MASTER OF TECHNOLOGY

U 2/6: Computational Intelligence – I

CA1: Regression using Neural Networks

Submitted to:

Dr. Zhu Fangming
Institute of System Science
National University of Singapore

Prepared by:

Gopesh Dwivedi (A0178338R)
Saurabh Semwal (A0178339N)
Tanmoy Chakraborty (A0178252B)

INSTITUTE OF SYSTEMS SCIENCE

Contents

Part I: Regression Using Neural Networks 3
1.1 Problem Statement and Dataset 3
1.4 Neural Networks and Ensemble Architecture 5
1.4.1 MLFF Neural Network using resilient backpropagation with weight backtracking 5
1.4.2 Radial Basis Function Neural Network 5
1.4.3 Ensemble Architecture 6
1.5 Results and Evaluation 6
1.6 Conclusion and Limitations .... **Error! Bookmark not defined.**

Part I: Regression Using Neural Networks

1.1 Problem Statement and Dataset

The dataset in consideration summarizes a heterogeneous set of features about articles published by Mashable in a period of two years. The goal is to predict the number of shares in social networks (popularity) which is a continuous variable. The articles were published by Mashable (www.mashable.com) and their content as the rights to reproduce it belongs to them. Hence, this dataset does not share the original content but some statistics associated with it. The original content be publicly accessed and retrieved using the provided URLs. There are **61 attributes** of which 58 are predictive attributes, 2 non-predictive, 1 target variable i.e. Number of shares.

Here is the description of the data set:

No.	Name	Description
1	url	URL of the article (non-predictive)
2	timedelta	Days between the article publication and the dataset acquisition (non-predictive)
3	n_tokens_title	Number of words in the title
4	n_tokens_content	Number of words in the content
5	n_unique_tokens	Rate of unique words in the content
6	n_non_stop_words	Rate of non-stop words in the content
7	n_non_stop_unique_tokens	Rate of unique non-stop words in the content
8	num_hrefs	Number of links
9	num_self_hrefs	Number of links to other articles published by Mashable
10	num_imgs	Number of images
11	num_videos	Number of videos
12	average_token_length	Average length of the words in the content
13	num_keywords	Number of keywords in the metadata
14	data_channel_is_lifestyle	Is data channel 'Lifestyle'?
15	data_channel_is_entertainment	Is data channel 'Entertainment'?
16	data_channel_is_bus	Is data channel 'Business'?
17	data_channel_is_socmed	Is data channel 'Social Media'?
18	data_channel_is_tech	Is data channel 'Tech'?
19	data_channel_is_world	Is data channel 'World'?
20	kw_min_min	Worst keyword (minimum shares)
21	kw_max_min	Worst keyword (maximum shares)
22	kw_avg_min	Worst keyword (Average shares)
23	kw_min_max	Best keyword (minimum shares)
24	kw_max_max	Best keyword (maximum shares)
25	kw_avg_max	Best keyword (Average shares)
26	kw_min_avg	Average keyword (minimum shares)
27	kw_max_avg	Average keyword (maximum shares)
28	kw_avg_avg	Average keyword (Average shares)
29	self_reference_min_shares	minimum shares of referenced articles in Mashable
30	self_reference_max_shares	maximum shares of referenced articles in Mashable
31	self_reference_avg_sharess	Average shares of referenced articles in Mashable
32	weekday_is_monday	Was the article published on a Monday?

33	weekday_is_tuesday	Was the article published on a Tuesday?
34	weekday_is_wednesday	Was the article published on a Wednesday?
35	weekday_is_thursday	Was the article published on a Thursday?
36	weekday_is_friday	Was the article published on a Friday?
37	weekday_is_saturday	Was the article published on a Saturday?
38	weekday_is_sunday	Was the article published on a Sunday?
39	is_weekend	Was the article published on the weekend?
40	LDA_00	Closeness to LDA topic 0
41	LDA_01	Closeness to LDA topic 1
42	LDA_02	Closeness to LDA topic 2
43	LDA_03	Closeness to LDA topic 3
44	LDA_04	Closeness to LDA topic 4
45	global_subjectivity	Text subjectivity
46	global_sentiment_polarity	Text sentiment polarity
47	global_rate_positive_words	Rate of positive words in the content
48	global_rate_negative_words	Rate of negative words in the content
49	rate_positive_words	Rate of positive words among non-neutral tokens
50	rate_negative_words	Rate of negative words among non-neutral tokens
51	avg_positive_polarity	Average polarity of positive words
52	min_positive_polarity	minimum polarity of positive words
53	max_positive_polarity	maximum polarity of positive words
54	avg_negative_polarity	Average polarity of negative words
55	min_negative_polarity	minimum polarity of negative words
56	max_negative_polarity	maximum polarity of negative words
57	title_subjectivity	Title subjectivity
58	title_sentiment_polarity	Title polarity
59	abs_title_subjectivity	Absolute subjectivity level
60	abs_title_sentiment_polarity	Absolute polarity level
61	**shares**	**Number of shares (target)**

Table 1: Data Dictionary of Regression Dataset

1.2 Feature Engineering

The dataset was checked for missing values, data inconsistencies and outliers. No missing values were found in the data. However, we scaled and normalized the predictor numerical variables. Also, the target variable was scaled between 0-1 to meet our requirements. We removed some redundant attributes which have near zero variance or have linear relationship i.e. high correlation. One such variable was "is_weekend" which is already present in attributes for Saturday and Sunday. We also split the data into 80:20 ratio for training and testing respectively.

