Machine Learning and Data Mining project: When the tweet goes viral..

Mattia Pividori[1] and Lorenzo Cabriel[2]

[1,2] problem statement, solution design, solution development, data gathering, writing

Course of AA 2021-2022 - Data Science and Scientific Computing

1 Introduction and problem statement

As a consulting firm, we have been hired by a renowned German international discount retailer chain to back their upcoming 2022 international advertising strategy aimed at launching a brand new line of fresh food (i.e. grocery items) via their official Twitter account. Company's plans are to publish one tweet per week starting in April till year's end, in English, and focusing on one/more of its existing or newly available food products, present across all its stores. To this end, we have been asked in first place to perform a study meant to understand the main drivers that make certain tweets about food 'viral' [14, 12] (i.e. spreading widely and quickly); secondly, the insights gained in the previous step will be used to implement a prediction tool to check each new tweet related to the ongoing campaign before its publication, in order to try maximizing the spreading/impact on the general public, so to hopefully help attaining an increase in our client's proceeds.
To align our interests with the goals of the company, our compensation arrangement is made by a variable part based on the count of how many tweets -among those related to the ongoing stategy- will become viral. As per our binding contract, the virality will be measured in terms of the number of single retweets received by each original tweet (i.e. not retweeted). Moreover, given that the retailer chain's most popular tweet to date counts 350 retweets and there are no hard rules defining what a 'viral' tweet is, for us here will specifically mean an original English-language tweet retweeted at least 1000 times, as we agreed on this target.

WARN1: Since often 'viral' brings also a time connotation (i.e. a quick spread), we warn the reader that we are not interested in this time dimension of the spread as, following our arrangement, the aim is to maximize our profit bringing ideally all the tweets related to the campaign within the 'popularity target', regardless of when. We will then use the terms 'viral' and 'popular' as synonyms in this note.
WARN2: We will model the virality of a tweet as a function of specific tweet features [4, 2], excluding all features solely tied to tweet's author which, in this specific case, are outside our control (e.g. number of account followers).

2 Dataset extraction and feature engineering

We created our reference dataset by scraping the stream of tweets from Twitter's inception (2006) to date, using Python's twint library [3] and applying 3 filters. The first controls for the presence of at least one food-related keyword in some entity fields (text, expanded/display urls for links/media, text for hashtags and screen name mentions); we scraped an English culinary dictionary index [13] to this end. Second filter is on the language of the tweet, which must be English. The last is on the minimum number of retweets: in fact, to form a balanced sample we want to bi-partition the space of our tweets into 'viral' and 'non-viral' elements with equal weight. So, as non-viral tweets outnumber viral ones, after a first download of the popular ones ($\geq$1000 retweets), we counted them and proceeded with a second run without min-retweets limit, adding instead a max-download limit for the number of tweets, set equal to the count of viral tweets previously collected. Eventually, we removed duplicates (by tweet id).

The final dataset counts 258730 tweets, balanced between viral and non-viral (50%-50%). Following an exploratory data analysis, we dropped some variables loosely related to the purpose of our study (e.g. *user_name, place, thumbnail, etc..*), and handled missing data via imputation of zeros, as context suggested us that all NA values were attributable to empty entity-list variables (NaN$\rightarrow$0).
The next step included text mining as we tried to extract sentiments from each tweet's text. Preliminary phase took out elements like urls/hashtags/mentions/links/photos from text and assigned them to separate columns of our dataset. Then, a text pre-processing step -via R's *tm* package- encompassed lowercasing, emoji/stop words/extra-spaces/numbers removal, the removal of anything other than English alphabet characters and stemming: such that we aimed at maximizing the amount of information contained in each short text, while reducing noise and computational complexity. By means of R's *syuzhet* package [8], we performed a first text parsing to vectorize the sentences of each tweet (*get_sentences*), followed by the the assessment of sentiments in each sentence (*get_nrc _sentiment*). After an additional punctuation-removal step, we also applied the *get_tokens* function, to tokenize text by words, as to obtain a more granular corpus with the aim of a more precise sentiment extraction. The result are 2 data frames in which each row represents a tweet, and each column represents one of the eight emotions (anger, fear, anticipation, trust, surprise, sadness, joy, disgust) and two sentiments (negative and positive) composing the categories of NRC classification [10]: each cell thus contains a score representing the count of how many instances of each NRC class occurred among each tweet's words/sentences.
Feature engineering followed, including transformations like: response variable from numerical to binary ($retweets \rightarrow viral$), text into characters count ($tweet \rightarrow tweet\_length$), publication date into its weekday ($date \rightarrow weekday$), publication time into its hour ($time \rightarrow hour$), counting of the occurrences of some items ($mentions$, $urls$, $photos$, $hashtags \rightarrow$ [...]$\_count$), "dummification" of two-class categorical variables ($quote\_url, video$).

