Nelson Joukov Costa de Oliveira

RETWEET PREDICTIVE MODEL IN TWITTER

Dissertation Report
Master in Informatics Engineering
advised by Prof. Dr. Bernardete Ribeiro and Prof. Dr. Catarina Silva.
and presented to the Departament Informatics Engineering
of the Faculty of Sciences and Technology of the University of Coimbra

January 2018

· U C ·

UNIVERSIDADE DE COIMBRA

University of Coimbra
Department of Informatic Engineering

Master's Degree in Informatics Engineering
Dissertation
January, 2018

Retweet Predictive Model in Twitter

Author
Nelson Joukov Costa de Oliveira
University of Coimbra
Department of Informatic Engineering
njoukov@student.dei.uc.pt

Advisor
Prof. Dr. Bernardete Ribeiro
University of Coimbra
Department of Informatic Engineering
bribeiro@dei.uc.pt

Jury
Prof. Dr. Paulo Rupino da Cunha
University of Coimbra
Department of Informatic Engineering
rupino@dei.uc.pt

Prof. Dr. Pedro Abreu
University of Coimbra
Department of Informatic Engineering
pha@dei.uc.pt

Abstract

Nowadays Twitter is one of the most used social networks with over 1.3 billion users. Twitter allows its users to write messages called tweets that can contain up to 140 characters. In this social network the so called retweeting is the key mechanism to information propagation. Twitter is widely used by brands, celebrities and news sources. Our main goal is to build a Retweet Predictive Model that predicts the popularity of the tweet and show how different text features affect its performance.

Predicting popularity can have many applications, such as allowing the users to build their message in a way that it has more chances of getting a certain popularity, which can help them to gain more followers. This in the case of marketing companies may have a great impact on their sales, since Twitter is widely used by them.

In our work we divided the popularity in 4 classes (0 retweets, 1 to 10 retweets, 10 to 100 retweets, 100 or more retweets) and analyzed how the prediction for different classes was affected for tweets with specific text features (for example tweets with Hashtags). We have introduced some new features in our model, such as the combination of sentiment analysis with trending topics, along with the use of media content like the presence of photos, GIFs and videos. In our results we show that the use of sentiment analysis with trending topics improves the performance of the model, and the prediction for different classes is affected differently by the features used in the text.

Finally, our novel Retweet Predictive Model built upon machine learning and the Twitter social media can have application in many fields such as marketing, politics or even finance. Its characteristics are unique, not only because it has the capability to predict the popularity of the tweets but also because it incorporates tweet's features such as sentiment scores and popular topics across Twitter. Moreover, it will be a useful tool in personalized Tweet Recommendation Systems

Keywords

Twitter, Machine Learning, Recommendation Systems, Retweeting.

Index

1 Introduction **15**
- 1.1 Motivation and Context . . . 15
- 1.2 Objectives . . . 16
- 1.3 Contributions . . . 17
- 1.4 Methodology . . . 17
- 1.5 Structure of the document . . . 18

2 Background **19**
- 2.1 ML algorithms . . . 19
  - 2.1.1 Supervised Learning . . . 20
- 2.2 Classification metrics . . . 22
  - 2.2.1 Metrics . . . 23
  - 2.2.2 Micro averaging vs Macro averaging . . . 24
- 2.3 Text Preprocessing Techniques . . . 25
  - 2.3.1 Tokenization . . . 25
  - 2.3.2 Stop Words Removal . . . 25
  - 2.3.3 Stemming . . . 26
- 2.4 Text Mining . . . 26
  - 2.4.1 Bag of Words . . . 26
  - 2.4.2 Term Frequency- Inverse Document Frequency . . . 27
  - 2.4.3 Part-of-speech tagging . . . 27
  - 2.4.4 Named Entity Recognition . . . 27
  - 2.4.5 Sentiment analysis . . . 28
- 2.5 Conclusions . . . 28

3 State of the art **29**
- 3.1 What does influence the retweets? . . . 29
- 3.2 Retweet Predictive Models . . . 32
  - 3.2.1 ML Classification models . . . 32
  - 3.2.2 Other approaches . . . 34
- 3.3 Tweet Recommendation Systems . . . 35

3.4 Technologies overview . . . 36
3.4.1 Twitter APIs . . . 36
3.4.2 Twitter text analysis tools . . . 37
3.5 Conclusions . . . 38

4 Proposed approach **41**
4.1 Problem definition . . . 41
4.2 Retweet Predictive Model . . . 42
4.3 Data collection . . . 44
4.3.1 Structure of the Twitter's data . . . 44
4.3.2 Datasets collection . . . 44
4.3.3 Trends Collection . . . 45
4.4 Database . . . 46
4.4.1 Database-creation . . . 46
4.4.2 Database-insertion challenges . . . 48
4.5 Features . . . 49
4.5.1 User Features . . . 49
4.5.2 Tweet Features . . . 50
4.5.3 Extra tweet features . . . 52
4.6 Conclusions . . . 53

5 Experimental Results **55**
5.1 Setup . . . 55
5.1.1 Data splitting and sampling . . . 55
5.1.2 Testing Sets . . . 57
5.2 Results analysis for the Twitter Streaming API dataset . . . 60
5.2.1 Global tests . . . 60
5.2.2 Tests featuring length of the message . . . 61
5.2.3 Test featuring tweets without replies . . . 63
5.2.4 Tests featuring Hashtags . . . 63
5.2.5 Tests featuring URLs . . . 65
5.3 Results analysis using the Twitter Search API dataset . . . 66
5.3.1 Global tests . . . 67
5.3.2 Tests for the number of retweets of a user last k tweets . . . 68
5.4 Results Discussion . . . 74
5.4.1 Twitter Streaming API . . . 74
5.4.2 Twitter Search API . . . 75

6 Conclusions & Future Work **77**

A Tables corresponding to the JSON Objects of the Twitter APIs **83**

B Results for the Class Balancing approach **87**

C Paper published in RECPAD 2017 **93**

List of Figures

4.1 Retweet Predictive Model overview 43
4.2 Database-scheme 48

List of Tables

2.1 Classification example (Confusion matrix) regarding the class C . 23
2.2 BOW example 26

4.1 Range of retweets of each class 42
4.2 Number of tweets for each class of the Twitter Streaming API dataset 45
4.3 Number of tweets for each class of the Twitter Search API dataset 45

5.1 Class 2 balancing 56
5.2 Class 3 balancing 57
5.3 Class 4 balancing 57
5.4 Global tests 58
5.5 Tests for the Twitter Streaming API dataset 59
5.6 Results for testing with tweets without using the extra features . 60
5.7 Results for testing with tweets using the extra features 60
5.8 Results for testing with tweets using the extra features with sub class balancing 61
5.9 Results for testing with tweets with less than 60 characters 62
5.10 Results for testing with tweets between 60 and 100 characters . . 62
5.11 Results for testing with tweets with more than 100 characters . . 63
5.12 Results for testing with tweets that are no replies 63
5.13 Results for testing with tweets without Hashtags 64
5.14 Results for testing with tweets with one Hashtag 64
5.15 Results for testing with tweets with one or more Hashtags 65
5.16 Results for testing with tweets without URLs 66
5.17 Results for testing with tweets with URLs 66
5.18 Results for testing with tweets without using the extra features . 67
5.19 Results for testing with tweets using the extra features 67
5.20 Results for testing with tweets using the extra features with sub class balancing 68
5.21 Results for testing using the number of retweets of the last tweet for each user 69

5.22 Results for testing using the number of retweets of the last 2 tweets for each user . . . 69
5.23 Results for testing using the number of retweets of the last 3 tweets for each user . . . 69
5.24 Results for testing using the number of retweets of the last 4 tweets for each user . . . 70
5.25 Results for testing using the number of retweets of the last 5 tweets for each user . . . 70
5.26 Results for testing using the number of retweets of the last 6 tweets for each user . . . 71
5.27 Results for testing using the number of retweets of the last 7 tweets for each user . . . 71
5.28 Results for testing using the number of retweets of the last 8 tweets for each user . . . 71
5.29 Results for testing using the number of retweets of the last 9 tweets for each user . . . 72
5.30 Results for testing using the number of retweets of the last 10 tweets for each user . . . 72

A.1 Table of the main characteristics in a structure of a tweet's JSON Object . . . 83
A.2 Table of the main characteristics in a structure of a JSON Object that contains the entities. . . . 84
A.3 Table of the main characteristics in a structure of a retweet's JSON Object. . . . 84
A.4 Table of the main characteristics in a structure of the JSON Object of a User. . . . 85
A.5 Number of retweets in different ranges of retweets of the class 3 . 85
A.6 Number of retweets in different early ranges of retweets of the class 4 86

B.1 Results for testing with tweets with less than 60 characters 87
B.2 Results for testing with tweets between 60 and 100 characters . . 87
B.3 Results for testing with tweets with more than 100 characters . . 88
B.4 Results for testing with tweets that are no replies 88
B.5 Results for testing with tweets without hashtags 88
B.6 Results for testing with tweets with one hashtag 88
B.7 Results for testing with tweets with 1 or more hashtags 89
B.8 Results for testing with tweets without URLs 89
B.9 Results for testing with tweets with URLs 89
B.10 Results for testing using the number of retweets of the last tweet for each user . . . 89

B.11 Results for testing using the number of retweets of the last 2 tweets for each user 90
B.12 Results for testing using the number of retweets of the last 3 tweets for each user 90
B.13 Results for testing using the number of retweets of the last 4 tweets for each user 90
B.14 Results for testing using the number of retweets of the last 5 tweets for each user 90
B.15 Results for testing using the number of retweets of the last 6 tweets for each user 91
B.16 Results for testing using the number of retweets of the last 7 tweets for each user 91
B.17 Results for testing using the number of retweets of the last 8 tweets for each user 91
B.18 Results for testing using the number of retweets of the last 9 tweets for each user 91
B.19 Results for testing using the number of retweets of the last 10 tweets for each user 92

List of Acronyms

API	Application Programming Interface
BOW	Bag Of Words
ML	Machine Learning
NB	Naïve Bayes
NER	Named Entity Recognition
SVM	Support Vector Machines
TF-IDF	Term Frequency-Inverse Document Frequency

Chapter 1

Introduction

This chapter introduces the main objectives and contributions of this work. Firstly, we will explain our motivation and context of this work. Then in the following sections, we will explain the objectives, contributions, methodology and structure of the report.

1.1 Motivation and Context

Twitter is a very popular micro-blogging social network founded in 2006. Authors of a study [1] say that Twitter plays a role of news media when announcing breaking news. In this social network a user can post a message (tweet), which can be shared (retweet) by other users. These users can also favorite a tweet and reply to it. Currently Twitter has over 1.3 billion accounts but it is estimated that only 550 million had ever made a tweet. Twitter is widely used by social media, brands and celebrities. It can influence the brand's reputation since 80% of Twitter users have mentioned a brand in a tweet. Since Twitter has 500 million tweets daily, this information can be used to marketing purposes, personal interests, like finding the most interesting content. Popular tweets can be propagated through multiple hoops (users) away from the user who tweeted, before they become viral. Since Twitter has 500 million tweets daily, this information can be used to marketing purposes, personal interests, like finding the most interesting content. Popular tweets can be propagated through multiple hoops (users) away from the user who tweeted, before they become viral. In Twitter the information propagates from the user to his followers, so every time that a user tweets, retweets or favorites a tweet it will appear on the time line of his followers. The difference here is that the retweet will stick to the user's tweet list. tweets can also spread based on the hashtag/topic of the tweet, since the Twitter allows a user to search tweets according to different criteria, which makes them more visible to the entire Twitter's

network.
Although Twitter has millions of tweets each day, not all of them become popular. Predicting this popularity can be very useful to marketing companies in order to increase the popularity and sales rate of a brand, movies, political campaigns to gather the attentions of the population. There are many ways to measure the popularity of a tweet such as the number of retweets, number of favorites and number of replies. The number of retweets measure is the most indicated one, since the more retweets a tweet gets, the more users will be able to see the tweet, so basically whenever a user retweets a tweet, he wants to show it to his followers, and when a user favorites a tweet it's like a sign of personal interest on the tweet. As for the reply count the reason why it's not a good measure it is because even if a tweet gets many replies it does not mean that it will get propagated through the Twitter's network. In the Twitter's network to be more precise only a small part of the tweets get retweeted. Obviously there are many reasons why certain tweets have more popularity than others. These reasons might be, the user who tweeted (celebrity, company, and others), if the topic of the tweet belongs to a trending one, the relation of the user and the follower, if the tweet has a hashtag or not, and many other things for example like the interests of the followers of a user. Building a Retweet Predictive Model will allow us to identify the popular tweets which can help users to personalize their tweets in order to have higher chances to create a popular message and become popular. Our main motivation is to find what are the reasons why some tweets are more popular than others, which might help us to recommend the use of hashtags and topics to add in the message, in order expand our network and provide better content to our followers.

