Neural Network Analysis of Dimuon Data within CMS

Shannon Massey
University of Notre Dame
SIST Intern, Summer 2015
Fermi National Accelerator Laboratory
Supervisors: Dr. Pushpa Bhat and Dr. Leonard Spiegel

August 7, 2015

Abstract

Today's high-energy particle physics experiments are heavily predicated on the ability to retrieve useful and interesting event data out of extremely large data sets. Discrimination between signal and background must be optimized in order to produce the best possible experimental results. The Toolkit for Multivariate Analysis (TMVA) within ROOT provides many different algorithms for the classification of signal and background events. We will analyze the Artificial Neural Network (ANN) methods within TMVA. More specifically, we will examine the implementation of multilayer perceptrons to classify $Z\rightarrow\mu\mu$ decay data and Drell-Yan process data.

Contents

Introduction

The Large Hadron Collider (LHC) is the world's largest and most powerful particle collider. Deep beneath the Franco-Swiss border, the 27-kilometer ring accelerates two high-energy particle beams to speeds nearing the speed of light. The two beams are then set to collide. With a design energy of 7 TeV per beam, the LHC would reach a center of mass collision energy of 14 TeV and a luminosity of 10^{34} cm^{-2} s^{-1}. Recently, the LHC has been running with a center of mass energy of 13 TeV at the collision with the goal of steadily increasing the luminosity to $2*10^{34}$ [1]. These collisions are analyzed by four main detectors: ATLAS, CMS, ALICE, and LHCb. The Compact Muon Solenoid (CMS) detector is a general-purpose detector. With a layered, modular design built around a large solenoid magnet, the CMS detector is designed to detect a variety of particles and phenomena produced in the collisions. In the searching for rare particles, an extremely large number of collisions are required. Since the vast majority of these events do not produce interesting effects, it is essential to have an accurate and robust classification method for discriminating between signal and background events and objects. Algorithms based on machine learning have provided relatively accurate means of classification in high-energy physics experiments. Within ROOT, the Toolkit for Multivariate Analysis (TMVA) provides many different "supervised" classification algorithms. We used a toy data set and Monte Carlo simulated $Z \rightarrow \mu\mu$ decay events to examine the how the architecture of a Multilayer Perceptron (MLP) affects data classification.

Toolkit for Multivariate Analysis

The Toolkit for Multivariate Analysis (TMVA) is a high-energy physics (HEP) oriented toolkit that has been integrated into ROOT. It is used for the processing, evaluation, and application of multivariate classification. TMVA includes a variety of "supervised learning" classification and regression algorithms [2]. These algorithms use training events, in which the featured inputs and desired outputs are provided to determine a mathematical model that will then make predictions about the classification of future data. We are interested in the algorithms that will help discriminate our data into two categories: signal and background.

Feed-forward neural networks (NNs), also known as multilayer perceptrons (MLPs), are popular in the classification of data. An MLP is a network of interconnected neurons arranged in layers. Each layer is linked by a set of weighted connections. These weights determine the relative importance of each neuron in the computation of the output value(s). Each neuron or node processes the information (values) it receives from multiple inputs using an activation function. Each neuron then provides a single output, and passes it to the next layer of neurons [3]. Figure 1 depicts a single neuron receiving weighted inputs, processing the values with the activation function, and the passing the output.

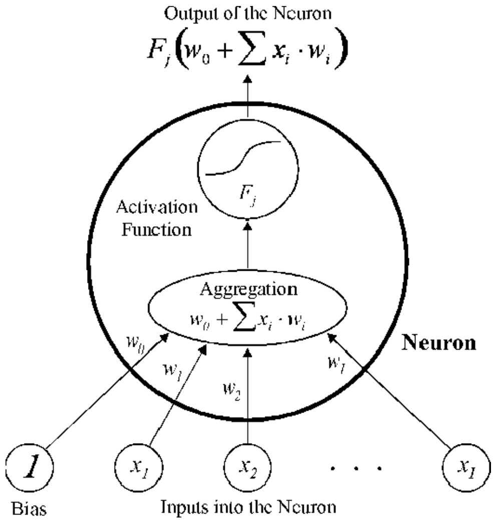

Figure 1: Picture of a single neuron receiving input values and producing an output

Once the output is given, the network uses a cost function to quantify the discrepancy in the modelled output and the actual output. The network then uses back propagation of errors to determine how much to tweak each weight between neurons. The network continues to adjust the values of each weight until a cost function is minimized or the number of training cycles is reached.