The predictors used are thus the following: $weekday$, $hour$, $tweet\_length$, $mentions\_count$, $urls\_count$, $photos\_count$, $hashtags\_count$, $quote\_url$, $video$, $anger$, $anticipation$, $disgust$, $fear$, joy, $sadness$, $surprise$, $trust$, $negative$, $positive$.
We then split the dataset into random train/test subsets(0.7/0.3): on both chunks we applied standardization feature-wise for numerical variables to ensure attributes with larger values not to over-influence models, plus a one hot encoding for categorical variables.

3 Solution design, development and assessment

This is a supervised learning problem which falls into the Binary Classification (BC) domain ($y \in \{0, 1\}$, where 1="viral" and 0="not viral").
Our solution encompasses two layers: using 5-fold cross-validation (CV) on the train set, we first assessed 6 different algorithms commonly employed for BC such as Random Forest (RF) [1], Logistic Regression (LR) [11], Decision Trees (DT) [7], Support-Vector Machine (SVM) [15], Naive Bayes (NB) [4] and k-Nearest Neighbors (kNN) [6], comparing their performances when their hyper-parameters were set to default values. Second, we selected the techniques which led to the most promising results in the previous step, running in sequence for each a random grid search (RGS) and a grid search (GS) on the train set to further improve the performance via hyper-parameter tuning [9]. In both latter methods parameters are optimized by cross-validated search over (hyper)parameter settings, difference is that in RGS we define a range of values from which the code can randomly pick and choose till finding an optimal set, while GS performs an exhaustive search over the specified parameter values for the estimator. We used RGS output as input for GS to only perform a fine methodological search once having already a good starting point, in order to save on processing power needed. All the above was implemented using Python's $scikit-learn$ library.
To evaluate the results of our experiments we need at first to choose a metric and a baseline reference rate. We chose as our naive classifier model and benchmark accuracy the Zero Rule classifier ($ZeroR$) which simply predicts the majority category (class) from the training set: the baseline accuracy rate was thus set to 0.50 as we have two perfectly balanced classes. All classifiers -among the 6 tested- scoring above the baseline threshold qualified for the second stage, where we ranked the performances of all classifiers passing the previous filter. Given the balanced nature of the problem and the not excessive cost of a specific error type, our primary measure was still the accuracy score but, to perform a finer discrimination in the ranking process (e.g. when two classifiers yielded similar accuracy), we also checked ROC-AUC, F1-score and Precision-Recall(PR)-AUC [5].
Finally, for the models surviving the second stage (3 out of the starting 6), after tuning their hyper-parameters via RGS and GS cross-validation, we performed a terminal evaluation on the test set aimed at confirming/challenging the previous ranking, so to end the selection picking the one model consistently best in/out of sample.

