MS&E 226 Project
Prediction of the number of shares of an article on social networks

GUILLAUME CHHOR
Stanford University
ICME
gchhor@stanford.edu

SIMON HAGEGE
Stanford University
MS&E
hagege@stanford.edu

December 16, 2018

Abstract

In this project, we try to address the challenge of predicting online news popularity based on some features extracted from the article itself. This kind of analysis can be useful for social companies in order to asses which articles should be highlighted in the newsfeed. In the first part of the project, we will describe the content of our dataset, notably spotting potential issues resulting from missing values and looking at statistics on the dataset across categorical variables as well as pairwise correlations between our variables. Building upon this data exploration, in the second part we will design our best model for classification and regression after having got rid of collinear covariates, logtransformed some of our variables, done feature selection using model scores and looked at interaction with categorical variables. For both of these tasks, we gave an estimate of the test error for each models, including a baseline defined beforehand, using cross validation and chose the best model accordingly. In this third part we will the test error of our best model for classification and regression, analyze a linear regression model from an inference perspective and finally discuss some applications of and improvements that can be brought to our model.

I. PART I

i. Dataset

The chosen dataset has been made available by the digital media Mashable and can be found online in the UC Irvine Machine Learning Repository *(link)*. This dataset allows the analysis of online news popularity based on features about articles published by Mashable in a period of two years. The proxy for the popularity of an article is the number of shares in social networks. Hence, in this regression task, the output variable Y we want to predict is the number of shares of an article in social networks, and the covariates are a set of statistics associated with this article. The raw dataset consists of 1 response variable and 60 explanatory variables for $39,644$ observations. Note that we can consider the response variable (popularity) as a continuous response variable by considering directly the number of shares in the social networks, or as a binary response variable (popular or not) by defining a threshold on the number of shares such that the number of shares Y (shares) exceed this threshold, then the article is considered as popular 1, otherwise, it is considered as impopular $Z = 0$. The analysis of this dataset will allow us to see if a relation can be established between the popularity of an article on the social networks and features that can be directly extracted from it. It will be notably interesting to determine which particular features of an article can be associated with its popularity or unpopularity, and going further, see if we can see what will make an article popular or not.

ii. Data cleaning and features analysis

Before explaining some of the features of the dataset, we get rid of the URL of the article feature that is extraneous to the data analysis. Then, we can see that there are some explanatory variables that can be merged together to form one categorical variable. Namely:

- the 8 features: `weekday_is_monday, tuesday, wednesday, thursday, friday, saturday, sunday, is_weekend`, can be merged in the categorical variable `day_of_week` variable that would take 7 values

- the 6 features: `data_channel_is_lifestyle, entertainment, bus, socmed, tech, world`, can be merged in the categorical variable `data_channel` variable that would take 7 values (some articles do not fit in any of the categories mentioned above, and will be categorized as undetermined)

To have more insights on the features we can separate them into different categories. For a given article we can extract features related to words, links and references, visual contents, time, keywords (topics) and category and finally features extracted by sentiment analysis of the text.

Category	Features	Type
Words	number of words in {title/content} rate of {unique/non-stop/unique non-stop words in content average word lenght	Numerical (x2) Numerical (x3) Numerical (x1)
Links & References	number of {links/links to other Mashable articles} {min/max/average} shares of referenced Mashable articles	Numerical (x2) Numerical (x3)
Visual Content	number of {images/videos}	Numerical (x2)
Time	day of the week time online	Categorical (x1) Numerical (x1)
Keywords & Category	number of keywords {min/max/average} shares of {worst/best/average} keyword article category	Numerical (x1) Numerical (x9) Categorical (x1)
Sentiment Analysis	closeness to LDA topic 0 to 4 {title/content} {subjectivity/sentiment polarity} rate {positive/negative} words in content rate {positive/negative} words among non neutral words {min/max/average} polarity of {positive/negative} words	Numerical (x5) Numerical (x4) Numerical(x2) Numerical (x2) Numerical (x6)

TABLE 1: COVARIATES SUMMARY

The values that consists in counts of and statistics on words, links and actual data associated with the article (date of publication and category) can be considered as reliable and accurate. However the features extracted by sentiment analysis can be subject to errors and variations depending on how we build them.

At first sight, there is no `NA/NULL` values in this dataset, but looking more closely at the data, we discovered that features that should be strictly different from zero were not for some rows, namely `n_tokens_content`, `n_tokens_title`, `num_keywords`, `average_token_length`, `min_negative_polarity`, `max_positive_polarity`, `avg_negative_polarity`, `avg_positive_polarity`. We remove all the $2,581$ rows where these features are zero. The dataset has now $37,063$ observations. We can ask whether the number of shares depends on the values of the categorical variables.

