Data Analytics Project 1: Predicting Popularity of Articles (N.03)

Aouina Chadha (aouina.chadha@usi.ch)
Marco Gabriel (marco.gabriel@usi.ch)

2024.04.16

1 Introduction

The goal of this project is to predict the number of shares (popularity) of news articles published by Mashable, given numerous other features such as day of publication and text length. The provided dataset contains a heterogeneous set of features about articles published in a period of two years.

2 Data Exploration

For each article, the dataset contains the URL, which is made of the date and title, and various numerical features. These attributes include the number of days between the publication and the dataset acquisition, the length of the article, the number of links to other articles, the Number of images, the channel/topic, the average shares of referenced articles, the weekday of publication, and the title's sentiment polarity.

2.1 Descriptive Statistics

A range of statistic measures of select features is shown in Table 1. Some features, such as the weekday of publication, are given in one-hot format. However, most are integers or floats.

Feature	**Std**	**Min**	**Mean**	**Max**
Days since publication	170	161	441	731
Article length	471	0	544	8474
Number of outbound links	4	0	3	74
Number of images	9	0	5	128
Avg. shares of referenced articles	26150	0	6819	843300
Title sentiment	0.26	-1	0.075	1
Shares	12614	1	3511	843300

Table 1: Descriptive Measures of Select Features

2.2 Visualizations

Exemplary, as shown in Figure 1 the distribution of the number of shares of the average keyword is highly skewed. There are very few keywords with average shares in the range of tens of thousands, whereas the vast majority of keywords average only a few thousand shares.

A similar distribution can be observed in the number of shares per article, visible in Figure 2. There are a few outliers with hundreds of thousands of shares, while the majority of articles only have a few thousand shares.

2.3 Observations

To find correlations, we used the Spearman and Pearson correlation coefficients. Just a low number of features show any correlation to the number of shares. Figure 3 shows all features with a Spearman correlation coefficient of at least 0.15.

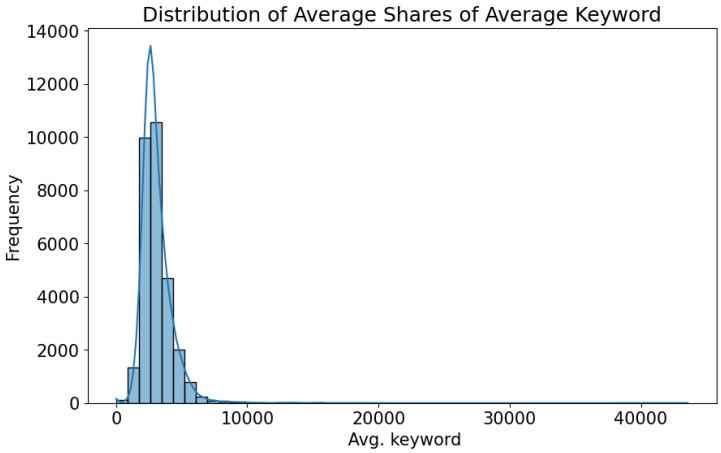

Figure 1: hist. of avg. shares for avg. keywords

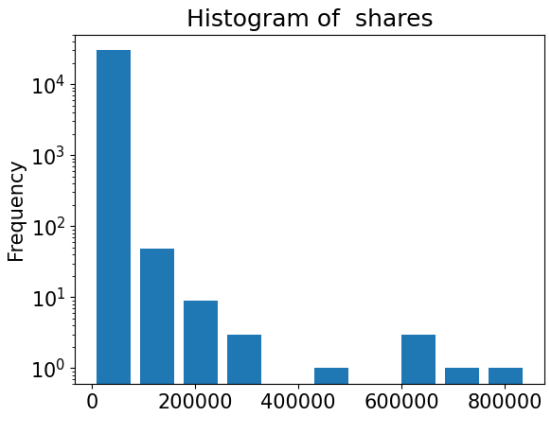

Figure 2: histogram of shares per article

A selection of these features are the average shares of an average keyword of an article, and average shares of referenced articles. These two features are shown in Figure 4. With a Spearman correlation coefficient of around 0.2 these are among the highest correlated features, yet the coefficients indicate only weak non-linear correlation.

