Predicting article Retweets and Favorites based on the title using Machine Learning

Flávio Henrique de Freitas

(Dated: September 10, 2018)

***Abstract*—Choosing a good title for an article is an important step of the writing process. The more interesting the article title seems, the higher the chance a reader will interact with the whole content. This project focus on predicting the number of Retweets and Favorites on Twitter from FreeCodeCamp's articles based on its titles. This problem is a classification task using Supervised Learning. With data from FreeCodeCamp on Twitter and Medium, it was used machine learning methods including Support Vector Machines (SVM), Decision Trees, Gaussian Naive Bayes (GaussianNB), K-Nearest Neighbors, Logistic Regression, Gradient Boosting and MultinomialNB to make the predictions. This study shows that the MultinomialNB model performed better for Retweets reaching an accuracy of 60.6% and Logistic Regression reached 55.3% for Favorites.**

***Keywords*— prediction, machine learning, social media, title, performance**

I. DEFINITION

A. Project Overview

Social networks websites have become an important communication tool and source of information. The hours spent in average connected per day in the past years is up to 6 hours [1] for adults and 9 for teenagers, while 30% of this time is on social networks [2]. During a normal navigation on such platforms, users are exposed to several posts such as friends' statuses, images, news and more. With such amount of information and variety of content, the time for the user to decide to interact with the content is very small. Gitte at al. [3] suggest that we take around 50 milliseconds to make a good first impression and this has proved to be very powerful in a wide range of contexts.

Besides being a place for connecting with friends and sharing moments of the user's life, a survey has shown that social networks are also used as a source of news and information by 67% of the users [4]. Part of these posts are articles that can be read on an external website. Typically such posts show the title of the article and sometimes a small part of its content and an image.

Considering the offer of content and competition with so many interesting posts, showing a proper title for the post affects the probability that a user will check the content. This measure has a strong impact on how many readers an article will have and how much of the content will be read. Furthermore, showing the user a content they prefer (to interact) increases the user satisfaction. It is thus important to accurately estimate the interaction rate of articles based on its title.

B. Related Work

In the literature is possible to find previous studies on the area of classifying the article focused on click-baits title detection [5] [6]. Click-bait headlines normally exploits the curiosity of the reader, proving enough information to make the reader curious, but not enough to fully satisfy the curiosity. In this way, the user is forced to click on the linked content to read the whole article.

Some other studies also investigate this subject using deep learning on cross-domain sentiment analysis [7].

C. Problem Statement

When an author writes a text, it is expected that their words will influence and bring value to the readers. While writing, the title is one of the important details that needs to be taken in consideration, because this will normally be the first contact place of their work. Thus, to create a good first impression, to have more people read the article and interact with it, choosing a good title is very important.

Some of the most used platforms to spread ideas nowadays are Twitter [8] and Medium [9]. On the first one, articles are normally posted including external URLs and the title, where users can access and demonstrate satisfaction with "Favorites" or "Retweet" (share) of the original post. The second one shows the full text with tags to classify the article and "Claps" (similar to Twitter's "Favorites") to show how much the users appreciate the content. A correlation between these two networks can bring us more valuable information.

The problem to be solved is a classification task using Supervised Learning: *Predict the number of Favorites and Retweets an article receives based on the title*

D. Evaluation Metrics

At least one evaluation metric is necessary to quantify the performance of the benchmarks and solution model. For this project, it will be used the accuracy, which is the number of correct predictions made as a ratio of all predictions made.

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions made}}$$

This metric only works well if there are similar number of samples belonging to each class. For this reason, we will divide the range of shares and favorites count in a way that respects this distribution.

II. ANALYSIS

A. Data Exploration

The data used to predict how titles will perform was gathered from the accounts of the non-profit organization FreeCodeCamp on Medium [10] and Twitter [11]. On both social platforms, it was possible to get public information about how the users interacted with the content, using as Favorites and Retweets from Twitter, and "Claps" from Medium.

Correlating the number of Favorites and Retweets from Twitter with a Medium article, is an attempt to isolate the effect of number of reached readers and number of Medium "Claps". Because the more the article is shared in different platforms, the more readers it will reach and the more Medium "Claps" it will receive. Using only the Twitter statistic, it is expected that the articles reached initially almost the same number of readers (that are the followers of the FreeCodeCamp account on Twitter), and their performance and interactions are limited to the characteristics of the tweet, for example, the title of the article, that is exactly what we want to measure.

The FreeCodeCamp account was chosen, because the idea is to limit the scope of the subject of the articles and predict better the response on a specif field. The same title can perform well in one category (e.g. Technology), but not necessarily in a different one (e.g. Culinary). Also this account posts as the Tweet content the title of the original article and the URL on Medium.