By looking at some basic statistics on the number of shares w.r.t the day of the week (see tabular below), we can see that far more articles are published during the weekdays than during the weekend.

Day of week	count	median	mean	min	max	std.
Monday	6229	1400	3607	1	690400	14785
Tuesday	6896	1300	3187	42	441000	9658
Wednesday	6945	1300	3313	23	843300	149654
Thursday	6790	1400	3161	5	306100	9591
Friday	5325	1500	3238	22	233400	7927
Saturday	2312	2000	4064	43	617900	14516
Sunday	2566	1900	3681	89	83300	6040

However, during the weekend the average and median article, is more shared on social networks. Whether the article is published during the weekend seems to be associated with a variation in the number of shares. We thus might be able to transform `day_of_week` into a binary feature `is_weekend` since not a lot of differences can be seen between the weekdays. The same analysis can be made with the categorical variable `type_of_channel` where we can notice for example a higher popularity for the Social Medial channel as well as a difference in number of article published by categories. In conclusion, the categorical variables seem to account for variation in the number of shares, we might transform them but have to keep them to do our analysis.

Category	count	median	mean	min	max	std.
Other	5218	1900	6041	22	843300	20122
Life style	2033	1700	3697	28	208300	9001
Entert.	6679	1200	2962	47	210300	7865
Business	5958	1400	3108	1	690400	15396
Social Media	2203	2100	3672	5	122800	5629
Tech	6984	1700	3104	64	663600	9220
World	7988	1100	2240	35	284700	5965

iii. Correlation analysis

In the original dataset description, the `timedelta` feature representing the number of days the article spent online, is considered as a non-predictive variable. Looking at the Pearson correlation coefficient with the logarithm of the target variable, a significant ($p-value = 5e-9$) positive correlation (0.03) can be found. Thus a correlation exists but is very small (see Figure 1). We might be able to discard this covariate in the future.

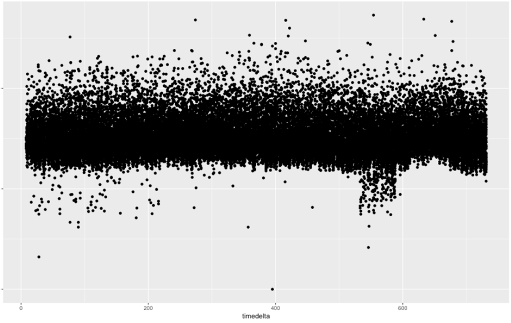

FIGURE 1: TIMEDELTA CORRELATION

To give a better sense upon the correlation between covariates and the influence each covariate can have on the response variable, we plot the correlation matrix. Note that as `day_of_week` and `type_of_channel` are factors to describe categorical variables, it makes no sense to calculate a correlation coefficients for these. Therefore, they will not be shown on the correlation matrix, even though these two are intuitively particularity relevant to determine the number of shares.

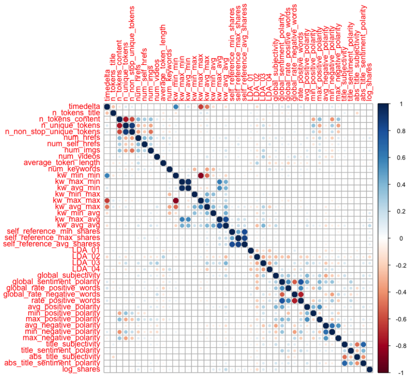

FIGURE 2: CORRELATION MATRIX

At first sight, some clusters of correlation are formed close to the diagonal, showing that some group of covariates are highly correlated. For instance, `n_unique_tokens`, `n_non_stop_words` and `n_non_stop_unique_tokens` present a strong positive correlation. This is understandable as the rate of non-stop unique words in the content can be calculated by the product of each rates of unique and non-stop words. Furthermore, a non-stop word is more likely to be unique than a stop one: this explains the correlation of the two first covariates. Similarly, considering different metrics (min to average, max to average, etc.) of the same data leads to positively correlated covariates on average. This is for instance the case for `kw` and `self_reference` which respectively refer to keyword and the number of reference to other Mashable articles.

NLP metrics are also closely linked. The `global_sentiment_polarity` is positively correlated to `global_rate_positive_words` and `avg_positive_polarity`; and negatively correlated to the equivalent for words. This makes perfect sense given the definition of polarity. Not surprisingly, `title_subjectivity` and `abs_title_sentiment_polarity` are also for instance positively correlated: the subjectivity of the article increases if the absolute polarity (the extreme polarity towards positive or negative) increases. A more striking correlation on these metrics is however the fact that `global_subjectivity` and `average_token_length` are positively correlated, which suggests that the longer are the words on average, the more likely the article is to be subjective. Interestingly, there is not any particular feature that seems correlated to the response variable. The best correlation that exists with shares is `kw_avg_avg`: the average keyword - i.e the keyword associated with articles that were shared a number of times close to the average share per article - influences the number of share. We would however expected that certain features are crucial in the determination of the number of shares and therefore that they would have shown a strong correlation with shares. Namely `timedelta` (the older an article is, the number of shares) or `max_positive_polarity` / `max_negative_polarity` (the more an article conveys an extreme positive or negative idea, the more likely it is to be shared).

