Model Selection

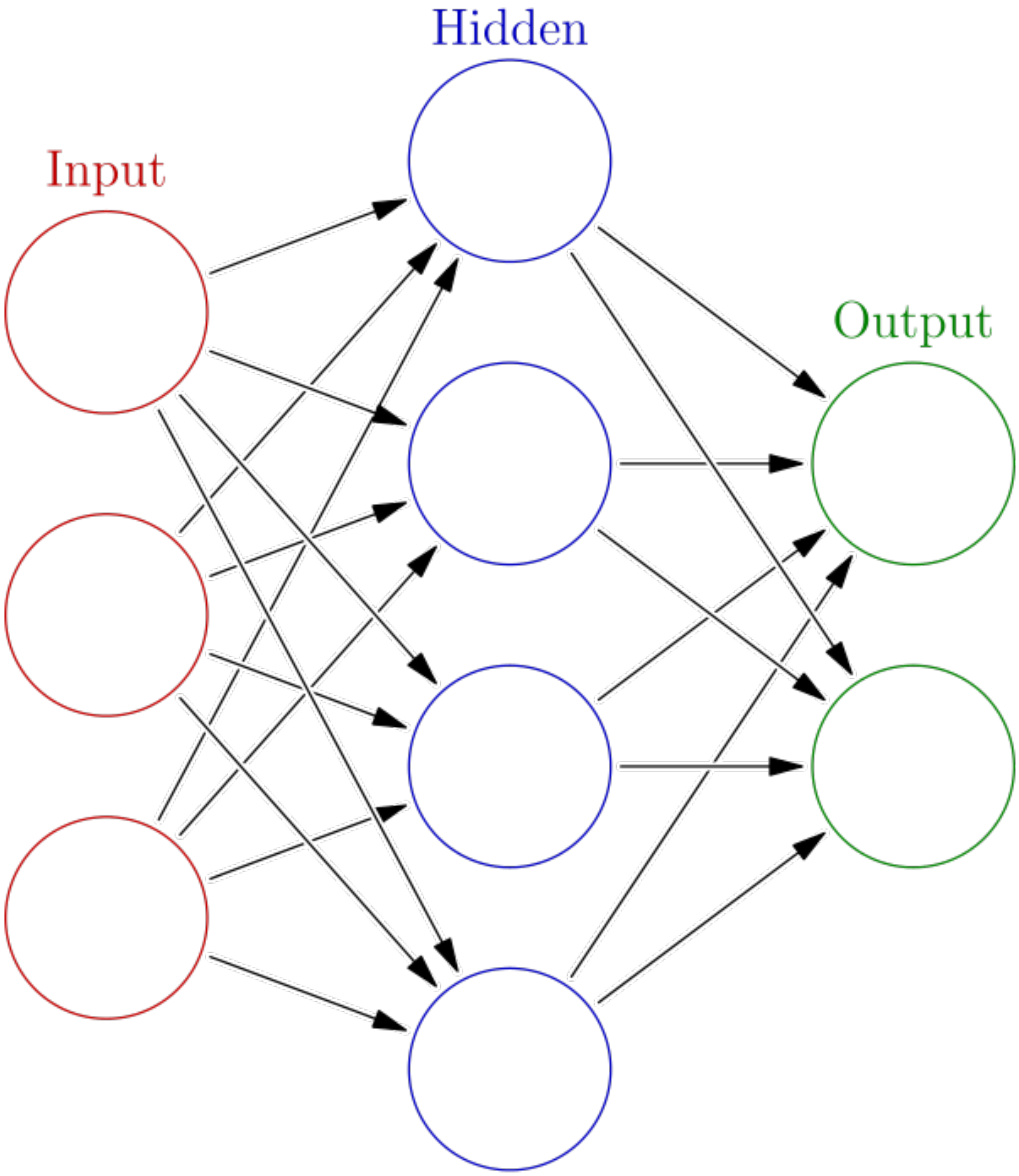
— Neural Network

What is Neural Network

Neural network is an information processing model that bases on a large amount of artificial neurons in various structures. Inspired by the features that biological nerves work, a general neural network consist of interconnected neurons and layers. Neurons are the units that can receive data, make computations, and give outputs. Layers are the collections of neurons, where neurons can link with those neurons located on different layers but not those on same layers. With different function of connection and diverse structure of layers, neural networks are widely used in the scenarios that people obtain massive data but can not capture the rules or patterns behind data, such as Big Data Processing or Computer Vision.

Figure 1. Basic Neural Network.

