Seminar Project

Prediction of Facebook post Metric Neural Network

Thi Thu Trang Nguyen-S0569071

26. Februar 2022

Dozentin: Dr. Alla Petukhina

Inhaltsverzeichnis

1 Introduction

Currently, the use of social networks is familiar and popular with almost everyone, especially young people. Facebook is now the largest social network in numerous countries with 2.74 billion monthly users worldwide1. It not only the most popular social network, but also one of the largest online sales channels. Many advertising and communication companies use Facebook to attract potential customers. In marketing communication, social networks are considered a tool that cannot be ignored to build a brand image and attract customers. Currently, more than 80 million companies are active on this platform.
In order to be able to use the platform optimally as a company, data analyses are unavoidable. In the following, we will use a given data set to examine the extent to which the number of interactions (likes, comments, sharing of a post) can be explained and predicted with the help of statistical modeling. In doing so, we want to answer the question of which influencing variables positively or negatively affect the number of interactions and also to test whether this Neural Network machine learning method is suitable for data analysis in this case. For this purpose, we will first conduct a descriptive analysis before we start with the statistical modeling using neural networks and finally we will summarize our results.

2 Data Analysis

This data set *dataset_Facebook.csv* comes from a Facebook page of a cosmetics brand and contains data regarding its posts published during the year 2014. The data set includes 500 rows and 19 columns. Seven of the variables were taken directly from Facebook, while the rest were obtained by Moro et al. (2016) using data mining and published in an article Moro at al.(2016). For simplicity, most features have been renamed in R abbreviations. An overview of the individual features can be found in the Appendix

3 Neuronal Networks

The method to be used for analysis is the neural network, which is also part of supervised learning. The goal of neural networks is to mimic the way the human brain works.
Linear combinations of incoming variables are used as features to model the target variable as a non-linear function of that feature. Algorithms are designed to mimic learning in the brain. It is a mathematical model for a nerve cell that receives signals from other cells and, depending on the activation, transmits them to mimicked cells ordered neurons. Artificial neurons are weighted and passed to (nonlinear) activation functions. The neural network is built on biological neural networks.

Definition 1 (ANN). *(Artificial Neural Network) is a network of artificial neurons (units that process and retransmit incoming information) used for machine learning (pattern recognition).*

Neural network has input data with each node, transforms this input data by calculating the total input with the corresponding weights at the inputs, which applies a nonlinear transformation function for variables to calculate the intermediate state. The above 3 steps form a layer and the transformation function is also known as the activation function. The output of this layer is an input of the back layer.

Definition 2 (ANN model). *In general, the target variable y(x,w) in a ANN can be represented as the result of a non-linear activation function, into which the weighted sum of the explanatory variables (features) enters. This can be done in several shifts. The explanatory variables are adjusted by weights whose parameters are calibrated. The general model for the output variable of a stratum is thus:*

$$y(x, w) = h(w^T(x))$$

where x is input variables and w is weights.

3.1 Architecture

The architecture of a ANN consists of three layers. These are input layers, hidden layers and output layers. Information is transmitted from the input to the output in the form of feed-forward. The input layer is a layer of the ANN.
The input layer has a number of neurons equal to the number of independent variables. In this layer, the input variables are represented by nodes. The hidden layer consists of neurons, receives input data from the neurons in the previous layer and transforms this input for the next processing layers. In a ANN there can be many hidden layers. In these layers the weighted input variables are processed. The output layer has a number of neurons equal to the number of groups in the classification problem or only one neuron in the case of the regression problem. The output variables are specified in this layer.

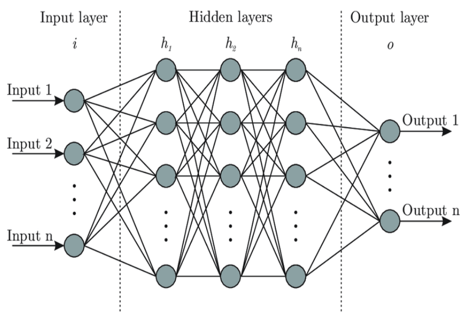

Abbildung 1: Basis architecture ANN

3.2 Activation Function

The activation function is a very important part of the neural network. It decides when a neuron is activated or not, whether the information received by the neuron is related to the given information or whether it should be ignored.
The activation function simulates the impulse transmission rate across the axon of a neuron. In an artificial neural network, the activation function acts as a non-linear component at the output of the neurons.
Activation functions are nonlinear functions applied to the output of neurons in the hidden layer of a network model and used as input data for the next layer.

Without nonlinear activation functions, a neural network with many layers is still as efficient as a linear layer. With only such simple computations, the neural network will actually not be able to recognize the complex relationships of the data.