Eventually, the correlation matrix shows associativity patterns within the data. For instance, `kw_max_max` is strongly negatively correlated to `kw_min_min`, which makes sense as if the max of shares on the best keywords increases, the number of shares on the worst keywords is more likely to decrease (on average and a fortiori on the min value). `kw_max_max` itself is strongly negatively correlated to `timedelta`, which leads `kw_min_min` to be positively correlated to `timedelta`. We can also find some association pattern on non-correlation: if A is correlated to B and B is not correlated to a large number of covariates, then A is not correlated on these. A similar scheme exists between `abs_title_sentiment_polarity` and `title_subjectivity`.

II. Part II

i. Regression Model

i.1 Design Matrix

Before getting into the model, it is important to have a full rank design matrix for covariates. Through our experiments, we discovered that some columns were collinear, linear combination of other columns or collinear to the intercept term. After analysis, we removed `LDA 00`, `n non stop words` and `rate negative words`.

i.2 Baseline

As a baseline, we are going to run a simple OLS model using all covariates from our dataset. We compute the Root Mean Squared Error (RMSE) on the training set and obtain a really high error of 1.249×10^4. In order to estimate the prediction error, we use 10-folds cross-validation and obtain an RMSE of 1.54×10^4. Both error are high which made us rethink the measurement of success and transform the data.

i.3 Log transformation of the outcome

Instead of calculating the error using RMSE, we have decided to measure the error using Root Mean Squared Logarithmic Error (RMSLE):

$$RMSLE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(Y_i + 1) - \log(\hat{Y_i} + 1))^2}$$

This metric makes more sense to estimate the error as we noticed that some number of shares are extremely large (the $\max$ value is $834,300$), which distorts the prediction for the majority of data. We therefore apply the $\log$ transformation to deal with these outcomes. The plot of residuals below tells us that this transformation is more relevant. After transformation, we obtained an $RMSLE$ on the training set of 0.8647 as well as an estimate of the prediction error using cross-validation of 0.8670. At this stage, we can notice that the estimate of the prediction error is higher than the training error: it takes into account the variance of the data, besides the irreducible error of the population model. Log transforming the data in R implies that the OLS model will minimize the $RMSLE$ instead of the $RMSE$. From now on, we will call $RMSE$ the $RMSLE$.

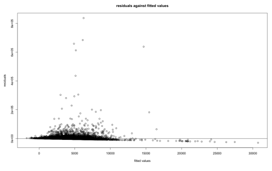

Figure 3: Residuals

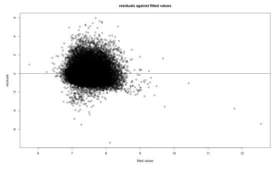

Figure 4: Residuals log

i.4 Log transformation of the features

We can also use the log transformation for positive features if they are more correlated with the outcome variable after this transformation. We compare the correlation between the positive features and the outcome, with the correlation of their log transformations and the outcome. If the log transformation yields a better correlation, we keep the log transformation of the feature over the feature itself.

i.5 Features selection

As a first step to build an efficient model, we know that simpler model using less covariates generalize better and show less sensitivity to the training data. As we are currently using 45 covariates, reducing this number can help us improve our model by reducing its variance, even though we may increase the bias. By looking at the correlation matrix, we can see that some covariates are highly correlated. Thus, we can determine groups of features that are correlated with each other (with a threshold on the absolute correlation of 0.85), and keep from each group the feature most correlated with the outcome variable.

To further select our features, we use the Akaike information criterion (AIC) that provides an estimate of the prediction error from the training set, penalizing the number of covariates often responsible for higher variance. Using this model score for feature selection, we end up with a model with a $RMSE$ of 0.8618 on the training set, and a cross validation error of 0.8641, but now with only 31 features. To double check this selection, we can use Lasso regression in order to zero-out coefficients as λ increases.

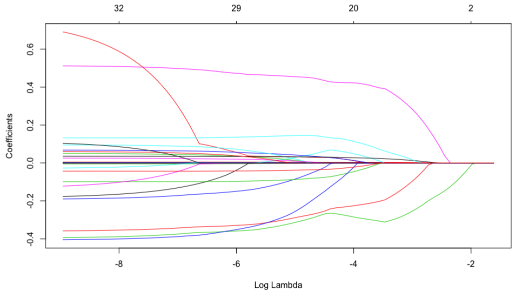

FIGURE 5: LASSO COEFFICIENTS

Eventually, after fitting a random forest model, we are able to plot the feature importances being computed by the model while fitting the data.