An MLP consists of three or more layers of neurons. The first layer is known as the input layer. It provides the feature variables of the data to the model. This is followed by one or more hidden layers. The final layer is called the output layer. This provides the response of the model. In this paper, the response is the classification of signal or background. A neural network can be viewed as mapping from a space of input variables onto a space of output variables [2]. Figure 2 provides an example of a multilayer perceptron.

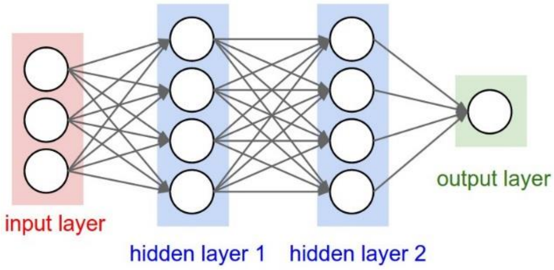

Figure 2: Details a multilayer perceptron with three input variables, two hidden layers, and one output variable

The TMVA includes three Artificial Neural Network (ANN) implementations: the Clermont-Ferrand neural network, the ROOT neural network, and the MLP neural network. The MLP neural network is recommended because it is the fastest and most flexible of the three implementations [2]. The configuration options for the MLPs give the user discretion over many of the MLP's attributes including the number of hidden layers, number of nodes in each hidden layer, neuron activation function, and number of training cycles. This enables the user to tailor the MLP to their specific set of input variables.

Academic Toy Data

To test the architecture of different MLPs, we used two different sets of data. For the first set, we used an example data set of four linearly correlated Gaussian distributed "toy" input variables provided by TMVA. Each variable is classified into signal and background events. Figure 3 depicts the distribution of each variable's signal and background inputs.

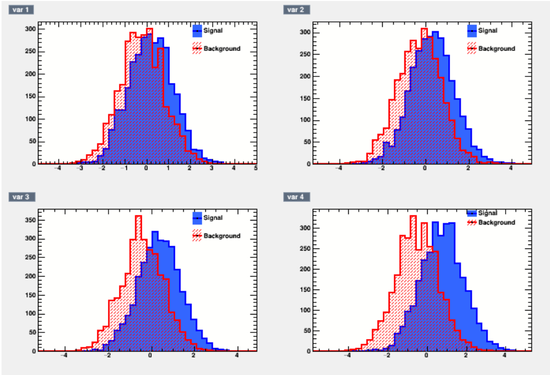

Figure 3: The graphs of arbitrarily named toy input variables var 1, var 2, var 3, and var 4. The blue gaussian histograms represent the signal events and the red gaussian histograms represent the background events for each variable

Through basic examination, it is easy to spot the slight separation between the signal and background events before applying the MLP algorithm to the inputs.

In order to visualize the effects of various MLP architectures, we decided to model eight different MLPs. We ran four of the MLPs with one hidden layer. One MLP was given four nodes, one was given nine, another was given fourteen, and the last was given nineteen nodes. The other four MLPs were given two hidden layers with the same amount of nodes as in hidden layer one. We allocated four, nine, fourteen, and nineteen nodes in each layer for the networks with two hidden layers. Since we modified the number of hidden layers and nodes within each layer, we were forced to keep the other MLP attributes constant. Every neuron used the sigmoid activation function (1), and we used 600 training cycles for each network.

$$Sigmoid\ Activation\ Function: \frac{1}{1+e^{-x}} \qquad (1)$$

With our MLP architectures set, we were then able to test the different neural networks. Figure 4a and 4b display the MLP outputs of each different type of network.