Source: Wikipedia: Artificial Neural Network



Implementation and Optimization of Model

Competition prediction based on neural network model is one of the main methods in this project. Considering efficiency on limited hardware and feasibility in specified time, we chose Keras as machine learning library to build our neural network, and use Theano as backend for Keras to make graph calculations and analysis.

To optimize the model and further get the best prediction accuracy, we examine the prediction performance of neural network in different inputs, parameters and layers. In this section, the model is tested under various selection of features, activation function, batch size, optimizer and number of layers, is revealed as follows.

1. Feature

   Features are the input of model, which are cleaned from original data and selected as many combinations to optimize the model.

Variable	**Description**
X_1	T1_AdjEM
X_2	T1_AdjO
X_3	T1_AdjD
X_4	T1_AdjT
X_5	T1_Luck
X_6	T1_SOSADjEM
X_7	T1_SOSOppO
X_8	T1_SOSOppD
X_9	T1_NCSOSAdjEM
X_{10}	T2_AdjEM
X_{11}	T2_AdjO
X_{12}	T2_AdjD
X_{13}	T2_AdjT
X_{14}	T2_Luck
X_{15}	T2_SOSADjEM
X_{16}	T2_SOSOppO
X_{17}	T2_SOSOppD
X_{18}	T2_NCSOSAdjEM

   Condition:

   Optimizer(with SGD, learning rate = 0.1, momentum = 0.1), Number of Layers = 1, Batch Size = 256, Action Function = ReLu.

	Feature	LogLoss	Accuracy	MSE
1	X_1 to X_{18} (all variables)	0.500986	0.761589	0.238411
2	X_1 , X_5 , X_{10} , X_{14}	0.486321	0.76067	0.23933
3	X_2 , X_3 , X_4 , X_{11} , X_{12} , X_{13}	0.501208	0.749816	0.250184

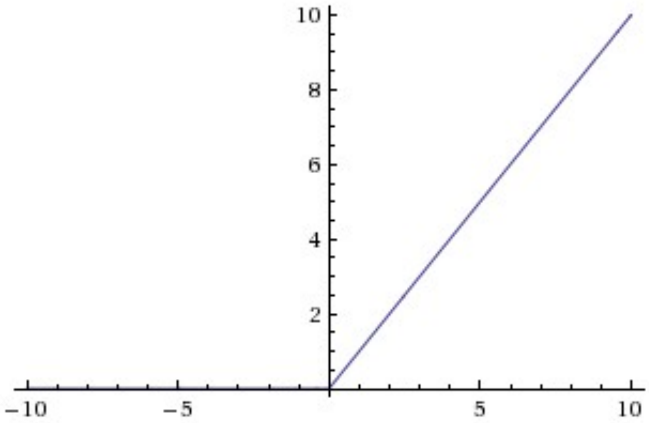
In this section, the combined input of X1, X5, X10 and X14 shows substantial outperformance over another two combined inputs, with a significantly lowest LogLoss and an almost highest Accuracy. Though the combined input of all variables has the best result of Accuracy, it's negatively impacted by the much higher LogLoss.

2. Activation Function

   Sigmoid and ReLU are two classical and popular activation function in neural network. Considering that they have no inflection point and are easy to make binary classification, this two function are applied and tested in this project.