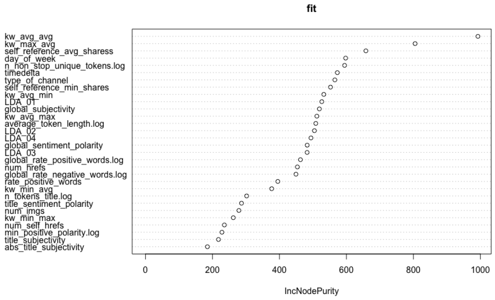

FIGURE 6: RANDOM FOREST IMPORTANT FEATURES

i.6 Interaction with the categorical variables

By adding interaction with the categorical variables we give more freedom to the model by adding the possibility to have different coefficients for each feature for each category. With this technique we will certainly diminish the bias at the cost of more variance. The resulting model has a $RMSE$ of 0.8447 and a cross-validation error of 0.8607. Thus in our case, it is a good sign since we managed to improve the cross validation error which is an estimate of the test error. This tells us that the type of the article as well as the day of publication are associated with a variation of the number of shares.

i.7 Model

In this section we evaluate the test error of our best model against the baseline. Indeed, the cross validation error can be an under estimate of the true test error since we use the same K-folds and made our choices according to the lowest cross validation error obtained. We obtain the following result for the hold out test set.

	Baseline	**Best model**
Training error	0.8647	0.8447
CV error	0.8670	0.8607
Test error	0.8728	0.8696

TABLE 2: ERRORS FOR BASELINE AND BEST MODEL

We have a slight improvement from the baseline model with a lower test error and a lower cross-validation error. Our model is able to generalize a bit better than the baseline model.

ii. Classification Model

In this section, we want a model to classify an article as popular or not. We define a threshold such that if the number of shares is above this threshold, the article is popular, otherwise it is not. The threshold is set as the median of the number of shares in the dataset (i.e. 1400)

ii.1 Evaluation

By choosing the median as the threshold for popularity, we have a balanced dataset. A good choice of loss for the classification task will be the 0-1 loss which is directly linked to accuracy.

ii.2 Baseline

After having converted the numbers of shares in binary values (1 for `shares` $\geq$ 1400, 0 otherwise). We run a basic logistic regression with all the features and obtain a 0-1 loss of 0.3485 on the training set and a cross validation error of 0.3492 which is an estimate of the test error for the baseline

ii.3 Feature selection using AIC model score

For feature selection we use the AIC model score. There is no formal justification for this practice, except that both scores provide a heuristic penalty for excessive model complexity. We end up with 33 selected coefficients. The new model gives us a 0-1 loss of 0.3484 and a cross-validation error of 0.3509. Our estimate of the test error via cross validation is slightly higher than the one of baseline, but we are using less covariates (33 instead of 44) which tells us that we increased our bias to lower the variance of our model.

ii.4 Interaction with the categorical variables

Again, by adding interaction with the categorical variable we give more freedom to the model to try to describe more closely the population model based on the training data. The resulting model has a 0-1 loss of 0.3329 on the training set and a cross-validation error of 0.3469. We managed to improve the cross validation error which is an estimate of the test error. We see that the training error is much lower than the cross-validation error which is a sign of a higher variance than the preceding model.

ii.5 Results on the test set

1. 0-1 loss

   In this section we evaluate the 0-1 loss on the hold out test set and compare the results in the table below for out best model (with feature selection using AIC and using the interactions with the categorical variables).

	Baseline	**Best model**
Training 0-1 loss	0.3485	0.3329
CV 0-1 loss	0.3492	0.3469
Test 0-1 loss	0.3415	0.3426

TABLE 3: 0-1 LOSSES FOR BASELINE AND BEST MODEL

   We did not get a better result than the baseline on the test set, despite having a slightly better result for the cross validation error. The cross validation error might thus have been an under estimate of the true test error since we made our choices according to the lowest cross validation error obtained using the same K-folds.

2. Analysis

- **Confusion matrices**

Actual / Predicted	**Popular**	**Not Popular**
Popular	2313	1368
Not Popular	1164	2568

TABLE 4: CONFUSION MATRIX FOR BASELINE MODEL

Actual / Predicted	**Popular**	**Not Popular**
Popular	2304	1377
Not Popular	1163	2569

TABLE 5: CONFUSION MATRIX FOR BEST MODEL

  We can see that the best model is more robust to false positive but less to false negative than the baseline.