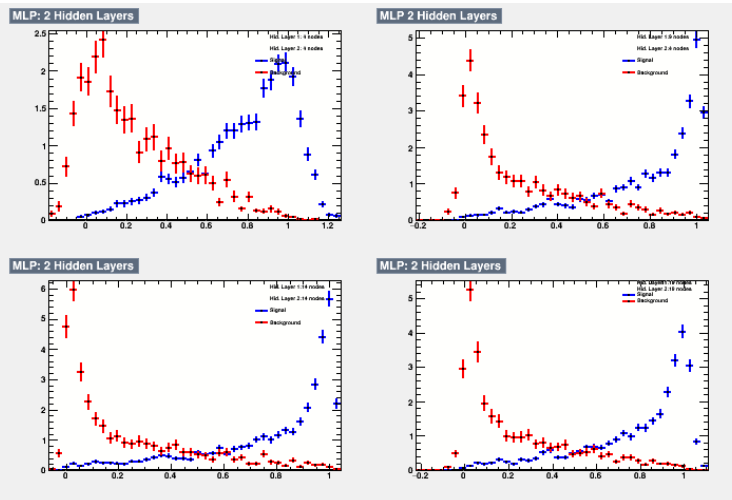

Figure 4a: Depicts the multilayer perceptrons for the four networks with a single hidden layer. The network architectures are shown as follows:

Top left: four nodes, top right: nine nodes, bottom left: fourteen nodes, bottom right:

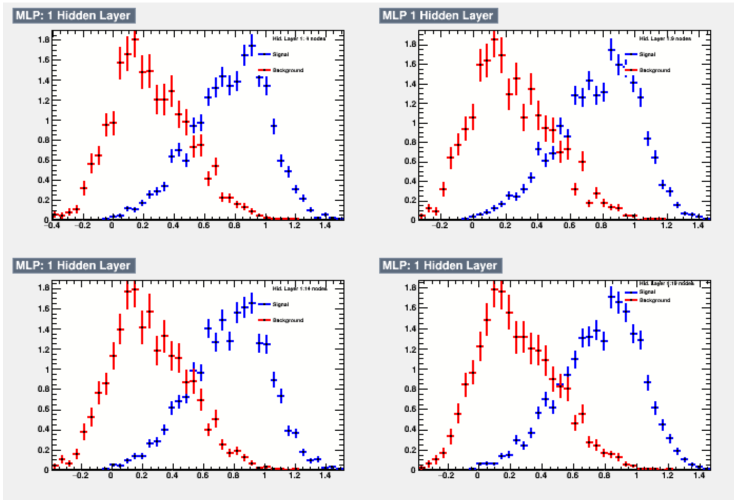

Figure 4b: Depicts the multilayer perceptron outputs for the four networks with two hidden layers. Each network has the same number of nodes in hidden layer one and hidden layer two. The network architectures are shown as follows

Top left: four nodes in each hidden layer, top right: nine nodes, bottom left: fourteen nodes, bottom right: nineteen nodes.

Examining these outputs by eye can give us a relative feeling of the effectiveness for the discrimination of an MLP network, however, we would prefer not to rely on such a subjective approach to determine if a network is adequate. In order to quantify the performance of a network we made use of a Receiver Operating Characteristic (ROC) curve. A ROC curve is graphical representation of discrimination values, and is used to compare classification methods. It is created by plotting certain discrimination values on the y- and x-axis. We implemented a C++ script that plotted the number of background events above each cut along the x-axis divided by the total number of background events (background efficiency) versus the number of signal events above each cut along the x-axis divided by the total number of signal events (signal efficiency). A high-arching ROC curve with an Area Under the Curve (AUC) of 1 would represent a network with complete discrimination between signal and

