ITCS 6156: Machine Learning
Assignment 2: Neural Networks

Submitted by: Ashwin Venkatesh Prabhu
UNCC ID: 800960400
Email: avenka11@uncc.edu

Collaborated with: Febin Zachariah
UNCC ID: 800961027
Email: fzhacari@uncc.edu

Neural Networks

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of many highly-interconnected processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Artificial neural networks (ANNs) provide a general, practical method for learning real-valued, discrete-valued, and vector-valued functions from examples. Algorithms such as Backpropagation use gradient descent to tune network parameters to best fit a training set of input-output pairs. ANN learning is robust to errors in the training data and has been successfully applied to problems such as interpreting visual scenes, speech recognition, and learning robot control strategies.

The program uses “neuralnet” and “nnet” libraries to implement neural network for optical recognition and amazon dataset. Mainly, three methods are used and they are listed below:

- neuralnet(): neuralnet method is used to train the data set using training algorithms like backpropagation, resilient backpropagation with or without weight backtracking or the modified globally convergent version. There are many arguments for this method. Some of the relevant ones are listed below:

formula	A representation of the model to be fitted. Example: response variables ~ sum of covariates
data	Data frame containing the variables specified in the formula
hidden	A vector of integers specifying the number of hidden neurons (vertices) in each layer. Example: hidden = 10 or hidden = c (3, 5)
threshold	a numeric value specifying the threshold for the partial derivatives of the error function as stopping criteria. Default value is 0.01
learningrate	a numeric value specifying the learning rate used by traditional backpropagation. Used only for traditional backpropagation
algorithm	a string containing the algorithm type to calculate the neural network. Possible values are ‘backprop’, ‘rprop+’, ‘rprop-‘, ‘sag’, ‘slr’. Default value is ‘rprop+’
err.fct	a differentiable error function. Possible values are ‘sse’ for sum of square errors or ‘ce’ for class entropy. Error function used for the below analysis is ‘sse’
act.fct	A differentiable activation function. Possible values are ‘logistic’ for logistic function and ‘tanh’ for tangens hyperbolicus function. Activation function for the below analysis is ‘logistic’
linear.output	Logical. If activation function should not be applied to the output neurons. Default value is True

  The output of the neuralnet method is stored into an ‘nn’ object. The out object consists of two paramters which are relevant here: net.result (a list containing the overall result of the neural network for each replication); result.matrix (a matrix containing error, reached threshold, steps taken, estimated weights, AIC, BIC for each replication. Each column represents one replication.)

- plot(nn): The results of the training process are visualized by passing the trained neural network object to the plot method. The result of this method represents the plot of a trained neural network including trained synaptic weights and basic information about the training process.

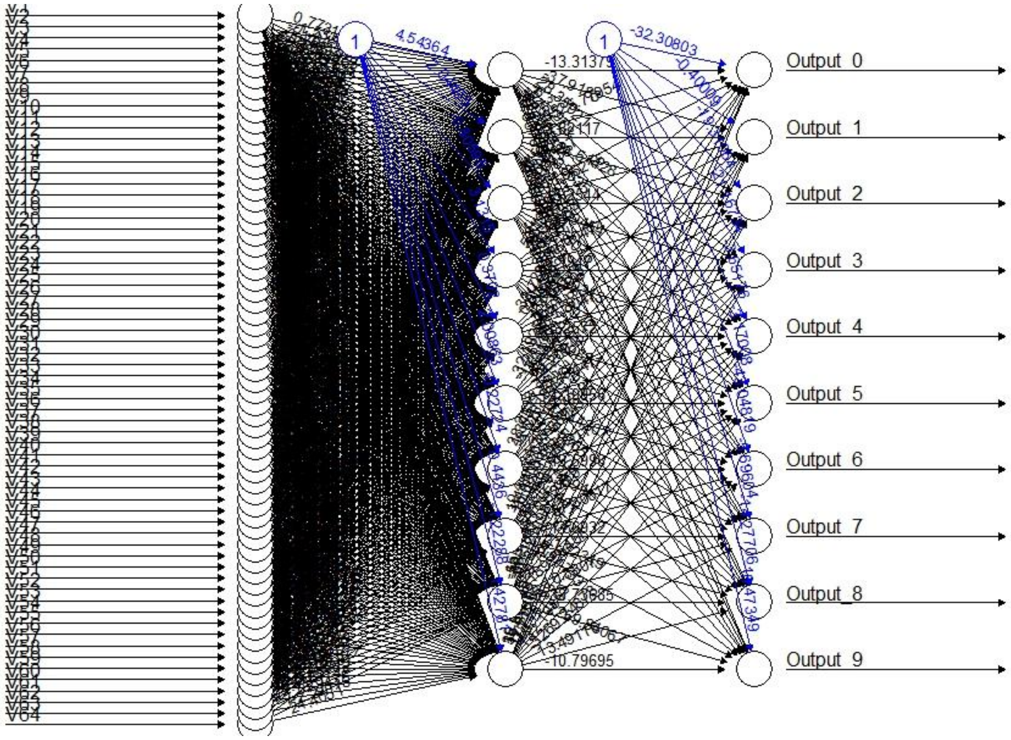
- compute(): This method calculates and summarizes the output of each neuron in input, hidden and output layers. It can be used to trace all signals passing the neural network for a given covariate combinations. A neural network is trained with a training data set consisting of known input-output pairs. It learns an approximation of the relationship between inputs and outputs and can then be used to predict outputs new observations relating to new covariate combinations. The method compute simplifies this calculation. It automatically redefines the structure of the given neural network and calculates the output for arbitrary covariate combinations.