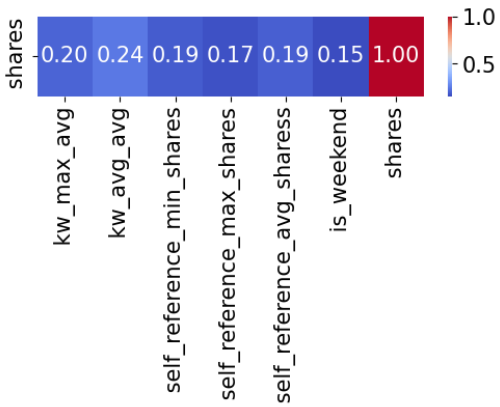

Figure 3: Spearman Correlation Matrix of the Most Correlated Features with Respect to Shares

3 Pre-process the data

We removed the url column as it only contains the urls which are the only non-numerical values in the dataset, and non-predictive.

We did not find any missing or corrupted data and therefore did not further process the data.

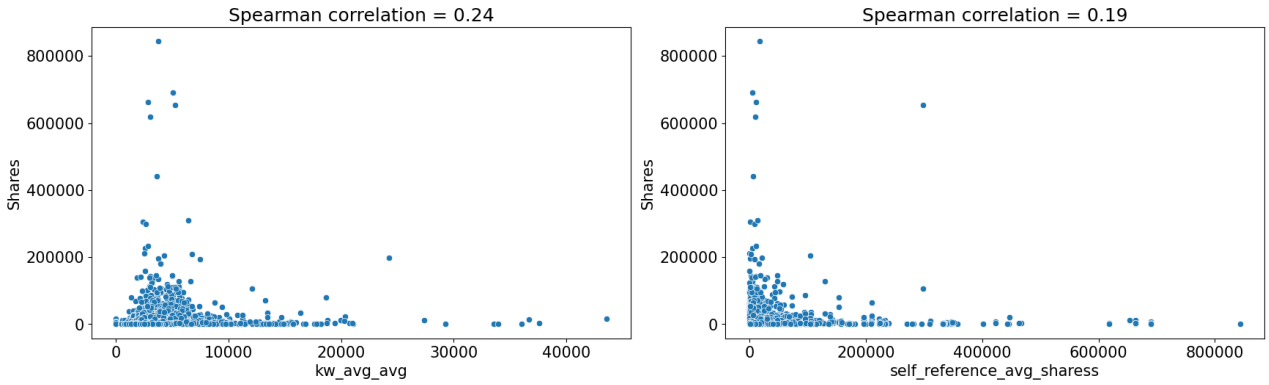

Figure 4: Two Features that are among of the highest correlated features with respect to the number of shares.