After getting the articles from FreeCodeCamp written on Medium and shared on Twitter, there is a dataset of 717 data points. Table I shows some examples of such correlation and table II explains the complete list of fields of the dataset.

TABLE I. Sample of the data points

Title	Retweet	Favorite	Claps
ES9: JavaScript's state of the art in 2018	15	48	618
Here's another way to think about state: How to visually design state in JavaScript	10	30	2
How to understand Gradient Descent, the most popular ML algorithm	4	14	102

TABLE II. Complete description of the dataset fields

Field	Description
Title	The content of the tweet, FreeCodeCamp normally uses the title of the article from Medium and sometimes the username of the author from Twitter
Retweet Count	How many times that tweet was "Retweeted" on Twitter
Favorite Count	How many times that tweet was marked as favorite on Twitter
Medium Claps	How many times that article was marked as favorite on Medium
Medium Categories	Which tags were used to classify the article on Medium
Created at	When the tweet was posted
URL	The website of the article on Medium

1. *Overall Statistic*

We will analyze here the high level statistics of the articles. Try to understand how many times the articles were in average retweeted, clapped or marked as favorites. Also understand the average length and number of words of the title.

TABLE III. Overall Statistic

	Favorite	Re-tweet	Claps	Text Length
count	711.00	711.00	711.00	711.00
mean	49.29	16.44	285.26	80.62
std	45.23	15.69	273.45	22.19
min	0.00	0.00	1.00	21.00
25%	20.00	7.00	6.00	65.00
50%	34.00	11.00	238.00	97.00
75%	63.50	20.00	471.50	97.00
max	298.00	125.00	997.00	146.00

From this statistic is possible to understand the order of magnitude of our dataset. Articles normally are retweeted and marked as favorite around tens of times and clapped hundreds of times. It is possible to check

B. Exploratory Visualization

This section will explore the data visualization of the existing dataset and analyze the possible metrics that will be used to understand the solution. We will identify the relationship between each one of the features with the overall performance of the article.

the maximum values from all the three variables, retweet and favorite hundreds and clap thousand of times. From these numbers we can define what is expected from our articles and the interaction with them. The length of the text goes from 21 to 146 characters, as expected, for a tweet content.

2. *Histogram and Box plots*

In this section we will check how the multiple features are distributed.

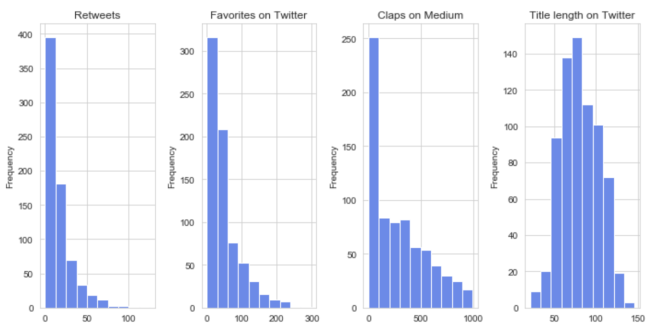

FIG. 1. Histogram

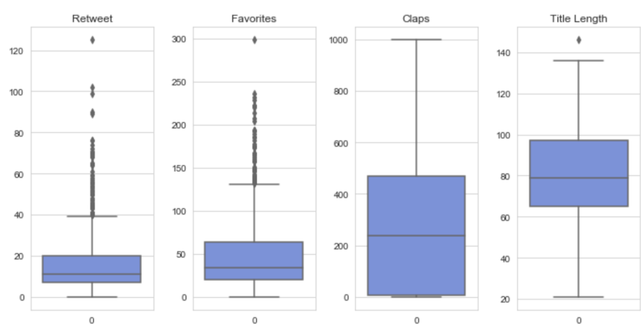

FIG. 2. Box plots

From these histograms together with the overall statistic and the box plots, we can notice that we have a Gaussian distribution for the text length and the average length is around 80 characters. Favorite, retweet and claps are positive-skewed, i.e. they are concentrated on the left part of the graph, meaning that a small part of the articles will over-perform about readers' interaction and biggest part of them will generate less interaction.

3. *Scatter Matrix*

Here we try to find a relationship between the multiple features that we gathered from Twitter and Medium.

We can notice for the image 3, we can notice a clear relationship between number of retweets and favorites.