1.2.1 Dimensionality Reduction

Due to large number of input variables, we applied principal component analysis to reduce the number of predictor variables. After reduction, 30 Principal components were used to train the model which contained 92.4% information value.

1.3 Tools Used

We have done the implementation in **R tool** using **"RSNNS"** and **"neuralnet" packages**. Neuralnet provides with the training of neural networks using the backpropagation, resilient backpropagation with or without weight backtracking. The Stuttgart Neural Network Simulator (SNNS) is a library containing many standard implementations of neural networks. RSNNS package wraps the SNNS functionality to make it available from within R. A radial basis function network is an artificial neural network that uses radial basis functions as activation functions. We have implemented RBF using RSNNS package.

1.4 Neural Networks and Ensemble Architecture

1.4.1 MLFF Neural Network using resilient backpropagation with weight backtracking

The first Neural Network that we trained is a Multi-Layer Feed Forward Neural network using resilient backpropagation with weight backtracking. In a MLFF network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (of each layer) and to the output nodes. There are no cycles or loops in the network.

Rprop, short for resilient backpropagation, is a learning heuristic for supervised learning in feedforward artificial neural networks. This is a first-order optimization algorithm. Rprop takes into account only the sign of the partial derivative over all patterns (not the magnitude), and acts independently on each "weight". For each weight, if there was a sign change of the partial derivative of the total error function compared to the last iteration, the update value for that weight is multiplied by a factor η^-, where $\eta^- < 1$. If the last iteration produced the same sign, the update value is multiplied by a factor of η^+, where $\eta^+ > 1$. The update values are calculated for each weight in the above manner, and finally each weight is changed by its own update value, in the opposite direction of that weight's partial derivative, to minimize the total error function. η^+ is empirically set to 1.2 and η^- to 0.5.

To overcome the inherent disadvantages of pure gradient-descent, RPROP performs a local adaptation of the weight-updates according to the behaviour of the error function. In substantial difference to other adaptive techniques, the effect of the RPROP adaptation process is not blurred by the unforeseeable influence of the size of the derivative but only dependent on the temporal behaviour of its sign. This leads to an efficient and transparent adaptation process.

We created a MLFF network with two hidden layers of size 10 neuron each, with error function as SSE

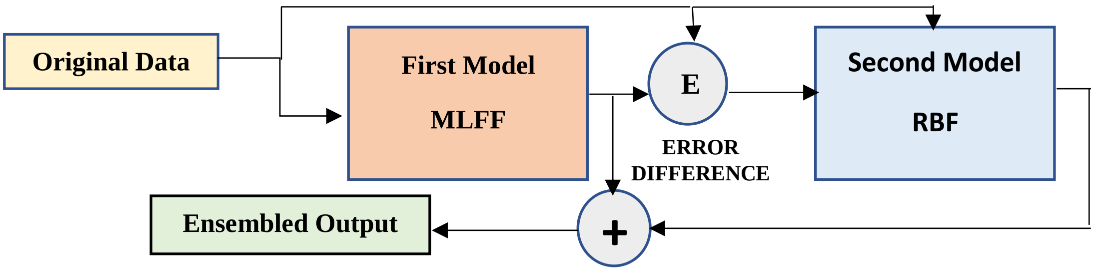
1.4.2 Radial Basis Function Neural Network

We implemented a RBF NeuralNet as the second neural network for regression. We used a single hidden layer with 40 nodes. RBF networks are feed-forward networks with one hidden layer. Their activation is not sigmoid (as in MLP), but radially symmetric (gaussian). Thereby, information is represented locally in the network (in contrast to MLP, where it is globally represented). Advantages of RBF networks in comparison to MLPs are mainly, that the networks are more interpretable, training ought to be easier and faster, and the network only activates in areas of the feature space where it was trained.

1.4.3 Ensemble Architecture

The goal of ensemble regression is to combine several models to improve the prediction accuracy on learning problems with a numerical target variable. The process of ensemble learning for regression can be divided into three phases: the generation phase, in which a set of candidate models is induced, the pruning phase, to select of a subset of those models and the integration phase, in which the output of the models is combined to generate a prediction.

We used an alternative approach to creating Ensembles. First, a model was generated using the original data. Second, another model was generated that estimates the error of the predictions of the first model and generates an ensemble that combines the prediction of the previous model with the correction of the current one. Finally, the output of the two ensembles is added up to give the final predictions.



1.5 Results and Conclusion

Model	Training Set RMSE	Testing Set RMSE
Base model on raw data	0.0311	0.0354
MLFF with Rprop on engineered data	0.0132	0.0138
RBF on engineered data	0.0134	0.0132
Ensembled Model	0.0127	0.0130

Table 2: Comparison of Various Models Architectures

RBF and MLFF+BP both being nonlinear networks can approximate arbitrary nonlinear functional mappings between multidimensional spaces. RBF are often considered to be superior to MLFF+BP in case of complex mappings but here using adaptive method for BP we got almost comparable results. We created an ensemble of these two models. MLFF+ rprop approximations were subtracted from the response values and the second model(RBF) learned this error. As a result, we did a summation of the result of first model and error approximations of the MLFF+RProp model learned by RBF. The final results were an improvement over the results of the individual models.