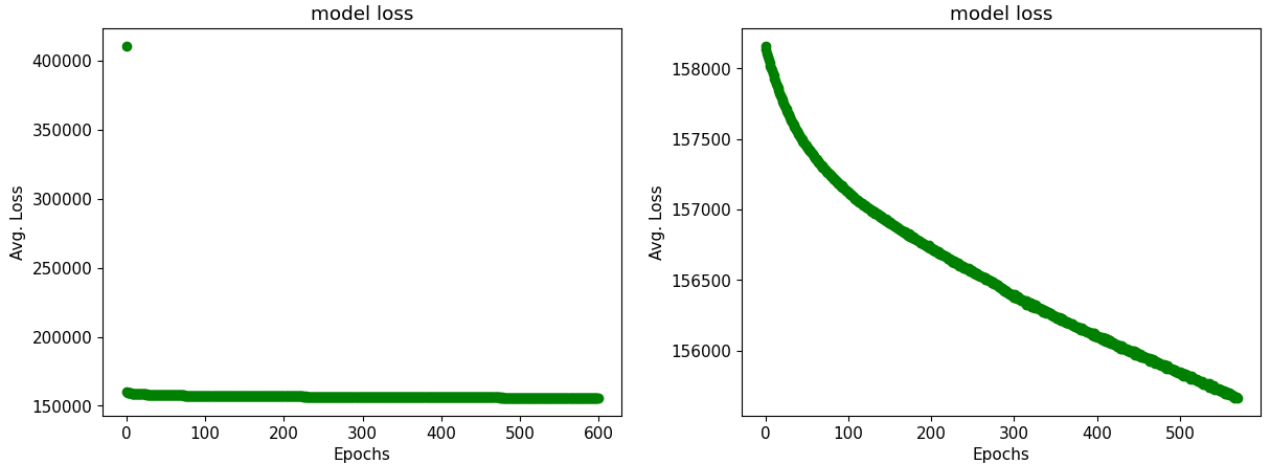

Figure 5: Example of a Model's Loss During Training. For Better Visibility, the Right Plot Excludes the First Epoch.

4 Prediction Models

To predict the target value, we began by fitting a linear model to the training data. We used the linear regression implementation of scikit-learn. Furthermore, we trained multiple Feedforward Neural Networks with different numbers of parameters using PyTorch. We chose this simple architecture, as the provided data is non sequential and not spatial, which eliminates the need for LSTM, convolutional, or similar layers. The performance of the linear model is used as a reference for the hyper parameter tuning. Viable hyper parameters must produce models that are at least as good as the linear model.

For determining the optimal learning rate, number of parameters and number of layers, we tested various values and compared their mean squared loss over the training data. We started with a sensible guess of 2 layers with 2000 neurons each, as this should be more than sufficient for an input of size 59. Afterwards, we trained multiple smaller models. We hoped that smaller models would generalize better, or at least achieve the same performance using fewer resources. These models ranged from 2 layers with 10 neurons each (Tiny), to 2 layers with 200 neurons (Small), up to 3 layers with 200 each (3 layer).

Considering the training process, we experimented with normalizing the input features, but as it did not improve the models, we stopped using it. However, we used batch normalization with batch size 1000, as it not only greatly increased the training speed, but also improved the training process. The resulting training process can be seen in Figure 5, which depicts the training loss of the 3-layer model during all epochs. The huge gap between the first and second epoch implies that most of the training is done in the first epoch. For this reason, the right diagram does not contain the first epoch for improved readability. However, the loss still decreases in the next hundreds of epochs, although slower.

As visualized in Figure 6, the "small" model, which is made of 12'200 parameters, reached the lowest training loss with a value of 154'000 or ≈390 squared. In comparison, the tiny models yield a loss of 168'000, which is roughly 410 squared. Thus we learn, that the 600 parameters used by the "tiny" model, is not enough to learn this data optimally.

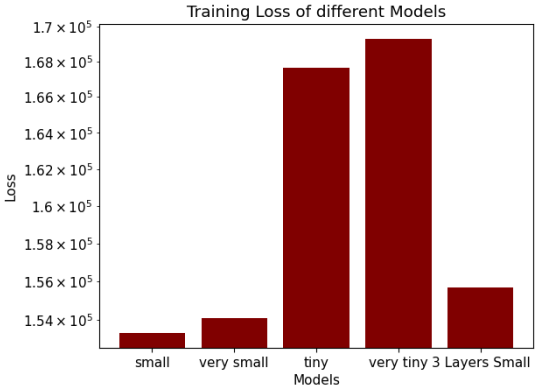

Figure 6: Mean Squared Error on the Training Data.

5 Evaluation and Comparison of Prediction Models

To evaluate the built models, we compared their loss on the test data. Figure 8 compares said loss. The relations are similar to the training loss, with the neural networks having only a few parameters performing the best. However, the accuracy is considerably worse in comparison to the training. With values of about 60 million, the mean squared error is over 100 times higher than during training. This was to be expected considering the low correlations found in the dataset.

