Homework Assignment 2
Softmax and Nearest Neighbor Classfication

Kailin Zheng kz882

March 2019

Submission Instructions You must typeset the answers to the theory questions using LaTeXor Microsoft Word and compile them into a single PDF file. For the programming assignment, complete the Jupyter notebook named `HW2.ipynb`. Create a ZIP file containing both the PDF file and the completed Jupyter notebook. Name it ⟨`Your-NetID`⟩`_hw2.zip`. Submit the ZIP file on NYU Classes. The due date is **Mar 5, 2019 11:55 PM**

1. Model Selection (10 points) Consider that we are learning a logistic regression $M^{(1)}$ and a support vector machine $M^{(2)}$, and we have partitioned the data into three subsets: a training set D_{train}, a validation set D_{val}, and a test set D_{test}.

The two models are iteratively optimized on D_{train} over T steps, and now we have T logistic regression parameter configurations (i.e. weights and biases) $M_1^{(1)}, M_2^{(1)}, \ldots, M_T^{(1)}$ and T support vector machine configurations $M_1^{(2)}, M_2^{(2)}, \ldots, M_T^{(2)}$, all with different parameters.

We now evaluate the expected cost for all the $2T$ models on training set, validation set, and test set. So we have $6T$ quantities $\tilde{R}^{(i)}_{\text{train},t}$, $\tilde{R}^{(i)}_{\text{val},t}$, $\tilde{R}^{(i)}_{\text{test},t}$ where $i \in \{1,2\}$ and $t = 1, \ldots, T$.

1. Which i and t should we pick as the best model? (5 points)
2. How should we report the generalization error of the model? (5 points)

Answer:

1. For logistic regression model, we should choose $p \in 1,2,...T$ such that $\tilde{R}^{(1)}_{val,p}$ is the smallest among all $\tilde{R}^{(1)}_{val}$.

For SVM, we should choose $q \in 1,2,...T$ such that $\tilde{R}^{(2)}_{val,q}$ is the smallest among all $\tilde{R}^{(2)}_{val}$.

To choose between the best logistic regression model and the best SVM, choose the smaller between $\tilde{R}^{(1)}_{test,p}$ and $\tilde{R}^{(2)}_{test,q}$.

2. Sanyam says on Piazza:

Generalization error means how well does your model perform on novel data (unseen during training/validation)?

The performance metric would in this case involve the cost. You have all those costs computed, how do you report the "Expected Cost" (in the limit of data).

We can never know the exact generalization error since we don't know every input/output in the reality. But we can make approximations.

We can only know empirical cost, but we have to get an idea of expected cost (report generalization error) and to prevent over-fitting. To prevent over-fitting, we can use cross-validation or early stopping to get a better empirical cost, which is equivalent to report the best generalization error as much as we can.

2. Multi-class loss gradient (10 points) The distance function of multinomial logistic regression is defined as (lecture notes Equation 1.21)

$$\begin{aligned}\Delta(y^*, \mathbf{M}, \mathbf{x}) &= -\log \mathbf{p}_{M^\star(\mathbf{x})} \\ &= -\mathbf{a}_{y^\star} + \log \sum_{k=1}^{K} \exp \mathbf{a}_k\end{aligned}$$

where $\mathbf{x} \in \mathbb{R}^d, \mathbf{W} \in \mathbb{R}^{K \times d}$, $\tilde{\mathbf{x}} = [\mathbf{x}; 1]$ (vector concatenation). $\mathbf{M}^\star(\mathbf{x})$ represents the output category from the reference machine, subscripts represent component of the vector, $\mathbf{a} = \mathbf{W}\tilde{\mathbf{x}}$ and $\mathbf{p} = softmax(\mathbf{a})$. Derive its gradient with respect to the weight vector $\mathbf{W}$ step-by-step.