1.2 Objectives

Our main goal is to build a Retweet Predictive Model as well as show how different factors (i.e. Hashtags and URLs) will affect it. The objective of this model is to predict how many retweets will a tweet have. Our model will be based on a machine learning approach, which will use data from the author of the tweet, the characteristics of the message itself which includes Sentiment Analysis and the popularity of the current Twitter's topics. The idea is to observe how the performance of different the classes of the model behave to a different personalized sets of tests, by that we mean tweets with different characteristics, such as tweets without hashtags and tweets with a short amount of text, among others. Studying the effectiveness of the model on different text features, will allow the user to build their message around the features that got the best results for popularity prediction. Another goal of our work is the introduction of sentiment analysis and topics of the tweets. Besides, we will also study how the past information about

the number of retweets of a user affects future predictions.

1.3 Contributions

The main contribution of this work is:

- An innovative Retweet Predictive Model using a model-based ML approach.

The Retweet Predictive Model developed presents the following novel characteristics:

- Ability to predict the popularity of a tweet
- Capability of the developed approach to be used in Tweet Recommendation Systems in order to detect first the popular tweets
- Usability for personalized testing based on tweet attributes such as Hashtags and URLs, which can be used by companies and people to expand their social network/reputation.

Another 2 contributions are:

- The paper presented at RECPAD (Encontro Nacional de Reconhecimento de Padrões) in October 2017.

  Nelson Oliveira, Joana Costa, Catarina Silva and Bernardete Ribeiro, **On the importance of User's Features in Retweeting**, Amadora, 27th October 2017.

- Paper submited for IEEE-International Joint Conference on Neural Networks (IJCNN), World Congress on Computational Intelligence (WCCI18), July, Rio de Janeiro, Brasil, 2018.

  Nelson Oliveira, Joana Costa, Catarina Silva and Bernardete Ribeiro, **Retweet Predictive Model for Predicting the Popularity of Tweets**, IEEE-International Joint Conference on Neural Networks (IJCNN), World Congress on Computational Intelligence (WCCI18), 8-13 July, Rio de Janeiro, Brasil, 2018 (submitted).

1.4 Methodology

This work was conducted over the course of two semesters. In the first semester we focused on the review of the literature, in order to become familiar with the

field and discuss related work. We also did some exploratory analysis and some preliminary tests with our Retweet Predictive Model.
The following list shows the plan for the first semester.

- State of the art review.
- Feature extraction in Twitter social networks streams.
- Preliminary tests on the Retweet Predictive Model.
- Model assessment on Twitter datasets.
- Writing of the intermediate report.

The second semester was focused on the implementation and evaluation of the Retweet Predictive Model. The following list presents the working plan for the second semester.

- Development of a Retweet Predictive Model.
- Tests and analysis of text features on the performance of the model.
- Writing of the Thesis report and Scientific paper.

1.5 Structure of the document

The remaining part of this document is structured as follows: In the next chapter we will give some background knowledge on ML algorithms, performance classification metrics, and some techniques used to extract features from text. This knowledge is important for a better understanding of this thesis. In chapter 3, we provide an overview of the factors that influence retweeting, some approaches used in the field of retweet prediction as well as some related work to Twitter Recommendation systems. To conclude the third chapter we will present technologies that were built for Twitter in order to facilitate data manipulation. In chapter 4 we will describe our model architecture and detail our dataset. Results will be presented in chapter 5. Finally in chapter 6 the conclusions and future work are presented together with possible extensions to our Retweet Predictive Model.
The document contains also 3 appendices. Appendix A presents the Tables corresponding to the JSON Objects of the Twitter APIs, along side with the tweets distribution. Appendix B contains the Tables of the remaining results of our Sampling methods. Finally Appendix C ends this thesis with the paper published in the conference RECPAD 2017.

Chapter 2

Background

In this chapter we will discuss the basic concepts that are required to understand the rest of this report. We first start by presenting the general definition of ML algorithms and give some examples. Then we will show the metrics used to evaluate the results of classification performed by ML algorithms. And finally, we will present some techniques to extract features and information from text, since it is important to use it in our model.

2.1 ML algorithms

ML algorithms are based on a process that learns from a given information (also know as features) and predicts an output. With the increase of the popularity of Big Data in the past years, these algorithms became widely used, since they allow to infer knowledge from data, which can have multiple applications in real world problems, such as stock market value prediction [2], predicting who is going to win the presidential elections [3, 4], or as simple as filtering spam messages from non-spam [5]. Since the information flow in Twitter generates a lot of data per day by a lot we mean more than 500 millions of tweets, this data can be used by these algorithms to multiple purposes regarding Twitter, such as Retweet Predictive Models [6, 7, 8].
ML algorithms can be divided mainly in two types [9], Supervised Learning and Unsupervised Learning. The first type is applied when we know target, so that we can predict for new data this target based on previous information. The second type is mainly used when we do not have these targets and we want to discover implicit relationships in our dataset. Since the main goal of our work is to build a model that predicts the class of a tweet using a complete dataset, by that we mean for each of the tweets that we have we know the label (number of retweets of each tweet), that implies the use of Supervised ML algorithms that we will describe

bellow.

2.1.1 Supervised Learning

Supervised Learning algorithms use a dataset that contains the input values (features) and the correspondent output value for each example, which allows the algorithm to learn from it and build the correspondent model that can predict the outcome for future examples.

Classification vs Regression

Supervised Learning algorithms can be divided in two type, according to the type of the output (i.e. a category or a continuous value). These types are classification and regression.
Classification as the name says the output corresponds to a class (category). A simple example can be classifying an image into violent and non-violent class based on its characteristics.
As for regression the output is a continuous value , a good example is predicting the price of a house based on, its size, location, number of rooms, if it has a pool etc.

Algorithms

There are multiple ML algorithms for supervised learning, each one of them has its unique properties. We can't say that there is one specific algorithm that is better than the others, since each real world problem is also unique. Also most of the algorithms can be applied to both categories of Supervised Learning. Below we will present some of the most used algorithms of this category. Most of these algorithms have a version for classification and for regression.

Support vector machines SVM [10] are optimization methods for binary classification tasks that map each set of features to a higher dimensional space where there is an optimal hyperplane that divides the existing classes. This decision boundary is chosen with the help of some training examples, called the support vectors, that have the widest separation between them and help maximizing the margin between the boundaries of the different classes. The decision surface is in the "middle" of these vectors. During the training phase, the coefficient vector w and the constant b that define the separating hyperplane are searched such that

the following error function is minimized:

$$minimize \quad \frac{1}{2}||w||^2 + C\sum_{i}^{N}\xi_i \quad s.t \quad y_i(w\cdot\phi(x_i)+b) \geqslant 1-\epsilon i, \quad \forall x_i,\ \xi \geqslant 0 \quad (2.1)$$

where C in the equation represents an arbitrary constant and ξ are slack variables that are used to penalize misclassified instances that increase with the distance from the margin. When the C has a bigger value, the penalty for misclassification is also greater. The vectors x_i represent the training sets, and ϕ represents a kernel function that maps the input vectors into a higher dimensional space. Regarding the classifications of the test sets, they are classified according to side of the hyperplane that they are in.

Decision Trees Decision Trees are a group supervised algorithms that learn rules based on the training data to classify new instances. The built trees are simple to understand and visualize. In the training phase, each node of the tree is recursively split by the feature that provides, for example, the best *Information Gain* at that level.

Random Forests Random forests are ensemble methods, i.e., to make the prediction they take into consideration the result of several classifiers which allows this method to improve its performance. Regarding Random Forests, a set of random trees with bootstrapped samples (samples drawn with replacement) from the initial training data are used. Every tree is constructed with the selection of m random features for the d features available, and then they recursively split the data by the best splits (similar to decision trees). As for the classification of the new information its made by doing the majority vote among all the trees in the forest.

Naïve Bayes The Naïve Bayes Classifier is a probabilistic model based on the Bayes theorem. For each class C_k the posterior probabilities are calculated and the class with highest value is selected to classify the set of features that is represented by the vector x. The likelihood, prior and evidence probabilities must be calculated by the following formulas:

$$posterior\ probability = \frac{prior\ probability \times likelihood}{evidence} \quad (2.2)$$

$$P(C_k|x) = \frac{P(C_k)\cdot P(x|C_k)}{P(x)} \quad (2.3)$$

The most difficult part is to compute the likelihoods $P(x|C_k)$, since the other factors are obtained directly from the data. $P(C_k)$ are the prior probabilities and the evidence $P(x)$ is a normalization factor which sometimes is omitted. Since the Naïve Bayes model assumes the independence among the features, the likelihoods can be easily obtained, thus reducing the cost and complexity of the calculation, as we show bellow:

$$P(x|C_k) = \prod_{i=1}^{n} P(x_i|C_k) \tag{2.4}$$

Logistic Regression Logistic regression is a regression model where the dependent variable is categorical, and the independent variables can be continuous an categorical.
The model consists of a vector β in a d-dimensional feature space. This vector represents the regression coefficients of each feature. β_0 is called the intercept value. Using the equation bellow, the model maps an array of features x to a real number z.

$$z = \beta_0 + \sum_{i=1}^{d} \beta_i x_i \tag{2.5}$$

After that it maps z to its correspondent probability p, using the logistic function as we can see below.

$$p(z) = \frac{1}{1 + e^{-z}} \tag{2.6}$$

Finally, this probability can be interpreted as a given set of features belonging to a class, for example in a binary classification we can use it as, if $p \geqslant 0.5$, the set of features will be classified to the class 1, and if $p < 0.5$ it will be classified to the class 0. In the case of where there are more than 2 classes, the regression coefficients are calculated for each one of them, and the final predicted class will be the one with the highest probability.

2.2 Classification metrics

Classification metrics are very important in order to evaluate the performance of an algorithm. Thus for better understanding the classification metrics explained in this section we must first detail the meaning of true positive (TP), false positive (FP), true negative (TN) and false negative (FN).

True positive - When a given class is correctly identified.

False positive - When a given class is incorrectly identified.

True negative - When a given class is correctly rejected.

False negative - When a given class is incorrectly rejected.

In Table 2.1, we can see an example regarding the class C. In the case if we have an object of the class C and the classifier predicts as C, then it is a True Positive. If we have an object of the class C and the classifier predicts it as another class (not C), then it is a False Positive, since it was incorrectly identified. If we have an object that is not from the class C (not C), and the classifier identifies it as C, then we can say that the object was incorrectly identifies, which is a False Positive. Finally if we have an object that is not C, and the classifier identifies it as not C, then we can say that it was correctly rejected, which is a True Negative for the class C.

Table 2.1: Classification example (Confusion matrix) regarding the class C

Predicted class / Correct class	**C**	**not C**
C	TP	FN
not C	FP	TN

2.2.1 Metrics

In order to evaluate the performance of the classifiers, there are many metrics that can be used. These metrics are essential to understand the results obtained in other studies presented in the chapter 3 and the results of this work, presented in the chapter 5 of this report. Below we present the most relevant metrics that are used to evaluate the performance of a classifier.

