CMME139

Introduction to Deep Learning

Assignment #2

Due Date: 30/05/2023 Tuesday (23:59)

Score Contribution: 30%, evaluated over 20 points.

Type: Individual Assignment

The ideal programming language to use in the assignment is R (expected programming language).

The codes are expected to be in R. However, it is also possible to use other programming languages provided that you also submit proper readme documentation.

Objectives

- **Develop an Artificial Neural Network (ANN)**
- **Use the developed ANN in an "unsanitary" problem**
- **Transform problem and data-specific techniques for your own problem**
- **Understand data structures used in ANN development**

Structure of the Assignment

This is an individual assignment. However, note that in order to develop a solution for this assignment, you will need to use the outcome of Assignment #1. I strongly suggest you use your own group's data in the assignment. However, if you were unable to finish Assignment #1, or do not trust in your results, you are welcome to ask to any other colleague for their results in Assignment #1 and use it. (There will be no grade deduction for this) If you do that, please make sure honor their work and acknowledge them in your report.

Overview

This assignment is straightforward: You will build a Neural Network (NN) model for the classification problem that we have used in the first assignment. Note that the resulting dataset of Assignment #1 is an attribute-based dataset. **Note that the case of Session 3 Workshop also involves an attribute-based dataset**. The problem at hand is a classification problem where, you attempt to predict the Rating class in the Mobile Games data ("Low" ratings or "High" ratings) **Note that we have also covered a classification solution in the Session 2 Workshop**.

As a result, a straightforward solution to this assignment is to examine the code in both workshops closely, possibly running them, and apply the techniques that you observe to the problem at hand.

Important Tip: Although this task will seem very straightforward, I suggest you especially examine the data structures supplied as inputs to Keras functions in both examples: Keras requires the input data structures of specific types, with specific properties. (For example, if you examine the examples, both workshop codes remove column names information from the matrices. The reason is that Keras do not accept matrices containing additional structural information)

Tip 2: As we have discussed in Session 2, Neural networks only accept numeric attributes. Hence you may need to convert your categorical attributes into numeric ones (by using the as.numeric() function).

The purpose of this assignment is to master the workflow of a neural network development task: At the end of this assignment, you will be able to understand the purpose of each step in the workflow, and more importantly understand the required data transformations better. (Although in Assignment #1 we have gone over all the tasks in data preparation, you will observe that everything is not straightforward when Neural Networks come into equation.)

Specifications of the Assignment

Here, the tasks in the assignment are provided in a step-by-step manner:

Task1: Importing (2 Points)

I strongly suggest you not to work on the same workspace that you have left after assignment #1: Working with Neural Networks require a training data, a testing data, training objectives, and testing objectives, and it can get very messy and very confusing fast.

One idea is to save your final data frame, clear your workspace, and just reload the data frame as a starting point. In order to do this, you can use the "save" and "load" functions. For example, assuming after assignment #1 your data is stored in a data frame named "df":

save(df, file = "mydata.Rda") would save your data frame (and that data frame only) into a file named "mydata.Rda", and **load(file="mydata.Rda")** would place the data frame into your workspace.

This strategy is also safer, faster, lighter, and more reliable than moving data frames by csv files or other alternatives.

Task 2: Data Preparation (5 points)

Ideally, your columns in your data frame need to be converted into numeric format, other than the objective (i.e., Ratings). Next, Ratings should be converted to numeric as one-hot variables (like the case in Mnist example).

Task 3: Build, compile the neural network, and fit training data (15 points)

Examine the Workshop codes; they contain everything you will need for this task.

I strongly suggest that you especially examine the data format of the inputs to Keras functions; most only accept a matrix as input, and matrices need to be organized properly (in terms of columns and rows). Additionally, they need to be stripped of all additional metadata such as column names.

Tip: I suggest you start by taking a small sample and start there. You would eliminate the range of possible errors only to Keras calls instead of doubting whether your data is faulty or not. Once your Neural Network properly works on this sample data, you can adopt it to the whole dataset.

Task 4: Parameter Tuning (5 points):

Try your neural network with several parameters. Test 2-layer and 3-layer networks, test neuron counts in the layers, test changing the dropout layer settings. Observe how loss and validation loss is behaving and try to find a good enough setting (good enough = not overfitting).

Pro Tip: There is no apparent end to this stage. Do not overdo it and waste your time for a 1% increase in accuracy or tiny improvement on loss. What you should expect is to achieve a setting where, training loss and validation loss behave similar to each other at later epochs (which means there is no apparent overfitting in your model).

Task 5: Check if you can perform better than traditional models (3 points)

Attempt to achieve a better performance than traditional model.

Note that, you will possibly achieve that relatively easily when you reach this state. However, a question to be answered is: How much does it improve? Is it worth it computationally? Is it worth it effort-wise? This is an observation that can only be developed manually.

What to Submit

We require **a zip file** containing (at least) 3 files:

- A report: A title page, what operations you have performed, describing your model and the parameters you have tested, your accuracy results, and loss and error graphs of (at least) your final test. Possibly your take and experiences from the assignment. (Again, don't overdo it. We will examine the effect of parameters in a later assignment more exhaustively. It does not have to be longer than this document!)
- Any Rda files you have moved (or other files if you use another method. But please share the code needed to run the NN in that case)
- Your code

What will be graded

- Success!
- Code quality and clarity
- Report quality
- Timeliness

Some Tips and Tricks

Keep your workspace clean. This assignment is straightforward, but it can be frustrating when something goes off. (It will go off, that is given and the reason this assignment is worth 7 days. Don't get frustrated, if you put enough effort, all will be resolved)

Good Luck!

I hope you will enjoy this assignment.

DL TEAM

Appendix – Some Useful Functions and Libraries

There is none. All you need is already in the Workshop codes!