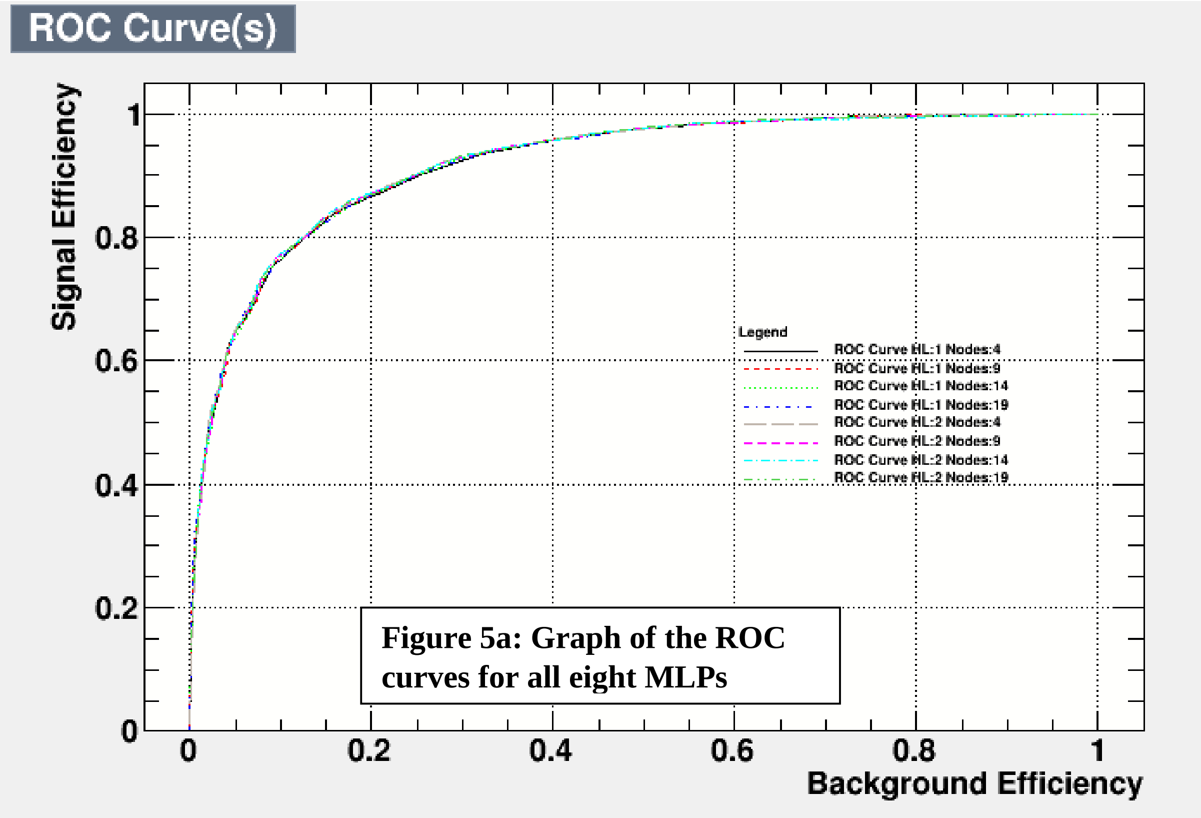

Figure 5a: Graph of the ROC curves for all eight MLPs

background (no overlap). A low-arching ROC curve nearing a slope modeled by the equation *y=x* with an AUC of .5 would represent a network that would have completely no discrimination between signal and background (complete overlap). With this method, we could compare the effectiveness of all MLP outputs to each other. Figure 5a shows the ROC curves for all eight MLPs superimposed on one plot. Figure 5b is a table of AUC values for each of the individual networks.

Toy Data: Area Under the Curve Table			
Total Hidden Layers:	Nodes in Layer One:	Nodes in Layer Two:	Area Under the ROC Curve (AUC)
One	Four	N/A	0.9195
One	Nine	N/A	0.9193
One	Fourteen	N/A	0.919
One	Nineteen	N/A	0.9193
Two	Four	Four	0.9199
Two	Nine	Nine	0.9191
Two	Fourteen	Fourteen	0.9186
Two	Nineteen	Nineteen	0.9202

Figure 5b: Table of AUC values for each of the eight neural networks

From the table above, it is evident that adding another hidden layer to the network provides little to no statistical improvements in the performance of discriminating signal from background. For this example, we can assume that the MLPs are able to accurately classify the data with only one hidden layer. With other data samples, this may not be the case.