3.2.1 Common activation function

There are many activation functions and researchers are still searching for new, more efficient activation functions. Linear function, SIGMOID function (logistic function, TANH function) and RELU function are currently the most commonly used activation functions.

Linear Function

Definition 3. *Linear Function has the form*

$$f(x) = x$$

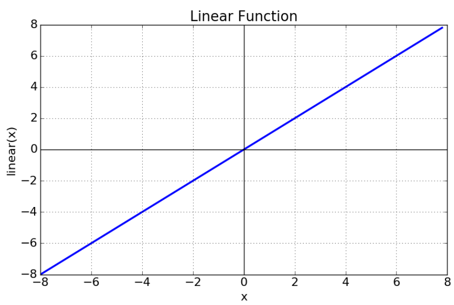

Abbildung 2: Linear function

The linear function is used in the output neurons. If a linear activation function is used in the connection between neurons, we will invisibly reduce all neuron layers to one, since the sum of the linear functions is still a linear function.

Sigmoid Function

The term sigmoid function is often referred to the special case logistic function. The sigmoid function takes a real number as input and converts it to a value between (0, 1). If the input is a negative real number, the output is asymptotic to

0. Conversely, if the input is a large positive real number, the output is a number asymptotic to 1. For inputs that are relatively close to 0, the sigmoid function converts 1 the input to a number between and 0 . Thus, for the sigmoid function, the 0 lower bound and is the 1 upper bound.

Definition 4. *The Sigmoidfunction is defined by*

$$sig(x) = \frac{1}{1 + e^{-x}}$$

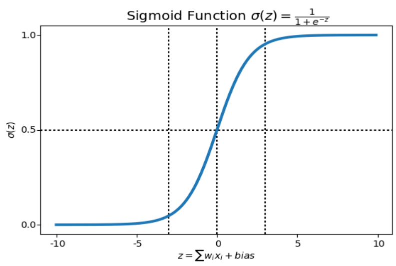

Abbildung 3: Sigmoid Function

Tanh Function

Definition 5. *Tanh function is defined by*

$$\frac{e^x - e - x}{e^x + e - x}$$

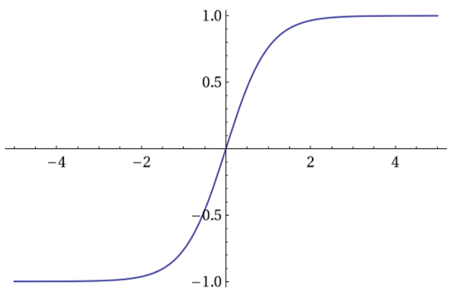

Abbildung 4: Tanh Function

Tanh function takes as input a real number and converts it to a value in the range (-1; 1). As with sigmoid, the Tanh function is saturated at both ends (the gradient changes very little at either end). However, the Tanh function is symmetric about the origin, so it overcomes a weakness of the sigmoid function.

Relu Function

Definition 6. *Relu Function (rectified linear unit) is defined by:*

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

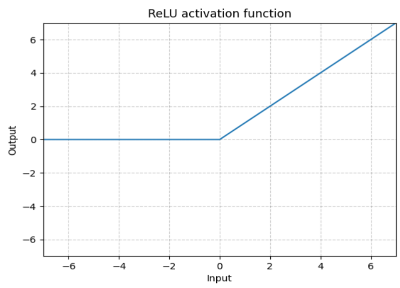

Abbildung 5: RELU Function

Some of its rather outstanding advantages over the logistic function and Tanh are fast convergence and faster calculations. Tanh and Sigmoid use the exp function and the formula is much more complex than ReLU, so it costs more to compute. In practice, we will use the sigmoid and tanh functions for the output layer only when the output needs values in the range (0,1) or (-1,1).

3.3 Implemented in R

3.3.1 neuralnet

With neuralnet, we used Tanh as the activation function. Because of the Char data type, the Input Type could not be read, so we used dummy variables for recording to convert it to an Int data type.

```r
library(neuralnet)
network = neuralnet(ti ~ Link + Status + Video + ptl + cat
+ pm + pd + ph + paid + lptr + lpti + leu +
lpconsumers + lpconsumption + lpiliked + lprliked+ lpleng ,
trainset, hidden=5 )
network$result.matrix
plot(network)
```

The neural network was trained with the neural network function with the 5 neurons in each layer.