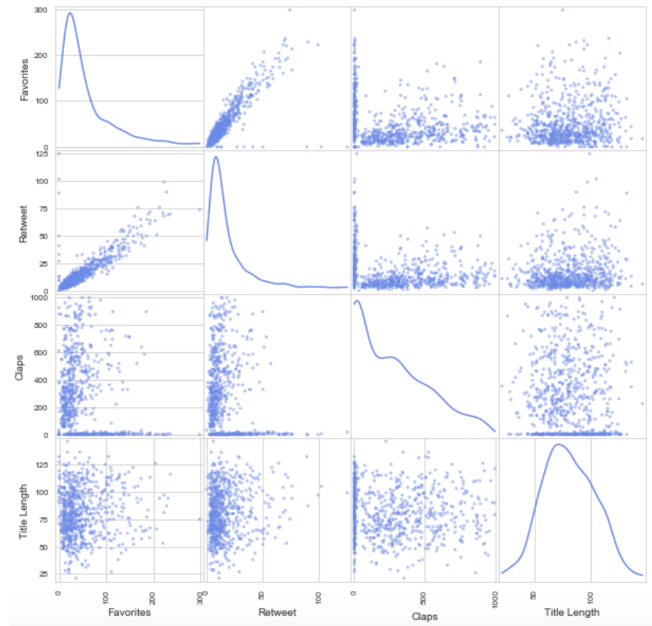

FIG. 3. Relationship between the features

They are directed connected, it means, the more retweets, the more favorites the article will receive and vice versa.

4. *Title length that performed better*

Here we analyze the relationship between the length of the title with its performance. For this experiment, we just considered the 25% top performers of each feature.

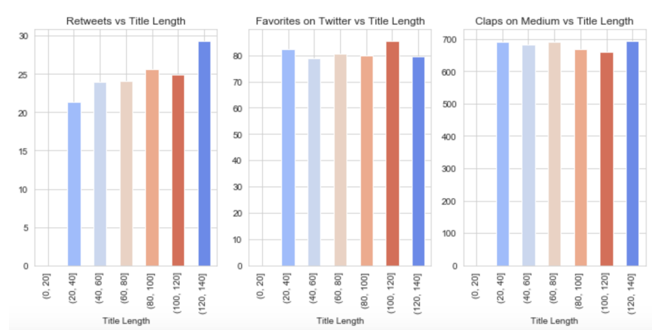

FIG. 4. Title length performance

To avoid being biased by outliers, we removed for each feature (favorites, retweets and claps) analysis the data points that don't fit the following formulas:

$$Outlier < Q_1 - 1.5 * IQR$$

$$Outlier > Q_3 + 1.5 * IQR$$

Where Q_1 and Q_3 are the first and third quartile and IQR is the Interquartile Range (IQR = Q_3 - Q_1).

We can notice from the graphics 4 that longer titles tend to perform a little bit better than shorter ones for retweets, but for claps on Medium and favorites it seems to influence even less.

After analyzing the title length and didn't reach any conclusion, we decided to investigate the number of words in the title.

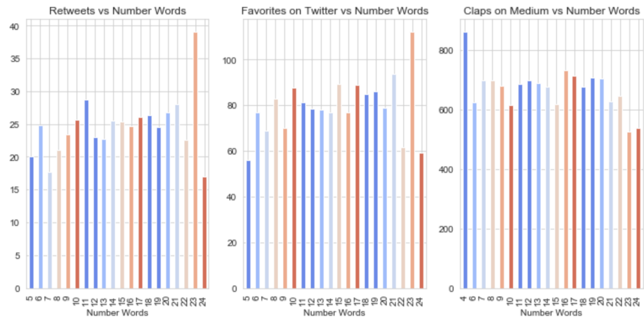

FIG. 5. Performance of the Number of words in the title

From this second experience showed on image 5, we reached the conclusion that neither number of words on the title affect considerably its performance.

5. *Categories that performed better*

Here we filtered the dataset and just analyzed the top 25% performers for each one of the features. We wanted to have a clear overview how the categories perform compared between them. The outliers were removed as explained in section II B 4.

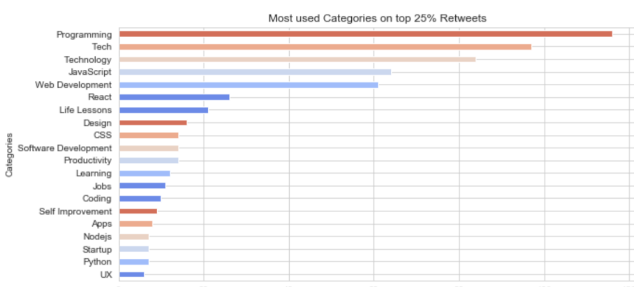

FIG. 6. Best Categories for Retweet

From this statistic we notice that articles created with the following categories can increase the number of retweets, favorites and claps: "Programming", "Tech", "Technology", "JavaScript" and "Web Development".