Dimuon Event Classification

Our second implementation of the MLP method utilized the invariant mass peak computed from simulated $Z \rightarrow \mu\mu$ decay data to represent signal events and the side-bands surrounding the invariant mass peak computed from simulated Drell-Yan process with $Z \rightarrow \mu\mu$ data as background events. In order to make sure we had the proper input variables for our MLPs, we received the final .root files after the desired variables and events had been selected using a Python script with a C++ analyzer. With a .root file consisting of the desired variables, we then computed the invariant mass of each event using the transverse momentum (p_T) of each muon and the difference in the angles eta ($\Delta\eta$) and phi ($\Delta\varphi$) with the following derivation from the Energy-Momentum relation:

$$M\mu\mu = \sqrt{2p_{t1}p_{t2}\,(\cosh(\Delta\eta) - \cos(\Delta\varphi))} \quad (2)$$

We then determined our signal and background events for the training of our MLPs. To designate the events used for the signal, we used the invariant mass of the dimuons from the pure $Z \rightarrow \mu\mu$ decay data. In order to isolate the events within the invariant mass peak, we used the entries within the range of 80 to 100 GeV. Figure 6a depicts the histogram of the pure $Z \rightarrow \mu\mu$ invariant mass.

For background events, we decided to use three different cases to test the effectiveness of classifications with MLPs. The first case we used the band of events from 60 to 80 GeV for background. In the second case, we used the band of events from 100 to 120 GeV. For the third case, we used both bands, 60-80 GeV and 100-120 GeV. Using both bands for background would be the most realistic case for actual experimental classification. Figure 6b depicts the invariant mass of the Drell-Yan process with dimuon decay.

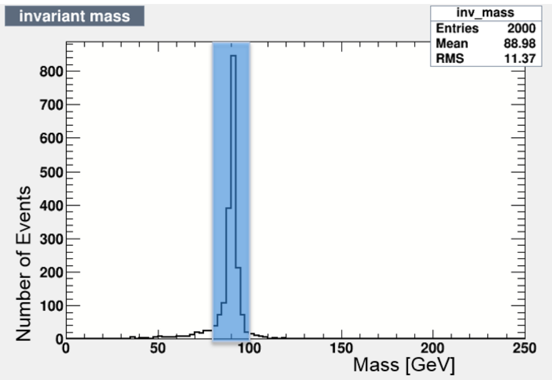

Figure 6a: Histogram of the pure $Z \rightarrow \mu\mu$ invariant mass. The blue band highlights the region of signal events we used to train the MLPs

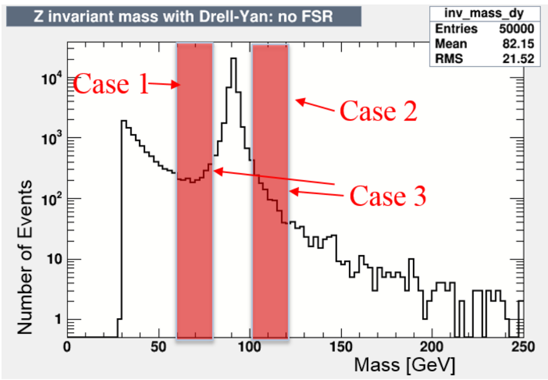

Figure 6b: Histogram of the invariant mass of the Drell-Yan process with a y-axis logarithmic scale. The red bands highlight the regions used for background events in the training of the MLPs. As it shows, Case 1 uses the band of events with lower mass for background. Case 2 uses the band of events with a higher mass for background. Case 3 uses both bands for background.

Figure 7a, 7b, and 7c depict the distribution of the input signal and background events for the three different cases.

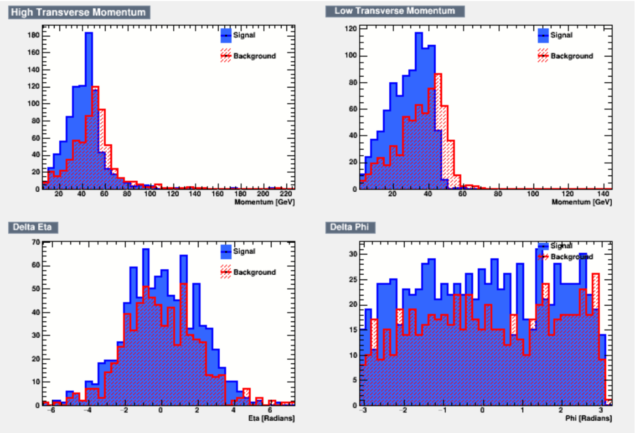

Figure 7a: Histograms of the signal (blue) and background (red) inputs for Case 1. The variables are shown as follows:

Top left: high transverse momentum, top right: low transverse momentum, bottom left: difference in eta, bottom right: difference in phi

