BARCELONA TECHNOLOGY SCHOOL

RETWEET PREDICTION PROBLEM

Author
ROBERTO ARENAL MIJARES
21/06/2021

INDEX

Introduction

Nowadays, people rely on twitter as their primary news source. The effect of this is that the quantity of opinions and thoughts about topics of interest is growing exponentially.

Taking into account the popularity that twitter has, an important aspect of this social media is the amount of people that these opinions and thoughts reach. To know that, the consumers of those ideas can express their reaction by liking, sharing, commenting or retweeting, and by doing one of these reactions, the spreading starts to trigger towards more users hence the popularity increases more and more.

In this research, of all the tools that a tweet has to be more popular, the chosen one to be the metric for popularity is the number of retweets. Meaning that by managing to predict the amount of retweets successfully that a tweet is going to produce, the user could know how many people are capable of reaching an opinion, information, and shared idea beforehand.

There have been lots of real use cases where the amount and popularity of tweets exponentially grows thanks to an emerging hot topic, and in this case, being Covid19 the most recent and trending subject. That is why in this research we are using COVID-19 related tweets to experiment with the modeling of retweeting behavior.

Related Work

Trying to predict the popularity of tweets is not a new thing, there have been several different experiments and approaches that although achieve good performance, there are still a lot of opportunities for improvement. This other research includes work that applies techniques like text processing which only focuses on analysing the content of the tweet [1]. And also working with no text but only users account information like # of Followers, Friends/Followees, Favorites, to mention some of them [2].

Our approach differs from the others in the way that we decided to combine either text processing (using entity score) and user's information. Plus the generation of 3 new features explained in more detail in a later section.

Methods

Data Exploration

(Find data exploration in jupyter notebook, will be available here in later version)

Feature Engineering

As feature engineering, we decided to convert our categorical features (entities, mentions, hashtags, URLs, Sentiment) into numerical variables. The specific details for each feature is explained below:

- Entities (the specific score for each entity is extracted and calculate the average and 0 if no entities found)
- Mentions (count the total number of mentions)
- Hashtags (count the total number of hashtags)
- Urls (count the total number of urls)
- Sentiment (the sentiment scores are splitted into positive and negative as new columns)

In the feature extraction phase, we made use of the timestamp feature to generate 3 more features and test their relevance to the model. We started by generating sequential id's of each user's tweets by applying sql row number function ordered by date. With this id, we can then know easily which is the previous tweet and extract our required columns. This will allow us to generate the next two new features:

- Activity of the user(Tweets per day).

This feature is calculated by dividing the total number of tweets at that given time by the # of days since the first tweet. This could potentially tell us that the more active the user is, the more probable that user gains more popularity hence the next tweet.

- Previous tweet popularity

This feature is generated by bringing the user's previous publication retweets. The intention with this new feature is taking advantage of the popularity of the previous tweet of the user as a signal that the next one could be as popular.

The third new feature was extracted directly from the date feature, which is Day Number of the week. This is in case that the other users are more active on weekends than in week days for example.

Here is the result filtered by one specific user.