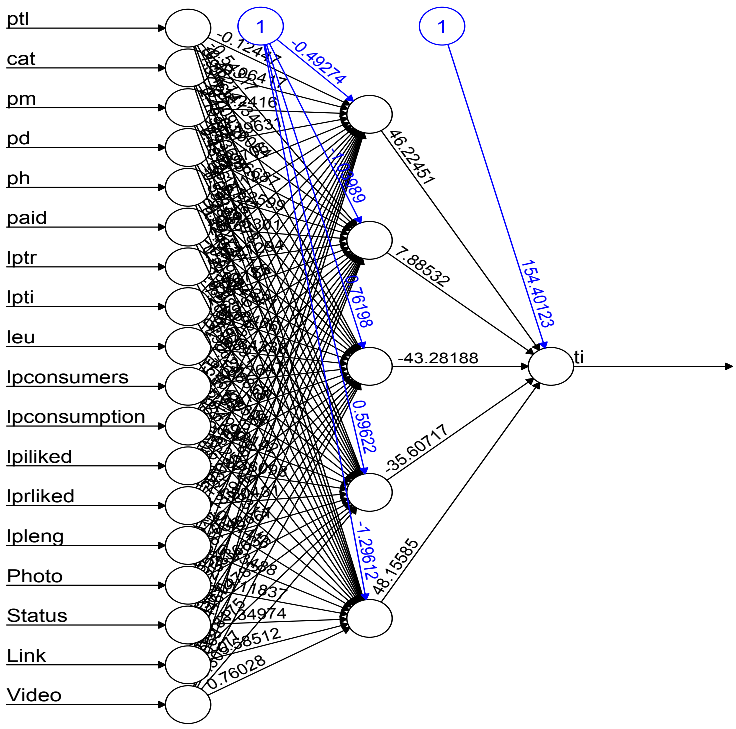

Abbildung 6: Neural network with 5 neurons

All variables except Likes, Comments, Shares were used as input neurons. Each input corresponds to an attribute of the data (pattern). Connection weights: This is a very important component of a KNN. It represents the importance (strength) of the input data for the information processing (data trans- formation process). Data from one layer has been derive to the other. Learning processing of KNN is the actual process of adjusting the weights of input data to get the desired result. The output variable here is Total Interactions. When there are so many connections, it is very hard to look whether the weight is positive or negative.
Therefore we use an alternative methode : nnet

3.3.2 nnet

`size = 3` here means the number of circles called neurons in the hidden layer. In `nnet` we can use the `size` instead of hidden in `neuralnet`

```
library(nnet)
library(NeuralNetTools)
ti.nnet <- nnet(ti ~ . - likes - shares - comments,
data = trainset,  size = 3, decay = 0.1)
```

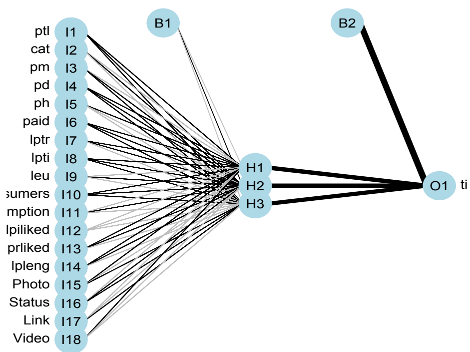

Abbildung 7: Neural networks with 3 neurons and distinct color

In this figure, you no longer see a number, but the color. Gray connections stand for negative numbers and black connections stand for positive numbers. `nnet` has an advantage to compare with`neuralnet`Because of the two colors you can clearly tell when a connection weight is positive or negative.
Neural networks are used to classify many inputs. The input attributes correspond to all variables with data type Int. There are 3 neurons in the hidden layer and the output is Total Interactions. Here are the parameters when neurons in the hidden layer is 3 equal:

```
#weights:  61
initial  value 9050580.182148
iter  10 value 9002583.768830
iter  20 value 9002351.562675
iter  20 value 9002351.499820
final  value 9002344.253513
converged
```

After running the neural network with each and 1,2,3,5neurons7 in the hidden layer, we see that the larger the number of hidden layers, the larger the weight, initital value, iter value, and final value.

Prediction on test data set

Based on the upper model, the prediction is now to be made on the test set.

Neuralnet

```
pred=predict(network,testset,type='response')
testti <- testset$ti
mean((pred - testti)^2)
[1] 9741.063
```

The MSE is now only at 9741.063. That's a lot less. There are also major differences between the Logistic and Tanh activation function. Using tanh as activation function we get a better result. We tried different values, for example 3, 4, 5 and

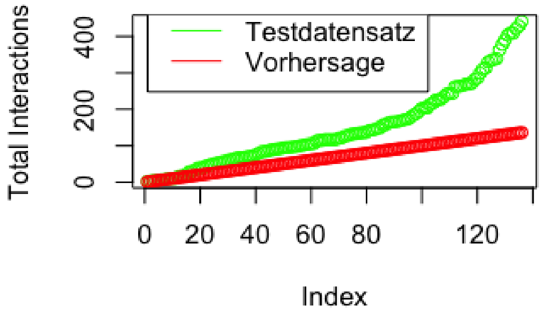

Abbildung 8: Test dataset and prediction

6 neurons in the hidden layer; Backpropagation and Resilient Backpropagation as an Algorithm; and the result shows that with neurons in the hidden layer = 5, Tanh function, neither backpropagation nor resilient backpropagation of the MSE is smallest.

