DS-UA 301
Advanced Topics in Data Science
Instructor: Parijat Dube
TAs: Astha Gupta, Rahul Sankar
Due: October 12, 2022

Homework 2

Problem 1 - *Perceptron* 15 points

Consider a 2-dimensional data set in which all points with $x_1 > x_2$ belong to the positive class, and all points with $x_1 \leq x_2$ belong to the negative class. Therefore, the true separator of the two classes is linear hyperplane (line) defined by $x_1 - x_2 = 0$. Now create a training data set with 10 points randomly generated inside the unit square in the positive quadrant. Label each point depending on whether or not the first coordinate x_1 is greater than its second coordinate x_2. Now consider the following loss function for training pair $(\bar{X}, y)$ and weight vector $\bar{W}$:

$$L = \max\{0, a - y(\bar{W} \cdot \bar{X})\},$$

where the test instances are predicted as $\hat{y} = \{\bar{W} \cdot \bar{X}\}$. For this problem, $\bar{W} = [w_1, w_2]$, $\bar{X} = [x_1, x_2]$ and $\hat{y} = (w_1 x_1 + w_2 x_2)$. A value of $a = 0$ corresponds to the perceptron criterion and a value of $a = 1$ corresponds to hinge-loss.

1. You need to implement the perceptron algorithm without regularization, train it on the 10 points above, and test its accuracy on 5000 randomly generated points inside the unit square. Generate the test points using the same procedure as the training points. You need to have your own implementation of the perceptron algorithm, using the perceptron criterion loss function. (6)
2. Change the loss function from perceptron criterion to hinge-loss in your implementation for training, and repeat the accuracy computation on the same test points above. Regularization is not used. (5)
3. In which case do you obtain better test accuracy and why? (2)
4. In which case do you think that the classification of the same 5000 test instances will not change significantly by using a different set of 10 training points? (2)

   References:

   - Perceptron Algorithm in Python
     Available at https://medium.com/hackernoon/implementing-the-perceptron-algorithm-from-scratch-in-python-48be2d07b1c0

Problem 2 - *Precision, Recall, ROC* 15 points

This question is based on a paper from ICML 2006 (reference below) that talks about the relationship between ROC and Precision-Recall (PR) curves and shows a one-to-one correspondence between them. You need to read the paper to answer the following questions.

1. Does true negative matter for both ROC and PR curve ? Argue why each point on ROC curve corresponds to a unique point on PR curve ? **(5)**
2. Select one OpenML dataset with 2 output classes. Use two binary classifiers (Adaboost and Logistic regression) and create ROC and PR curves for each of them. You will have two figures: one containing two ROC and other containing two PR curves. Show the point where an *all positive classifier* lies in the ROC and PR curves. An all positive classifier classifies all the samples as positive. **(10)**

Reference paper:

- Jesse Davis, Mark Goadrich, The Relationship Between Precision-Recall and ROC Curves, ICML 2006.

Homework 2

Problem 3 - *Linear Separability* 10 points

Consider a dataset with two features x_1 and x_2 in which the points $(-1,-1),(1,1),(-3,-3),(4,4)$ belong to one class and $(-1,1),(1,-1),(-5,2),(4,-8)$ belong to the other.

1. Is this dataset linearly separable ? Can a linear classifier be trained using features x_1 and x_2 to classify this data set ? You can plot the dataset points and argue. **(2)**
2. Can you define a new 1-dimensional representation z in terms of x_1 and x_2 such that the dataset is linearly separable in terms of 1-dimensional representation corresponding to z ? **(4)**
3. What does the separating hyperplane looks like ? **(2)**
4. Explain the importance of nonlinear transformations in classification problems. **(2)**

Problem 4 - *Softmax Activation Function* 10 points

Consider the softmax activation function in the output layer, in which real-valued outputs $v_1,\ldots,v_k$ are converted into probabilities as follows:

$$o_i = \frac{\exp(v_i)}{\sum_{j=1}^{k}\exp(v_j)} \quad \forall i \in \{1,\ldots,k\}.$$

1. Show that the value of $\frac{\partial o_i}{\partial v_j}$ is $o_i(1-o_i)$ when $i=j$ and $-o_i o_j$ when $i \neq j$. (5)
2. Assume that we are using cross-entropy loss $L = -\sum_{i=1}^{k} y_i \log(o_i)$, where $y_i \in \{0,1\}$ is the one-hot encoded class label over different values of $i \in \{1,\ldots,k\}$. Use the result in part 1 to show the correctness of following equation:
$$\frac{\partial L}{\partial v_i} = o_i - y_i. \text{ (5)}$$

Problem 5 - *Weight Initialization, Dead Neurons, Leaky ReLU* 25 points

Read the two blogs, one by Andre Pernunicic and other by Daniel Godoy on weight initialization. You will reuse the code at github repo linked in the blog for explaining vanishing and exploding gradients. You can use the same 5 layer neural network model as in the repo and the same dataset.