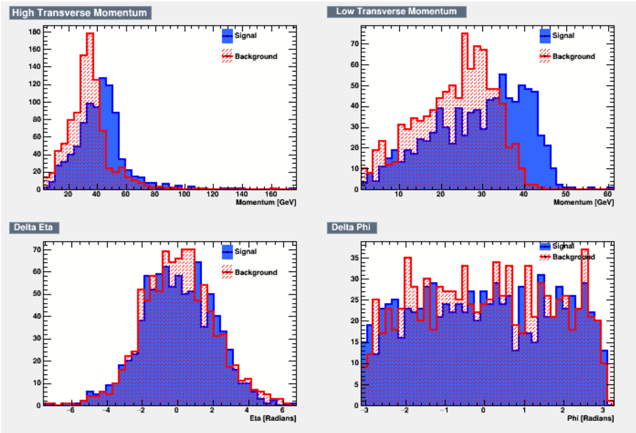

Figure 7b: Histograms of the signal (blue) and background (red) inputs for Case 2. The variables are shown as follows:

Top left: high transverse momentum, top right: low transverse momentum, bottom left: difference in eta, bottom right: difference in phi

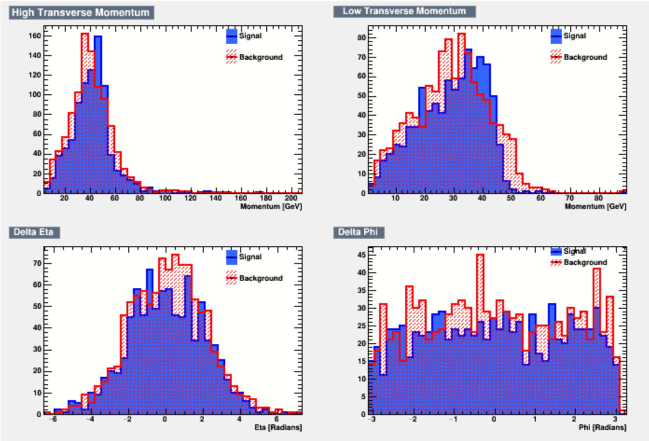

Figure 7c: Histograms of the signal (blue) and background (red) inputs for Case 3. The variables are shown as follows:

Top left: high transverse momentum, top right: low transverse momentum, bottom left: difference in eta, bottom right: difference in phi

For each case, we implemented two MLPs. The first network consisted of the input layer, one hidden layer with four nodes, and the output layer. The second network had two hidden layers with four nodes in each layer. We kept the number of training cycles constant at 5,000 cycles to give our networks more practice modelling the data sets. Each neuron was designated a sigmoid activation function. Figure 8a, 8b, and 8c depict the three MLP outputs after training each network.

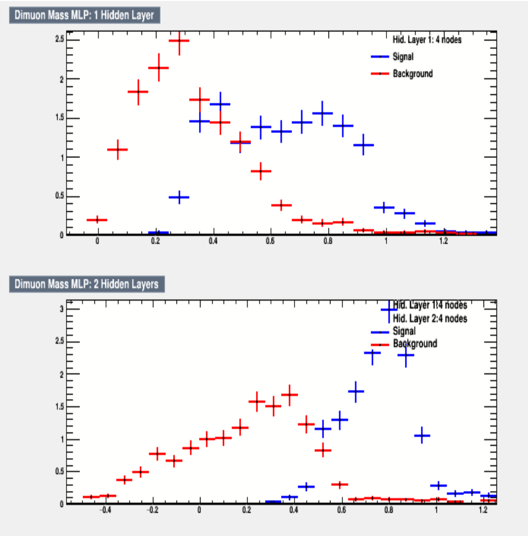

Figure 8a: The signal (blue) and background (red) histograms of the MLP output for Case 1 (60-80 GeV background).

Top: One hidden layer with four nodes, bottom: two hidden layers with four nodes in each layer.

