Predicting Online News Popularity

Federico Sisci
Politecnico di Torino
Student id: s304657
s304657@studenti.polito.it

Gioele Costa
Politecnico di Torino
Student id: s318947
s318947@studenti.polito.it

***Abstract*—In this paper we propose our approach to the problem of predicting the number of shares of online articles. First we focus on data exploration in order to manage possible issues in the dataset, then we compare the performance of basic regression models so as to choose a starting optimal estimator. Finally we improve our regressor through feature selection and we perform the predictions on the evaluation set.**

I. Problem overview

In the era of digital information, quantitatively predicting how much an article will be shared means understanding which are the main factors that drive the spread of information nowadays. This may help yielding contents that reach more people and also studying how online communication affects people's opinion.

In our case, the given dataset is a collection of 39632 articles published by Mashable, a US news website, in a period of two years [1]. It is divided into two parts:

- the *development set* with 31715 rows. This part is used for training and validation so it contains the target column;
- the *evaluation set* with 7917 rows. This part must be used as unseen data so it does not contain the target column.

It is composed by 49 attributes (47 predictive and 2 non predictive) listed below.

Feature	Type (#)
Words	
Number of words in the title	number (1)
Number of words in the article	number (1)
Average word length	number (1)
Rate of non-stop words	ratio (1)
Rate of unique words	ratio (1)
Rate of unique non-stop words	ratio (1)
Reference	
Number of links	number (1)
Number of self links	number (1)
Min, average and max number of shares of self links	number (3)
Keyword	
Number of keywords	number (1)
Worst keyword	number (3)
Average keyword	number (3)
Best keyword	number (3)

Feature	Type (#)
Media	
Number of images	number (1)
Number of videos	number (1)
Others	
URL of the article	nominal (1)
Days between publication and dataset acquisition	number (1)
Day of the week	nominal (1)
Article category	nominal (1)
Natural Language Processing	
Closeness to LDA topics	ratio (5)
Subjectivity scores	ratio (3)
Polarity scores	ratio (9)
Rate of positive words	ratio (2)
Rate of negative words	ratio (2)
Target	
Number of shares	number (1)

Many papers treat this problem as a classification task by partitioning the number of shares into two or more classes of popularity. Then, they predict the class of new articles (i.e, their popularity) by means of classification models. Here we address the problem as a regression task, therefore we want to predict a numerical target.

II. Proposed approach

A. Preprocessing

In this part we perform some data exploration in order to improve the quality of the data. First we discard the features *url* and *timedelta* since these are metadata and so not predictive. Then, for each attribute we compute some statistics that allow us to discover both missing values and outliers in the features. For instance, this is the output for the category *media* given certain percentiles.

	num_imgs	num_videos
count	31705.000000	31707.000000
mean	4.569153	1.256631
std	8.375140	4.147667
min	0.000000	0.000000
1%	0.000000	0.000000
25%	1.000000	0.000000
50%	1.000000	0.000000
75%	4.000000	1.000000
99%	38.000000	21.000000
max	128.000000	91.000000

As we can notice for both the features there are both outliers and missing values. Values from the 99% are very far from the rest, these represents outliers to handle. We achieve that through the *boxplot method*: we assign to outliers values at lower or higher percentiles in order to carry them into a better range of values. Given the large number of attributes, we mostly do this work on features grouped by categories, although some cases required us to analyze them individually. However, we only treat extreme outliers, since a little bit of noise improves the generalization ability of the model. In fact, we observe that deleting less outliers on the target feature makes us reach a large better score on the public leaderboard, despite a little worse score on the development set. This probably means that we were overfitting the development set. In Fig. 1 we can look at the boxplots of the features *num_imgs* and *num_videos* before and after handling outliers.

Regarding missing values, we notice them from the count value since it differs from the total number of rows in the dataset (39632). Our choice is to fill them with the median value of the respective column: we prefer it to the mean because it is less sensitive to the presence of outliers.

Finally, categorical features *datachannel* and *weekday* are turned into numerical values by means of *One-Hot Encoding*. This leads to a number of columns equal to 58.