6. *Words that performed better*

We repeated the same strategy of limiting the 25% performers for the words on the title of the article. We wanted to understand if there are words that can boost the interaction from the readers. The outliers were removed as explained in section II B 4.

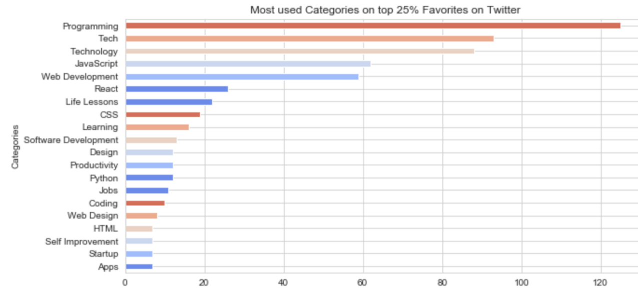

FIG. 7. Best Categories for Favorite

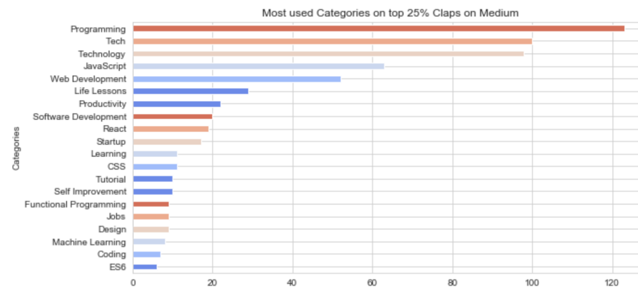

FIG. 8. Best Categories for Claps

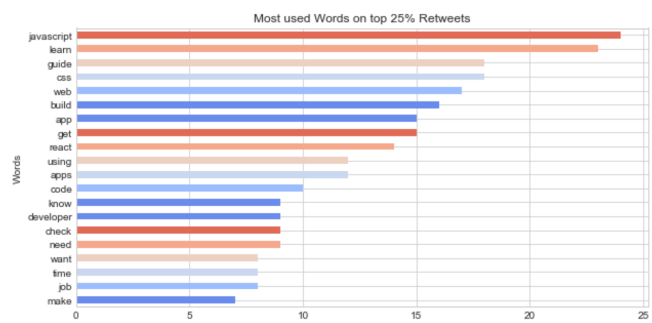

FIG. 9. Best Words for Retweet

In this lexical analysis, we can notice that some words get much more attention on the FreeCodeCamp community than others, we noticed if we want to make our articles reach further in numbers talking about JavaScript, React or CSS will increase this change. Using the words "learn" or "guide" to describe will also increase this probability.

C. Algorithms and Techniques

Classification is a common task of machine learning (ML), which involves predicting a target variable taking in consideration the previous data [12]. To reach such classification, it is necessary to create a model with the previous training data, and then use it to predict the

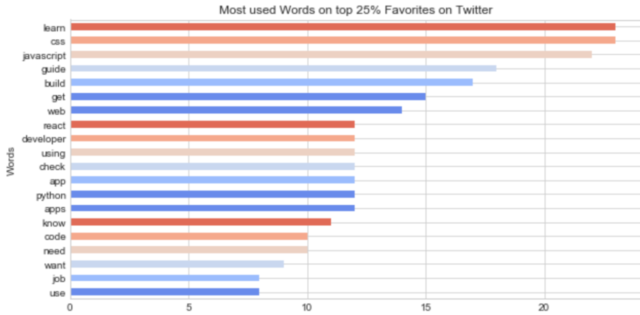

FIG. 10. Best Words for Favorite

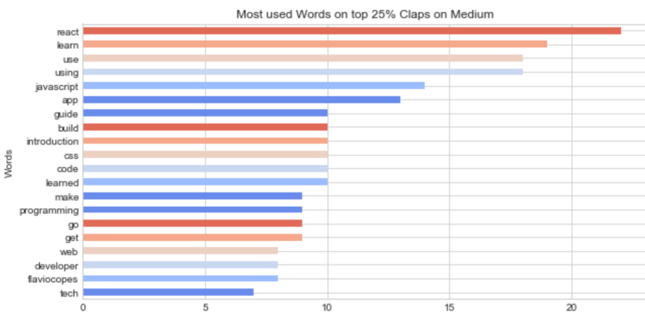

FIG. 11. Best Words for Claps

value of the test data [13]. This process is called Supervised Learning, since the data processing phase is guided toward the class variable while building the model [14].

Predicting number of shares and favorites of an article can be treated as a classification problem, because the output will be discrete values (range of shares and favorites). As input, the title of the articles with each word as a token t1, t2, t3, ... tn.