Accuracy - This metric measures the percentage of correct predictions (True Positives and True Negatives) of the algorithm across the whole testing set.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.7}$$

$TP =$ True Positives of a given class $\quad TN =$ True negatives of a given class
$FP =$ False Positives of a given class $\quad FN =$ False Negatives of a given class

Precision - This metric measures the proportion of correctly classified instances (True Positives) among all the positive instances classified in a class (True Positives and False Positives), by other words it means if the classifier gives a positive prediction, what is the probability of being correctly classified.

$$Precision = \frac{TP}{TP + FP} \tag{2.8}$$

$TP =$ True Positives of a given class $\quad FP =$ False Positives of a given class

Recall - This metric measures the percentage of correctly identified instances of a class, in other words among all the instances of a given class, what is the percentage of instances that the classifier was able to find.

$$Recall = \frac{TP}{TP + FN} \tag{2.9}$$

$TP =$ True positives of a given class $\quad FN =$ False negatives of a given class

F1-Score (also known as F1, F-score and F-measure) - This metric combines precision and recall and provides a break-even between them. It is calculated as the harmonic mean between the two metrics (Precision and Re).

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{2.10}$$

2.2.2 Micro averaging vs Macro averaging

The metrics showed above for *Recall* and *Precision* are applied to each class, but there are times when we want to use them to evaluate to whole model. There are two ways that we can make that, the first one is using Macro Averaging and the second one is using Micro Averaging.

Micro Averaging - This technique uses the total number of True positives, True Negatives, False Positives and False Negative of all the classes to calculate the metrics that we showed in the previous subsection. Micro Averaging is mostly used when we just want to know the global performance of the model, even if the classes are completely unbalanced (number of instances of each class is very different).

$$Precision^{\mu} = \sum_{i=1}^{n} \frac{TP_i}{TP_i + FP_i} \tag{2.11}$$

TP_i = True Positives of class i $\quad$ FP_i = False Positives of class i

$$Recall^{\mu} = \sum_{i=1}^{n} \frac{TP_i}{TP_i + FN_i} \tag{2.12}$$

TP_i = True Positives of class i $\quad$ FN_i = False Negatives of class i

Macro Averaging - This technique uses the metrics calculated for each class, and then computes their average. Macro averaging is mostly used when we have an unbalanced dataset, giving us the global performance of the model, but unlike Micro Averaging it takes into account the classes with small number of examples, giving them an equal importance.

$$Precision^{M} = \frac{\sum_{i=1}^{n} P_i}{n} \tag{2.13}$$

$P_i = Precision$ of class i $\quad$ n = number of classes

$$Recall^{M} = \frac{\sum_{i=1}^{n} R_i}{n} \tag{2.14}$$

$P_i = Recall$ of class i $\quad$ n = number of classes

2.3 Text Preprocessing Techniques

Text preprocessing techniques play a very important role in text mining, that will be explained in the next section. These techniques consist in preparing the text to be further analyzed in order to extract useful information from it. There are multiple ways that we can prepare the text, either is by reducing the size of the text, such as Tokenization, Stop Words removal and Stemming.

2.3.1 Tokenization

Tokenization consists in extracting the tokens from a piece of text. These tokens can be words, symbols, isolated letters, etc.
For example the phrase , "I ate a hamburger" is divided in the following tokens, "I", "ate", "a" and "hamburger".

2.3.2 Stop Words Removal

The goal of this technique is to remove word from the text that usually don't add useful information to it, such as 'are', 'and', 'that'.

2.3.3 Stemming

Stemming [11] is the process of transforming a word into a common representation by removing the suffixes and prefixes.
For example, the following words, "presented", "presenting" and "presentation" could be reduced to "present" which is a common representation of the words above.
However there are two big types of errors that can occur in stemming which are:

- Over stemming - When a pair of words with a different stem are stemmed to the same common word.
- Under stemming - When a pair of words that theoretically should be stemmed to same common word, but instead is stemmed to a different word.

2.4 Text Mining

Text mining is the act of extracting useful information from text documents. This information might help us to find common patterns through these text documents, so that we can use them as a feature while building a prediction model, as has been done in other studies [6, 7, 8]. There are different types of techniques regarding text mining that are described in the subsections below.

2.4.1 Bag of Words

This technique consists in counting the number of words of a text document according to a given dictionary. The process of counting words allows us to find correlations between different documents such as topics. For example as we can see in the Table 2.2, where we have a dictionary consisting of 4 words ("car","auto","insurance","best") and their corresponding number of occurrences on each one of the 3 documents. Once we have these occurrences we can normalize them and use them as as features in our model.
Since BOW provides these word counting, some studies used it in order to find Twitter's trending topics [12, 13].

Table 2.2: BOW example

Word	**Document 1**	**Document 2**	**Document 3**
car	27	4	24
auto	3	33	0
insurance	0	33	29
best	17	0	17

2.4.2 Term Frequency- Inverse Document Frequency

TF-IDF is another technique to represent a text document by its words. The goal of TF-IDF is measuring the relevance of a given word and not its frequency. The first step of the algorithm is obtain the number of occurrences of a word in a document. Then we multiply it by log of total number of documents divided by the number of documents that contain that word. This technique allows us to get rid of words that are common to the most part of the documents and found the most relevant ones, which can be used to summarize these documents or tweets [14]. This technique is also used in finding Twitter's trending topics [15].

$$w_{i,j} = tf_{i,j} \times log(\frac{N}{df_i}) \tag{2.15}$$

$w_{i,j}$ = weigth of the word of i in document j
$tf_{i,j}$ = number of occurrences of the word i in document j
df_i = number of documents containing the word i
N = total number of documents

2.4.3 Part-of-speech tagging

This technique consists in assigning the parts of a speech(adverb, adjective, noun, verb...) to every word in a text.
For example for the phrase "Today is Sunday", the results should be "Today" and "Sunday", as noun and "is" as a verb.

2.4.4 Named Entity Recognition

As the name suggests this technique consists on recognizing different types of entities present in a text. This entities can be such as People, Places, Companies, Brands etc. A simple example is considering the following sentence: "Donald Trump is the new president of United States of America", once we apply a NER algorithm to this sentence we can get a result as we can see below.

People - Donald Trump
Place - United States of America

This technique was also used in some researches involving Twitter datasets [16], [17].

2.4.5 Sentiment analysis

Sentiment Analysis is another technique that is used to extract information of a document text. It was used in multiple Twitter studies [8], [18], [19], [12]. This technique is also very used to identify a users opinion on a given topic. It consists on identifying the sentiment value, that can be: positive, neutral or negative, but depending on the approaches these values can be translated into numbers, for example a positive value divided in a range 1 to 5. The process of sentiment analysis can use emoticons (i.e. ":)", ":(", ":D"), and words, of a text as a feature in order to classify its sentiment value. For example words like happy can indicate a positive sentiment associated with the text, and words like sad, can indicate a negative sentiment.

2.5 Conclusions

In this chapter we first started by introducing some basic concepts of ML. Followingly we discussed some metrics regarding the evaluations of ML classification algorithms. Then we introduced concepts about text mining that will be used later in our work.

Chapter 3

State of the art

In this chapter we will present a review of the state of the art regarding aspects that affect retweeting, Retweet Predictive Models and Twitter recommendation systems. We will also make an overview of the current frameworks, APIs, and tools for Twitter.

3.1 What does influence the retweets?

Predicting the number of retweets can be a hard task to do, since there are multiple factors that play a role. Many studies were carried out to identify these factors and their impact on retweeting.

An exploratory analysis was done by Suh et al.[20] that studied how different features influence the retweet rate (average of retweets per tweet). The authors used Principal Component Analysis in order to find the correlation between features, which showed that there are 3 groups of features that are related among them. For the first group, the features that are highly correlated are the number of followers, number of followees and number of statuses, which have a positive impact in the number of retweets. In the second group, the features are URLs and Hashtags which positively influence the number of retweets, and number of mentions in a tweet which has a negative impact in the number of retweets. The last group separates two different kind of users, the ones who have a high number of retweets and followers and the ones who have a lot of tweets and favorites. The first kind of users have positive impact on retweets while the second has a negative impact. In this study they also built a Logistic Regression model to analyze the influence of the features in the retweet probability. These results showed that the number of followers of the tweet's author is the most influential feature, followed by the number or URLs and Hashtags, which in part was confirmed by Zhang et al.[7]

when analyzing the *Information Gain* of the features for their predictive model, where the number of followers and number of times the user was listed were the most meaningful ones.

In another study, Cha et al.[21] used a dataset with users that only obtained 10 or more retweets. This study analyzed 3 characteristics that influence the number of retweets, which were: number of followers, the number of retweets of a user, and the number of mentions that contain a user's name. The results showed that the users who obtained a lot of mentions were also the ones who got often retweeted, which were public figures and news sources. This can be related to another analysis made by Petrovic et al.[6], were it was showed that 96% of the tweets of verified accounts (celebrities, politicians) get retweeted, comparing to only 6% of non-verified accounts. This study also contradicted other studies [20] [8] that were made, because since it says that the number of followers may not be the most influential feature, not all the users who have thousands of followers are either active or involved with the Twitter's community.

Kwak et al.[1] compared the top 20 Twitter users with the highest PageRank with the top 20 most followed users and the top 20 users who had most number of retweets. PageRank is an algorithm that measures the importance of a web-page (in this case adapted to a user), through the quantity and quality of links associated with it. When comparing the first 2 rankings they found that 18 out of 20 users were the same in the 2 lists. But once they compared the first 2 lists with the one which had the most retweeted users, there were very few users who were in the first 2 lists and in this last list. This can also lead us to conclude, like Cha et al.[21], that the number of followers may not be the most influential feature.

Sentiment analysis is also proved to be a good feature to use Jenders, et al.[8]. This study used a framework called SentiStrenght to analyze the sentiment value of a message, this framework returns the positive and negative sentiment score for the message (from 1 to 5 and from -1 to -5, respectively). These values were used to calculate emotional divergence (that was introduced by [22]) of the message. Emotional divergence is defined by the following formula:

$$\frac{positive\ score + |negative\ score|}{10}$$

It showed that tweets with higher emotional divergence have a higher retweet probability. Another metric regarding sentiment analysis was also used, which was the *Sentiment Valence* which indicates whether the message has predominantly a positive, neutral, or negative sentiment. They found that tweets with a negative sentiment valence have a higher probability of being retweeted.

Besides sentiment analysys, in the study, the authors showed that the average number of retweets grows with the number of followers. This makes perfect sense, because when the number of followers is higher the tweet itself gets more chance to be seen by other users, which makes it easier to be retweeted.
In this study they also analyzed the length of the tweet message. The authors state a few interesting hypothesis that when length of the tweet grows it means that it is more interesting so it will have more retweets. However when analyzing the results for the average number of retweets per length of the tweet message it showed that when the length of tweet is greater than 120 characters the average number of retweets tend to decrease. The explanation that was found for that is because only relatively new users try to use all of 140 characters available for a tweet and the most experienced ones use abbreviations. This study also showed that the average number of retweets regarding the number of Hashtags used in a tweet differs. For example for the tweets with one Hashtag the study showed an average of nearly 150 retweets, and for tweets without Hashtags the average was only half of that amount. As for the tweets with more than one Hashtag the average of retweets decreases as the number of Hashtags in the tweets increases.

Regarding the influence of a photo in a tweet, Zhao et al.[23] studied the influence of images on the Chinese social network called Weibo, which is similar to Twitter. This study showed that among 111 million collected tweets (43.3% pure text tweets), the ones that got retweeted 1000 or more times, 65.8% contained images, 11.5% contained an URL, 16.4% contained both and only 6.3% were pure text tweets. This study can be related to Twitter since some statistics in [24] show that tweets with images have 150% more retweets, but these statistics do not say how much data they used.

To summarize this section, we saw that among different factors that affect the number of retweets, multiple studies point to the fact that the number of followers is the principal one, which is comprehensible because users with more followers when they tweet their message is seen by more users than those with a few number of followers. Other factors like the number of mentions that a user got also are interesting, since when there are a lot of tweets that reference someone means that person is either very popular or it's involved with the Twitter's community. As for the text characteristics, the Hashtag seems to be the most relevant factor.

3.2 Retweet Predictive Models

Currently there are some approaches to identify the number of retweets. This number of retweets can be continuous or contained in a class (i.e. belonging in a certain range of values), the so called classification approaches. In this section we will describe the classification models that used ML algorithms, which aim to identify the class of a tweet. We will also briefly review other 2 approaches that are Statistical and Collaborative Filtering which differ from the classical classification approaches.

3.2.1 ML Classification models

Since in our model we will use a classification approach, we decided do detail these models in order to identify what can be done to improve the results obtained by these studies and discuss them in the conclusion section of this chapter.