As judging the performance of a model using the squared loss can be unintuitive, we also show the targets and predictions over the Test data made by the best model in Figure 7, as well as the relative error of the predictions. We see, that while the model is far from perfectly predicting the targets, most predictions are within a factor of 10 of the correct value. Considering the distribution of the number of shares, we consider the model to predict them with moderate accuracy.

6 Conclusion

We found the data to be relatively uncorrelated with peak correlation coefficients with the number of shares at around 0.2. Thus, we hypothesized that no model would be capable of predicting the number of shares very precisely. After training multiple models, we found that on the Test data, the best model achieved a mean squared error of around 60 million, which we consider to reflect moderate precision, given that the mean value of shares is 3511 and there are multiple outliers in the range of hundred thousand.

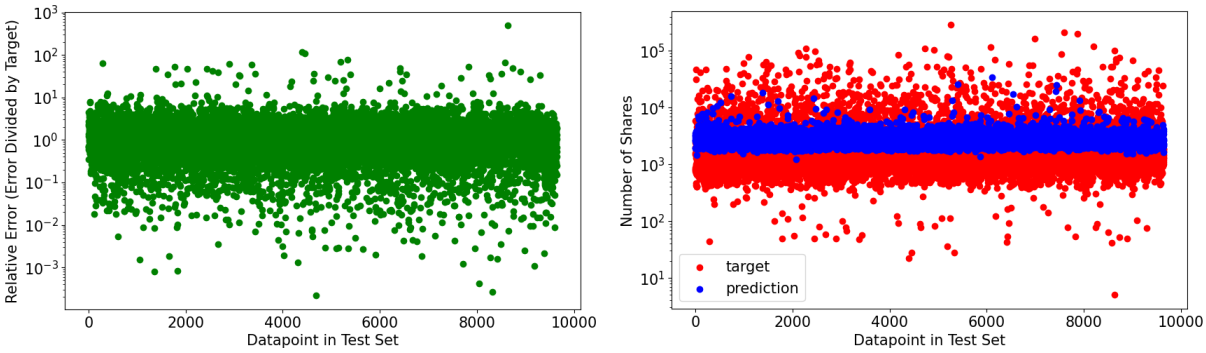

Figure 7: The Left Side Shows the Relative Error (Error Divided by Target). The Right Side Shows a Plot of the Test Targets and the Predictions. All Data is from the "Small" Model.

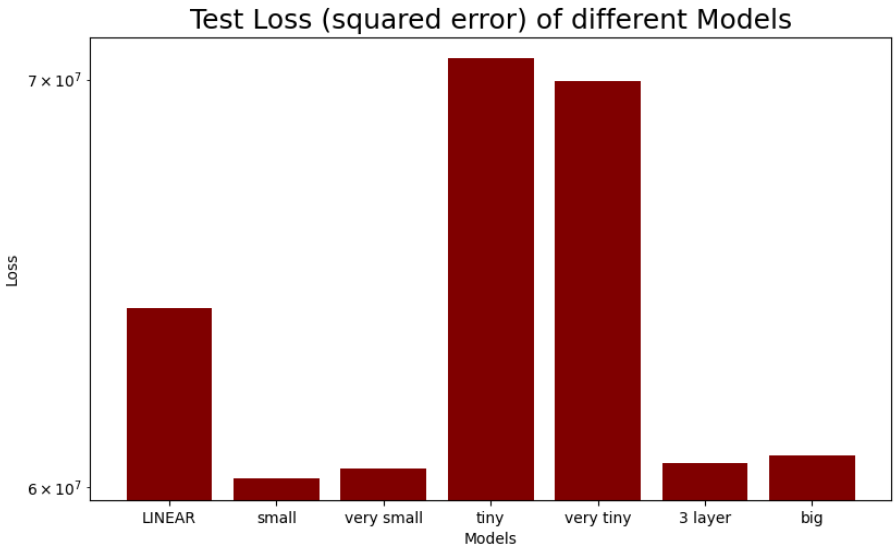

Figure 8: Mean Squared Error on the Test Data