For this task we will evaluate the following algorithms: Support Vector Machines (SVM), Decision Trees, Gaussian Naive Bayes (GaussianNB), K-Nearest Neighbors and Logistic Regression. In the end, it will be compared the performance of each one of them and one will be chosen. To estimate accuracy, it will be used a 5-fold cross validation, that splits the dataset in 5 parts, 4 of training and 1 of testing. The implementation of this project was made using Python, Numpy [15] and Scikit [16].

D. Benchmark

This project run the same testing and training data for multiple algorithms, the comparison between them was used to evaluate the overall performance. The overall benchmark was made comparing our data with the Logistic Regression results.

III. METHODOLOGY

A. Data Preprocessing

1. Data cleaning

The first part of the data processing was to clean the dataset. After downloading the tweets, we removed the ones that didn't have any URL (that points to the Medium article) or title. Data points with values of favorites, claps or retweets that were not positive numbers or zero were also excluded.

Words that were Twitter users were replaced by the character '@' (that could be used on the statistics) and words that were wrong non ASCII characters were also removed.

Some of the data points have same URL, it means, that they shared more than once on the account of Twitter. After analyzing each one of the duplicates, we noticed that there were two types of retweets: same URL and same title; and same URL and different title. We removed the ones of the first type. For the second type, we left, because the titles were completely rewritten and it can be considered as one different data point.

For the remaining data points, we removed the ones that are considered outliers, as explained in II B 4. We reached the numbers: Retweet and Favorites have 711 items (658 without outliers) each; Claps has the same number with or without outliers, 711.

2. Assigning classes to the dataset

For this project, we decided to classify the number of retweets and favorites in ranges. We wanted to make use of the properties of the Classification family of the Supervised Learning algorithms.

To avoid the Class Imbalance Problem [], we divided the dataset in similar groups, as shown in image 12.

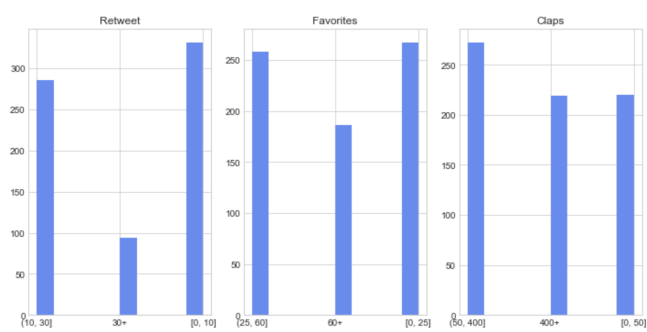

FIG. 12. Range Distribution

The defined ranges for our features are:

1. Retweets: 0-10, 10-30, 30+

2. Favorites: 0-25, 25-60, 60+
3. Claps: 0-50, 50-400, 400+

Where the range 'x-y', means bigger than x and less equal than y. The first range item of each feature also contains the zero on the range.

3. Bag of words

To be possible to analyze the title in each data point, we need to map each word into a number. This is necessary because Machine Learning models normally don't process raw text, but numerical values. To reach this, we used a bag of words model [17]. In this model, it is taken in consideration the presence and often the frequency of words, but the order or position is ignored.

For the calculation of the bag of words, we will use a measure called Term Frequency, Inverse Document Frequency (TF-IDF) [18]. The goal is to limit the impact of tokens (words) that occur very frequently.

At this step, we processed the collection of documents and built a vocabulary with the known words. We reached a vocabulary of 1356 words for retweets, 1399 words for Favorites and 1430 words for Claps.

B. Implementation

1. Training and Testing Data Split

Before starting the training and the evaluation of the models, we split the dataset into test and training sets. Retweets and Favorites they have a total of 658 data points each with 526 (80% approximately) as training and 132 as testing points. Claps has 711 data points with 568 (80% approximately) as training and 143 as testing points.

2. Training and Evaluating Models

For evaluating the model and realizing the prediction, it was chosen to start with the models: Support Vector Machines (SVM), Decision Trees, Gaussian Naive Bayes (GaussianNB), K-Nearest Neighbors and Logistic Regression.

During the implementation tests, we noticed that these models were not satisfying what we expected, so we also trained and predicted using Naive Bayes classifier for multinomial models (MultinomialNB) and Gradient Boosting for classification.

The classification process followed the steps:

1. Load the data (title, favorites, retweets and claps count)
2. Clean the title removing not desired words III A 1
3. Filter the outliers from the dataset II B 4
4. Classify the features in ranges III A 2
5. Divide the dataset in training and test
6. Create a bag of words using TF-IDF to the tiles III A 3
7. Train the model and calculate the accuracy

3. Model Performance Metrics

We separated the dataset into learning and validation set. A validation set is important to reduce the risk of over-fitting of the chosen model. To avoid discarding relevant data points, we used a cross-validation strategy.