Answer:

$$\begin{aligned}\Delta(y, M, x) &= -\log p_{M^*(x)} \\ &= -a_y^* + \log \Sigma_{k=1}^{K} \exp(a_k) \\ &= -W_{y^*}^T \tilde{x} + log \Sigma_{k=1}^{K} \exp(W_k^T \tilde{x})\end{aligned} \tag{1}$$

Gradient descent with respect to W_{y^*} is:

$$\begin{aligned}\frac{\partial \Delta(y, M, x)}{\partial M_{y^*}} &= -\tilde{x} + \frac{1}{\Sigma_{k=1}^{K} \exp(W_k^T \tilde{x})} \exp(W_{y^*}^T \tilde{x}) \tilde{x} \\ &= (-1 + \frac{1}{\Sigma_{k=1}^{K} \exp(W_k^T \tilde{x})} \exp(W_{y^*}^T \tilde{x})) \tilde{x} \\ &= -(1 - \frac{1}{\Sigma_{k=1}^{K} \exp(W_k^T \tilde{x})} \exp(W_{y^*}^T \tilde{x})) \tilde{x}\end{aligned} \tag{2}$$

Since by definition, $\frac{1}{\Sigma_{k=1}^{K} \exp(W_k^T \tilde{x})} \exp(W_{y^*}^T \tilde{x}) = p(C = y^* | x)$, substitute it into equation (2), we have:

$\frac{\partial\Delta(y,M,x)}{\partial M_{y^*}} = -(1-p(C=y^*|x))\tilde{x}.$

Gradient descent with respect to W_y (meaning it's the wrong label) is:

$$\begin{aligned}\frac{\partial\Delta(y,M,x)}{\partial M_y} &= -\tilde{x} + \frac{1}{\Sigma_{k=1}^{K}\exp(W_k^T\tilde{x})}\exp(W_y^T\tilde{x})\tilde{x} \\ &= (-0 + \frac{1}{\Sigma_{k=1}^{K}\exp(W_k^T\tilde{x})}\exp(W_y^T\tilde{x}))\tilde{x} \\ &= -(0 - \frac{1}{\Sigma_{k=1}^{K}\exp(W_k^T\tilde{x})}\exp(W_y^T\tilde{x}))\tilde{x}\end{aligned} \tag{3}$$

Since by definition, $\frac{1}{\Sigma_{k=1}^{K}\exp(W_k^T\tilde{x})}\exp(W_y^T\tilde{x}) = p(C=y|x)$, substitute it into equation (3), we have:
$\frac{\partial\Delta(y,M,x)}{\partial M_y} = -(1-p(C=y|x))\tilde{x}.$

Then we can combine them into a single vector equation

$$W\Delta(y,M,x) = -(y^* - p)\tilde{x}^T$$

where

$$y^* = \begin{bmatrix} 0 \\ 0 \\ \dots \\ 1 \\ \dots \\ 0 \end{bmatrix} \tag{4}$$

where 1 is in the y^* row.

3. Nearest-neighbour classifier by RBFN (20 points) A nearest-neighbour classifier can be constructed as a radial basis function network (lecture notes section 1.7) by selecting all the input vectors in a training set $D_{tra} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)\}$ as basis vectors. In a multi-class classification setting, with a set of $\mathscr{C} = \{0, 1, \dots, K-1\}$ categories, provide a description on how a weight matrix could be built.

Answer:

Assume we have N basis vectors $r^1, r^2, ..., r^N$ with respect to the centroids of K clusters. With these basis vectors, we now transform each input vector x into an N-dimensional vector:

$$\phi(\mathbf{x}) = \begin{bmatrix} \exp\left(-(\mathbf{x}-\mathbf{r}^1)^2\right) \\ \vdots \\ \exp\left(-(\mathbf{x}-\mathbf{r}^N)^2\right) \end{bmatrix} \tag{5}$$

This transformation of every training input vector is equivalent to building a new training set consisting of pairs of a transform input vector and its correct class:

$D_{tra} : (\phi(x_1), y_1)...(\phi(x_N), y_N)$.

Once the transformation is done, as we have the new set D_{tra} and $\phi = 0, 1, ..., N-1$, the category distribution is defined over N event with set of L probabilities. So we can write weight vector w: $W = argmax\Sigma(logp(y_i \phi x_i)$

4. Programming Assignment (40 points) Complete the implementation of the radial basis function network and k-nearest neighbor classification using Python and scikit-learn. The completed notebooks must be submitted together with the answers to the questions above.

Please follow all instructions in Jupyter notebook file. When submitting Jupyter notebooks, make sure to save outputs as well.