Optical Recognition of handwritten digits dataset

Implementation Details:

1) The program is written in R programming language.
2) Program uses "neuralnet" and "nnet" libraries to create a neural network.
3) The datasets for training as well as test do not have a header, so the attributes are name V1 to V65. V1 to V64 represent feature attributes and V65 is for class attribute.
4) Main steps explaining the program:
    - Train data available in "optdigits_raining.csv" is loaded into "DigitRecog_data" variable
    - Train data is sample for cross validation. 80% of the data is used for training and 20% of the data is used for cross validation
    - The class variable here is the last column, namely, 'V65'. The class variable can have values 0 to 9. This column is split into 10 columns, each representing one possible value of the class variable (Eg: Output_0, Output_1, etc.). Each of these columns can have two possible values, 0 or 1. 0 means the observation does not fall under class 0, and 1 means the observation falls under that class. For each observation, only one of these columns will have value 1 and the rest will have value 0.
    - class_ind method from nnet package is used to split the class variable column into 10 different columns
    - Neural network is created using neuralnet method. Formula used for creating the neural network is "Output_0 + Output_1 + Output_2 + Output_3 + Output_4 + Output_5 + Output_6 + Output_7 + Output_8 + Output_9 ~ V1+V2+V3+…..+V64". Data used will be 80% of the train data.
    - After the neural network is created, the 'nn' object is used on the 20% of the train data kept aside for cross validation. compute method is used for this purpose.
    - Now the test data is loaded. compute method is used on the test data using the trained 'nn' object. The output for this is stored in the outtest variable.

5) Plot for the neural network is shown below.



6) Conclusion: The accuracy achieved for digit recognition using neural networks is on an average 85 to 90%. This is a better accuracy rate than that of decision tree algorithm which had about 75% accuracy approximately.

