Name – PRABHAKAR KUMAR

Roll – IRM2017008

Heart Disease classification using SVM(SMO) with Radial Kernel

Support Vector Machine is a supervised machine learning algorithm which can be used for both classification or regression challenges. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features we have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.

SVM algorithms use a set of mathematical functions that are defined as the kernel. The function of kernel is to take data as input and transform it into the required form. The kernel functions return the inner product between two points in a suitable feature space. Thus by defining a notion of similarity, with little computational cost even in very high-dimensional spaces.

In order to get a proper implementation of the SMO or the Sequential Minimal Optimization algorithm, we have used the following simplified version of SMO, whose pseudo code is given as below:

- Initialize $\alpha_i = 0, \forall i, \quad b = 0$.
- Initialize $passes = 0$.
- **while** $(passes < max\_passes)$
  - $num\_changed\_alphas = 0$.
  - **for** $i = 1, \ldots m$,
    - Calculate $E_i = f(x^{(i)}) - y^{(i)}$ using (2).
    - **if** $((y^{(i)}E_i < -tol \;\; \&\& \;\; \alpha_i < C) \;||\; (y^{(i)}E_i > tol \;\; \&\& \;\; \alpha_i > 0))$
      - Select $j \neq i$ randomly.
      - Calculate $E_j = f(x^{(j)}) - y^{(j)}$ using (2).
      - Save old α's: $\alpha_i^{(\text{old})} = \alpha_i$, $\alpha_j^{(\text{old})} = \alpha_j$.
      - Compute L and H by (10) or (11).
      - **if** $(L == H)$
        continue to next i.
      - Compute η by (14).
      - **if** $(\eta >= 0)$
        continue to next i.
      - Compute and clip new value for α_j using (12) and (15).
      - **if** $(|\alpha_j - \alpha_j^{(\text{old})}| < 10^{-5})$
        continue to next i.
      - Determine value for α_i using (16).
      - Compute b_1 and b_2 using (17) and (18) respectively.
      - Compute b by (19).
      - $num\_changed\_alphas := num\_changed\_alphas + 1$.
    - **end if**
  - **end for**
  - **if** $(num\_changed\_alphas == 0)$
    $passes := passes + 1$
  - **else**
    $passes := 0$
- **end while**

Once implemented we have used different values of C and Gamma and calculated the different metrics like the accuracy over the testing data and also the area under the ROC curve generated by the trained model, and the result obtained is as follows:

Value of C	Gamma	AUC	Accuracy
1	1	0.5425	0.5494
1	25	0.5629	0.5714
1	100	0.5561	0.5604
10	25	0.5935	0.5934
10	100	0.5952	0.5934