Cross-validation splits the training dataset in k folds, being k - 1 folders used to train the model and the last one to test it. This strategy will repeat multiple times and the overall performance is the average of the computed values.

To estimate the model's accuracy, we used a 5-fold cross validation that split the dataset in 5 parts, 4 of training and 1 of testing.

C. Refinement

During the models implementation a lot of steps were tested and some of them needed to be modified to reach better performance. For choosing a better parameter, we interacted over the options and decided for the one that optimized the accuracy.

Ranges of Favorites and Retweets: We tried ranges with different number of elements and also different values for the ranges. Some of them were under performing, while others reached close values. The chosen one divided the ranges with similar number of data points and also offer a good overview of the feature analyzed.

New models: We increased the number of models to be tested in three. The classifiers previous chosen were not reaching a desired accuracy, we decided to add new models to try to make better predictions. The new models have a worse performance than the first ones.

Outliers: We made some tests to discover if we should keep the outliers for the training or remove them. During the tests we discover if we keep the outliers, the accuracy was always worse.

Bag of words: To create the bag of Words, we had the option of choosing the CountVectorizer or TfidfVectorizer. During the simulation we got better results with the last one, TfidfVectorizer.

Clean words: Another step during implementation was to decide if we should keep the title in the original way or remove the undesired words. The tested to remove the name of the Twitter users that appeared in some titles, some wrong characters that appeared on our dataset during the crawling process. After checking the results, we decided to clean the data.

Model's parameters: For each model tested, we calculated the accuracy for the default model (without any parameter, just the default ones) and also we tried to come with better parameters to evolve the accuracy. To test the combination of the new parameters, fine tune the model, we used grid search (GridSearchCV).

IV. RESULTS

A. Model Evaluation and Validation

The tables IV, V and VI describe the accuracy values we reached with the proposed model. The final accuracy for each of the features are: Favorites is 55.3%, Retweets is 60.6% and Claps is 49%.

TABLE IV. Accuracy Favorites (%)

ID	Model Name	Default	Tuned
0	Benchmark	53.15	–
1	**LogisticRegression**	**55.30**	45.45
2	GaussianNB	46.21	46.21
3	DecisionTreeClassifier	43.18	45.45
4	SVC	51.51	51.51
5	KNeighborsClassifier	44.70	46.21
6	MultinomialNB	40.91	45.45
7	GradientBoostingClassifier	47.73	48.48

TABLE V. Accuracy Retweets (%)

ID	Model Name	Default	Tuned
0	Benchmark	57.34	–
1	LogisticRegression	49.24	53.79
2	GaussianNB	59.09	49.24
3	DecisionTreeClassifier	56.06	54.54
4	SVC	55.30	57.58
5	KNeighborsClassifier	57.58	55.30
6	**MultinomialNB**	47.72	**60.61**
7	GradientBoostingClassifier	56.82	55.30

The parameters that we obtained by the grid search of the models and features are explained in the tables VII, VIII and IX.

TABLE VI. Accuracy Claps (%)

ID	Model Name	Default	Tuned
0	Benchmark	42.65	–
1	**LogisticRegression**	46.85	**48.95**
2	GaussianNB	41.26	41.26
3	DecisionTreeClassifier	37.06	44.06
4	SVC	49.65	49.65
5	KNeighborsClassifier	39.16	41.95
6	MultinomialNB	42.66	42.66
7	GradientBoostingClassifier	44.76	37.76

TABLE VII. Tuned Parameters Favorites

ID	Parameters
1	C=1, class_weight=None, dual=False, fit_intercept=True, intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1, penalty='l2', random_state=None, solver='liblinear', tol=0.0001, verbose=0, warm_start=False
2	priors=None
3	class_weight=None, criterion='gini', max_depth=None, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, presort=False, random_state=None, splitter='best'
4	C=1, cache_size=200, class_weight=None, coef0=0.0, decision_function_shape='ovr', degree=1, gamma='auto', kernel='linear', max_iter=-1, probability=False, random_state=None, shrinking=True, tol=0.001, verbose=False
5	algorithm='auto', leaf_size=50, metric='minkowski', metric_params=None, n_jobs=1, n_neighbors=20, p=2, weights='uniform'
6	alpha=0.5, class_prior=None, fit_prior=True
7	criterion='friedman_mse', init=None, learning_rate=0.1, loss='deviance', max_depth=10, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=2, min_samples_split=0.5, min_weight_fraction_leaf=0.0, n_estimators=100, presort='auto', random_state=None, subsample=1.0, verbose=0, warm_start=False

B. Justification

The benchmark used on this project was Logistic Regression model, with ID 0 and named Benchmark on the tables IV, V and VI.

