Introduction to machine learning project: Twitter food popularity

Giovanni Zanin[1] and Ludovica Caiola[2]

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

Course of AA 2022-2023

1 Problem statement

The aim of the project is to build a ML system that will predict the popularity of a tweet regarding food, evaluating the popularity in relationship to the sum of the tweet's interactions (composed by: Likes, Retweets and Replies). If the interactions are greater than 40, a tweet will be considered popular and it will be classified as Positive, otherwise not-popular and it will be classified as Negative. This threshold was chosen because it represented a reasonable percentage of positive tweets (about 2%). For every tweet, the chosen features regarded both information about the tweet itself and about its author.

2 Data

2.1 Data collection

The dataset has been created by taking 10 tweets a day from the year 2022, for a total of 3650 tweets. In order to obtain tweets regarding 'food content', it was created a dictionary of 12 words in this way:

- the texts of 4000 general tweets with hashtag "#twitter" were collected;
- the texts of 4000 food related tweets containing the word "food" were collected;
- all tweets' texts were processed (eliminating punctuation, stop-words, linked URLs and numbers) to create two vocabularies of words: one for general tweets and one for food related tweets;
- after considering only words that appeared many times in the food vocabulary (at least 38) and removing food-unrelated words that appeared more

than few times in the general vocabulary (at least 4), 12 words regarding food were selected: "food", "eat", "foodfess", "foodporn", "delivery", "cooking", "restaurant", "eating", "hungry", "rice", "kitchen" and "dinner".

At least one of the obtained words had to be contained in tweets' text to collect them.

2.2 Pre-processing

Each observation in the dataset represents a tweet. The initial columns were: *inReply* (whether the tweet is a reply to another tweet or not), *cashtags* (number of cashtags), *mediaCount* (number of media contents in the tweet), *mentionsCount* (number of mentions in the tweet), *label* (e.g.: "official", "automated"...), *accountCreation* (days since user's account was created), *len* (length of the tweet), *userFollowersCount* (number of followers of the user), *verified* (whether the user is verified or not), *usersNumTweets* (number of tweets already posted by the user), *weekday*, *hashtags* (number of hashtags), *replies*, *retweets*, *likes*, *language*. Pre-processing operations included:

- the substitution of "replies", "retweets" and "likes" columns with a boolean column (representing the response of the ML problem), indicating: **True**, if the tweet had more than 40 total interactions; **False**, otherwise
- to permit the use of `sklearn` libraries, one-hot encoder was adopted to create multiple dummy variables from two multi-categorical variables ("label" and "language").

The food tweets selected were randomly chosen independently from their popularity, in order to collect as realistic as possible data. This led to a very unbalanced dataset as positive responses were only the 2%.

3 Proposed solution

To find an effective learning technique many algorithms were evaluated on the dataset:

- **Decision Tree classifier** with default parameters values (minimum number of observations to branch a node: n_{min}=2);
- **Random Forest classifier** with number of trees $n_{tree} = 500$ and default number of variables to retain at every learning: $n_{vars} = \lceil \sqrt{p} \rceil$ (where p is the number of independent variables in the dataset i.e. 38);
- **Decision Tree classifier** with the tuning of the hyperparameter n_{min}: the results obtained for each possible value of this parameter given in the grid were compared with 10 cross-fold validation and the score function $f1$, that considers both precision and recall($f1 = 2 \times \frac{pr \times rec}{pr + rec}$);

- **Random Forest classifier** with n_{tree} =500 and the tuning of the hyperparameter n_{vars}, using 10-CV and $f1$ for comparisons;
- **Tree Bagging classifier** with n_{tree} =500;
- **K-Nearest Neighbors classifier** with $k = 5$ neighbors and d=*p-norm*;
- **Dummy classifier** used as a baseline of comparisons.

4 Experimental evaluation

4.1 Procedure

Every learning technique in this project was evaluated using 10-CV and calculating the average value of some scoring functions over the 10 test folds. These functions included Precision (portion of true positives on all positive predictions), Recall (portion of positives correctly predicted), Accuracy, Weighted accuracy (preferred to simple Accuracy in an unbalanced dataset like this) and Area under the curve (area under the ROC).