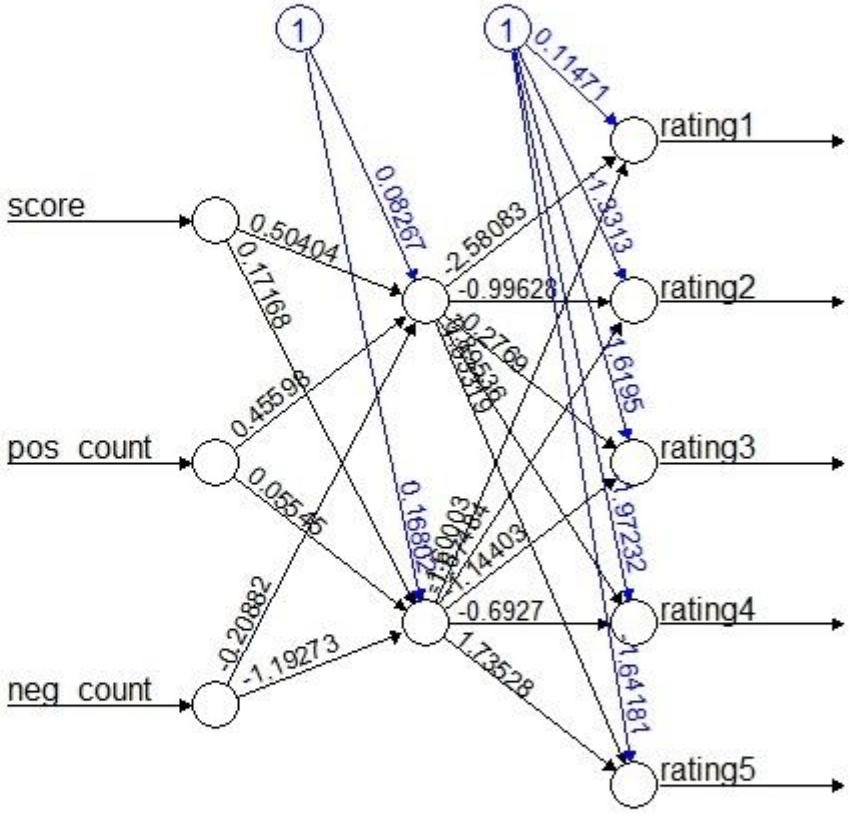
Amazon Reviews sentiment analysis

Implementation Details:

1) The program is written in R programming language.
2) Program uses “dplyr”, “plyr”, “stringr”, “tm”, “neuralnet”, “nnet” libraries.
3) The datasets have 3 headers, namely, “name”, “review”, “rating”. Class attribute for this dataset is “rating”.
4) A set of positive words in “positive_words.txt” file is loaded into positive_words variable while running the program.
5) A set of negative words in “negative_words.txt” file is loaded into negative_words variable while running the program.
6) Main steps explaining the program:
    - Amazon train data set is loaded and stored into a variable named ‘amazontrain’. The review column of the data set is passed as a parameter to tm_map method to create a corpus. This is important to remove unnecessary stop words and stem the reviews to get the words which will make a difference while calculating the sentiment score.
    - Once the positive and negative words are loaded into their respective variables, “score.sentiment” method is called to evaluate the sentiment score for each review in the training dataset.

- Score.sentiment method assigns a sentiment score to each review. A negative sentiment score means there are more number of negative words in the review than the positive words. A positive sentiment score means there are more number of positive words in the review than the negative words. If the sentiment score value is 0, then the review is neutral. An important point to note is that, a considerably good review with a rating 3 stars can have a negative sentiment score, if the review contains more number of negative words. Hence, the correlation between rating and sentiment score is loosely defined.
- Train data is sampled for cross validation. 80% of the data is used for training and 20% of the data is used for cross validation
- Neural network is created using neuralnet method. Formula used for creating the neural network is “rating1 + rating2 + rating3 + rating4 + rating5 ~ score + pos_count + neg_count”. Data used will be 80% of the train data.
- After the neural network is created, the ‘nn’ object is used on the 20% of the train data kept aside for cross validation. compute method is used for this purpose.
- Now the test data is loaded. compute method is used on the test data using the trained ‘nn’ object. The output for this is stored in the outtest variable.

7) Plot for neural network is shown below.



8) Conclusion: Here, I tried to use neural networks resilient backpropagation with backtracking weights algorithm to predict the rating of a review from the test data set. There is not much improvement in the accuracy of prediction compared to decision tree algorithm (There is a significant improvement for digit recognition data using neural network algorithm over decision tree algorithm). On further analysis, I feel there is a need to improve the function which calculates

the sentiment score for better results.

References

1) https://cran.r-project.org/web/packages/neuralnet/neuralnet.pdf
2) http://www.di.fc.ul.pt/~jpn/r/neuralnets/neuralnets.html
3) https://www.linkedin.com/pulse/neural-networks-using-r-jeffrey-strickland-ph-d-cmsp-asep
4) https://www.youtube.com/watch?v=JNtcslLZYsY
5) https://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html
6) https://cran.r-project.org/web/packages/nnet/nnet.pdf
7) Textbook – Machine Learning by Tom M. Mitchell