador (a) sobre los datos de prueba

c) Datos de entrenamiento y un clasificador mejor entrenado

c) Aplicación de un clasificador mejor sobre los datos de prueba

16

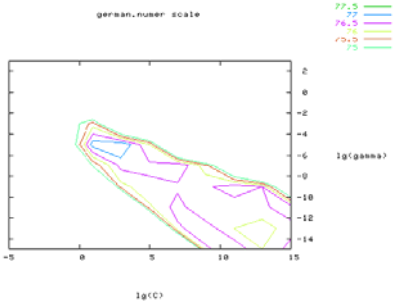
Guía práctica para la experimentación

- Validación cruzada y búsqueda de rejilla
  - Se usa búsqueda de rejilla para probar pares de (C,γ)
    - Probar secuencias crecientes exponencialmente de C y γ (por ejemplo, $C = 2^{-5}, 2^{-3},..., 2^{15}$, $\gamma = 2^{-15}, 2^{-13},..., 2^{3}$)
    - El tiempo computacional de hacerlo así en lugar de por métodos avanzados no es mucho mayor ya que sólo hay dos parámetros
    - La búsqueda por rejilla se puede paralelizar ya que cada par (C, γ) es independiente pero no muchos de los métodos avanzados
    - Primero usar una rejilla gruesa e identificar la mejor zona
    - Después usar una búsqueda de rejilla más fina sobre esa zona
  - Se entrena de nuevo todo el conjunto de entrenamiento con el mejor par (C, γ) para obtener el clasificador final

17

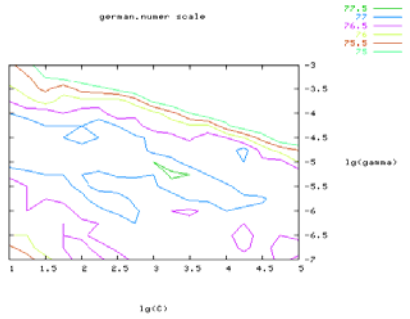
Guía práctica para la experimentación

Ejemplo de búsqueda de rejilla



Guía práctica para la experimentación

Ejemplo de búsqueda de rejilla



Guía práctica para la experimentación

- En algunas situaciones el procedimiento propuesto no es suficientemente bueno, puede que se necesiten técnicas de selección de atributos
- Este método funciona bien para datos que no tienen muchos atributos

20

Biblioteca LIBSVM

- Implementa los tipos de entrenamiento y núcleos mas comunes.
- Permite clasificación multiclase.
- Implementa el procedimiento para realizar validación cruzada
- Métodos para obtener máquinas que proporcionen además la probabilidad de la clasificación.
- Incluye técnicas para reducir el coste computacional.
- Implementaciones en C++ y JAVA.

21

Ejemplo con LIBSVM

- Física de astropartículas
  - Original sets with default parameters
    $./svm-train train.1
    $./svm-predict test.1 train.1.model test.1.predict
    Accuracy = 66.925%
  - Scaled sets with default parameters
    $./svm-scale -l -1 -u 1 -s range1 train.1 > train.1.scale
    $./svm-scale -r range1 test.1 > test.1.scale
    $./svm-train train.1.scale
    $./svm-predict test.1.scale train.1.scale.model test.1.predict
    Accuracy = 96.15%
  - Scaled sets with parameter selection
    $python grid.py train.1.scale
    . . .
    2.0 2.0 96.8922
    (Best C=2.0, γ=2.0 with five-fold cross-validation rate=96.8922%)
    $./svm-train -c 2 -g 2 train.1.scale
    $./svm-predict test.1.scale train.1.scale.model test.1.predict
    Accuracy = 96.875% (3875/4000) (clasificación)

22

Ejemplo con LIBSVM

- Vehículo
  - Original sets with default parameters
    $./svm-train train.3
    $./svm-predict test.3 train.3.model test.3.predict
    Accuracy = 2.43902%
  - Scaled sets with default parameters
    $./svm-scale -l -1 -u 1 -s range3 train.3 > train.3.scale
    $./svm-scale -r range3 test.3 > test.3.scale
    $./svm-train train.3.scale
    $./svm-predict test.3.scale train.3.scale.model test.3.predict
    Accuracy = 12.1951%
  - Scaled sets with parameter selection
    $python grid.py train.3.scale
    . . .
    128.0 0.125 84.8753
    (Best C=128.0, γ=0.125 with five-fold cross-validation rate=84.8753%)
    $./svm-train -c 128 -g 0.125 train.3.scale
    $./svm-predict test.3.scale train.3.scale.model test.3.predict
    Accuracy = 87.8049% (36/41) (clasificación)

23

Bibliografía

- C.J.C. Burges, *A Tutorial on Support Vector Machines for Pattern recognition,* 1998
- N. Cristianini y J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods.* Cambridge University Press, 2000
- C.-W. Hsu, C.-C. Chang, C.-J. Lin, *A Práctical Guide to Support Vector Classification,* 2003.
- C.C. Chang y C.J. Lin, *LIBSVM: a Library for Support Vector Machines*, Marzo 2004.

24