MLSM 2020 - Quiz 3
02/10/2020 - 14/10/2020 at 23:59

Questions based on Lecture 4 and 5

(1) (1.0 pt.) In the hard-margin SVM we have the constraints $y_i \mathbf{w}^T \mathbf{x}_i \geq 1$ for all $i = 1, \ldots, m$. For a pair, $(y, \mathbf{x})$, of a new sample example we have $y\mathbf{w}^T\mathbf{x} < 1$ but $y\mathbf{w}^T\mathbf{x} > 0$. What does it mean?

(1) This example is wrongly classified.
(2) *** This example is classified correctly but it is within the margin.
(3) We can not decide on which class contains this example.

Answer: An example is correctly classified if it satisfies the inequality $y\mathbf{w}^T\mathbf{x} > 0$, thus the positive examples are above, and the negative examples are below the separating hyperplane. Since the SVM is able to provide that $y\mathbf{w}^T\mathbf{x} > 0$, thus $\mathbf{w}$ is well defined, in this way all examples can be classified.

(2) (1.0 pt.) In the soft-margin SVM we have the constraints $y_i \mathbf{w}^T \mathbf{x}_i \geq 1 - \xi_i$ for all $i = 1, \ldots, m$. If a pair, $(y_i, \mathbf{x}_i)$, of a training example is correctly classified which set of values contains the corresponding slack variable ξ_i?

(1) $\xi_i < 0$
(2) $\xi_i = 0$
(3) *** $0 \leq \xi_i < 1$

Answer: We have $\xi_i \geq \max(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i)$, therefore $\xi_i \geq 0$. $\xi_i < 1$ because if $\xi_i \geq 1$ then the example is on the separating hyperplane, or it is on the opposite side of that hyperplane, thus it is not classified correctly,

(3) (1.0 pt.)

We are given only positive examples in an SVM problem. We might include a new example in which the input is the zero vector, $\mathbf{0}$, and its label is equal to -1. It is called as one-class classification problem.

What does the SVM do if the new example, $\mathbf{0}$, is not included? Assume that there is no bias term in the SVM.

(1) *** It tries to separate all positive examples from the $\mathbf{0}$ even if it is not included into the examples.
(2) The result has no any meaning.
(3) All slacks will be 0.

Answer: If no bias then the separating hyperplane will contain the zero vector. The SVM algorithm will stop when the optimality conditions are satisfied. In that case, all the points satisfy the constraints $y_i \mathbf{w}^T \mathbf{x}_i \geq 1 - \xi_i, i = 1, \ldots, m$. At the optimum there are also to parallel hyperplanes defined by the $\mathbf{w}$, one is which contains the zero vector, the separating hyperplane, and the other one corresponds to the margin and it is defined by the constraints $y_i \mathbf{w}^T \mathbf{x}_i \geq 1 - \xi_i, i = 1, \ldots, m$. There is only one margin related hyperplane because we have only positive cases. The two hyperplanes are separated by the margin, which is 1 after scaling by the inverse of the length of $\mathbf{w}$. The zero vector is on the separating hyperplane, and all positive examples are above or on the margin related hyperplane in sense of the soft margin by the optimality, therefore they are separated from the zero vector by the margin.

(4) (2.0 pt.)

Let 6 points be given in the plane, $X = \{(-1,0),(2,1),(2,-1),(1,0),(-2,-1),(-2,1)\}$, and the corresponding labels in the same order $y = \{1,1,1,-1,-1,-1\}$. Compute $\mathbf{w}$ by the algorithm given by the Slide "Stochastic gradient descent algorithm for soft-margin SVM ". Process the examples in the given fix order instead of randomly drawing them. The parameters defined on the slide are set in the following way. The penalty weight is set by $\lambda = 1$, and the learning speed is given by $\eta = 0.1$.

Which of these coordinates can give the solution for $\mathbf{w}$? Round the numbers up to 2 decimals, and take the closest one.

(1) $(-0.54, 0.2)$
(2) $(0.66, 0.12)$
(3) *** $(0.52, -0.02)$

```
## ######################################################
import numpy as np
## Load the data
X=np.array([(-1,0),(2,1),(2,-1),(1,0),(-2,-1),(-2,1)])  ## input
y=np.array([1,1,1,-1,-1,-1])                            ## output
mexample,ndim=X.shape                                   ## size of the data
## initialize w, eta, lambda
w=np.zeros(ndim)     ## normal vector of the hyperplane
eta=0.1              ## learning speed, step size
xlambda=1.0          ## balancing constant between loss and regularization
for i in range(mexample):
  ## process all sample examples sequentially
  if y[i]*np.dot(w,X[i])<1:       ## is it under the margin
    Jgrad=-y[i]*X[i]+xlambda*w    ## compute the gradient
  else:
    Jgrad=xlambda*w               ## compute the gradient for correctly classified case
  ## update w
  w=w-eta*Jgrad                   ## update the normal vector
  print(w)
print('Normal vector of the hyperplane:',w)
## ######################################################
```