In 2011 Petrovic et al.[6], proposed to predict if a tweet is going to be retweeted or not. They used a dataset of 21 million tweets gathered during the month of October of 2010. This data was splitted in 90% for training and 10% for testing. To make the predictions, two models were built. Both of the models used a *Passive-Aggressive algorithm*, which whenever it receives a new tweet from the dataset "it tries to classify it correctly with a certain margin, while keeping the decision boundary as close as possible to the old one". In the first model the authors assumed that the tweets are sequential through the dataset. In the second approach taken by the authors a *Time-sensitive* modeling technique was performed. This means that for every hour there is a separated model that was trained with the tweets that occurred in that hour. They built this second model under the assumption that each hour of the day has different trending words.
The features used in both models can be divided in two parts: **Social features related to the author of the tweet**: number of followers, friends, statuses,favorites, number of times the user was listed, is the user verified, and is the user's language English.
Tweet features: number of Hashtags, mentions, URLs, trending words, length of the tweet, novelty, is the tweet a reply, and the actual words in the tweet.
Another important part in this article is that there were conducted social experiments. These experiments tested how well a person could perform on identifying if a tweet will be or will not be retweeted. But only 2 persons were used to do this classification task on 202 pairs of tweets. The accuracy in these experiments was 76.2% and 73.8% respectively, when only the text was showed to the persons, and 81.2% and 80.2% when the text and social information about the user who tweeted was showed. Comparing these results with those obtained using a Time-sensitive

model which were 69.3% and 82.7%, it shows that the approach can make predictions as well as a human. Although the accuracy values where high, the macro F1-score values of both models were poor, 37.6% for the first model and 46.6% for the Time-sensitive model. Knowing that in a real world most of the tweets do not get any retweets, we can imply that these accuracy values were high due to the high amount of tweets with no retweets that are in the dataset. And the low values for F1-score are due to the fact that these models most of the times identify the tweets that are retweeted as tweets that are not retweeted.

Later in 2012, Zhang et al [7], used a feature-weighted approach, in this approach some extra features were introduced; the **Social features** added were: number of account days and length of the user's screen name. The added **Text features** were: number of characters, number of words, the time when the tweet was posted. Although they added these features, some of them were left out like the words itself and the number of trending topics. The authors built 3 models, the first one used Logistic Regression Classifier, the second one used a SVM Classifier, and the third one also used a SVM Classifier. The difference was that for this third model (feature-weighted) the weights of the features were previously calculated before initializing the model, using *Information Gain* to measure the impact that each one of them has on retweeting, instead of automatically adjusting the weights of the features during the training process. On the contrary of what was done in [6], this study grouped the number of retweets in 4 classes. The first one contains the tweets which did not get any retweets. The second class represents the tweets which got more than 0, and less than 10 retweets. The third class represents the tweets which got more than 10, and less than 100 retweets. Finally the last class represents the tweets that got at least 100 retweets. This study also showed that among the 10 million English written tweets, 88% belonged to the first class.
Comparing the results of the 3 models, the third one got the better results, 76.58% for precision, 65.34% for Recall and 70.56% for Macro-F1, while the first and the second got respectively 60.16% and 70.81% for precision, 52.37% and 60.26% for recall and finally 55.99% and 65.11% for Macro-F1.

Following these two studies, Janders et al.[8] used Sentiment analysis in his work as was suggested by [1]. He proposed to identify if a tweet belongs to a given interval controlled by a certain Threshold that separates viral tweets from non viral tweets. The used Threshold values were: 50, 100, 500 and 1000. There were two models that were built one that used Naive Bayes and another that used Logistic Regression. Both of these models were trained with all of the Thresholds mentioned above, for each one of them the authors extracted from their dataset 20 thousand tweets with equal number of samples for viral tweets and non-viral

tweets. The feature values used in these models were discretized because the Naive Bayes model can only learn when the values of the features are often repeated, and some tweet features can have a large range of values such as number of followers, which means for example that instead of using the actual number of followers, this feature was divided in 3, the first one is when the number of tweets is less than 10 thousand, the second one is when the value is between 10 000 and 300 000 and the last one is when the number of followers is greater than 300 000. The features used here were similar to the ones used by [6], although as in [1], features related to the words itself were left out.
Regarding the features that were introduced here related to sentiment analysis of the tweet, **these features were**: positive sentiment, negative sentiment , emotional divergence and sentiment valence of the tweet. Analyzing the results, the macro F1-score values for NB model were 91.6%, 92.7%, 94,7% and 95.1%. For the Logistic Regression model the results were 93.6%, 94%, 96.3% and 96.8%. The order of these results correspond to the Thresholds 50, 100, 500 and 1000. Discussing these results the authors state that the Logistic Regression model got a better performance because it does not assumes conditional independence among the features. Since from the Logistic Regression model the authors can extract the weighs of the features, which means the importance that the feature has in the prediction. The most important feature was the number of followers, following by the number of mentions and URLs.
Although the performances presented by these models were good, there are some major drawbacks. The reasons are the small amount of tweets used in the experiments and the balancing of viral and non viral tweets on the testing dataset. This might lead to better results but it won't represent the reality of Twitter data distribution since the real data is not balanced.

To finalize this subsection, we showed the approaches of 3 different studies. Among these studies most of them used in majority the same features, such as number of followers, statuses, friends and number of times the user was listed regarding the user information. For the text characteristics the most common features were, the number of Hashtags, URLs, mentions and length of the text. Other features we also uses regarding the text, such as number of trending topics and features regarding sentiment analyses, but none of the studies applied these two techniques together.

3.2.2 Other approaches

In this subsection we will briefly show other Retweet Predictive Models that are not based on the traditional ML. These models can be such as statistical, or based on collaborative filtering which is used in recommendation systems.

A statistical approach was conducted by Zhao et al.[25] to predict the final number of retweets through time, using information cascades. This model was based on the theory of *self-exciting point processes*. A point process is a counting process which in this case counts number of retweets through time of a given tweet. Self-exciting processes assume that the future evolution of the process is influenced by the previous instances, which in this case is the number of retweets. To predict the spreading the authors used the intensity of a tweet (i.e. expecting number of retweets in the next unit of time). Intensity is calculated with rate of viewing and with the infectiousness of a tweet (i.e. the retweet probability). It is important to mention that this model only requires as a features the number of followers of the user who posts the tweet.

In another study Zaman et al.[26] implemented a collaborative filtering approach to predict the probability of a tweet being retweeted by a follower. The authors used the social features (name, number of followers, number of followers) of the user who tweeted the tweet and also the social features of the users who retweeted the tweet. It was also used the positive and negative feedback, which was obtained using the active users (followers of the user who tweeted) that retweeted (positive feedback) and do not retweeted (negative negative feedback) the tweet. The biggest flaw of this approach was that the authors used as training data the first hour after the tweets which when usually most of the retweets happen [25].

To conclude we presented two different approaches to predicting the number of retweets, where we observed that both of them predict the number of retweets using the information of the tweet's behavior, instead of making the prediction with a set of features previously gathered before the time of the tweet.

3.3 Tweet Recommendation Systems

Many studies investigated recommendation systems in Twitter. These systems use features such as the relationship between the user that receives a tweet and the author of the tweet. They also use the interest of a user in a given topic and information related to the tweet itself. So unlike retweet prediction models that we showed in the section 3.2, which try to directly identify the "number" of retweets of a tweet, mostly through direct features (i.e. without exploring the followers and followees) extracted from the tweet itself and the social features of the users. These recommendation systems focus on personalized retweet probabilities (i.e. the probability of a given user retweet a certain tweet).
The problem here is that it is difficult to build a model for all the Twitter's

data, because if we want to analyze all the users, their tweets, their followers and friends, we will end up with huge amount of data. Yan et al.[27] presented a graph-theoretic model for tweet recommendation system, in this study they started from collecting the information about 23 users, and expanded all their relations with other users resulting in 364 million tweets, 56 million retweets and 9 million users. Other authors studied these systems, Uysal et al.[28] made a recommendation system for retweeting, and targeted users. This study used 2 approaches, the first one is ranking the incoming tweets. For each one of the users, the tweets are ranked according to their probability of being retweeted. For the second approach that is ranking targeted users, for each tweet, the users are ranked according to the probability of retweeting that tweet, in other words it is the probability if followeres who receive a tweet of a user to retweet it. Similar to it Chen et al.[29] presented a collaborative ranking model, for recommending useful tweets, which used collaborative filtering based recommendation by collecting preferred information from many users.

3.4 Technologies overview

Since Twitter is one of biggest social networks, there are many tools and frameworks that allow the user to gather information about it and process it. In the following subsections we present two type of technologies. In the first subsection we show how the Twitter data can be obtained. In the second we show some tools that allow to process the text from the tweets.

3.4.1 Twitter APIs

Currently Twitter offers a list of APIs that allows us to access multiple Twitter data. This data can be, information about tweets, user profiles and the posts that a user made. There are different types of APIs according to their purposes, bellow we will introduce the two most common ones.

Twitter Search API

The Twitter Search API[1] allows a user to access twitter data that went through the Twitter's network. The API allows to search through this data by, specific keywords, user name, Hashtags, location, and many others . Basically the search API allows us to search like the twitter search bar, but the API is used to gather the data, and search bar is used to navigate through the tweets (on the browser or mobile app).

[1] `https://dev.twitter.com/rest/public/search`

Although we can gather all this information, there are rate limits imposed by Twitter. Regarding to searching tweets of a user, the maximum amount is 3200 tweets, if the search keyword is different than a user name, for example an Hashtags the maximum amount of tweets is 5000. Besides this amount limit, Twitter Search API also has a limit regarding the number of searches that a user can make in a period of time. These limits currently are 180 in a time window of 15 minutes. There is also another important aspect is that this API filters the tweets by relevance instead of completeness which means that it might show more important tweets over less important.

Twitter Streaming API

Twitter Streaming API[2] gives the possibility to gather the real-time information that is going over Twitter's global stream of Tweet data, which is different from the Search API since only old data could be accessed. In this API the user can collect the tweets according to different criteria (Hashtags, locations, user names and many others), which means that if a tweet belongs in the criteria it will get collected by the user.
Although this API does not have rate limits, there is a big problem associated with it, since the percentage of tweets that are retrieved depends on the amount of tweets found according a given and the amount of Twitter's network current traffic. It is estimated that this amount is near 1% [30] of the actual real-time tweets.

3.4.2 Twitter text analysis tools

One of the challenges in Twitter is the extraction of useful information of the tweet message. A significant amount of times the text presented in a tweet is noisy. This makes the task of text mining more difficult in this type of data. Putting that in mind, multiple tools have been made specifically to deal with this noisy text, which usually contains emoticons and has many mistakes, such as not following the grammar rules, besides that, this noisy text often provides little context. Bellow we present some of these tools and what they can offer us.

TwitterNLP

TwitterNLP[3] is a Python library that offers a Natural Language Processing pipeline for performing Tokenization, POS, Chunking and NER. This library was built with

[2] `https://dev.twitter.com/streaming/overview`
[3] `https://github.com/aritter/twitter_nlp`

features obtained from Twitter-oriented POS taggers. In order to improve the detection of NER, this tool uses entities from the FreeBase, which is a knowledge base. Looking inside of the NER categories, this framework offers the following entities, person, Geo-location, company, facility, product, band, movie, TV-show and others.

TweetNLP

TweetNLP[4] is a Java tool that provides a Tokenizer and a POS Tagger with the available models, trained with Twitter data, manually processed by its authors. Besides that, TweetNLP also has the functionalities to identify mentions, URLs and emoticons inside a text.

TwitIE

The TwitIE [5] pipeline is a plugin for the GATE framework, which is an architecture and flexible framework used to design and develop natural language applications. This plugin was developed for micro-blogging text and its main task is to extract named entities from tweet messages due to the typical characteristics of this type of messages that we have mentioned before. TwitIE reuses a set of components from GATE, like the sentence splitter and the gazetteer lists (lists of cities, organizations, days, etc), but for Tokenization, POS tagging and NER it adapts to the Twitter's kind of text. Like TweetNLP, TwittIE can also identify mentions, URLs and emoticons.