- **ROC curves and AUC**

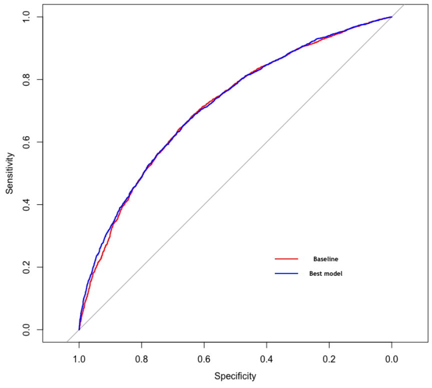

FIGURE 7: ROC CURVE

We compute an area under the curve (AUC) of 0.714 for the best model and of 0.710 for the baseline. This let us think that our best model can be more relevant than the baseline if we were to change the threshold for classification (instead of taking 0.5 as the default, i.e by default classify 1 if $P(\hat{Y} = 1|X) \geq 0.5$), to vary the trade-off between the number of false positive and false negative.

III. Part III

i. *Prediction on the test set*

We apply here our best models for classification and regression on the held out test set and compare the result with our estimate of the test error from cross validation.

i.1 Regression Model

For our best regression model the results are summarized in Table 1.

	Best model
Training error	0.8447
10-fold CV error	0.8607
Test error	0.8696

Table 6: Errors for best model

To estimate the test error we used a 10-fold cross validation. There are two reasons why the cross validation error can be a biased estimate of the true test error. The first one is that we used the same 10 folds to estimate the test error for each of our models and selected the best model according to the lowest 10-fold cross validation error. The estimate of the test error by cross validation for this chosen model is thus an under estimate of the true test error because there is a selection bias (i.e. we selected the model that had the lowest estimate of the test error). The second is that when doing 10-fold cross validation, we successively train our model on 9/10 of the training data and evaluate on the train data left, and the estimate is the average of the 10 errors resulting from this procedure. But when training our final model, we train it on the entire training data, and then we have the test error by evaluating this model on the test set. Thus this cross validation estimate can be an over estimate of the test error because we do not train on the entire set of training data. But the dataset is quite large ($\approx$ 37000), so we expect this second effect to be minimal in comparison to the first one.
Thus as we can see in the results, the cross validation error is an under estimate of the test error. And as expected, the training error is lower than both the cross validation error and test error as the model is built so that it minimizes the training error.

i.2 Classification Model

In this section we evaluate our best classification model (with feature selection using AIC and using the interactions with the categorical variables) by computing the 0-1 loss on the hold out test set and we compare it to our estimate of the test error using cross validation in Table 7.

	Best model
Training 0-1 loss	0.3329
10-fold CV 0-1 loss	0.3469
Test 0-1 loss	0.3426

Table 7: 0-1 losses for best model

The analysis on the potential bias of the estimate of the test error via cross validation made in the previous part still holds. However here, the cross validation error seems to be an over estimate of the test error. It could be for the second reason mentioned above (i.e. we have an over estimate of the test error because we did not train the cross validated models on the whole training set) or just because the test set was more favorable for the model in for this particular split train/test. It is hard to make a strong statement here as the errors are still very close.
As before, the error on the training set is lower as we built the model with minimizing the error on this particular subset of the data as objective.

ii. *Inference*

For this part and for simplicity of analysis we use the linear regression model from part II resulting from the elimination of very correlated features, log-transformation of certain features and then feature selection using AIC. This models relies on 32 covariates for which we will do an inference analysis.

ii.1 Statistical significance of coefficients

Under a lot of assumptions on the population model, notably that it is linear, includes all the right covariates and has independent and identically distributed errors with mean zero and a constant variance we can interpret statistical significance as follows:

- a coefficient is said to be significant when the Student t-test is statistically significant for this coefficient with the null hypothesis being that this coefficient is equal to zero. In other words, if the coefficient is significant, it means that if the true coefficient in the population model were zero, then it is very unlikely to see this estimate of the coefficient (i.e. we have reasonable evidence that this coefficient is not zero).

- however a coefficient that is not significant does not mean that it is not important to the model, but rather that we do not have enough evidence to know that this coefficient is not zero.

We report below the coefficients[1] with lowest p-value from our best model fitted on the training data.

[1] Note that *kw_min_min:* is actually the interaction covariate *kw_min_min:type_of_channel5*, that *type_of_channel6:* is *type_of_channel6:average_token_length.log* and that *type_of_channel3:* is *type_of_channel3:num_keywords.log*

Covariates	Estimate	$\hat{SE}$	t value	Pr(>|t|)
kw_avg_avg	2.8e-04	3.4e-05	8.4	6.4e-17
(Intercept)	5.4	7.5e-01	7.16	8.5e-13
kw_max_avg	-3.4e-05	5.5e-06	-6.2	4.9e-10
kw_min_min:	1.9e-03	3.3e-04	5.8	8.7e-09
type_of_channel3:	4.6e-01	8.6e-02	5.3	1.2e-07
kw_avg_max	-1.1e-06	2.0e-07	-5.2	1.8e-07
type_of_channel6:	-1.6	3.1e-01	-5.2	2.1e-07
self_ref_min_shares	7.6e-06	1.5e-06	5.0	6.8e-07

When running this analysis, we found that 13.4% of the covariates are below the 99% significance level and 21.3% of the covariates are below a 95% significance level. These results are to be analyzed carefully as they are under the heavy assumptions on the population model mentioned above, but gives us a good idea of the importance of each coefficient and how sure we are about it.