4.2 Results and discussion

The table below shows the result for each classifier, measuring the effectiveness in terms of "Precision", "Recall", "Weighted accuracy" and "Area Under the Curve". In the results obtained with the complete dataset decision tree resulted to be the one with the best performances. Then, concerning the improvement of the solutions, it was introduced an algorithm of feature ablation which permitted to identify the importance of each variable, evaluating (with Decision Tree classifier, 10-CV and $f1$) the variations of performance removing each feature. At this point the same classifiers were applied to the new dataset without the two less important variables. Despite some improvements in other algorithms (the most relevant variations are highlighted in the table), Decision Tree classifier remains the best solution.

	on complete dataset				**post feature selection**			
classifier	Prec.	Rec.	wAcc	AUC	Prec.	Rec.	wAcc	AUC
DT	0.11	0.10	0.54	0.54	⇧0.18	0.11	0.55	0.55
RF: $n_{tree} = 500$[1]	0.10	0.01	0.51	0.51	0.10	0.01	0.51	0.51
DT: $n_{min} = 36$[2]	0.36	0.12	0.55	0.55	⇩0.31	0.12	0.56	0.56
RF: $n_{tree} = 500$[1], $n_{vars} = 16$[2]	0.15	0.05	0.53	0.53	⇧0.23	⇧0.08	0.54	0.54
Tree Bagging	0.27	0.08	0.54	0.53	⇩0.23	0.08	0.54	0.54
kNN	0.27	0.06	0.53	0.53	0.27	0.06	0.53	0.53
Dummy	0.00	0.00	0.50	0.50	0.00	0.00	0.50	0.50

[1]: parameter manually specified; [2]: obtained with hyperparameter tuning; if not specified, parameters are kept with default values

The technique with the best results seems to be the Decision Tree classifier with $n_{min} = 36$ applied on the entire dataset. In fact, by comparing the two decision trees used for each case, the one with $n_{min} = 36$ has a less deep tree than the one with $n_{min} = 2$ and the overfitting is minimized.
This technique allows to predict, besides just the class, the probability of a tweet belonging to each class. In binary classification, predicting which class an instance belongs to consists in verifying if its probability of being positive exceeds or not a certain threshold τ. In the experiments above τ is implicitly set to 0.5, but in some cases there could be other convenient values. For example: if the cost of a false negative prediction is larger than the one of false positive (i.e. recall is more important than precision), the threshold τ should be lowered in order to predict as positive also the instances having a predicted probability of being positive lower than 0.5 (like 0.4, 0.3 and so on). This improves the true positive rate (i.e. the recall) at the cost of some values of true negative rate. The graphic below shows how recall has better results with low values of τ.

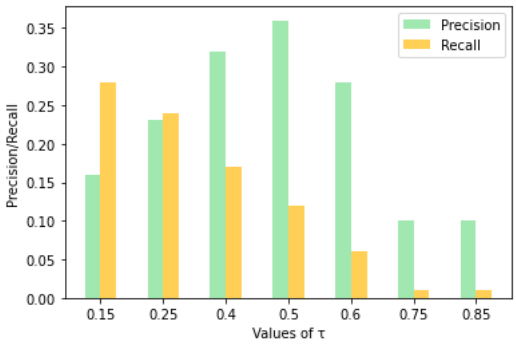

Figure 1: Bar plot that shows precision and recall values by varying τ for decision tree classifier tested on the complete dataset

The results obtained are partially satisfying as the values of precision and recall obtained are not so high. Better performances could possibly be obtained by considering more features in the dataset or by using different techniques/parameters (for example an higher value of the parameter n_{tree} for the Random Forest classifier would slightly improve the effectiveness, but it would worsen the efficiency).
Despite this, the project also shows the fact that it's hard to find tweets' features strongly correlated with their popularity as the quantity of like, retweets and replies is influenced by many unpredictable external factors that doesn't concern the tweet itself or the user who wrote it.