ASSIGNMENT ON ARTIFICIAL NEURAL NETWORK AND MACHINE LEARNING (CPE593)

BY

GWAM ONYELUKACHUKWU NORA ENG1508326

OMOREGIE OSAKPOLOR BLESSING ENG1603917

USIWOMA OGHENOVO ENG1503613

OPENE VICTORIA EKENEDINICHUKWU ENG1503604

OKONKWO NKEMDILIM RITA ENG1503596

GODWIN MONICA OGHENERO ENG1503570

OKORUWA VICTOR

OKOLIE TOOCHUKWU CHARLES ENG1503595

ADEBIYI EMMANUEL OLUWAGBEMIGA ENG150349

NWOKOYE ONYINYECHI MARYJANE ENG1503585

BENEDICT ODOH IKECHUKWUKA ENG1603904

EHIMA CHRISTIAN CHUKWUEMEKA ENG1503565

EBU JACOB SUNDAY ENG1503561

**CROSS VALIDATION**

Cross Validation is a method used to estimate the performance or accuracy of a model. The goal of cross-validation is to evaluate a model's performance on data it was not trained with and flag problems like overfitting and selection bias. It determines how a model will perform on a dataset it has not been trained on. We can train several models using different hyperparameters and select the model with the best performance. When applied to several neural networks with different hyperparameter values (such as the number of hidden nodes, learning rate and so on), we can then select the best set of hyperparameter values to use.

**TYPES OF CROSS VALIDATION**

**1.0 EXHAUSTIVE CROSS VALIDATION**

Exhaustive cross validation methods are cross-validation methods which learn and test on all possible ways to divide the original sample into a training and a validation set.

**1.1 Leave-P-Out Cross Validation**: When using this exhaustive method, we take p number of points out from the total number of data points in the dataset (say n). While training the model we train it on these (n – p) data points and test the model on p data points. We repeat this process for all the possible combinations of p from the original dataset. Then to get the final accuracy, we average the accuracies from all these iterations. This is an exhaustive method as we train the model on every possible combination of data points. Remember if we choose a higher value for p, then the number of combinations will be more and we can say the method gets a lot more exhaustive.

**1.2 Leave-One-Out Cross Validation** - This is a simple variation of the Leave-P-Out cross validation method, here the value of p is set as one. This makes the method much less exhaustive as now for n data points and p = 1, we have n number of combinations. Although If the sample data is too large, this can still take a lot of time. But it would still be quicker than the Leave-p-out cross-validation method.

**2.0 NON-EXHAUSTIVE CROSS VALIDATION**

Non exhaustive cross validation is a type of cross validation which does not compute all ways of splitting the original sample data set. It is an approximation of leave-p-out cross validation.

**2.1 K-Fold Cross Validation** - In this type of cross validation, the data sets available for training and testing are split into K folds/sections where each fold is used as a testing set at some point. This type of cross validation is not suitable for an imbalanced data set. The following steps are used for K-fold cross validation:

* Equally split the entire data set into K folds.
* For each K fold in the data set, train your model on K-1 folds, then test the model using the Kth fold.
* Record error seen on each prediction.
* Repeat until each of the K fold has served as a test set.
* The average of the K recorded errors is called the cross validation error and would serve as the performance metric of the model.

**2.2 Repeated Random Sub Sampling Cross Validation**: This type of cross validation is also known as Monte Carlo cross validation. It splits the dataset randomly into training and validation. The number of iterations is not fixed and decided by analysis. The result is the average of the values gotten in each iteration. One disadvantage of this type of cross validation is some samples may not be selected for either training or validation.

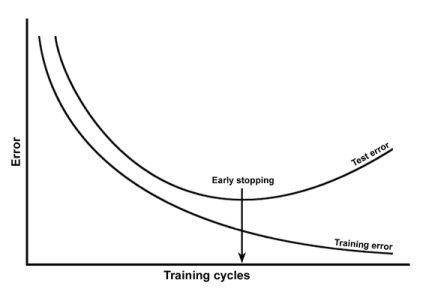
**2.3 Holdout Cross Validation**: Here, the dataset is randomly split into training and validation depending on data analysis. Generally, the split of training data is more than validation data. The more the data, the better the model. The disadvantage of this cross validation is it is not suitable for an imbalanced dataset and a lot of data is isolated from training the model.

**TRAINING SET AND TEST SET AS REGARDS MACHINE LEARNING**

The Training Set is a dataset of examples which is used to fit the parameters of the neurons of an Artificial Neural Network (ANN). It is the actual dataset that is used to train the model. On the other hand, the Test Set is a dataset used to provide an unbiased evaluation of a final model fit on the training dataset. The test set is used to judge the performance (accuracy, sensitivity, specificity, F-measure and so on) of the classification produced by the model. It must be independent of the training set but usually follows the same probability distribution as the training set.

The division of large datasets into training and testing sets is usually done while training Artificial Neural Networks as it helps to mitigate issues like overfitting, which is a scenario where which the predictions of a model correspond too closely to the data from which it was trained with.

**TESTING A NEURAL NETWORK**

To train up a neural network you need a huge pool of data, data which has been manually processed to set an output. Get a broad and diverse set of data. You run it through your neural network with it switched to learning mode.

A graph showing overfitting and two important error lines which are explained below

***The first step*** is to switch your neural network over from a learning operation to a running operation. You then run through the same training data you’ve just used through your system to observe the error rate you get from comparing the neural network output with the expected result from your data. This is the “training error” line in the above graph of overfitting. This is the first step of verification of your neural network and hopefully you’re seeing such a trending line. However, if you’re feeding data into your system, replaying the training cycles and you’re not seeing the “training error” line go down, then your neural network is struggling to find a pattern. If you can’t find such pattern, review your neural network, the quality of your data and the size of the data pool.

***The second verification step***is to test it a bit more. Now for this you can’t simply use data that you’ve already used for training — the neural network has learned to cope with that explicit case. You need some additional data around to do this. Typical results from this stage of verification can be seen above as the “test error” line, and notice how they always have a higher error rate than “training error” data. That shouldn’t be unexpected — the neural network trained on explicit data provided and re-verified in the “training error” data line.