Interestingly, the statistically most significant coefficient is `kw_avg_avg` which is consistent with our AIC and Random Forest analysis and confirmed our intuition stated in Part 1. Namely, the number of shares is mainly driven by the average number of shares counts among the articles that share average keywords article category. This influence is easily interpretable as the keywords define the topic of the article and directly influence the number of shares. The second most important feature is the intercept term, which represents the number of shares an article would get provided all other covariates were set to zero. Under that condition, this would be 221 shares, given that we are predicting the $\log(shares)$. This figure is however, quite surprising as the intercept term is expected to evaluate the mean of the outcome feature, which is 1758 shares on the training data. This covariate is indeed less interpretable given the complex nature of some covariates. One point to mention here is also the fact that interaction covariates appear at level of p-values really low, supporting the proposal of including them in the model.

In a second analysis, we fit the chosen model on the test data[2] and plot the coefficients with lowest p-value:

Covariates	Estimate	$\hat{SE}$	t value	Pr(>|t|)
(Intercept)	8.3	1.6	5.2	2.1e-07
type_of_channel4:	-1.6	4.1e-01	-3.8	1.3e-04
type_of_channel5:	9.7e-01	2.9e-01	3.3	8.6e-04
kw_avg_max:	1.5e-06	4.5e-07	3.3	9.0e-04
type_of_channel3:	-9.5e-01	3.0e-01	-3.2	1.3e-03
rate_pos_words:	5.0	1.6	3.2	1.4e-03
kw_avg_avg	2.5e-04	8.3e-05	2.9	3.3e-03
type_of_channel6:	-1.9	6.5e-01	-2.9	3.3e-03

We found that only 5.0% of the covariates are below the 99% significance level and 12.4% of the covariates are below a 95% significance level. Besides the p-values and t Student statistic, the estimates changed for the most significant coefficients we had on the training data. More generally for a given covariate, the coefficient associated with it will be less significant when fitting the model on the test set. This can be explained by the fact that we have much less data in the test set than in the training set, we have thus less data ($n = 7413$ for the test data and $n = 29650$ for the train data) to build our estimates resulting in less evidence to know whether a coefficient is zero. More precisely, we can actually quantify the ratio between the two t-statistics. We know that the t-statistic expression is:

$$t = \frac{\hat{Y} - \theta_0}{\hat{SE}} = \sqrt{n}\frac{\hat{Y} - \theta_0}{\hat{\sigma}}$$

Having 4 times more data in the training set, we can thus expect the t-statistics to be $\sqrt{4} = 2$ times larger for the coefficients of the model fitted on the training data. When looking at the median of the ratio of the t-statistics between the two models, across the coefficients we find 1.74 which is close to 2. This leads the test to have lower power.
Another reason for which the significance of the coefficients could be different (not only by a factor of 2) between the test and training sets is that we could have been victim of post selection inference, this point will be developped further in the section *iii.*.

As our chosen model does not include all covariates we had available, we run an inference analysis on the training data with a model including all covariates. We report below the coefficients[3] with lowest p-value.

Covariates	Estimate	$\hat{SE}$	t value	Pr(>|t|)
kw_avg_avg	2.7e-04	3.4e-05	8.0	1.6e-15
(Intercept)	5.4	7.8e-01	6.9	4.84e-12
kw_max_avg	-3.2e-05	5.5e-06	-5.9	3.5e-09
kw_min_min:	1.9e-03	3.4e-04	5.6	2.5e-08
type_of_channel6:	-1.7	3.2e-01	-5.3	1.0e-07
type_of_channel3:	4.6e-01	8.7e-02	5.3	1.2e-07
kw_avg_max	-1.0e-06	2.1e-07	-5.0	4.8e-07
self_ref_min_shar	7.2e-06	1.5e-06	4.7	2.5e-06

We found that 12.2% of the covariates are below the 99% significance level and 21.1% of the covariates below a 95% significance level. Interestingly, the most significant covariates of the model are similar with the case where we only selected covariates from our final model. For these covariates, we can notice that the coefficient estimates are in line with those from the final model as well as the standard error, which supports our covariates selection. Comparing both results, we can also view that the statistical significance is higher in our final model than in the model using all covariates. This makes particular sense,