In all the scenarios our models predicted better than the benchmark, for this reason we can assume the decisions made on the step III C leaded us to a better model.

TABLE VIII. Tuned Parameters Retweets

ID	Parameters
1	C=10, class_weight=None, dual=False, fit_intercept=False, intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1, penalty='l2', random_state=None, solver='lbfgs', tol=0.0001, verbose=0, warm_start=False
2	priors=None
3	class_weight='balanced', criterion='gini', max_depth=20, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, presort=False, random_state=None, splitter='best'
4	C=1, cache_size=200, class_weight=None, coef0=0.0, decision_function_shape='ovr', degree=1, gamma='auto', kernel='linear', max_iter=-1, probability=False, random_state=None, shrinking=True, tol=0.001, verbose=False
5	algorithm='auto', leaf_size=10, metric='minkowski', metric_params=None, n_jobs=1, n_neighbors=10, p=2, weights='uniform'
6	alpha=1, class_prior=None, fit_prior=True
7	criterion='friedman_mse', init=None, learning_rate=0.5, loss='deviance', max_depth=10, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=2, min_samples_split=1.0, min_weight_fraction_leaf=0.0, n_estimators=50, presort='auto', random_state=None, subsample=1.0, verbose=0, warm_start=False

TABLE IX. Tuned Parameters Claps

ID	Parameters
1	C=2, class_weight=None, dual=False, fit_intercept=True, intercept_scaling=1, max_iter=10, multi_class='ovr', n_jobs=1, penalty='l2', random_state=None, solver='liblinear', tol=0.0001, verbose=0, warm_start=False
2	priors=None
3	class_weight=None, criterion='gini', max_depth=10, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, presort=False, random_state=None, splitter='random'
4	C=1, cache_size=200, class_weight=None, coef0=0.0, decision_function_shape='ovr', degree=1, gamma='auto', kernel='linear', max_iter=-1, probability=False, random_state=None, shrinking=True, tol=0.001, verbose=False
5	algorithm='auto', leaf_size=30, metric='minkowski', metric_params=None, n_jobs=1, n_neighbors=20, p=2, weights='uniform'
6	alpha=0.5, class_prior=None, fit_prior=True
7	criterion='friedman_mse', init=None, learning_rate=0.5, loss='deviance', max_depth=3, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=0.5, min_weight_fraction_leaf=0.0, n_estimators=100, presort='auto', random_state=0, subsample=1.0, verbose=0, warm_start=False

V. CONCLUSION

A. Free-Form Visualization

The image 13 shows the evolution of the development of this project. We started with the benchmark. From there we started adding and testing features and treating the dataset. The steps are describe by the table ??.

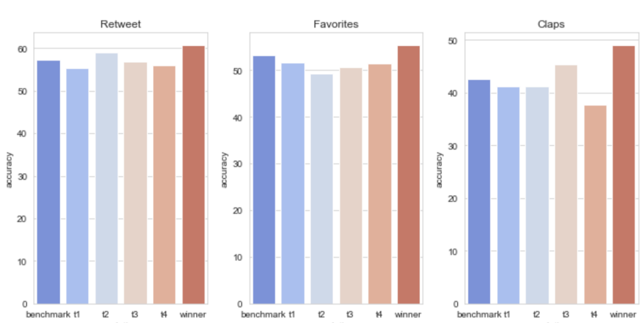

FIG. 13. Evolution Accuracy

During the evolution of the metrics, some variables deprecated the accuracy value, but in further steps, it made it grow. The evolution happened in the following steps:

1. Benchmark
2. t1: Parameters from the model tuned
3. t2: t1 + Removed outliers
4. t3: t2 + Cleaned the words
5. t4: t3 + Added TF-IDF
6. Winner: t4 + Used stopwords

B. Reflection

In this project, we developed classifiers to understand how many times an article will receive interaction like Favorites and Retweets (both on Twitter) and Claps (on Medium). We also presented a list of words that have high change to impact positively with the readers, when used on the title or the article. We classified and extracted information about the Categories used on Medium that are commonly presented on our top performers. Number of words and length of the title were also discussed and presented an optimal number to increase the success numbers.