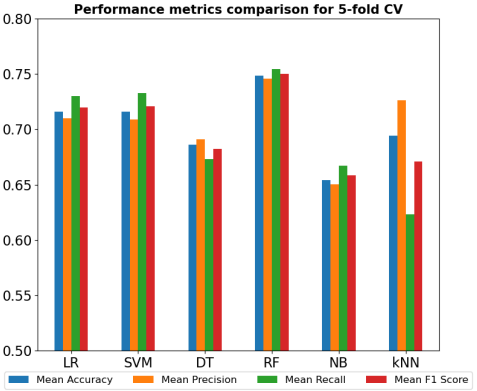

Figure 1: Cross validation mean scores

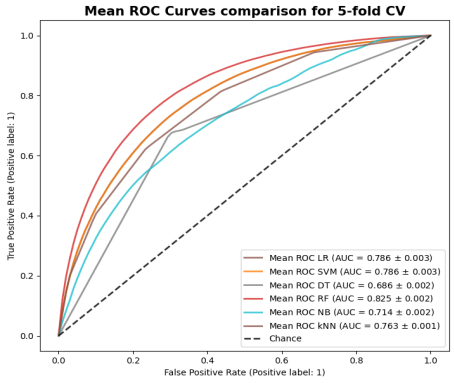

Figure 2: Mean AUC-ROC on validation set

4 Results and conclusions

Mean results for CV scores achieved by the 6 classifiers are presented in Fig.1. We notice that (i) all yield a mean accuracy above the *ZeroR* threshold thus passing the first screen and (ii) the best scores are found for RF (i.e. 0.749 +/- 0.001 accuracy, 0.825 +/- 0.002 ROC-AUC, 0.751 +/- 0.001 F1-score, 0.814 +/- 0.001 PR-AUC), followed by SVM and LR very close to each other (about 0.716 mean accuracy both). In general, being samples quite balanced across target classes, we find scores for accuracy, precision, recall and F1-score quite similar. In Fig.2 we can see the same pattern repeating for the 6 average ROC-AUCs: they validate accuracy's ranking, with the plus of being threshold and scale invariant.
We thus discarded DT, NB, kNN and proceeded further performing RGS and GS on the three selected models. Results on the test sample confirmed overall dominance for RF algorithm over SVM and LR, with an accuracy for these tuned models on average about 0.01-0.02 higher than default models: in Tab.1 we present a classification report for the final RF model, highlighting its overall accuracy of 0.767 (ROC-AUC 0.85) on test set, quite consistent with previous CV estimates.
To draw more insights from our model of choice, we portrait in Fig.3 the variable importance (%) for the top ten most relevant features of the RF, so how much -on average- each variable decreases the impurity of the split. This highlights the fact that for reliable predictions of viral tweets, it is important to consider a combination of features covering both structural, content-based elements like text length (17% importance), but also other aspects such as sentimental ones (positive/negative about 2% both). Moreover, it is curious to observe the relevance of *video* and *Saturday* dummies, which may help producing actionable insights for our client. Variables omitted in plot such as all hours' dummies seem not particularly significant, so future refinements of current study could safely drop them while evaluating the inclusion of other features. Examples of such may be the relevant information contained in images, videos and links which we have not processed here, due to computational reasons. To improve predictions' quality we would also suggest controlling for collinearity among features (e.g. 'positive' sentiment against trust/joy emotions), as well as using a larger dataset -possibly unbalanced- to allow examining other relevant facets of problem at hand.

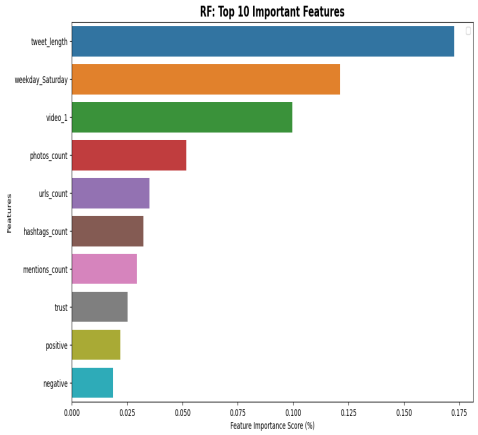

Figure 3: Tuned RF variable importance

	precision	recall	f1-score	support
0	0.77	0.75	0.76	38784
1	0.76	0.78	0.77	38835
accuracy			0.77	77619
macro avg	0.77	0.77	0.77	77619
weighted avg	0.77	0.77	0.77	77619

Table 1: Tuned RF results on test set

References

[1] Oliveira N. Costa J. Silva C. and Ribeiro B. Retweet predictive model for predicting the popularity of tweets. *International Conference on Soft Computing and Pattern Recognition*, pages 185–193, 2018, December.

[2] Fiok K. Karwowski W. Gutierrez E. and Ahram T. Predicting the volume of response to tweets posted by a single twitter account. *Symmetry*, 12(6):1054, 2020, June.

[3] Poldi F. and Zacharias C. Twint - twitter intelligence tool. `https://github.com/twintproject/twint/wiki`, 2019. Accessed: 2022-01-10.

[4] Jenders M. Kasneci G. and Naumann F. Analyzing and predicting viral tweets. *Proceedings of the 22nd international conference on world wide web*, pages 657–664, 2013, May.

[5] Davis J. and Goadrich M. The relationship between precision-recall and roc curves. *Proceedings of the 23rd international conference on Machine learning*, pages 233–240, 2006, June.

[6] Gao S. Ma J. and Chen Z. Popularity prediction in microblogging network. *Proceedings of the 16th Asia-Pacific Web Conference*, pages 379–390, 2014, September.

[7] Cheng J. Adamic L. Dow P.A. Kleinberg J.M. and Leskovec J. Can cascades be predicted? *Proceedings of the 23rd international conference on World wide web*, pages 925–936, 2014, April.

[8] Matthew L. Jockers. Introduction to the syuzhet package. `https://cran.r-project.org/web/packages/syuzhet/vignettes/syuzhet-vignette.html`, 2020. Accessed: 2022-01-21.

[9] William Koehrsen. Intro to model tuning: Grid and random search. `https://www.kaggle.com/willkoehrsen/intro-to-model-tuning-grid-and-random-search`, 2018. Accessed: 2022-01-22.

[10] Saif M. Mohammad. Nrc word-emotion association lexicon. `http://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm`, 2010. Accessed: 2022-01-20.

[11] Hong L. Dan O. and Davison B. D. Predicting popular messages in twitter. *Proceedings of the 20th international conference companion on World wide web*, pages 57–58, 2011, March.

[12] Simon Rogers. What fuels a tweet's engagement? `https://blog.twitter.com/official/en_us/a/2014/what-fuels-a-tweets-engagement.html`, 2014. Accessed: 2022-01-18.

[13] Linda Stradley. Linda's culinary dictionary. `https://whatscookingamerica.net/glossary`, 1997. Accessed: 2022-01-10.

[14] Zeng B. Feng R. Hou Y. and Mahmood Z. Predicting popularity. `http://belindazeng.github.io/goingviral`, 2015. Accessed: 2022-01-15.

[15] Zhang Y. Xu Z. and Yang Q. Predicting popularity of messages in twitter using a feature-weighted model. *International Journal of Advanced Intelligence*, 20, 2018.