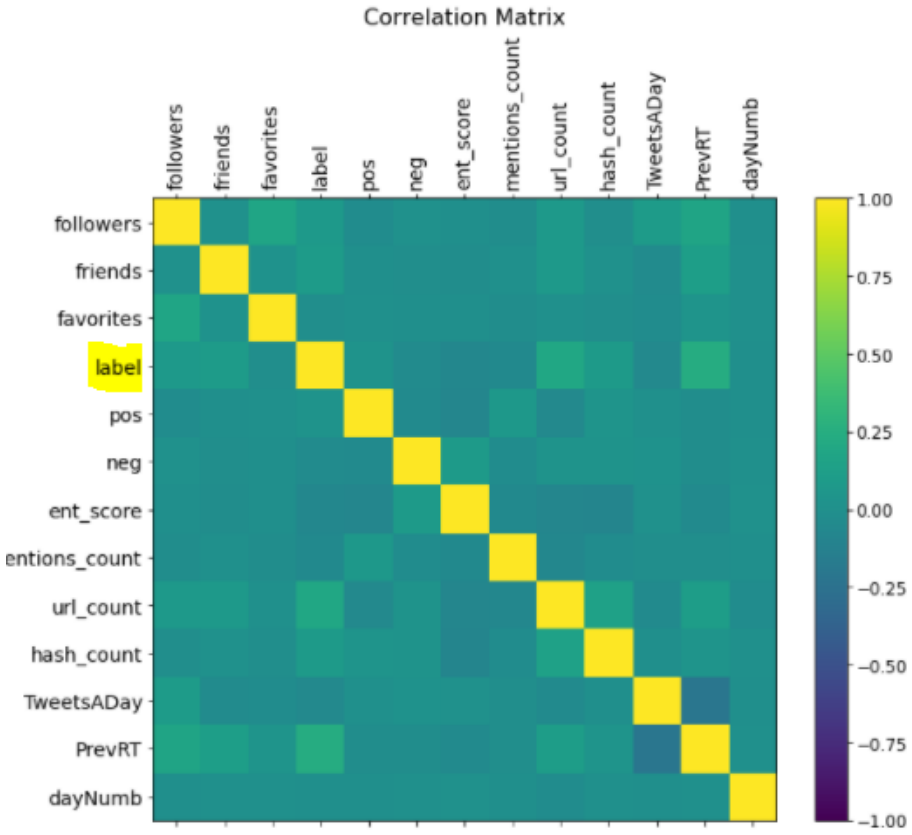
username	Rnumb	Date	TweetsADay	PrevRT	dayNumb	label
d8eb2a8123c267ebb4e4b317ad238f77	1	2019-10-01 03:53:58	1.0	0.0	2	0.0
d8eb2a8123c267ebb4e4b317ad238f77	2	2019-10-01 14:23:55	2.0	0.0	2	2.0
d8eb2a8123c267ebb4e4b317ad238f77	3	2019-10-04 05:08:04	1.0	2.0	5	0.0
d8eb2a8123c267ebb4e4b317ad238f77	4	2019-10-05 09:14:16	1.0	0.0	6	1.0
d8eb2a8123c267ebb4e4b317ad238f77	5	2019-10-08 17:57:36	0.7142857142857143	1.0	2	2.0
d8eb2a8123c267ebb4e4b317ad238f77	6	2019-10-09 20:25:50	0.75	2.0	3	0.0
d8eb2a8123c267ebb4e4b317ad238f77	7	2019-10-10 19:06:43	0.7777777777777778	0.0	4	1.0
d8eb2a8123c267ebb4e4b317ad238f77	8	2019-10-11 18:49:36	0.8	1.0	5	0.0
d8eb2a8123c267ebb4e4b317ad238f77	9	2019-10-12 18:31:53	0.8181818181818182	0.0	6	1.0
d8eb2a8123c267ebb4e4b317ad238f77	10	2019-10-14 12:14:59	0.7692307692307693	1.0	1	1.0
d8eb2a8123c267ebb4e4b317ad238f77	11	2019-10-16 21:33:37	0.7333333333333333	1.0	3	1.0
d8eb2a8123c267ebb4e4b317ad238f77	12	2019-10-19 01:25:06	0.6666666666666666	1.0	6	1.0
d8eb2a8123c267ebb4e4b317ad238f77	13	2019-10-19 05:54:38	0.7222222222222222	1.0	6	1.0
d8eb2a8123c267ebb4e4b317ad238f77	14	2019-10-22 18:37:06	0.6666666666666666	1.0	2	2.0
d8eb2a8123c267ebb4e4b317ad238f77	15	2019-10-23 02:44:37	0.6818181818181818	2.0	3	1.0
d8eb2a8123c267ebb4e4b317ad238f77	16	2019-10-24 10:06:02	0.6956521739130435	1.0	4	2.0
d8eb2a8123c267ebb4e4b317ad238f77	17	2019-10-24 13:39:18	0.7391304347826086	2.0	4	1.0
d8eb2a8123c267ebb4e4b317ad238f77	18	2019-10-26 22:54:14	0.72	1.0	6	2.0
d8eb2a8123c267ebb4e4b317ad238f77	19	2019-10-27 02:44:31	0.7307692307692307	2.0	7	1.0
d8eb2a8123c267ebb4e4b317ad238f77	20	2019-10-27 05:22:42	0.7692307692307693	1.0	7	1.0

Correlation Matrix

As one of our feature importance validation, after getting our final 12 features we applied a feature correlation matrix. This helped us to get an idea of either knowing the relation between features and label, and also see if there are features highly related within each other, causing multicollinearity.