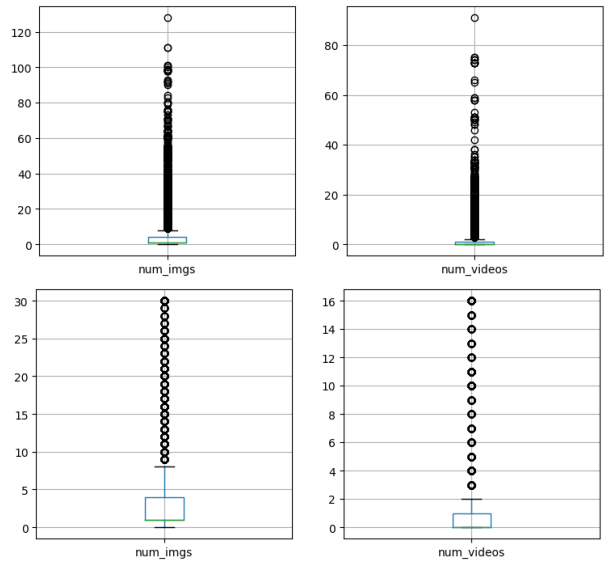

Fig. 1. Boxplots of the features *num_imgs* and *num_videos* before (above) and after (below) handling outliers.

A further step would be applying *feature selection* through feature correlation analysis and *feature reduction* by means of principal component analysis (PCA). For feature selection we compute the pairwise correlation among all the attributes and for each couple with a correlation $\geq 90\%$ we discard one of the two columns involved. In the following figure it is reported the heatmap of the correlation matrix. The lighter is the cell, the higher is the correlation between the respective features.

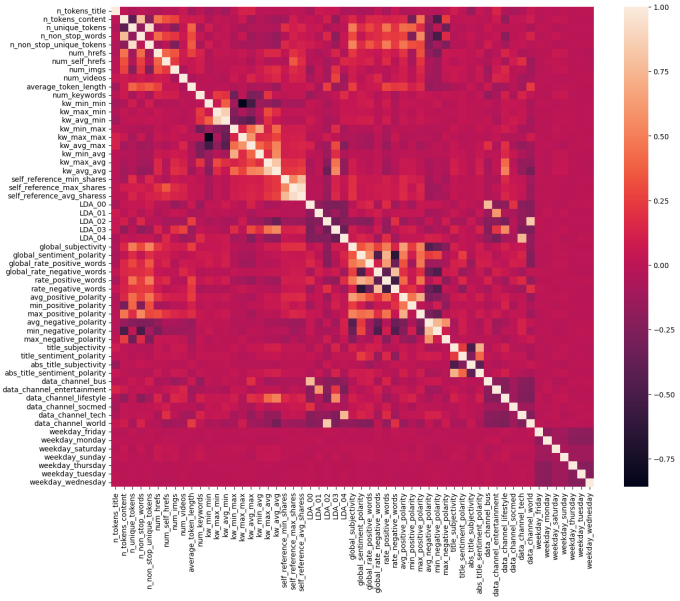

Fig. 2. Heatmap of the correlation matrix.

In order to apply PCA, we search for the number of components that explains the 95% of the total variance. As we can see in the presented graph, which shows the cumulative explained variance against the number of components, this number is 36.

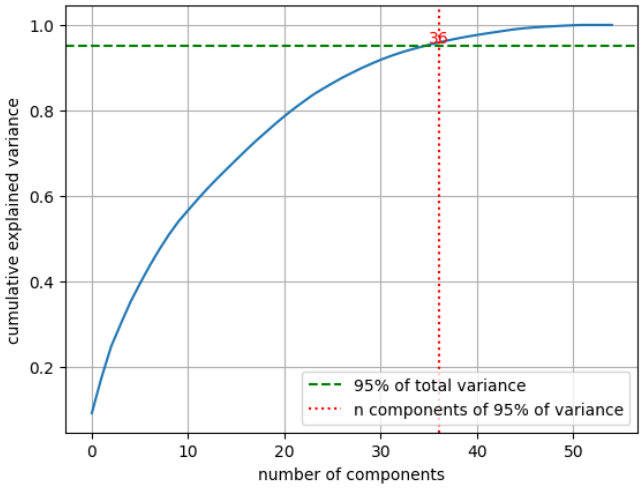

Fig. 3. The graph of the cumulative explained variance as a function of the number of components.