SentiStrength

SentiStrength [6] is a tool that was specifically developed for sentiment analysis of short messages from social media, such as tweets. The tool is capable to process multiple language like English, Spanish, Dutch, Russian, Portuguese and many others. SentiStrentgh takes as an input a simple text message and returns the positive and negative sentiment score. For positive score the values go from +1 to +5 and for the negative score the values go from -1 to -5.

3.5 Conclusions

In this chapter we started with a analysis of the features that most affect retweeting. Here we observed that many studies state that the number of followers is the

[4] `http://www.cs.cmu.edu/~ark/TweetNLP/`
[5] `https://gate.ac.uk/wiki/twitie.html`
[6] `http://sentistrength.wlv.ac.uk/`

most influential feature, what is understandable, because the tweet will be exposed to more users who can retweet it. This automatically increases the range of users that can see this tweet. Although many studies state that number of followers is the most influential one there are some disagreement between authors, since some state that the number of mentions that a user gets has a higher influence, it implies that he has a better relationship with his followers and the general public. Other factors like number of, Hashtags, user mentions and URLs in a tweet were also stated in many studies as having a high influence on retweeting, since they add information to the tweet.
Secondly we presented the ML classification approaches to retweeting, where we could observe that as was expected the most influential features were used in these models, but the number of mentions that a user has wasn't used in any of these models, since to obtain them it is necessary to get all the tweets that mention a given user. Other features were also used related to sentimental value of the tweet, and the most frequent words of the tweet. However none of these models makes a specific analysis to the content of the text (only individual words [6]), such as, if the text is related to a trending topic, since it may help the tweet to be seen by others that are not the followers of the user who tweeted, which increases the probability of being retweeted. Also as Zhang et. al [7] said, no work have been done of text content related to trending topics and sentiment analysis. Other case of possible features that were not used in these models, like presence of images, which was showed in one of the studies and in some Twitter statistics (that tweets with images has a higher probability to be retweeted). To continue none of the models applied NER, which might be interesting do analyze.
We also briefly showed that there are other ways to predict the number by showing a statistical model and a collaborative filtering approach. The problem here was that both of them needed to observe the tweet for some time before making the final prediction.
Then we reviewed some tweet recommendation systems which suggest tweets to users. These recommendations are personalized, since they use the relations among the user and the author of the tweet, the interests of the users and other features related to the tweet itself and the author's authority.
To conclude, after analyzing the recommendation systems we did a brief technology overview related to Twitter, where we showed some technologies that allow us to extract data from Twitter as well as extract information from text contained in the tweet.

Chapter 4

Proposed approach

In this chapter we start by describing the problem that is addressed this work. After that we will make a description of our Retweet Predictive Model. Finally we will describe some key aspects of our work, like gathering and aspects of the data, how we built our database and the features used in our work.

4.1 Problem definition

Predicting the number of retweets is an hard task, since there are many factors that affect it. As we showed in the chapter 3, mainly if a given user wants to tweet a message which will become more popular, he has to have a high amount of followers.
We find that predicting the popularity of a tweet inside lower ranges of retweets would be far more interesting than addressing the task focusing on more popular tweets which will naturally belong to the users with a higher amount of followers.
We formulate the problem as a multi class classification, although some studies [6, 8] looked at this problem as a binary classification, by that we mean predicting if a tweet will have more or less retweets than a certain Threshold. Performing multi class classification will allow us to simultaneously look and compare tweets from different "ranges" and understand how the model behaves in each one of them. We defined 4 classes as already was done by Zhang [7] , each one of them representing a different range of number of retweets. We aim to test different text features (number of Hashtags, number of URLs, and length of the text and if a tweets is a reply) which allows us to observe how the performance of the model changes regarding different values of these features values among the different levels of popularity. With this we can find relations between these values of the features and the different classes of tweet popularity.
In the Table 4.1 below we present the 4 classes that we used to represent the

popularity of a tweet. The first one represents the tweets that do not have any retweets, this class is the dominant one across all the datasets that we will later describe. The second class represents the tweets that have between one and 10 retweets in other words this class represents the tweets with few number retweets. For the third class we selected a bigger interval that represents all the tweets that have a range of retweets between 10 and 100. Finally for the last class it represents the tweets that have 100 or more retweets.

Table 4.1: Range of retweets of each class

Class	**Range**
1	[0 retweets]
2	[1, 10 retweets[
3	[10, 100 retweets[
4	[100, + ∞ retweets[

4.2 Retweet Predictive Model

In the first stage we collected data from the Twitter's network is collected (1). In this case the collected data is from the Twitter Streaming API for the first dataset, and from Twitter Seach API for the second. Besides that we built a WebCrawler to gather data regarding Twitter's past trends. After collecting all this data we inserted it into a database (2). We will explain in more detail these first two steps in the sections 4.3 and 4.4. The data collected from the datasets and inserted into the database, already has most of the features used in this work and in studies of other authors. Regarding the feature extraction part (3), in this case we used a Sentiment Analysis tool, we had to extract them from the text that was previously stored. Regarding feature selection we selected manually some features as already was done in other studies, although we have added some new, like features related to the sentiment value and media content in a tweet (4). After having all the features that we gathered/extracted we have sampled and splitted the data from each dataset into multiple training/testing sets (4). We used this training data to fit it into a ML algorithm (5), in our case we used Random Forests in order to build the model (6). Finally when the model was built with , we tested it with multiple testing sets (7) and finally we analyzed these results for the given testing sets (8).

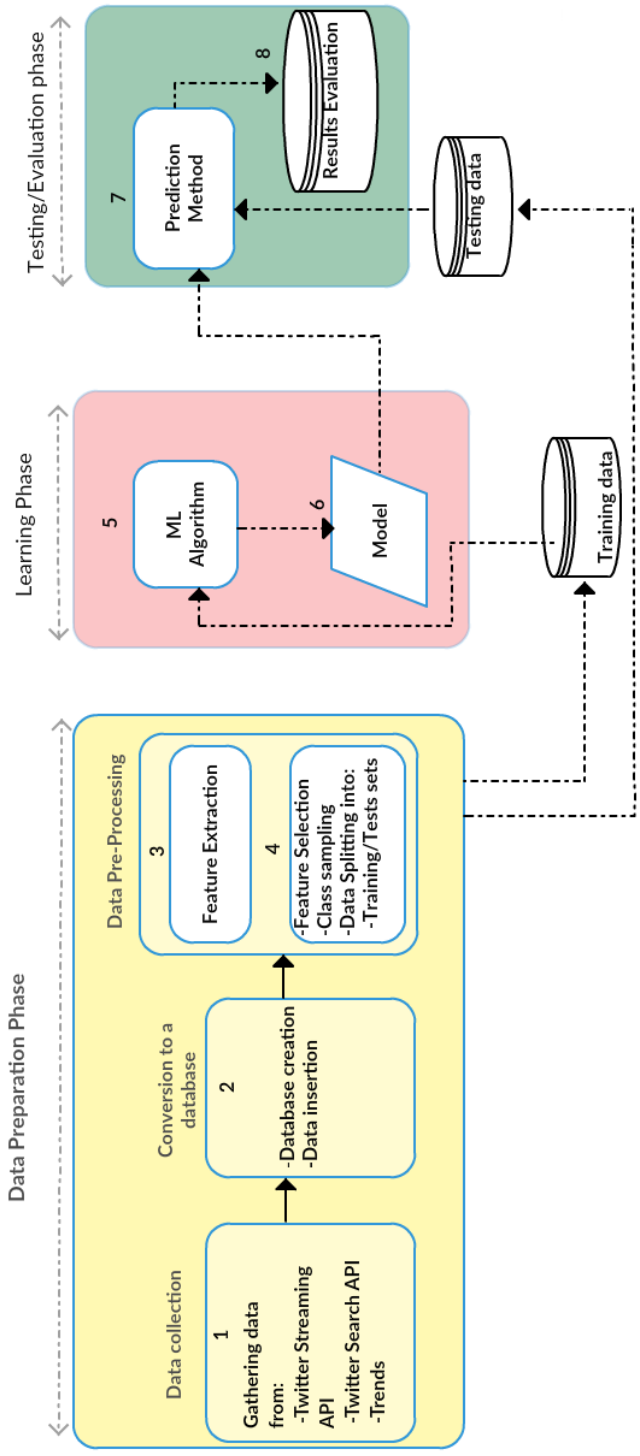

Figure 4.1: Retweet Predictive Model overview

4.3 Data collection

In this section we will describe how we collected our data. Firstly, we are going to show how the structure of the data provided by the Twitter's APIs is organized, and then how we collected our 2 datasets, one through the Streaming API and the other with the Search API. Finally we will show how we obtained the Twitter's trends for these 2 datasets.

4.3.1 Structure of the Twitter's data

When using the Twitter Streaming API and the Twitter Search API, the data that is retrieved by these two APIs comes in a JSON format. Each JSON object is composed by a pair, key and value, the key is the name of a field by which we can obtain its value, for example like the number of retweets, or the name of the user. The value as the names indicates, represents the value of the of the key, so for the key number of retweets, the value can be 0. It is important to mention that the value can also be another JSON object. There are 4 main JSON Objects retrieved by these APIs, the main one is the tweet, and then inside a tweet there are other JSON Objects, like the user, the retweet and the entities. We can see these JSON Objects in a table format, in the appendix A. where we show in each one of the tables the key and the correspondent meaning of the value.

4.3.2 Datasets collection

In a first phase we collected a Twitter dataset, corresponding to the month of July of 2016 from the `https://archive.org/` website which has many of Twitter datasets for different months and years, collected with the Twitter Streaming API. As we said earlier on chapter 3 Twitter Streaming API only provides 1% of all of the current data that is going through the Twitter's network. One of the problems here is that there are some retweet messages that are gathered that its original tweet message was not caught. For example, a user posts a tweet, but the API will stream only a retweet of this message. With this in mind when we later update the number of retweets of a given tweet, the final value might not be very accurate. Putting this in mind, and the fact that we also wanted to analyze the influence of the number of retweets of k last tweets of an user we collected another dataset with the Twitter Search API.
Below we present more information about the class distribution of these 2 datasets.

Twitter Streaming API dataset

Twitter-stream dataset it's the main dataset in our work. This dataset contains the tweets gathered using the Twitter Streaming API. The dataset has 12470144 (English) tweets that go from 1st of July, 2016 until 15nth of July. In the Table 4.2 below we can see the distribution of the tweets among the 4 different classes. It is clearly visible that the number of tweets of each class it's completely unbalanced.

Table 4.2: Number of tweets for each class of the Twitter Streaming API dataset

Class	Number of tweets
1	8684496
2	2276806
3	999511
4	509331

Twitter Search API dataset

In order to analyze the influence of the user's activity regarding his last numbers of retweets and how they affect the predictability for a new tweet that a user posts. We have collected from 153k users contained in the previous dataset, their last 200 tweets using the Twitter Search API. From this 200 tweets we have filtered the data in order to obtain only the pure tweets of each user and excluding the retweets from tweets that a user has previously retweeted. So for each tweet that we have, we also have the number of retweets of each of the user's last 10 tweets.

Table 4.3: Number of tweets for each class of the Twitter Search API dataset

Class	Number of tweets
1	10088923
2	2803163
3	1118919
4	626580

4.3.3 Trends Collection

As we know trends are normally popular topics at a certain time. These topics have an influence across the Twitter network. Trends differ from city to city, from country to country, although there are some trends that are mutual across various countries. The Twitter Search API allows us to get the current trends for any local that we choose, the problem here is that in our case the data was previously

collected and not at the same time as the current trends. Fortunately there is a website `https://trendogate.com/` , that keeps track of these trends for each possible city and local for a given day since 2015-03-01. Another problem is that not every user has the location where he tweets the tweet, in fact only a small percentage has these values. So by observing this, and since our tweets are all written in English we selected the countries where the official language is English, like Canada, USA, UK and Australia. Besides these countries we also selected the global trends for other 3 countries such as India, Indonesia, and Brazil, which are among the ones with more Twitter users. Getting the trends manually from this website it's almost an impossible task to accomplish, if we take into account the combinations of days and places. To mitigate this problem, we built a WebCrawler using a Python's library called BeautifulSoup[1]. This library allows us to navigate through a web site and filter the information that we want, which in this case were the trends. With that we built a script that automatically giving a place and a date returned the trends with specifications and furtherly inserted them in the database.