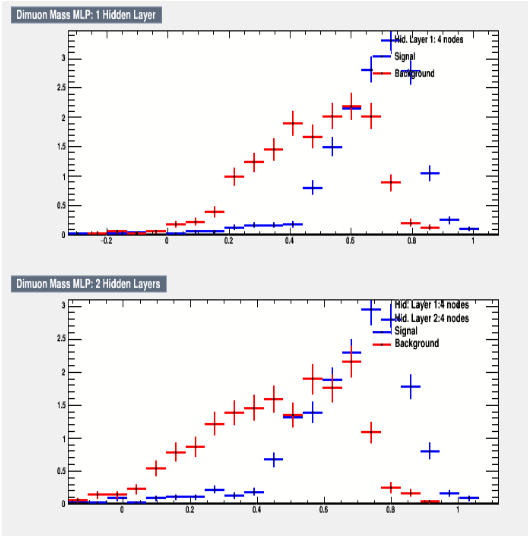

Figure 8b: The signal (blue) and background (red) histograms of the MLP output for Case 2 (100-120 GeV background).

Top: One hidden layer with four nodes, bottom: two hidden layers with four nodes in each layer.

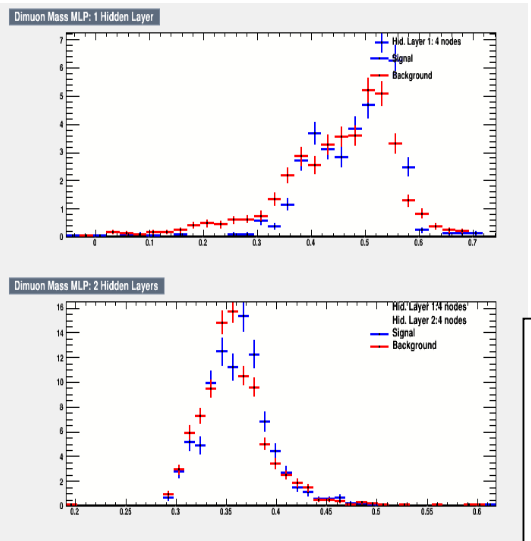

Figure 8c: The signal (blue) and background (red) histograms of the MLP output for Case 3 (60-80 GeV and 100-120 GeV background).

Top: One hidden layer with four nodes, bottom: two hidden layers with four nodes in each layer.

In order to more accurately determine the six networks' ability to discriminate signal from background, we implemented our ROC curve, in which allows us to calculate the AUC for each. Figure 9a depicts the each ROC curve, and Figure 9b illustrates the AUC value for each MLP output.