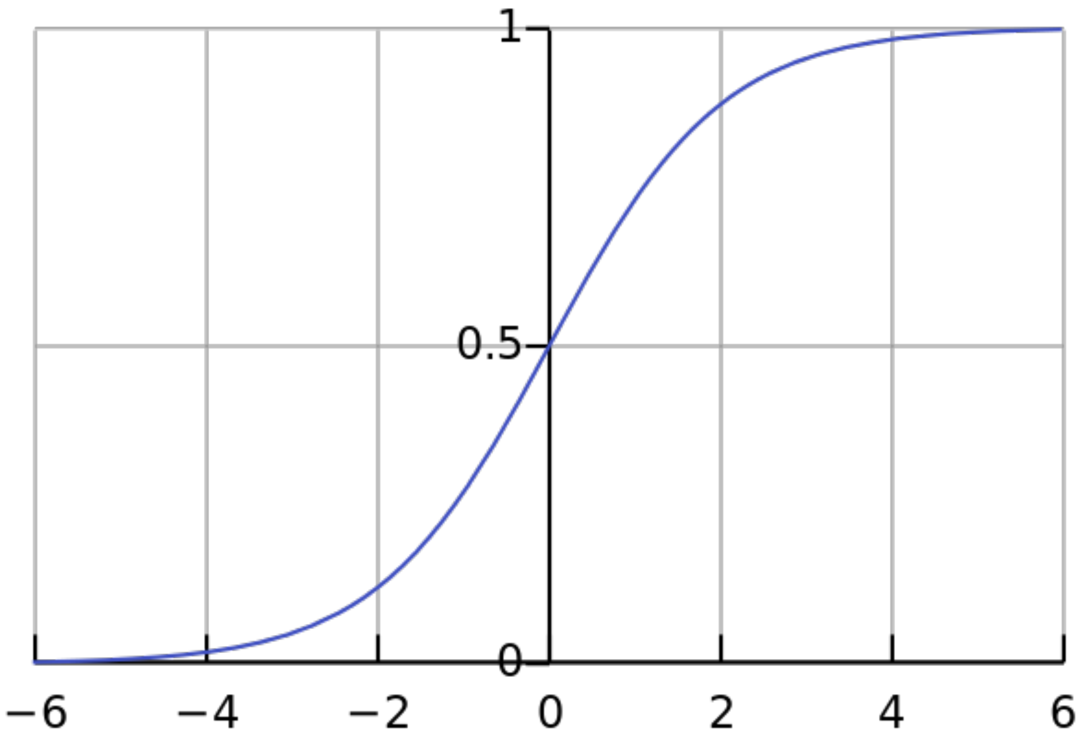
   ReLU

$$f(x) = \max(0, x)$$



   Sigmoid

$$S(t) = \frac{1}{1 + e^{-t}}.$$



Condition:

Optimizer(with SGD, learning rate = 0.1, momentum = 0.1), Number of Layers = 1, Batch Size = 256, Features Selected {X_1 , X_5 , X_{10} , X_{14} }.

	Activation Function	LogLoss	Accuracy	MSE
1	Sigmoid	0.48795	0.759934	0.240066
2	ReLU	0.485487	0.760302	0.239698

As for results of activation function, Accuracy of ReLU is high than that of Sigmoid. In addition, ReLU has a lower LogLoss than Sigmoid though slightly, which is a good indicator of performance. Therefore, it is obvious that in this section ReLU outperforms Sigmoid.

1. Batch Size

   Batch size is the number of data to be calculated at one time. Batch size can affect the speed of convergence and performance of optimized point.

Condition:

Optimizer(with SGD, learning rate = 0.1, momentum = 0.1), Activation Function = ReLu, Number of Layers = 1, Features Selected {X_1 , X_5 , X_{10} , X_{14} }.

	Batch Size	LogLoss	Accuracy	MSE
1	2560	0.48584	0.761405	0.238595
2	256	0.485487	0.760302	0.239698
3	32	0.485506	0.763613	0.236387

Obviously, the Batch Size of 2560 shows a disadvantage due to its LogLoss, which is much higher than any other Batch Size. Compared with Batch Size of 32, Batch Size of 256 has a slightly lower Accuracy, but a shorter time of possessing according to previous discussion. Moreover, it has the lowest LogLoss, thus coming 1st among others.

1. Optimizer

   Optimizer is entries that regularize and modify the target function to make it reach a good convergence point within reasonable time.

Condition:

Number of Layers = 1, Activation Function = ReLu, Batch Size = 256, Features Selected {X_1 , X_5 , X_{10} , X_{14} }

	Learning Rate	Momentum	LogLoss	Accuracy	MSE
1	0.1	0.1	0.485487	0.760302	0.239698
2	0.1	0.5	0.485729	0.761221	0.238779
3	0.2	0.5	0.485515	0.76067	0.23933

The first parameter group of optimizer, with 0.1 of learning rate and 0.1 of momentum, performs the best LogLoss and a relatively lower Accuracy. In the second group, the momentum was added up to 0.5, which results an improve in Accuracy. However, LosLoss is negatively affected with a substantial increasing. The third group maintains the adjusted level of momentum, but increases Learning Rate to 0.2, resulting both LogLoss and Accuracy increasing. Overall, the first group has the best performance.

1. Number of Layers (Fully Connected layers)

   Number of layers has affection on efficiency of feature capturing and speed of training.

Condition:

Optimizer(with SGD, learning rate = 0.1, momentum = 0.1), Activation Function = ReLu, Batch Size = 256, Features Selected {X_1 , X_5 , X_{10} , X_{14} }

	Number of Layers	LogLoss	Accuracy	MSE
1	1	0.485528	0.761221	0.238779
2	2	0.485792	0.761957	0.238043
3	3	0.486321	0.76067	0.23933

Considering the number of layers, NN with one layer has the lowest LogLoss and a relatively high accuracy, thus outperforms NN with two layers and NN with three layers.