Mohammed J. Zaki

Dmcourse /
Assign4

Assign4 (Due Date: 5th Nov, before midnight)

This assignment has two parts, the first is for all sections, whereas the second is for CSCI6390 only .

Logistic Regression

You will implement the stochastic gradient descent (SGD) algorithm for logistic regression, given a training set and a testing set. You may assume that the last column is the class or dependent variable. The fields are all comma separated. The training data is thus a set of n points $\{(\mathbf{x}_i, y_i)\}$ for $i = 1, 2, \ldots, n$. Or we can denote all points as the $n \times d$ matrix $\mathbf{X}$, and the dependent variable as the $n \times 1$ vector $\mathbf{y}$.

The gradient of the weight vector $\mathbf{w}$ at a training point $\mathbf{x_k}$ is given as

$$\nabla_k = f(-y_k \mathbf{w}^T \mathbf{x}_k)\ y_k\ \mathbf{x}_k$$

where $f(z)$ is the logistic function, given as

$$f(z) = \frac{1}{1 + exp(-z)}$$

The SGD algorithm is therefore as follows:

Input: $\mathbf{X}, \mathbf{y}, \epsilon, \eta$
$\mathbf{X} = (\mathbf{X}, \mathbf{1})$ //Map points to 1 higher dimension by adding a column of 1s
$d = d + 1$
$\mathbf{w}$ = random d dimensional vector in the range [0,1]
repeat
  $\mathbf{w}_{prev} = \mathbf{w}$
  for $k = 1, 2, \ldots, n$ in random order **do**
    $\mathbf{w} = \mathbf{w} + \eta \cdot \nabla_k$
until $\|\mathbf{w} - \mathbf{w}_{prev}\| \leq \epsilon$

Next, predict the class for the points $\mathbf{z}_j$ in the testing set, using the following approach:

$\hat{y}_j = +1$, if $f(\mathbf{w}^T \mathbf{z}_j) \geq 0.5$ or
$\hat{y}_j = -1$, if $f(\mathbf{w}^T \mathbf{z}_j) < 0.5$.

After this compute the accuracy of the prediction using the true class y_i for each test point $\mathbf{z}_j$.

Kernel Regression (CSCI6390 Only)

You will implement kernel regression using linear , quadratic (homogeneous), or gaussian kernels.

Recall that the weight vector in feature space is given as $\mathbf{w} = \sum_{i=1}^{n} \alpha_i \phi(\mathbf{x}_i$

There is a closed form solution for computing the α_i values, i.e., for $\vec{\alpha}$ when considering all n points, given as:

$$\vec{\alpha} = \mathbf{K}^{-1}\mathbf{y}$$

You can use np.linalg.pinv to compute the inverse of the kernel matrix $\mathbf{K}$.

After computing $\vec{\alpha}$ use the testing set to predict the $\hat{y}_j$ values for each test point $\mathbf{z}_j$, as follows:

$$\hat{y}_j = \sum_{i=1}^{n} \alpha_i \mathbf{K}(\mathbf{x}_i, \mathbf{z}_j)$$

Finally, compute the "accuracy" by checking if $sign(\hat{y}_j)$ matches the true value y_j for test case $\mathbf{z}_j$; we are assuming here that the dependent attribute is a class variable with values $\{+1, -1\}$.

What to turn in

Write a script named **assign4-LAST-FIRST.py** where **LAST** and **FIRST** are your last and first names, respectively. The script will be run as fo llows:
assign4.py TRAIN TEST eps eta (**for CSCI4390**) or
assign4.py TRAIN TEST eps eta [linear | quadratic | gaussian] [spread] (**for CSCI6390**)
The spread value is used only for gaussian kernel.

Show your results on the following training and testing files:
Attach:Concrete_Data_RNorm_Class_train.txt for training set, and
Attach:Concrete_Data_RNorm_Class_test.txt for testing set. Save them to a text file and submit with script. Print out the $\mathbf{w}$ value for logistic regression, and print out the accuracy values.

Page last modified on November 04, 2016, at 12:52 PM