Report for Project SVM

11610511 Shiyi Li

December 2018

1 Preliminaries

In machine learning, support vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Nowadays, SVM has been widely used in text categorization, image categorization, Handwritten character recognition and some other applications.

2 Methodology

The basic definition of each variables will be introduced in the methodology.

2.1 Representation

Several variables should be defined in advance in the algorithm.

- Gradient descend
  - filename: address of training data
  - x: training data
  - y: labels of training data
  - epochs:
  - learning_rate: learning rate of
  - w:
- SMO: Sequential minimal optimization
  - filename: address of training data
  - data: training data
  - label: labels of training data
  - C: regularization parameter
  - tolar: numerical tolerance
  - column: length of dataset
  - α: Lagrange multipliers for solution
  - b: threshold for solution
  - E: difference between predict value and true value
  - ker: training data after kernel transition

2.2 architecture

For Gradient descend, we use loss function to evaluate to predict value and true value. If the loss is larger than 0, gradient descend is used to renew the function to decrease the difference between predict value and true value.

- Class SVM:
  - get_loss: evaluate the difference between predict value and true value
  - cal_sgd: gradient descend function
  - train: train data and renew the predict function
  - predict: get predicted value

For SMO, kernel transition is used to map the data to higher dimension at first. Use KKT condition to update Lagrange multipliers and threshold to get better predict function[4].

- Class SMO:
  - add_edge: record the edge and its weight
  - kernel: kernel transition for data

- cal_value: calculate the difference between predicted value and true value
- update_value: store the difference
- select_j: select one Lagrange multipliers
- KKT: update Lagrange multipliers and threshold
- update_a: update Lagrange multipliers
- update_b: update threshold
- smo: control the iteration and condition of smo
- predict: get predicted value

2.3 Algorithm

Here shows the exact algorithm of Gradient descend and SMO.

Algorithm 1 Gradient descend

Input: max_iteration

Output: the lowest couple x_n, $f(x_n)$ found

choose x_0 randomly
while $f(x_{n+1} - f(x_n)) > 0$ **do**
  choose a decreasing y_n (generally $1/n$)
  compute $x_{n+1} = x_n - t_n \bigtriangledown f(x_n)$
end while
do some random restarts

(1): $f(x) = \sum_{i=1}^{m} \alpha_i y^i < x^i, x > +b$
(2): If $y^i \neq y^j$, $L = max(0, \alpha_j - \alpha_i)$, $H = min(C, C + \alpha_j - \alpha_i)$
(3): If $y^i = y^j$, $L = max(0, \alpha_j + \alpha_i - C)$, $H = min(C, \alpha_j + \alpha_i)$
(4): $\eta = 2 < x^i, x^j > - < x^i, x^i > - < x^j, x^j$
(5): $\alpha_j = \alpha_j - y^i(E_i - E_j)/\eta$
(6): $L \geq \alpha_j \leq H$
(7): $\alpha_i = \alpha_i + y^i y^j ({\alpha_j}^{old} - \alpha_j)$
(8): $b_1 = b - E_i - y^i(\alpha_i - {\alpha_i}^{old}) < x^i, x^i > -y^j(\alpha_j - {\alpha_j}^{old}) < x^i, x^j >$
(9): $b_2 = b - E_j - y^i(\alpha_i - {\alpha_i}^{old}) < x^i, x^j > -y^j(\alpha_j - {\alpha_j}^{old}) < x^j, x^j >$
(10): If $0 < \alpha_i < C$, $b = b_1$. If $0 < \alpha_j < C$, $b = b_2$. $b = (b_1 + b_2)/2$, otherwise

Algorithm 2 SMO

Input: C, tolar, max_iteration

Output: α, b

initialize $\alpha_i = 0$, $b = 0$
initialize iteration=0
while iteration ¡ max_iteration **do**
  num_changed_alphas = 0
  for $i = 1, ..., m$ **do**
    calculate $E_i = f(X^i) - y^i$ using (1)
    if $((y^i E_i < -tolar \& \alpha_i < C) || (y^i E_i > tolar \& \alpha_i > 0))$ **then**
      select $j \neq i$ randomly
      calculate $E_j = f(x^j) - y^j$ using (1)
      save old α's: ${\alpha_i}^old = \alpha_i$, ${\alpha_j}^old = \alpha_j$
      compute L and H by (2) or (3)
      if $L == H$ **then**
        continue to next i
      end if
      compute η by (4)
      if $\eta \geq 0$ **then**
        continue to next i
      end if
      Compute and clip new value for α_j using (5) and (6).
      if $|\alpha_j - {\alpha_j}^old| < 10^{-5}$ **then**
        continue to next j
      end if
      determine value for α_i using (7)
      Compute b_1 and b_2 using (8) and (9) respectively.
      Compute b by (10)
      $numchangedalphas = numchangedalphas + 1$.
    end if
  end for
  if ($numchangedalphas == 0$ **then**
    $iteration = iteration + 1$
  else
    $iteration = 0$
  end if
end while

3 Empirical Verification

3.1 Dataset

Dataset includes: train_data.txt. The data and label in the file is used to train the model. The data is

Table 1: Performance measure

Dataset	Algorithm	Correct rate
train_data.txt(linear)	GD	0.997
train_data.txt(linear)	SMO	0.888

used to

3.2 Performance measure

The version of Python is 3.7.0. The data and label in the file is used to train the model. The data is used to evaluate the performance again through comparing the predicted value and the true value. We get correct rate finally.

3.3 Hyperparameters

In gradient descend, learning_rate = 0.01 and epochs = 200. In SMO, θ for kernel transition is 20, max iteration is 10000, C is 200 and tolar is 0.00001[3].

3.4 Experimental results

Experimental result is showed in Table 1.

3.5 Conclusion

If the dataset is linear and parameters of gradient descend are well selected, gradient descend performs better than SMO, no matter in correct rate and time complexity. However, gradient descend can not deal with nonlinear dataset[2]. While SMO maps the dataset to higher dimension and the hyper plane can divide the dataset better.[1]

References

[1] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[2] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks for anomaly detection in time series. In *Proceedings*, page 89. Presses universitaires de Louvain, 2015.

[3] Sandra Oliveira, Friderike Oehler, Jesús San-Miguel-Ayanz, Andrea Camia, and José MC Pereira. Modeling spatial patterns of fire occurrence in mediterranean europe using multiple regression and random forest. *Forest Ecology and Management*, 275:117–129, 2012.

[4] John Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. 1998.