4.4 Database

In this section we will start by describing how we stored and organized our data. Then we will describe some challenges that we had regarding the data insertion into the database.

4.4.1 Database-creation

An important part that allows us to access the data that we previously gathered is to have it stored in a database. This database must have a scheme that is suitably designed. This can help us to improve the speed of the process when we select the data to use it in our model. Although we have 2 datasets the database scheme is equal for both of them. There is only one difference is that the data is stored in 2 different databases. As we can see on the figure 4.2 we have 4 Tables that represent our database scheme. As we said before there is a JSON Object retrieved by the Twitter's APIs called Retweet, which in this case is only used in the Twitter Stream API dataset. We used this object to insert the tweets that were contained inside of it, as well as updating previous number of retweets of a given tweet. In the case of the dataset collected with the Twitter Search API, the only retweets that appear in it are the ones that were made by a given user which from we have collected the tweets. Therefor since we only wanted the tweets from

[1] `https://www.crummy.com/software/BeautifulSoup/`

the user, we ignored these retweets. The first table is called User, this Table has a primary key "user_id", which is the ID that is assigned to an user by Twitter. Regarding these characteristics we have inserted the following ones, name of the user, description, number of followers, number of friends, number of statuses, number of times that the user was listed, date when the user created the account, all of these characteristics were extracted from the structure of the User JSON Object. For the Table Tweet, it contains all the tweets that we gathered. This Table has a primary key "tweet_id" and a foreign key "user", and the following characteristics of the tweet, number of retweets, number of Hashtags, number of URLs, number of user mentions, type of tweet (if it is a reply or not), number of photos, number of GIFs, number of videos and finally the time when the tweet was created. Regarding the Table Sentiment, it has a primary key "id" and a foreign key "tweet_id". This Table has the following attributes, positive score and negative score of a tweet obtained with sentiment analysis that we will later explain in this chapter. The reason why we had these features separated from the Table tweet, was because to extract them we had to have all the tweets previously inserted, we will explain it later in the section 4.5. For the final Table Trend, it contains all the gathered trends for the all the dates from the tweets in the database. With this information we later can see if a tweet contains any trend.

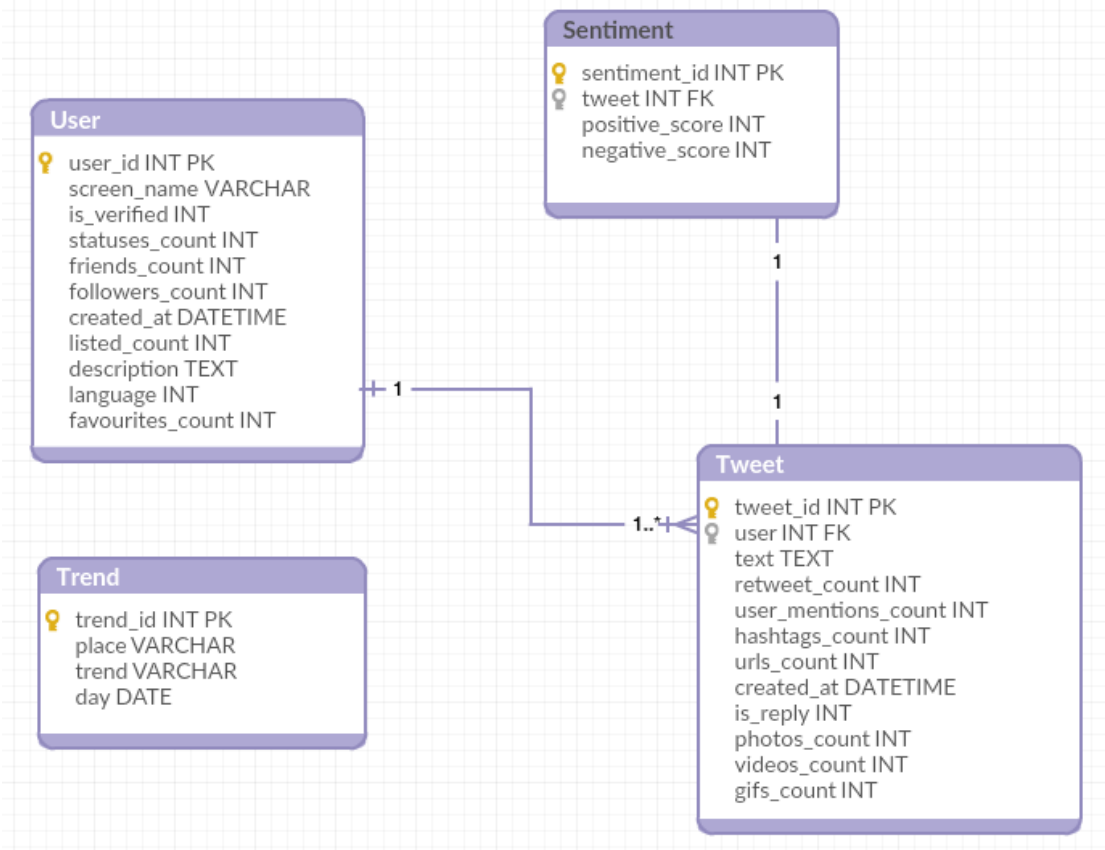

Figure 4.2: Database-scheme

4.4.2 Database-insertion challenges

As we said before we used 2 datasets, the first one was obtained with the Twitter Streaming API, and the other one was obtained with the Twitter Search API, each one of them was inserted into it's own database with equal database's schemes as we said before. For both of the datasets we only inserted the tweets written in English. The main thing to keep in mind is that in the second dataset we may have tweets that were tweeted before 2015-03-01, so it means that we will have to leave them out. This happens because the trends that we gathered are only available since that day. Also since we will study the influence of the past number of retweets of each user in this dataset (from 0 to 10), although we had crawled 200 tweets for each user some of them might be tweets that he retweeted, so we only inserted the tweets from the users that had more than 10 tweets written by themselves.

4.5 Features

Selecting a good set features is an important process to represent a prediction model. The features should allow the model to separate as much as possible the different classes by giving it the necessary information. In the world of ML there are many techniques that allow a person to select the best features (by removing redundancy, irrelevancy) typically when there are hundreds of features and the information known about them isn't much. In our work since we have previous knowledge about our features, due to the studies that were conducted regarding Twitter, we decided to select them manually. In our model we can divide the features in 2 main groups, user features and tweet features. The first one gives us the information about the user, for example the number of followers and followees. The second group gives us information about the tweet message, for example the length of the message, and how many Hashtags are in it. Most of our features already have been used in other Retweet Predictive Models. We will show our features in the subsections bellow.

4.5.1 User Features

User features are the ones o are related to the author of the tweet. We have already mentioned them in the chapter 3, but we will talk about them again in order to explain our selection. These features are the most influential ones, like the number of followers of the users. Is widely known that users who have lots (thousands and millions) of followers can easily get more retweets that the ones who have only a few amount.

Number of followers

This feature is among the most important ones, it indicates the popularity of the user who tweeted the tweet. Number of followers generally has a great correlation with the number of retweets, since the higher the number of followers of the user, any tweet that he will do will have a higher visibility, thus the probability of it being retweeted is higher. Which in other words we can say that more followers means more retweets, as we have already discussed in chapter 3.

Number of friends/followees

This feature indicates how many users a given user follows. Users that have more followers tend to follow less users.

Number of favorites

This feature represents the number of tweets that an user marked as a favourite. Many studies say that the influence of this features provides little benefit.
Suh et al.[20] says that although this feature does not provide much benefit to the the understanding of the retweeting practice.

Number of times the user was listed

This features represents how many lists follow an user. A list allows an user to group other users (by his interests, for example a list that follows basketball teams, photography), which will allow him to see separately the tweets from that group of users on his timeline. An user that is listed a lot means that users are interested in his content, so like the number of followers this feature can also be a good indicator of popularity of a tweet.

Number of account days

This feature indicates how many days has an account. Usually an account gets more followers with time, which logically thinking indicates that an account that is has been on twitter for a longer period of time has a potential to get more retweets.

Verified user

This feature represents whether the user is verified or not. Twitter has a policy that only people such as celebrities, artists, public figures have a verified account. So by that users that have verified accounts tend to have more followers and by having more followers they will have more retweets.

Number of statuses

This feature shows how many tweets a user has on his time-line and in this case the tweets are the sum of his own tweets and the retweeted tweets.

4.5.2 Tweet Features

Tweet features are the ones related to the tweet message itself. These features are also important. Like we did for the user features, some of these features already have been explained on chapter 3, but we will explain them one more time.

Hashtags

Hashtags already were shown in chapter 3 as having a good correlation with the number of retweets. Since tweets with Hashtags are more likely to get more retweets, although as the number of Hashtags grow the number of retweets it is expected do decrease.

URLs

Other features like URLs impact the number of retweets but it may differ from the URL itself, since different domains may differently affect the number of retweets. In our model we only consider these feature as the number of URLs that the message contains.

Mentions

This feature represents the number of mentions in a tweet. Typically tweets with mentions are directed to an user, such a reply or a tweet specifically addressing to an user.

Text Length and number of words

These features, respectively indicate the length and number of words in a tweet. The main reason why we divided it in these two features, is because we can have a text with 140 characters and 20 words, and a text with the same number of characters and 40 words. This can indicate that the first text uses more robust words, and the second one uses shorter words.

Reply

Reply represents whether the tweets is a response to another tweet or not. Typically when a tweet is a reply on average it will have less retweets than a normal tweet.

Hour of the tweet

This feature indicates the hour when the tweet was tweeted, different hours have different numbers of tweets and usually tweets that were made between 11 PM and 12 PM are the ones which got more retweets [31].

Visual Content

Visual content to our knownledge has not been explored yet in Retweet Predictive Models. Studies [32] say that people tend to engage more when there is an image, or a video in a given post rather than a solo text, which can keep their attention to further read it. This can lead furtherly to a reetweet. So by saying that we will added into our model the following 3 features which can be obtained directly from the characteristics of the message through its JSON Object as we previously said.

- **Videos** - This feature measures how many videos are in a tweet.
- **Photos** - This feature measures the number of photos in a tweet.
- **GIFs** - This feature measures the number of GIFs in a tweet.

4.5.3 Extra tweet features

In our model have used some extra features related to the content of the tweet. These features are related to the Sentiment of the message, and a binary feature that represents if a tweet contains a trending topic.

Sentiment features

Sentiment features represent the sentiment associated with a text message, which can be classified in many ways, such as positive, neutral and negative. These features already have been applied in a previous studies as we showed on chapter 3, where they already showed that negative tweets have higher probabilities of being retweeted. To extract the sentiment values from the text we used a framework called SentiStrenght that we have already mentioned in chapter 3. In this case we had to select the text from each tweet in our database, in order to extract these values, since the framework takes as an input a text file (one line per text message), and returns a text file with the positive sentiment and the negative sentiment (for each line in the text). Hence we defined for each tweet the following two features:

- **Positive sentiment score** - Positive sentiment of the tweet from 1 to 5.
- **Negative sentiment score** - Negative sentiment of the tweet from -1 to -5.

Trends

Trends are essentially popular topics on Twitter, which can be an Hashtag or a given word. As we previously said trends may differ from different locations and

days, but the main problem here is that only a few users show their location when they tweet, so in this case since our data only contains English tweets, we collected the trends from countries where the English is the main language and from other 3 countries that we have mentioned in section 4.3.

4.6 Conclusions

Firstly we started by introducing a problem definition. We showed that we will take a multi classification approach, in order to compare the performances among the different classes with different testing sets regarding different text features. Followingly, we described the 3 phases of our model, data preparation, learning phase and testing/evaluation phase. The first phase is responsible for gathering the data and preparing it for the model. The second phase is responsible for building the model using this data that was processed. The final one is responsible for the testing and evaluation of the model. Followingly we described the data that we used and how we collected and stored it. To conclude we described and explained the features that we used in our model.

Chapter 5

Experimental Results

In this chapter we will describe our experimental results, we will start by describing the setup of our work, where we explain how we splitted and sampled our data, how we built the training and testing sets. After explaining our setup we will show and analyze the results that we have obtained for the 2 datasets.

