CS 229 Machine Learning, spring 2020

Homework 3:

Neural Networks

Due Saturday March 14, 11:59pm

Submit by the **blackboard system**

**Question1: (2.5 points) Property of derivatives of Error function**

1. (1pts) Show the derivative of the error function

(5.21)

with respect to the activation ak for an output unit having a logistic sigmoid activation function

satisfies

(5.18).

1. (1.5 pts) Show the derivative of the error function

(5.24)

with respect to the activation ak for output units having a softmax activation function

satisfies

(5.18).

*Hint:* for each *,* its true label has  *,* and *.* That is to say, for the activation ak activated by an , the corresponding could be either 1 or 0.

**Question2: (7.5 points) Implementation of NN (using Back-Propagation)**

**Code:**

Write your code in any programming language, and submit your results together with well-commented program source code. **An executable** file is **required**. You should **write your own code**, rather than just downloading existing code from the Internet.

**Data:**

Generate a set of data **points (x,y),** by choosing a **nonlinear function f(x)** and evaluating **y=f(x)+noise** for **random x** values, where each **x** is a **real vector of at least 2 elements (2-dim vector, or high than 2-dim)** and each **y is a real scalar or vector**. As an alternative, you can usea data set you find online or are using in other research. You should clearly statewhat your data set is, or how you generated it.

**Note:**

For example, **x**=[x1; x2] is a 2-dim vector, f(**x**) can be x1+(x2)2+ (2\*x1-cos(x2)) 2.

f(**x**) should not be a simple one, like f(**x**)= f(x1) +f(x2), e.g., f(**x**)= x1 + (x2)2 +b, because, in such cases, the two inputs can be considered as one input is x1, and the other input is (x2)2. Then f(**x**) = f(**x** \* [1, 0]) + f(**x** \* [0, 1]). The obtained weights **W** will be a very simple set.

**Task:**

Implement a two-layer (or more) neural network with back-propagation.

The network should have **2 or more inputs**. The inputs connect to **M** neurons in the **hidden layer**, each of which takes a weighted sum of its inputs plus a bias and then applies the **an activation function.**

The network should have 1 or more outputs. Each output of the network is a weighted sum plus bias of the outputs of the hidden layer (need no activation function or use **identity function f(x)=x** for the **output** because this is a regression problem).

During learning, weights are updated by gradient descent (you can use either batch or stochastic gradient descent) to decrease error function value.

1) (4 pts) **Describe all the parameters** you have, including the number of inputs, outputs, and hidden neurons, the number of weights. Find a learning rate that allows it to learn to a small error. **Plot a figure of how the error decreases during learning.**

2) (1 pts) **Test** the NN you learned by a **different** set of data **points (x,y)** (different from training set, but y is still generated by f(x)+noise). Show the root mean squared error (RMSE) when applying the iteratively trained NN on the training set (in one color), and on the testing test (in the other color) --- calculate RMSE in each iteration after updating the NN parameters (weights) for both training data and testing data.

3) (1 pts) How will the training error and testing error be different if you re-train the NN by different initializations of weights? You can compare the training error and testing error decreasing curve in 3 independent runs with different initialized weights.

4) (1 pts) how will the NN behave if you set ***M*** (the number of hidden units) differently? For example, you can show the training error curve and the testing error curve when NN has M/2 hidden units, or has 2\*M or even more.

**Bonus: (3 points)**

Implement **two types of gradient descent optimization strategies** discussed in class (**e.g., choosing two from momentum, AdaGrad, RMSProp, AdaDelta or Adam)**.

Run them and see if they learn faster, or learn a better solution (in terms of testing error). Describe whether they improved speed, accuracy, or both, and why you think that occurred. (It’s OK if you discover your “improvement” had no effect or even made it worse. The important thing is to test it and explain the results).