[2] Note that *type_of_channel4:* is actually the interaction covariate *type_of_channel4:n_non_stop_unique_tokens.log*, that *type_of_channel5:* is *type_of_channel5:global_rate_negative_words.log*, that *kw_avg_max:* is *kw_avg_max:day_of_week3*, that *type_of_channel3:* is *type_of_channel3:average_token_length.log*, that *rate_positive_words:* is *rate_positive_words:type_of_channel5* and that *type_of_channel6:* is *type_of_channel6:average_token_length.log*

[3] Note that *kw_min_min:* is actually the interaction covariate *kw_min_min:type_of_channel5*, that *type_of_channel6:* is *type_of_channel6:average_token_length.log* and that *type_of_channel3:* is *type_of_channel3:num_keywords.log*

as in the case of having all covariates, the individual significance of one single coefficient is more likely to decrease (and hence the p-value to increase) due to collinearity between certain features. If some covariates are collinear, their individual significance will be lower.

ii.2 Confidence interval using the bootstrap

We can use the bootstrap (case resampling) to have an estimate of the sampling distribution of the estimates of the coefficients for the linear regression. In particular, this will allow us to compute confidence intervals for our regression coefficients.

We can draw at random $B = 1000$ new samples of the size of the training set from the training set with replacement and fit a linear regression model on each of this bootstrap sample. After doing so, we can have the bootstrap distribution of each coefficient, estimate for example the standard error of the sampling distribution and compare it with the ones given by R.

Using a normal interval method, we report the following results obtained for few coefficients:

Covariates	Coefficients	$\hat{SE}$	CI at 95%
(Intercept)	5.4	9.5e-01	[3.5,7.2]
timedelta	9.6e-05	1.3e-04	[-1.6e-04,3.6e-04]
num_hrefs	1.8e-03	2.2e-03	[-2.5e-03,6.0e-03]
num_self_hrefs	-5.7e-03	6.9e-03	[-1.9e-02,7.8e-03]
num_imgs	1.2e-03	3.2e-03	[-5.0e-03,7.3e-03]
kw_min_min	-1.3e-03	4.1e-04	[-2.2e-03,-5.4e-04]
kw_avg_min	1.9e-06	4.6e-05	[-8.9e-05,9.3e-05]
kw_min_max	2.7e-07	4.0e-07	[-5.2e-07,1.1e-06]

As a comparison, see below the corresponding confidence intervals and $\hat{SE}$ returned by R.

Covariates	Coefficients	$\hat{SE}$	CI at 95%
(Intercept)	5.4	7.5e-01	[3.9,6.8]
timedelta	9.6e-05	1.5e-04	[-1.3e-04,3.2e-04]
num_hrefs	1.8e-03	1.6e-03	[-1.4e-03,4.9e-03]
num_self_hrefs	-5.7e-03	5.5e-03	[-1.6e-02,5.1e-03]
num_imgs	1.2e-03	2.4e-03	[-3.6e-03,5.9e-03]
kw_min_min	-1.3e-03	3.4e-04	[-2.0e-03,-6.7e-04]
kw_avg_min	1.9e-06	3.3e-05	[-6.2e-05,6.6e-05]
kw_min_max	2.7e-07	3.6e-07	[-4.3e-07,9.7e-07]

We can notice that the standard error given by the bootstrap are on average higher than the one given by R. To visualize this, we can look at the ratios between the standard errors of the coefficient in R and the standard error of the bootstrap distributions to see how close they are:

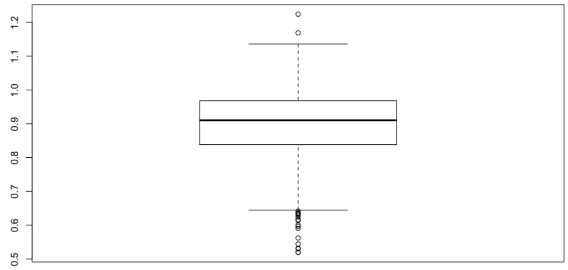

FIGURE 8: RATIO BETWEEN $\hat{SE}$ OF COEFFICIENTS FROM R AND $\hat{SE}$ OF THE BOOTSTRAP DISTRIBUTION FOR COEFFICIENTS

Differences might occur when the assumption on the normality of the distribution of the estimate of the linear regression coefficients does not hold. In this case, the estimate of the standard error from the bootstrap distribution would be more accurate. More generally, if the linear normal model assumptions are violated then the bootstrap method will be more robust to modeling errors.

ii.3 Potential problems of the analysis

On the coefficients significance analysis, we noticed that the p-values of coefficients estimates were higher against test set than training set. Besides having a lower power hypothesis test, this can also be explained by post-selection inference. As we selected our covariates and best model on the training set, we optimized their significance and favorably biased our selection of p-values. Thus, on the test set, the significance will change for many covariates, in reasonably the p-value increases. Therefore, we need to remain wary while analyzing statistical significance on our final model.