5.1 Setup

In this section we will describe our experimental setup, in which we will show the used algorithm how we divided our data for training and testing and the correspondent testing sets that we used to analyze the performance of our model.

5.1.1 Data splitting and sampling

Data splitting means dividing the data into training and testing sets. In our work we decided to use constantly 80% for the training data and 20% for the testing data. As we showed on chapter 4 both of our datasets are clearly unbalanced, which will affect the performance of the model. In our work we applied two techniques to mitigate this problem. The first one we called Class balancing, and the second one we called Sub Class balancing.

Class Balancing

One way to mitigate this problem is to oversample the minority classes while undersampling the majority classes, by saying that we mean, selecting the same amount of samples of each class for the training data.
In the case of the first dataset we chronologically divided it into two parts. The first one contains the tweets from the days 1 until 12, we used them to build the

training set. The second one contains the tweets from the days 12 to 15, we used them to build our testing sets. In the first part of the data we had 413288 tweets for the class 4 which is the one with the less number of tweets in it, so the minority class in our case. From this number of tweets of the class 4 we decided to select randomly 300k, which in this case will lead us to select randomly 300k tweets for each one of the remaining classes in order balance the number of samples of each class. As for each one of the testing sets we also selected randomly 300k tweets (since the used ratio was 80% for training and 20% for testing).
For the second dataset we also decided to use 300k of tweets of each class for training and 300k randomly for testing. In this case we did not explicitly splitted the dataset into two parts, we just selected the tweets from the available data for each on of sets (training and testing), but in a way to make sure that we do not get the same tweet repeated in the training data and in the testing data.

Sub class balancing

As we observed in our data, that as we advance with higher values of the number of retweets, there are less and less tweets in each class. This also happens inside of the classes for example if we look at the class 3 which represents the tweets with 10 or more retweets and less than 100, the first interval for which we count the number of tweets within that range (10-20) has almost 10 times more than the range (90-100) as we can see in Table A.5, so by saying that if we randomly select x amount of tweets of that class most of them will be contained in the first intervals. This might affect the performance of the model when classifying the tweets that are "near" the border (90-100 number of retweets and 100-110 number of retweets). In order to avoid these differences among each class, we randomly selected, the following quantity of tweets for each one of the "sub-classes" that we selected. For the class 1 we continued to select randomly 300k tweets, since it is only represented by tweets with 0 retweets. Regarding the second class we divided it into 2 sub classes that we show in the Table 5.1 bellow with the number of randomly selected tweets for each interval.

Table 5.1: Class 2 balancing

Interval	**Sample**
1-5	150k
5-10	150k

In respect to the class 3 since it covers the tweets from 10 to 100, we divided it into 9 sub classes. In the Table 5.2 we show the different intervals that we selected, and the given amount of tweets that we randomly selected from each one of them.

Table 5.2: Class 3 balancing

Interval	**Sample**
10-20	50k
20-30	50k
30-40	40k
40-50	30k
50-60	20k
60-70	20k
70-80	20k
80-90	20k
90-100	20k

In respect to the class 4 since it covers the tweets with 100 or more retweets, we divided it into 11 sub classes. In the Table 5.3 we show the different intervals that we selected, and the given amount of tweets that we randomly sampled from each one of them.

Table 5.3: Class 4 balancing

Interval	**Sample**
100-120	35k
120-140	35k
140-160	30k
160-200	25k
200-240	25k
240-280	25k
280-340	25k
340-400	25k
400-520	25k
520-100	25k
1000+	25k

5.1.2 Testing Sets

Before going into detail of these specific tests for each one of the datasets, we have made 3 main testing sets for both of them. So as we said we are going to compare the results of the different testing sets among them (like testing with tweets that contain one Hashtag, testing with tweets without Hashtags). But we will also compare it with global testing sets. For each time that we trained the model we

ran all of our tests and repeated this process with a total of 30 times using the **Random Forests** algorithm.

Global tests

This group of testing sets is used for both of the datasets, here our goal is firstly to observe if the addition of the extra features mentioned in the subsection 4.5.3 of the previous chapter, if they improve the performance of the model for each dataset. After that we want to observe if our method for Sub class balancing would bring improvements to the model.

Table 5.4: Global tests

Features	**Balancing Approach**
Without extra features	Class balancing
With extra features	Class balancing
With extra features	Sub class balancing

As we show above in the Table 5.4 we have 3 combinations for the global tests, the first one we test the model without the extra features using the class balancing approach. The second test that we performed was testing the model with the extra features with the class balancing approach. The final test that we have performed was testing the model with the extra features using the sub class balancing method.

Tests for the Twitter Streaming API dataset

There are many ways that we can evaluate the performance of a model. In this case since this dataset has the tweets that represent the general overview of the flow of information in Twitter, we decided to test our model with multiple testing sets, by that we mean that besides doing the classic testing like selecting a random number of tweets (for example like 20% of the data for testing), we selected randomly a group of tweets within certain characteristics of the tweet message in order to compare it among them, and see which ones have better results for different classes. For example using random sampling for the testing set we have a F1 score of 50% for the class 4, and when using a testing set where the all of the tweets have one Hashtag the F1 score for the sames class is 52%.
Bellow we present the different testing sets that we have selected. These testing sets are performed for each one of the sampling methods that used.

Table 5.5: Tests for the Twitter Streaming API dataset

Features tested	Test(s)
Reply	Tweets that are not replies
Length of the tweet message	Tweets between 0 and 60 characters (Short texts) Tweets between 60 and 100 characters (Medium texts) Tweets with more than 100 characters (Long texts)
Number of Hashtags	Tweets without Hashtags Tweets with 1 Hashtag Tweets with 1 or more Hashtags
Number of URLs	Tweets without URLs Tweets with 1 or more URLs

As we can see in the Table 5.5 we tested 4 different text features, that were, if a tweet is a reply, the length of the tweet message, the number of Hashtags in a tweet and the number of URLs in a tweet. For the first feature, we only tested the tweets that are not replies, since that in the available testing data we do not have the same amount of tweets that are replies. For the length of the tweet, we divided it into 3 parts according to its length, the first one represents the short tweets, the second one represents the medium tweets, and the last one represents the long tweets. Regarding the feature number of Hashtags, we also divided into 3 parts, the first one is the tweets without Hashtags, the second one is the tweets that contain 1 Hashtag, and the third one is the tweets that contain at least one Hashtag, meaning that it contains tweets that may have more than one Hashtag. For the final feature, we divided it into 2 tests, the first one represents the tweets without URLs, and the second one represents the tweets that have at least 1 URL, since in our testing set we do not have enough tweets that contain exactly 1 URL, we decided to group them.

Tests for the Twitter Search API dataset

The main goal of this dataset is to analyze how the number of retweets of a user's k last tweets affects the performance of the model. By saying that we will test the model with this dataset, with testing sets that have different k last tweets of a user. With that in mind we will have the following cases, training data with the addition of the number of retweets of a user last tweet, and then testing data also with this number of retweets of the user last tweet. And this logic will follow, by using the number of retweets of the user last 2 tweets for training a testing, until we reach the usage of the last 10 of the user's number of retweets of his last 10 tweets.

5.2 Results analysis for the Twitter Streaming API dataset

In this section we will present the results for the Twitter Streaming API dataset. Firstly we will start by analyzing the global tests that we did, then we will analyze the results of the different testing sets that we have performed. From now on when we mention performance we will refer to the F1 score.

5.2.1 Global tests

The following 3 Tables contain the results of the experiments for these groups of tests.
The first test that we have applied was by testing the model without the extra features that we have used, which we show in Table 5.6 bellow. We can see that the performance was 80.76% for the class 1, 39.40% for the class 2, 44.07% for the class 3 and for the class 4 it was 58.07%.

Table 5.6: Results for testing with tweets without using the extra features

Class	Number of tweets	Precision	Recall	F-score
1	211996.10 ± 241.15	91.34 ± 0.08	72.37 ± 0.09	80.76 ± 0.06
2	53986.70 ± 183.91	32.41 ± 0.13	50.22 ± 0.18	39.40 ± 0.12
3	23452.77 ± 150.89	37.66 ± 0.31	53.10 ± 0.31	44.07 ± 0.29
4	10564.43 ± 80.74	49.06 ± 0.34	71.12 ± 0.52	58.07 ± 0.33

Now using the extra features as we show below in the Table 5.7, we can see that there was an improvement of the performance for all the classes. For the classes 1 and 2 the improvement was just little bit, 0.04% and 0.05% respectively, but for the classes 3 and 4 the improvement was 0.43% and 0.53%.

Table 5.7: Results for testing with tweets using the extra features

Class	Number of tweets	Precision	Recall	F-score
1	211996.10 ± 241.15	91.36 ± 0.06	72.43 ± 0.15	80.80 ± 0.10
2	53986.70 ± 183.91	32.47 ± 0.15	50.26 ± 0.20	39.45 ± 0.15
3	23452.77 ± 150.89	37.92 ± 0.23	53.84 ± 0.31	44.50 ± 0.21
4	10564.43 ± 80.74	49.89 ± 0.38	71.01 ± 0.34	58.60 ± 0.32

For the final global test we tested the model with the extra features in order to compare it to the normal evenly random selection for each class. In the Table 5.8

we show the results for training with this sub class balancing approach, and testing with randomly selected tweets as we did before. We can see that for the first class the performance was 85.19%, which is a 4.39% improvement, as for the class 2 the improvement was 6.29%. Regarding the class 3 there was an improvement of 12.68% and for the last class the improvement was 19.58%.

Table 5.8: Results for testing with tweets using the extra features with sub class balancing

Class	Number of tweets	Precision	Recall	F-score
1	211996.10 ± 212.07	90.83 ± 0.11	80.21 ± 0.08	85.19 ± 0.07
2	53986.70 ± 140.65	41.25 ± 0.19	51.32 ± 0.55	45.74 ± 0.26
3	23452.77 ± 128.80	49.20 ± 0.60	68.25 ± 0.34	57.18 ± 0.31
4	10564.43 ± 45.44	70.51 ± 0.35	87.72 ± 0.41	78.18 ± 0.15

Tests conclusions From the first 2 Tables (Table 5.6 and Table 5.7) we observed that adding these extra features improved the performance of the model, especially in the classes 3 and 4, which is logical since these extra features add useful information to the model. As for the application of sub class balancing the improvements were very noticeable. One of main reason is probably that when we select samples from the different sub classes we select more tweets that are on the limit ranges of the classes. So for example selecting randomly 300k tweets for the class 3, most of them will be contained in the 10-20 interval, and way less in the 90-100. So by selecting more tweets from that interval the model is more capable to distinguish them from the tweets of the class 4.

5.2.2 Tests featuring length of the message

In order to analyze how the length of the message affects the performance of the model among the classes, we divided the length in the following intervals 0-60, 60-100 and 100-140. The first interval is to represent the tweets with short messages, the second one is to represent the tweets with an medium length of the message, and the last one is to represent the longest tweets in terms of text.

Regarding the tests with less than 60 characters presented on the Table 5.9, we can see that for the class 4 the performance of the model improved 2% when comparing with the same class on the Table 5.8 that represents the global performance of the model. As for the classes 2 and 3 there was a decrease of the performance 5.97% and 2.67% respectively. Finally for the first class there was a improvement of 1.85%.

Table 5.9: Results for testing with tweets with less than 60 characters

Class	Number of tweets	Precision	Recall	F-score
1	231214.20 ± 176.16	90.00 ± 0.06	84.27 ± 0.20	87.04 ± 0.08
2	46331.00 ± 162.10	36.89 ± 0.18	43.14 ± 0.62	39.77 ± 0.29
3	13842.00 ± 56.34	47.62 ± 0.48	63.74 ± 0.41	54.51 ± 0.34
4	8612.80 ± 60.67	72.22 ± 0.27	90.53 ± 0.42	80.34 ± 0.24

Analyzing the Table 5.10 that represents the tweets between 60 and 100 characters, when comparing to the global model on the Table 5.8,the only class that had an increase on performance was the class 1 with an improvement of 0.17%. For the classes 2 and 3 the performance decreased 0.98% and 0.78% respectively. For the class 4 it only decreased 0.17%. Comparing these results with the ones from the Table 5.9, we can see for the class 1 that the performance got worse by 1.70%, but for the classes 2 and 3 there was an improvement of 4.99% and 1.89% respectively. For the class 4 the performance got worse by 2.33%.