1. Explain vanishing gradients phenomenon using standard normalization with different values of standard deviation as given in the reference. Train the model with tanh and sigmoid activation functions. Next, show how *Xavier (aka Glorot normal) initialization* of weights helps in dealing with this problem. You should plot the gradients at each of the 5 layers for all 4 experiments to answer this question. (10)
2. The dying ReLU is a kind of vanishing gradient, which refers to a problem when ReLU neurons become inactive and only output 0 for any input. In the worst case of dying ReLU, ReLU neurons at a certain layer are all dead, i.e., the entire network dies and is referred as the dying ReLU neural networks in Lu et al (reference below). A dying ReLU neural network collapses to a constant function. Show this phenomenon using any one of the three 1-dimensional functions in page 13 of Lu et al. Use a ReLU network with 10 hidden layers, each of width 2 (hidden units per layer). Use minibatch of 64 and draw training data uniformly from $[-\sqrt{7},\sqrt{7}]$. Perform 1000 independent training simulations each with 3,000 training points. Out of these 1000 simulations, what fraction resulted in neural network collapse. Is your answer close to over 90% as was reported in Lu et al.? (10)

3. Instead of ReLU consider Leaky ReLU activation as defined below:

$$\phi(z) = \begin{cases} z & \text{if } z > 0 \\ 0.01z & \text{if } z \leq 0. \end{cases}$$

Run the 1000 training simulations in part 2 with Leaky ReLU activation and keeping everything else same. Again calculate the fraction of simulations that resulted in neural network collapse. Did Leaky ReLU help in preventing dying neurons ? (5)

References:

- Andre Perunicic. Understand neural network weight initialization.
  Available at https://intoli.com/blog/neural-network-initialization/
- Daniel Godoy. Hyper-parameters in Action Part II — Weight Initializers.
- Initializers - Keras documentation. https://keras.io/initializers/.
- Lu Lu et al. Dying ReLU and Initialization: Theory and Numerical Examples .

Problem 6 - *Batch Normalization, Dropout, MNIST* 25 points

Batch normalization and Dropout are used as effective regularization techniques. However its not clear which one should be preferred and whether their benefits add up when used in conjunction. In this problem we will compare batch normalization, dropout, and their conjunction using MNIST and LeNet-5 (see e.g., http://yann.lecun.com/exdb/lenet/). LeNet-5 is one of the earliest convolutional neural network developed for image classification and its implementation in all major framework is available. You can refer to Lecture 3 slides for definition of standardization and batch normalization.

1. Explain the terms co-adaptation and internal covariance-shift. Use examples if needed. *You may need to refer to two papers mentioned below to answer this question.* (5)
2. Batch normalization is traditionally used in hidden layers, for input layer standard normalization is used. In standard normalization the mean and standard deviation are calculated using the entire training dataset whereas in batch normalization these statistics are calculated for each mini-batch. Train LeNet-5 with standard normalization of input and batch normalization for hidden layers. What are the learned batch norm parameters for each layer ? (5)
3. Next instead of standard normalization use batch normalization for input layer also and train the network. Plot the distribution of learned batch norm parameters for each layer (including input) using violin plots. Compare the train/test accuracy and loss for the two cases ? Did batch normalization for input layer improve performance ? (5)
4. Train the network without batch normalization but this time use dropout. For hidden layers use dropout probability of 0.5 and for input layer take it to be 0.2 Compare test accuracy using dropout to test accuracy obtained using batch normalization in part 2 and 3. (5)
5. Now train the network using both batch normalization and dropout. How does the performance (test accuracy) of the network compare with the cases with dropout alone and with batch normalization alone ? (5)

References:

Homework 2

- N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R.Salakhutdinov . Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Available at at https://www.cs.toronto.edu/ rsalakhu/papers/srivastava14a.pdf.
- S. Ioffe, C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. Available at https://arxiv.org/abs/1502.03167.