For both cases, *feature selection* and *feature reduction*, we observe a worsening in performance, therefore we decide to keep the original set of features and don't consider this two steps.

B. Model selection

Once the data is cleaned we are ready to perform some basic model evaluation. After splitting the dataset in training, validation and test set, we carry out model selection only on the first two. Then, we standardize the features and we evaluate five models with their default parameters: *Linear Regressor*, *Ridge*, *Lasso*, *Random Forest Regressor* and *Support Vector Regressor*. The evaluation is based on the **Root Mean Square Error (RMSE)** metric and its formula is the following:

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)}$$

where n is the number of samples, i.e. the number rows in the dataset, $\hat{y}_i$ is the predicted value and y_i is the true value.

These are the resulting scores:

Regression model	**RMSE**
Linear Regressor	4531.9227136601
Ridge	4531.6851014266
Lasso	4531.6212371616
Random Forest Regressor	4650.8293193519
SVR	4949.7808834823

Since Lasso achieves the best RMSE score and given that this kind of regularization zeros some coefficients, this may suggest that there is only a subset of features that is really significant for the problem. Therefore we use Lasso to carry out a new *feature selection* on the entire dataset. Before that, we want to introduce a different variant of Lasso:

LassoLarsIC [2]. It is an information-criteria based model that uses the *Akaike Information Criterion* (AIC) or the *Bayes Information Criterion* (BIC) to penalize the over-optimistic scores of different Lasso models in k-fold cross-validation. Since the default setting makes use of the AIC criterion, this is the formula considered:

$$AIC = n\log(2\pi\sigma^2) + \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sigma^2} + 2d$$

where d is the degree of freedom, σ^2 is an estimate of the noise variance, y_i and $\hat{y}_i$ are respectively the true and predicted targets, and n is the number of samples. This regressor achieves an RMSE score of 4531.48 which is less than the previous scores, so we rely on it as the final estimator.

Now we are ready to perform *feature selection*. To do that we use a cross validation function by *Scikit-learn* that performs a recursive feature elimination: given an estimator that computes weights on the features, it recursively select the features considering smaller and smaller set of them until it reaches the best score given a certain metric. As we previously anticipated, in this process we use Lasso as estimator. This operation selects 45 features from a total of 57 leading to an average RMSE score of 4511.81 (from cross validation with Lasso on training+validation data). It follows the set of deleted features:

n_unique_tokens	num_videos
num_keywords	kw_min_max
kw_max_avg	LDA_01
global_sentiment_polarity	max_positive_polarity
abs_title_sentiment_polarity	min_negative_polarity
data_channel_lifestyle	weekday_friday

C. Hyperparameters tuning

In this part we use a *grid search cross validation* to tune the parameters of the model LassoLarsIC. In this process, all the possible combinations of parameters are tested in order to define the best configuration. For each one, the dataset is splitted into five folds: in turne, each of them is used as the test set while the remaining ones are used to train the model, then the average score is computed. These are the parameter tuned and the tested values for each of them:

Hyperparameter	**Values**
criterion	*AIC, BIC*
fit_intercept	*True, False*
positive	*True, False*

III. Results

The resulting best configuration is the default one with criterion = *AIC*, fit_intercept = *True* and positive = *False*. Now we fit the model on the development set and we make the prediction on the evaluation set. The followings are respectively the obtained scores on the development set and on the public part of the evaluation set:

- development set score: 4504.689
- public score: 5938.345

IV. Discussion

In conclusion, we adopte several strategies to maximize the effectiveness of the pipeline. We notice that the most relevant part is the data pre-processing step, therefore we explore several techniques to improve the quality of the dataset. In addition, we do a selection of the features trying to identify the most relevant ones for the problem. Of course a better proceeding would select the best model performing a grid search cross validation among the previous models and tuning some of their hyperparameters. Following this approach we observe that our available computational power is not sufficient to make the algorithm converge. Despite that, since our performance on the public leaderboard, we are satisfied with the presented pipeline.

References

[1] V. P. C. P. Fernandes, Kelwin and P. Sernadela, "Online News Popularity." UCI Machine Learning Repository, 2015. DOI: https://doi.org/10.24432/C5NS3V.

[2] scikitlearn.org, "Information-criteria based model selection." https://scikit-learn.org/stable/modules/linear_model.html#lasso-lars-ic.