As we can see in the correlation matrix, there is no risk of multicollinearity and few of the features have a considerable relation with our dependent variable. Being the top 5; friends, PrevRT, URLs, followers and entity score.

followers	friends	favorites	label	pos	neg	ent_score	mentions_count	url_count	hash_count	TweetsADay	PrevRT	dayNumb
1.0	0.0113	0.1722	0.076	-0.0271	0.0087	-0.0063	-0.019	0.0767	-0.0139	0.0945	0.1664	-0.0061
0.0113	1.0	0.0179	0.0981	-6.0E-4	-0.0079	-0.0161	0.006	0.0713	0.0137	-0.0379	0.1192	0.0055
0.1722	0.0179	1.0	-0.0094	0.0025	-0.0039	-2.0E-4	-0.017	9.0E-4	-0.0103	-0.0243	0.0313	0.0041
0.076	0.0981	-0.0094	1.0	0.0281	-0.0329	-0.0761	-0.0541	0.1881	0.0895	-0.0573	0.236	-7.0E-4
-0.0271	-6.0E-4	0.0025	0.0281	1.0	-0.0398	-0.0841	0.0698	-0.0573	0.0372	6.0E-4	-0.0183	-0.0015
0.0087	-0.0079	-0.0039	-0.0329	-0.0398	1.0	0.0886	-0.0271	0.0245	0.0237	0.0187	-0.0215	0.0022
-0.0063	-0.0161	-2.0E-4	-0.0761	-0.0841	0.0886	1.0	-0.0487	-0.0838	-0.0909	0.0123	-0.0391	0.0099
-0.019	0.006	-0.017	-0.0541	0.0698	-0.0271	-0.0487	1.0	-0.0725	-0.0243	-0.0126	-0.0128	-0.0013
0.0767	0.0713	9.0E-4	0.1881	-0.0573	0.0245	-0.0838	-0.0725	1.0	0.1464	-0.0395	0.1168	-0.0238
-0.0139	0.0137	-0.0103	0.0895	0.0372	0.0237	-0.0909	-0.0243	0.1464	1.0	-0.0	0.033	0.0023
0.0945	-0.0379	-0.0243	-0.0573	6.0E-4	0.0187	0.0123	-0.0126	-0.0395	-0.0	1.0	-0.212	-0.0044
0.1664	0.1192	0.0313	0.236	-0.0183	-0.0215	-0.0391	-0.0128	0.1168	0.033	-0.212	1.0	0.0049
-0.0061	0.0055	0.0041	-7.0E-4	-0.0015	0.0022	0.0099	-0.0013	-0.0238	0.0023	-0.0044	0.0049	1.0



A second analysis of feature weighing and importance is done later as part of the models in the experimentation phase.