Nnet

```
set.seed(100)
ti.nnet = nnet(ti ~ . - likes - shares - comments, data = trainset,
size = 6, act.fct = 'logistic', linout = T, decay = 0.01)
pred=predict(ti.nnet, newdata = testset)
testti <- testset$ti
mean((pred - testti)^2)
[1] 8703.595
```

When forecasting with `nnet` we get different results. Logistic and Tanh as activation functions result in the same MSE. We tried different values, for example 3, 4, 5, 6, 7, 8, 15 and 18 neurons in the hidden layer and with 6 neurons in the hidden layer we get a better result for the MSE.
It is even better than `neuronalnet` which scored 8703.595. This is then the best MSE result for neural networks.

The MSE is so big because in R you can only make one hidden layer. I may need other programs to get better results.

4 Evaluation

With neural networks it is difficult to answer the question. On the other hand the signs of the coefficients make it easy to see which influencing variables have a positive or negative effect on the target variable but it is complicated because they are sent through hidden layers and so it is no longer possible to see in the output layer which effect an input variable has on the output variable. To answer the research question, we can say that posting a video has the strongest positive effect on the number of interactions.Posting a status or action post has the opposite effect.

Overall, i can say that based on my analyses, i have already found a model that can explain and predict our target variable well, but it is by no means a perfect model. A possible approach is the modeling with the help of a Poisson regression. This is because it is designed to model count variables and could therefore provide a better result than a multiple linear regression.

At the end the main reason that brought me to this topic is because I am fascinated by the metric data from Facebook and the neural network machine learning model

Feature	Explaination	Source	Designation in R
Page total likes	Number of people who like your page	Facebook	ptl
Type	Media type (photo, video, status, link)	Facebook	type
Category	Content of Characterization: 1= Action(special offers), 2= Product(direct advertising), 3= Inspiration(content is not directly related to the brand	Facebook	cat
Post Month	Month in which post was posted (1= January , 2= February,. . .)	Facebook	pm
Post weekday	On which day of the week the post was posted (1= Sunday, 2= Monday,. . . 7= Saturday)	Facebook	pd
Post hour	At what time was posted	Facebook	ph
Paid	Whether the company paid Facebook to make the post to promote (1= yes, 0= no)	Facebook	paid
Lifetime post total reach	Number of individual people who saw the post	Moro at al.	lptr
Lifetime post total impression	How often the post was viewed in total	Moro at al.	lpti
Lifetime engaged users	Number of individual person who have clicked on the post	Moro at al.	leu
Lifetime post consumers	Number of person (multiple possible) who have clicked on the post	Moro at al.	lpconsumers
Lifetime post consumption	Total number of clicks on a post	Moro at al.	lpconsumtion
Lifetime post impressions by people who have liked your page people people who have liked your page	How often the post was seen by people who like the page	Moro at al.	lpiliked
Lifetime post reach by people who like your page	Number of individual people who saw the post, because they like the page	Moro at al.	lprliked
Lifetime people who have liked your page and engages with your post	Number of individual people who like your page and have clicked on the post	Moro at al.	lpleng
Comments	Number of comments of the post	Moro at al.	comments
Likes	Number of likes on the post	Moro at al.	likes
Shares	How many times a post was shared	Moro at al.	shares
Total Interactions	Sum of “likes”, “shares” and “comment”	Moro at al.	ti

5 Literature

https://www.agorapulse.com/blog/facebook-post-consumers-and-post-consumption
https://medium.com/technologymadeeasy/for-dummies-the-introduction-to-neural-netwo
https://victorzhou.com/blog/intro-to-neural-networks/ https://ml4a.github.
io/ml4a/how_neural_networks_are_trained/ https://en.wikipedia.org/wiki/
Activation_function https://www.datacamp.com/community/tutorials/neural-network-mod
https://rviews.rstudio.com/2020/07/20/shallow-neural-net-from-scratch-using-r-part
https://www.analyticsvidhya.com/blog/2017/09/creating-visualizing-neural-network-i
https://moodle.htw-berlin.de/course/view.php?id=35125 http://www.math-evry.
cnrs.fr/_media/members/aguilloux/enseignements/m1mint/moro2016.pdf https:
//sites.google.com/site/scmoro/facebook