INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, ALLAHABAD

VI Semester BTech in Information Technology

Report - Group Assignment 2

Data Mining and Warehousing

Group Members:
Mudit Goyal (IIT2018132)
Shiv Kumar (IIT2018134)
Moksh Grover (IIT2018186)
Karan Chatwani (IIT2018194)
Shubham (IIT2018200)

INTRODUCTION

A Support Vector Machine, or SVM, is a supervised learning algorithm that is non-parametric. SVMs can be used for sorting, regression, and a variety of other functions. Many hypotheses make use of Support Vector Machines to achieve their goals. We are also unaware of their learning success on restricted groups of distributions despite these various attempts. It is not easy to wipe out the clutter.To be clear, it is unclear what conditions SVM will guarantee high learning rates in. For such non-trivial distributions, certain techniques such as Tsybakov's noise assumption and local Rademacher averages can help achieve learning rates up to the order of n-1. The Gaussian RBF kernel's approximation properties can also be calculated by using a geometric assumption for the dispersion.

We establish learning rates to the Bayes risk for support vector machines (SVMs) using a regularization sequence
$\lambda n=n-\alpha$, where $\alpha \in (0,1)$ is arbitrary. Under a noise condition recently proposed by Tsybakov these rates can become faster than $n - 1/2$. In order to deal with the approximation error we present a general concept called the approximation error function which describes how well the infinite sample versions of the considered SVMs approximate the data-generating distribution. In addition we discuss in some detail the relation between the “classical” approximation error and the approximation error function. Finally, for distributions satisfying a geometric noise assumption we establish some learning rates when the used RKHS is a Sobolev space.

ALGORITHM

Step 1: Draw randomly N1 training samples {zi =(xi,yi) ,i=1,2,3,..N1} from the data set D.Use the SVMC algorithm to train the samples of size N1 ,and obtain a preliminary learning model f0,set m+=0 and m-=0.

Step 2: Draw randomly a sample from D and denote it by the current sample zt.Set m+=m+ +1
If the label of zt is +1 ,or set m- = m- +1 if the label of zt is -1.
Step3 : Draw randomly a sample from D and denote it by the candidate z*.
Step 4 : Calculate the ratio α of e^(-l(f0,z)) at the candidate sample(zx) and the current sample (zt), α= e^(-L(f0,zx))/e^(-L(f0,zt)),where L(f,z)=(f(x)-y)^2.
Step5: if &>=1 accept the candidate sample z*. Set Z(t+1) = z(*), m+ = m+ + 1 if the label of zt is +1.or set m_= m_+1 if the label if zt is -1.
Step 6: If m+ < m/2 or m_ < m/2 then return to Step 3 : else stop it. Here m is the number of samples