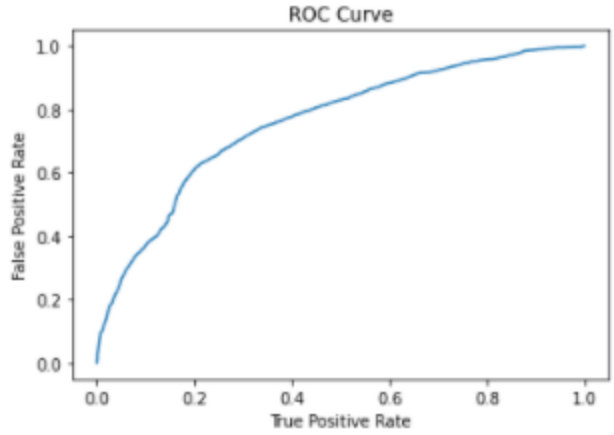
Experimentation

Binary Classification

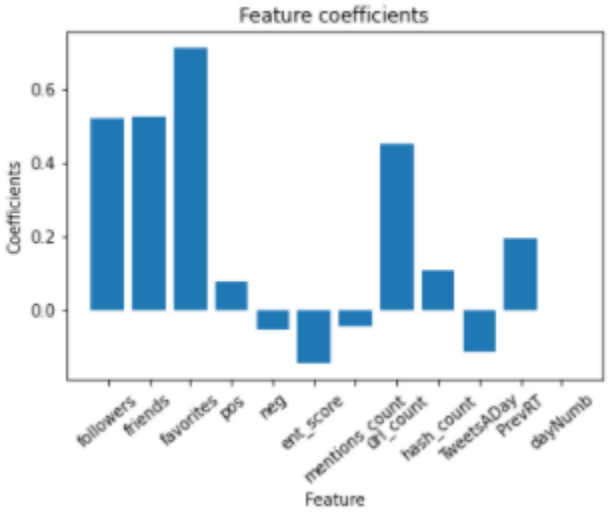
Analysing the high amount of tweets that have 0 retweets, which is 50% of the data, it has been decided to tackle this first set of experiments as a binary classification problem. By turning the rest of the samples that are not 0, we end up having an automatic balanced dataset.

Our first experiment was to use a **logistic regression** model. After performing 10 fold cross validation, the resulting best model was trained with the parameters; maxIter=10, regParam=0.01, elasticNetParam=0.08. Choosing Area Under ROC as a performance metric, we got the following result.

$$AUROC = \int_0^1 \frac{TP}{P} d\left(\frac{FP}{N}\right)$$

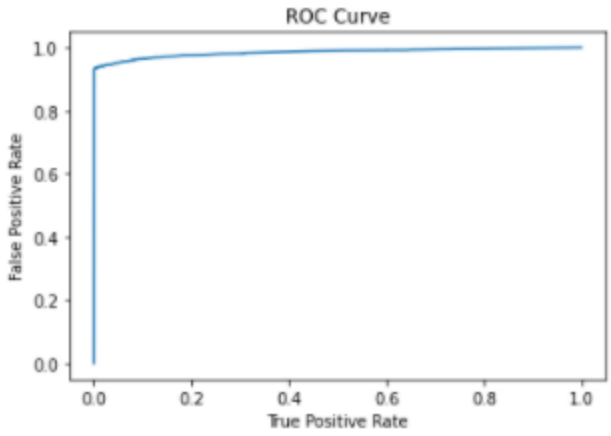



```
Training set areaUnderROC: 0.7603540015500718
Test Area Under ROC 0.732917480686653
```

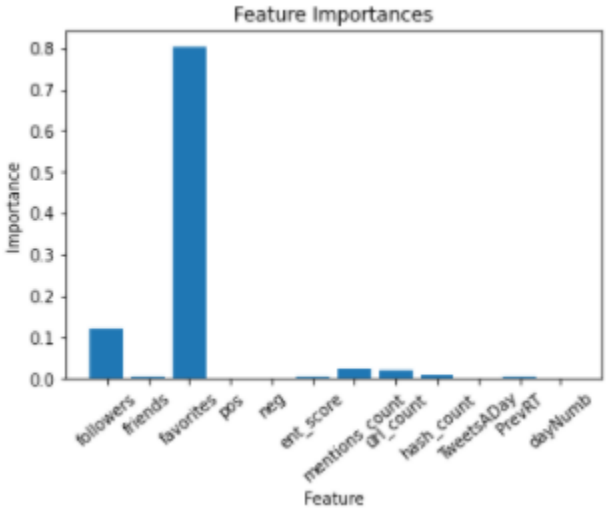


Since logistic regression applies feature selection, the hyperparameters set when training the model added both L1 and L2 regression as regularization methods. This resulted in the reduction/nullification of the non important features. In this case, the feature Day Number was nullified.

