

Support Vector Machine Approach to Automated Cardiac SPECT diagnosis

8th Semester Project Presentation

Varun B Patil

2012

Dept. of Computer Science & Engineering

Sri Jayachamarajendra College of Engineering, Mysore



“Support Vector Machine Approach to Automated Cardiac SPECT diagnosis”

by

Dharmatheja Bhat (4JC08CS023)

Varun B Patil (4JC08CS118)

Supreeth Nag V.P (4JC08CS107)

Vinay M (4JC08CS124)

Under the guidance of

Dr. M.P Pushpalatha

Associate Professor,

Department of Computer Science & Engineering,

SJCE, Mysore

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

June 2012

Algorithm

- The SVM machine learning algorithm aims to find the best relationship between the features and its class so as to minimize the error in classification.
- The algorithm we employ is called **SMO(Sequential Minimal Optimization)** and it finds the best possible class to which a particular instance of SPECT data belongs to. The term "best class" here means the class which minimizes the classification error or likewise improves the classification accuracy.
- The two classes possible for any SPECT instance are: indicates possible cardiac disease, does not indicate cardiac disease(future enhancements to the project will include more fine grained classification or multiclass classification). w SVM works for non-linear decision boundaries:

Algorithm

- The SVM algorithm we have implemented can either use a **Linear Kernel** or a **Gaussian Kernel**.
- A **Linear Kernel** can be imagined to be a **straight-line seperation** between the classes (or a planar seperation in the case of multidimensional data).
- The **Gaussian Kernel** on the other hand can be visualized as a **non-linear seperation** between classes which also means a non-planar or curved seperation between classes of multidimensional data.

$$K_{linear}(x^i, x^j) = x^i * x^j$$

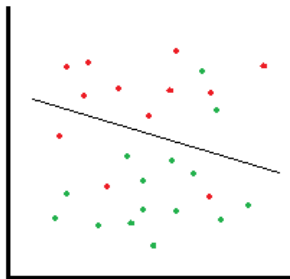
$$K_{gaussian}(x^i, x^j) = \exp\left(-\frac{\|x^i - x^j\|^2}{2\sigma^2}\right)$$

Algorithm

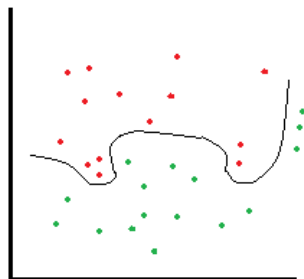
Which one do you use ?

- With choice comes the problem of deciding which one to use for the particular problem at hand.
- A Linear Kernel is usually employed when the number of features is more than the number of samples and a Gaussian Kernel is employed when the number of samples is more than the number of features.
- And also, the kernel used depends on whether the data itself is linearly-seperable or not.

Algorithm



Linear Kernel



Gaussian Kernel

Algorithm

The cost function in SVM is:

$$\sum_{i=1}^m (y^i * cost_1(z) + (1 - y^i) * cost_0(z)) + \left(\frac{\lambda}{2} * \sum_{i=1}^m (\Theta_i^2)\right)$$

where $z = \Theta^T x$

- $cost_1()$ graph has 0 value for $z \geq 1$ and a straight line for $z < 1$
- $cost_0()$ graph has 0 value for $z \leq -1$ and a straight line for $z > -1$

So, to minimize cost function, we want z to be ≥ 1 when the class is 1 and we want z to be ≤ -1 when the class is 0.

Algorithm

By convention, we optimize:

$$J = (C * A) + B$$

where

$$A = \sum(y^i * cost_1(z) + (1 - y^i) * cost_0(z))$$

$$B = \frac{1}{2} \sum(\Theta^2)$$

$$cost_1(z) \approx -\log(z)$$

$$cost_0(z) \approx -\log(1 - z)$$

Algorithm

- We see that the definitions for $cost_0()$ and $cost_1()$ above are nothing but straight line approximations to the logarithmic curves. This is done in order to reduce execution time.
- By minimizing the above equation $J = C * A + B$, we learn the optimal parameters Θ
- Once we know Θ , we can predict the class as follows:

$if(\Theta^T x) \geq 0$, predict class = 1

$if(\Theta^T x) \leq 0$, predict class = 0

Algorithm

- We noted earlier that C and σ are two important parameters that need to be decided by examining the Cross Validation set.
- Informally, the C parameter is a positive value that controls the penalty for misclassified training examples. A large C parameter tells the SVM to try to classify all the examples correctly (large margin). However, large C means that it is more susceptible to outliers and does not give a natural fit for the data.

Algorithm

- SVM's are large margin classifiers meaning that the separation between classes is maximum. In other words, the distance of the separating boundary from the nearest data point is large. Such a large margin SVM classifier is obtained by setting C to a large value.
- A large value for C value means that the term A should be made close to 0 to make J less. In other words, if C is made large, then we only need to optimize $\frac{1}{2}\Sigma(\Theta^2)$
- The only problem is that large C means greater susceptibility to outliers. So we have no choice but to make C small.

Algorithm

- σ is a parameter of the Gaussian Kernel which determines how fast the "similarity measure" approaches 0 for data points that are further apart. If σ is large, the similarity changes slowly and can cause underfitting. If σ is small, the similarity changes fast and can cause overfitting.
- In practice, C and σ are selected from a set of highly possible values for both C and σ where the values vary approximately by a factor of 10, by testing all possible combinations of C and σ for the one that gives the least error in classification of the CV set. In our implementation, the values of C and σ take on the following set of values

$$C = [0.01, 0.03, 0.1, 0.3, 1, 3, 10, 30]$$

$$\sigma = [0.01, 0.03, 0.1, 0.3, 1, 3, 10, 30]$$

How SVM works for non-linear decision boundaries:

- Pick landmarks l_1, l_2, \dots (usually these are nothing but the data points themselves).
- Now the similarity (or kernel) function = $\exp\left(-\frac{\|x - l_i\|^2}{2\sigma^2}\right)$
- The hypothesis here is $\Theta_1 * f_1 + \Theta_2 * f_2 + \Theta_3 * f_3 + \dots$
- f_1 is called the similarity of the data points to the landmark 1 (i.e., l_1) and so on.
- When the data point is very close to the landmark, we get the corresponding feature almost equal to 1. When the data point is very far from the landmark, we get the corresponding feature almost equal to 0. That is, as the data point moves farther from the landmark, the value of feature decreases.

Algorithm

- How fast the feature value decreases depends on the σ for gaussian kernels.
- If σ is large, the feature value decreases slowly as the data point moves away from the landmark. This is called high bias or lower variance(underfitting).
- If σ is small, the feature value decreases fast as the data point moves away from the landmark. This is called high variance or lower bias(overfitting).
- Using the kernels approach, the cost function is also changed. Instead of $z = \Theta^T x$, we use $z = \Theta^T f$