Another issue that can be addressed in the analysis is the interpretation of coefficients from a causality perspective. For the causation analysis to be true, we need that all covariates are independent (orthogonal ideally) of other variables so that we can be in situation of exogeneity. This potential problem justify why we enforced a correlation analysis on the covariates in the preprocessing part, so that the design matrix is invertible.

Running many simultaneous hypothesis tests on coefficients is also an inherent risk of our analysis. Among the significant covariates we selected, there is indeed the risk that we select non-significant ones. With a 5% cut-off, we are indeed willing to accept a 5% rate of false positives, i.e coefficients that look large due to random chance, despite the fact that there is really no underlying relationship. Thus, among the 86 covariates we assessed as significant, there might be 0.05×403 = 20 covariates which appeared significant due to a random chance (403 covariates, including interaction covariates were selected in our final model).

In order to ensure that the probability of declaring even one false positive is no more than 5%, we can apply multiple testing corrections such as the Bonferroni correction. Using this method, we declare as significant any hypothesis where the p-value is lower than $0.05/p$ where p is the number of hypothesis tests being carried out. In our analysis, $p = 403$, therefore the Bonferroni correction enforces the α to be 1.2×10^{-4}. We would therefore only retain 16 significant coefficients.

ii.4 Causality

On the final model, the most significant feature is `kw_avg_avg`, associated with an estimate of $2.8e - 04$. In terms of causal interpretation, this means that a one unit change in X_i will result in a $2.8e - 04$ change in the outcome. This links is however an association and not pure causation, as it is clear that both data are correlated. It is however less obvious that X_i is orthogonal to all other covariates or that X_i does not convey omitted variable bias, and in other words, that X_i leads to a direct causal effect on Y. Under that uncertainty, it is tricky to carry out proper causal inference reasoning on our model.

IV. Discussion

i. Practical use of the model

This model could be used to predict the popularity of an article published online based on some of its key features. It should not be used for inference but could be use as a first step to see what features of an article could be potential candidates as features influencing the number of shares it will have. Then to confirm the actual effect that these potential features have on the popularity, we should run a randomized controlled trial. Then if we can actually state that a feature has an impact on the number of shares of an article, we can use it to make decision on the design of an article to try to increase its popularity.

It would be hasty to infer causality from our prediction model, as there could be confounding variable influencing both a covariate and the outcome causing a spurious correlation useful for prediction but not for inference.

More practically, this analysis can be used by writers and editors to forecast whether their article will be successful and go viral. On the other hand, social and information networks such as Facebook or LinkedIn may use this model to highlight articles in the customer's newsfeed.

ii. Hold up over time

The popularity of articles published online depends of a lot of factors, and since the online world is always changing, we expect that the models will need to be refitted quite often. For example, we could imagine that the apparition or disappearance of a new distribution channel (e.g. a new social network platform) would impact drastically the number of shares of an article and thus the model would become obsolete. A new trend could also appear with a new infatuation from people about a topic (e.g. technology). This would completely change the evolution of the number of shares with respect to the considered covariates, making the refitting necessary. Inherently, our model depends our a specific time frame considered, and therefore on a particular context.

iii. Points to be aware of

In our data analysis, we would reasonably report on the following concerns and warnings:

- First, we would communicate on the data preprocessing step and data transformation, namely explain the features retained for the final model as well as their transformations (outcome, some of the covariates)
- Second, it is important to point out that our model does not perform really better than baseline OLS, even after feature engineering and inference analysis. One of the reason for this is that the prediction task is particularly complex as it really depends on factors closely linked to human emotions or subjective perception of the article, which can hardly be described in numbers. In order to tackle this human aspect in the way an article goes viral, we should wait for the increase of the accuracy and plurality of NLP metrics.
- Eventually, it would be necessary to notice that our final model might suffer from post-selection inference as we favorably biased the selection of some covariates. Moreover, the issue of multiple hypothesis testing makes sense, given the important number of covariates.

iv. Data collection process

Even though the data collection process seemed reliable to us, we would include more covariates to better understand the nature of the population model. Having the raw text of every articles for instance could let us apply NLP algorithms in order to build our own features of the text analysis. It could have also been interesting to have covariates on the author of every article (popularity, influence, number of articles written). Similarly, it could be interesting to know about the readership of every articles: does an article goes more likely viral when addressed (and read) to a certain category of readers?

v. What would we do differently?

If we were to tackle the same dataset again, we would probably perform unsupervised methods such as PCA to perform better feature selection. We would definitely also try other regression and classification techniques such as k-nearest neighbors, gradient boosting or support vector machines to see whether the estimate of prediction error can be decreased.