In our second experiment, a **Random Forest Classification** was implemented. This time, after a 10 fold cross validation, a model with 20 trees, maxDepth= 5 and maxBins=32 was chosen. Choosing Area Under ROC as a performance metric, we got a higher result in comparison with the previous logistic regression model.




```
Training set areaUnderROC: 0.9848651986942859
Test Area Under ROC 0.9825409837871908
```



Multi-Label Classification

After performing a binary classification, we are confident on predicting if a tweet will be retweeted or not. But now, what if we want to know if that tweet is going viral or will have just a few retweets. This is why we are now performing a multi-label classification.

For this approach, we will start by dropping the tweets with 0 retweets.

The classes were chosen in a way that all of them were as balanced as possible. In this case we end up having 4 classes as we can see in the next table.

```
+----------+-------+
|labels_cat|  count|
+----------+-------+
|       80+| 613089|
|     20-80| 830181|
|      5-20|1052344|
|       1-4|1382673|
+----------+-------+
```

As we can see, they are still a little bit unbalanced. But this is something we are going to fix with undersamplic technique. For this, the lowest sampled class was taken as reference and used to take the same number of random samples (without replacement) for the other classes.

```
+----------+------+
|labels_cat| count|
+----------+------+
|       1-4|613067|
|       80+|613089|
|     20-80|613248|
|      5-20|613390|
+----------+------+
```

For this multi label classification problem, a random forest was implemented.
The metrics chosen to measure the model's performance are the following;

Accuracy

$$ACC = \frac{TP}{TP+FP} = \frac{1}{N}\sum_{i=0}^{N-1} \hat{\delta}\left(\hat{\mathbf{y}}_i - \mathbf{y}_i\right)$$

Weighted precision	$PPV_w = \frac{1}{N}\sum_{\ell \in L} PPV(\ell) \cdot \sum_{i=0}^{N-1} \hat{\delta}(\mathbf{y}_i - \ell)$
Weighted recall	$TPR_w = \frac{1}{N}\sum_{\ell \in L} TPR(\ell) \cdot \sum_{i=0}^{N-1} \hat{\delta}(\mathbf{y}_i - \ell)$
Weighted F-measure	$F_w(\beta) = \frac{1}{N}\sum_{\ell \in L} F(\beta, \ell) \cdot \sum_{i=0}^{N-1} \hat{\delta}(\mathbf{y}_i - \ell)$

Since we are working with multi-label classification, our metrics are now a weighted sum of each one of the labels.

```
print('Accuracy: ', evaluator.evaluate(prediction, {evaluator.metricName: "accuracy"}))
print('False Positive Rate: ', evaluator.evaluate(prediction, {evaluator.metricName: "falsePositiveRateByLabel"}))
print('True Positive Rate: ', evaluator.evaluate(prediction, {evaluator.metricName: "truePositiveRateByLabel"}))
print('F1 Score: ', evaluator.evaluate(prediction))
print('Precision: ', evaluator.evaluate(prediction,{evaluator.metricName: "precisionByLabel"} ))
print('Recall: ', evaluator.evaluate(prediction, {evaluator.metricName: "recallByLabel"}))
```

```
Accuracy:  0.7151898734177216
False Positive Rate:  0.04849539915443919
True Positive Rate:  0.7942264988897113
F1 Score:  0.716174619727391
Precision:  0.8462145110410094
Recall:  0.7942264988897113
```

Discussion

In this report, we did not work with any NLP techniques as we transformed every categorical feature into numerical. Even though we got a decent performing model, there could be a high improvement if we take advantage of the text that we find for example in the hashtags or entities columns.

As potential improvements, a more extensive multi-label classification is left to be done in order to have a comparison between different models and get to choose the one with better performance.

Conclusion

In order to make use of as much data as possible, the approach of first developing a binary classification algorithm was chosen. The result of this first phase will tell us if a tweet and user's attributes are relevant enough to be retweeted or not. Once we know that it will be retweeted, the next phase comes in. A multi-label classification was performed based on 4 ranges (0-4, 5-20, 21-80, 80+).

References

[1] Prediction of Likes and Retweets Using Text Information Retrieval. (2019).

Elsevier. https://www.sciencedirect.com/science/article/pii/S1877050920304129

[2] *RETWEET PREDICTIVE MODELING IN TWITTER*. (2018, enero).

https://estudogeral.uc.pt/bitstream/10316/83564/1/RPMTwitter.pdf