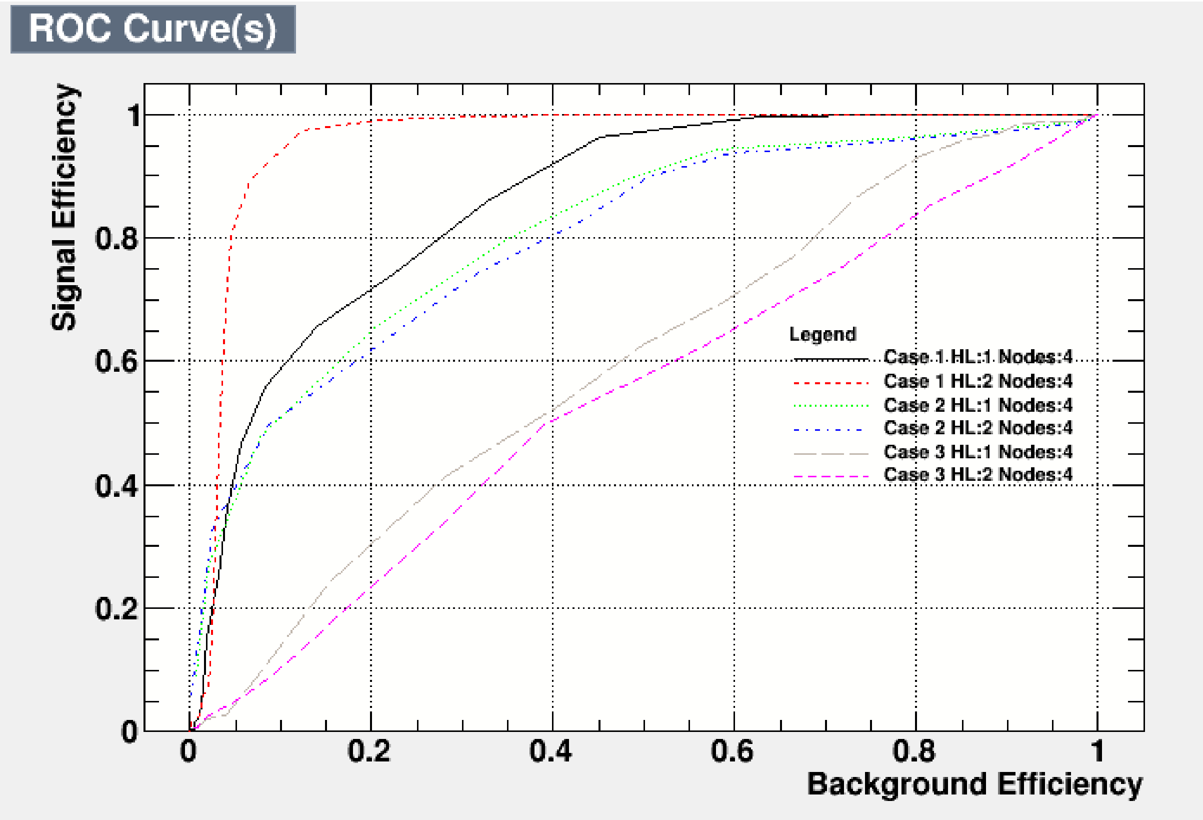

Figure 9a: ROC Curves for the six MLP outputs.

Dimuon Mass MLP Attributes Table (Case 1)			
Hidden Layers:	Nodes in Layer One:	Nodes in Layer Two:	AUC
One	Four	N/A	0.8607
Two	Four	Four	0.9577

Dimuon Mass MLP Attributes Table (Case 2)			
Hidden Layers:	Nodes in Layer One:	Nodes in Layer Two:	AUC
One	Four	N/A	0.809
Two	Four	Four	0.797

Dimuon Mass MLP Attributes Table (Case 3)			
Hidden Layers:	Nodes in Layer One:	Nodes in Layer Two:	AUC
One	Four	N/A	0.595
Two	Four	Four	0.543

Figure 9b: Tables of the AUC values for each network.

After examining the ROC curves and AUC tables, it is difficult to recognize the change in MLP outputs after adding an extra hidden layer. In Case 1, the AUC improves significantly, but in the other two cases, the AUC decreases slightly.

Conclusion

As experiments in particle physics continue to work to provide answers regarding our physical world, the equipment and analysis methods have to become more complex. These experiments depend on the ability to extract a tiny sample of events and objects from massive data sets. It is vital that the methods of discrimination provide accurate and well-defined results. In our use of the multilayer perceptron method with TMVA, we saw an improvement in discrimination after adding an additional hidden layer in Case 1. In Case 2, the two networks provided nearly the same amount of discrimination. Adding a second layer in Case 3 actually gave the MLP output less separation than the network using only one hidden layer. As the work and research behind machine learning improves, neural networks and other classification methods will improve. With this research, deeper MLPs with the ability to learn more complex data features will add to the tools available to scientists for event discrimination.

Sources

[1] The Large Hadron Collider. cern.ch. Retrieved August 4, 2015

[2] Speckmayer, P., Höcker, A., Stelzer, J., & Voss, H. (2010). "*The toolkit for multivariate data analysis*", TMVA 4. J. Phys.: Conf. Ser. Journal of Physics: Conference Series, 032057-032057.

[3] Bhat, P. (2011). "*Multivariate Analysis Methods in Particle Physics*". Annu. Rev. Nucl. Part. Sci. Annual Review of Nuclear and Particle Science, 281-309.

Acknowledgements

I would like to offer my special thanks to the SIST Committee for selecting me to partake in this internship. I would like to acknowledge my supervisors Dr. Pushpa Bhat and Dr. Leonard Spiegel for their support, encouragement, and guidance throughout the summer. I would also like to thank Dr. Sudeshna Banerjee for her feedback regarding my presentation, Graham Stoddard (Northern Illinois University) for his comprehensive assistance throughout the summer, and Grace Cummings (Virginia Commonwealth) for her help with the .root event simulation files. Finally, I would like to express my deepest gratitude to my mentor Dr. Elliott McCrory for his effort and time he put into building and maintaining this exceptional program.