Besides the mathematical analysis used to extract important characteristics of the dataset, we also developed and trained models to predict the how an article would perform. To achieve this machine learning project, some features and characteristics were used:

1. Bag of words to tokenize the words of the title
2. Term Frequency, Inverse Document Frequency (tf-idf) to translate the frequency of words in the dataset
3. Clean the dataset and each title before training the model (remove Twiter users and invalid characters)
4. Grid search to search for the best model parameters
5. Remove outliers before processing the data
6. Test the dataset to discover a good relation between train and test data points
7. Test and split the number of Favorite, Retweet and Claps in ranges
8. Use stopwords to remove common terms of the language

Following these steps listed here this methodology and framework can be used to classify any kind of article and subjects that are created on Medium and shared on Twitter. This solution is not limited by the context neither the subject of the articles and can be easily reproduced to other datasets.

The hard part of the project was to reach a higher accuracy than the one found with simple models, it was necessary multiple reiterations and several modifications on the initial assumption. Reaching the 61%, 55% and 49% is not the ideal solution, but it can clearly lead to the creation of a good title.

C. Improvement

For future work we can think about some aditional improvements: Adding more features to the original dataset making possible to relate more information to the success of the article. For example, we can correlate the words of the title, with trendy words of the month; Bring more data points to train our model, would also increase the accuracy of the solution; and try to use the position of the word on the title to classify its importance.

REFERENCES

[1] *eMarketer Report. (2017). US Time Spent with Media: eMarketer's Updated Estimates for 2017. Accessed 9 Aug. 2018. Available at: https://www.emarketer.com/Report/US-Time-Spent-with-Media-eMarketers-Updated-Estimates-2017/2002142.*

[2] *Common Sense Media. (2015). The Common Sense Census: Media Use by Tweens and Teens. Accessed 9 Aug. 2018. Available at: https://www.commonsensemedia.org/research/the-common-sense-census-media-use-by-tweens-and-teens.*

[3] *Lindgaard, Gitte Fernandes, Gary Dudek, Cathy M. Brown, Judith. (2006). Attention web designers: You have 50 milliseconds to make a good first impression! Behaviour and Information Technology, 25(2), 115-126. Behaviour IT. 25. 115-126. 10.1080/01449290500330448.*

[4] *Shearer, E., Gottfried, J. (2017). News use across social media platforms 2017. Accessed 9 Aug. 2018. Available at: http://www.journalism.org/2017/09/07/news-use-across-social-media-platforms-2017/.*

[5] *Chen, Yimin, Niall J. Conroy, and Victoria L. Rubin. "Misleading Online Content: Recognizing Clickbait as "False News"" ResearchGate. ACM WMDD, 9 Nov. 2015.*

[6] *Lex, Elisabeth, Andreas Juffinger, and Michael Granitzer. "Objectivity Classication in Online Media." ResearchGate. Proceedings of the 21st ACM Conference on Hypertext and Hypermedia, 13 June 2010.*

[7] *Glorot, Xavier, Antoine Bordes, and Yoshua Bengio. "Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach." WUSTL. Proceedings of the 28 Th International Conference on Machine Learning, 2011.*

[8] *Twitter. Accessed 13 Aug. 2018. Available at: https://www.twitter.com.*

[9] *Medium. Accessed 13 Aug. 2018. Available at: https://www.medium.com.*

[10] *freecodecamp. Accessed 13 Aug. 2018. Available at: https://medium.freecodecamp.org/.*

[11] *freecodecamp.org. Accessed 13 Aug. 2018. Available at: https://twitter.com/freecodecamp.*

[12] *N. Abdelhamid, A. Ayesh, F. Thabtah, S. Ahmadi, W. Hadi. MAC: A multiclass associative classification algorithm J. Info. Know. Mgmt. (JIKM), 11 (2) (2012), pp. 125001-1-1250011-10 WorldScinet.*

[13] *I.H. Witten, E. Frank, M.A. Hall. Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, Burlington, MA (2011).*

[14] *Thabtah, S. Hammoud, H. Abdeljaber, Parallel associative classification data mining frameworks based mapreduce, To Appear in Journal of Parallel Processing Letter, March 2015, World Scientific, 2015.*

[15] *NumPy. Accessed 14 Aug. 2018. Available at: http://www.numpy.org/.*

[16] *scikit-learn. Accessed 14 Aug. 2018. Available at: http://scikit-learn.org/stable/.*

[17] *Nahm, U. Y.; Mooney, R. J. Text mining with information extraction. In: AAAI 2002 Spring Symposium on Mining Answers from Texts and Knowledge. Bases. [S.l.: s.n.], 2002. v. 1.*

[18] *Baeza-Yates, R.; Ribeiro-Neto, B. et al. Modern information retrieval. [S.l.]: ACM press New York, 1999. v. 463.*