Table 5.10: Results for testing with tweets between 60 and 100 characters

Class	Number of tweets	Precision	Recall	F-score
1	214178.60 ± 246.31	91.13 ± 0.03	80.28 ± 0.14	85.36 ± 0.08
2	51769.40 ± 148.03	40.24 ± 0.23	50.42 ± 0.28	44.76 ± 0.16
3	22283.80 ± 78.94	48.05 ± 0.45	68.27 ± 0.25	56.40 ± 0.35
4	11768.20 ± 73.83	70.02 ± 0.40	88.05 ± 0.30	78.01 ± 0.25

Analyzing the results when testing the tweets with more than 100 characters on the Table 5.11 and comparing it to the global results in the Table 5.8, the performance for the class 1 decreased 2.57% and for the class 4 it decreased 1.1%. For the classes 2 and 3 there was an improvement. 3.09% for the class 2 and 4.3% for the class 3. If we compare these results regarding the classes 2 and 3 with the ones on the Tables 5.9 and 5.10 we can see that there is an increment of the performance as we use more characters for the classes 2 and 3, and a decrease of performance for the classes 1 and 4. Interestingly the inverse phenomena happens when we use less characters, the performance for the class 1 and 4 increases and for the class 2 and 3 decreases.

Table 5.11: Results for testing with tweets with more than 100 characters

Class	Number of tweets	Precision	Recall	F-score
1	190635.80 ± 310.29	91.91 ± 0.09	75.03 ± 0.16	82.62 ± 0.09
2	62533.00 ± 138.41	44.92 ± 0.15	58.54 ± 0.23	50.83 ± 0.07
3	32624.00 ± 157.02	50.51 ± 0.21	70.39 ± 0.09	58.81 ± 0.12
4	14207.20 ± 120.07	69.97 ± 0.24	85.79 ± 0.29	77.08 ± 0.22

Tests conclusions From these tests regarding the length of the message we observed that when using less characters is easier to predict the tweets for the classes 1 and 4, and when using tweets with more characters it's easier to predict tweets of the classes 2 and 3 and harder for the classes 1 and 4.

5.2.3 Test featuring tweets without replies

We tested the model with tweets without replies in order to compare it to the global model 5.8. It's important to state that in our dataset the number of tweets with replies was small, so we decided to compare the global results to the model with tweets without replies, since the tweets selected for the global results contain tweets that are replies.
We can observe that the performance of the model with tweets that are no replies on the Table 5.12 is better for the class 4 by 0.26% and is lower for the remaining classes. One possible reason for that might be since the tweets with replies are the ones which got less retweets, so in this case it makes sense since we leave them out of our testing set the performance of the classes with less retweets falls.

Table 5.12: Results for testing with tweets that are no replies

Class	Number of tweets	Precision	Recall	F-score
1	204578.60 ± 241.45	90.14 ± 0.08	77.91 ± 0.20	83.58 ± 0.12
2	55440.40 ± 222.27	40.09 ± 0.25	49.90 ± 0.38	44.46 ± 0.23
3	25384.60 ± 114.11	48.13 ± 0.21	68.07 ± 0.45	56.38 ± 0.12
4	14596.40 ± 139.01	70.55 ± 0.54	88.33 ± 0.36	78.44 ± 0.36

5.2.4 Tests featuring Hashtags

We tested the model with tweets with Hashtags and tweets without Hashtags to see if there is any relevant difference in the performance. We can observe that when Hashtags are used the performance of the model drops for the classes 1 and 4 and increases for the classes 2 and 3.

Analyzing the tweets the Table 5.13 and comparing them to the general results on the Table 5.8 we can see that for the class 1 there was an improvement of the performance by 1.17%, for the classes 2 and 3 there was decrease of the performance of 2.52% and 0.67%. Finally for the class 4 the improvement was 0.75%.

Table 5.13: Results for testing with tweets without Hashtags

Class	Number of tweets	Precision	Recall	F-score
1	220067.00± 205.57	90.75± 0.08	82.37± 0.08	86.36± 0.05
2	49700.60± 116.26	39.57± 0.17	47.61± 0.44	43.22± 0.24
3	19566.80± 52.33	48.48± 0.29	67.45± 0.42	56.41± 0.21
4	10665.60± 80.20	71.26± 0.69	88.45± 0.25	78.93± 0.52

Regarding the Table 5.14 where we present the results when testing with tweets with one Hashtag, we can see that if we compare the performance to the global results in the Table 5.8 there was a decrease of 8.15% for class 1. For the classes 2 and 3 there was an improvement around 5%, and 2%. In the last class the performance decreased 5.81%. Comparing the performance of these results from the Table 5.14 with the results when no Hashtags are used (Table 5.13), we can observe that for the class 1 there is a drop of around 8% and for the classes 2 and 3 the improvement of the performance was 8.33% and 1.8%. Regarding the class 4 the performance decreased 3.61%.

Table 5.14: Results for testing with tweets with one Hashtag

Class	Number of tweets	Precision	Recall	F-score
1	175128.20 ± 65.80	90.96 ± 0.06	69.28 ± 0.22	78.65 ± 0.14
2	69156.60 ± 111.58	45.37 ± 0.27	59.69 ± 0.37	51.55 ± 0.11
3	38244.60 ± 75.92	50.07 ± 0.30	69.51 ± 0.29	58.21 ± 0.12
4	17470.60 ± 49.41	66.83 ± 0.33	86.28 ± 0.17	75.32 ± 0.26

Since in the data that we left for selecting the testing samples did not contain enough tweets with 2 Hashtags, we randomly selected the tweets which had at least 1 Hashtag. Comparing the performance of these results in the Table 5.15 where we tested the model with tweets that had at least 1 Hashtag with the global results in the Table 5.8, we can see that was a decrement of 4.96% regarding the class 1. For the classes 2 and 3 there was an improvement of 6.38% and 1.43% respectively. For the class 4 there was a drop of 2.46%.
If we compare these results with the ones in the Table 5.13 (tweets without Hashtags), the performance of the classes 1 and 4, decreased 6.13% and 3.21% re-

spectively. For the classes 2 and 3 the performance improved 8.9% and 2.2%, respectively.

Finally comparing these results with the ones in the Table 5.14 (tweets without Hashtags), the performance increased for all of the classes, 1.58% for the class 1, 0.7% for the class 2, and 0.4% for the classes 3 and 4.

Table 5.15: Results for testing with tweets with one or more Hashtags

Class	Number of tweets	Precision	Recall	F-score
1	184168.00 ± 110.47	91.42 ± 0.07	71.48 ± 0.37	80.23 ± 0.24
2	67551.20 ± 144.78	45.37 ± 0.28	61.23 ± 0.42	52.12 ± 0.10
3	34167.80 ± 88.68	50.63 ± 0.26	69.58 ± 0.34	58.61 ± 0.16
4	14113.00 ± 16.70	67.72 ± 0.14	85.87 ± 0.16	75.72 ± 0.11

Tests conclusions In general we observed that the tweets which belong the classes 2 and 3, the prediction improved when we used Hashtag(s), as for the class 1 and especially for the class 4 the results got worse. Knowing that in average tweets with Hashtags have more retweets as we can see in on the number of tweets for the classes 2, 3 and 4 is greater than when using tweets without any Hashtags (Table 5.13). But it is interesting the class 4 is the one which got worse results than before, since the popular users are the ones who got more retweets. That may suggest using an Hashtag is more relevant to a user who has less retweets than a user who has more retweets.

5.2.5 Tests featuring URLs

We tested the model with tweets without URLs and tweets that have at least one URL to see if there is any relevant difference in the performance of the model.

The Table 5.16 shows us the results when we tested the model using only the tweets without URLs, we can observe on this Table by comparing it to the global results on the Table 5.8 that for the class 1 there is decrement of performance approximately 1.59%. For the classes 2 and 3 there was also a decrement of 1.28%, and 0.8% respectively. Finally for class 4 there was an improvement of 0.26%.

Table 5.16: Results for testing with tweets without URLs

Class	Number of tweets	Precision	Recall	F-score
1	204578.60 ± 241.45	90.14 ± 0.08	77.91 ± 0.20	83.58 ± 0.12
2	55440.40 ± 222.27	40.09 ± 0.25	49.90 ± 0.38	44.46 ± 0.23
3	25384.60 ± 114.11	48.13 ± 0.21	68.07 ± 0.45	56.38 ± 0.12
4	14596.40 ± 139.01	70.55 ± 0.54	88.33 ± 0.36	78.44 ± 0.36

Analyzing the Table 5.17 where we showed the results for the tweets with one more URLs since that in the training data there was not enough tweets with 1 URL to make the sample of 300k. We can observe when comparing it to the global results in the Table 5.8 for the first 3 classes there was an improvement of 2.53% for the class 1, 2.76% for the class 2 and 2.2% for the class 3. Regarding the class 4 there was a decrement of 1.61%.
Analyzing the Table 5.17 where we showed the results for the tweets with one more URLs since in the training data there was not enough tweets with 1 URL. We can observe when comparing it to the global results in the Table 5.8, for the first 3 classes there was an improvement of 4.14% for the class 1, 4.27% for the class 2 and 4.78% for the class 3. Regarding the class 4 there was a decrement of 5.7%.

Table 5.17: Results for testing with tweets with URLs

Class	Number of tweets	Precision	Recall	F-score
1	224748.60 ± 337.25	92.00 ± 0.07	83.81 ± 0.11	87.72 ± 0.07
2	50604.20 ± 206.50	43.68 ± 0.27	54.51 ± 0.26	48.50 ± 0.12
3	18629.80 ± 51.62	51.99 ± 0.64	69.23 ± 0.15	59.38 ± 0.46
4	6017.40 ± 95.05	69.88 ± 0.31	84.68 ± 0.57	76.57 ± 0.36

Tests conclusions Here we analyzed how the presence of URLs affected the model, we saw that comparing to the global results on the Table 5.8 for the class 1 there was an improvement when using URLs as well for the classes 2 and 3, for the class 4, when not using URLs the performance got better, while when URLs were used it got worse.

5.3 Results analysis using the Twitter Search API dataset

In this section we will present the results for the Twitter Search API dataset. Firstly we will start by analyzing the global tests that we did. Then we will

analyze the results of the different testing sets that we have performed. Finally we will make a general conclusion for all of the performed tests with this dataset. In this case for the testing sets we decided to select randomly 211500, for the class 1, 54000 for the class 2, 24000 for the class 3 and 10500 for the class 4. The reason why we kept these values fixed instead of randomly selecting them from all of the tweets that were not used in the training data, was because we tried do maintain approximately the proportion according to the number of tweets in each class when randomly selected for the tests of the global results in the Twitter Streaming API dataset.

5.3.1 Global tests

The following 3 Tables contain the results of the experiments for these groups of tests.
The first test that we have made testing the model without the extra features that we have used, which we show in Table 5.18 bellow. We can see that the performance was 84.61% for the class 1, 46.59% for the class 2, 59.55% for the class 3 and for the class 4 it was 72.34%.

Table 5.18: Results for testing with tweets without using the extra features

Class	**Number of tweets**	**Precision**	**Recall**	**F-score**
1	211500.00 ± 0.00	92.23 ± 0.05	78.15 ± 0.07	84.61 ± 0.06
2	54000.00 ± 0.00	39.47 ± 0.17	56.86 ± 0.23	46.59 ± 0.19
3	24000.00 ± 0.00	53.73 ± 0.17	66.79 ± 0.18	59.55 ± 0.09
4	10500.00 ± 0.00	65.07 ± 0.33	81.43 ± 0.43	72.34 ± 0.31

Now using the extra features as we show below on the Table 5.19, we can see that there was an improvement of the performance for all the classes. For the classes 1 and 2 the improvement was 0.13% and 0.11% respectively, and for the classes 3 and 4 the improvement was greater 0.38% and 0.55% respectively.

Table 5.19: Results for testing with tweets using the extra features

Class	**Number of tweets**	**Precision**	**Recall**	**F-score**
1	211500.00 ± 0.00	92.27 ± 0.04	78.34 ± 0.06	84.74 ± 0.05
2	54000.00 ± 0.00	39.69 ± 0.17	57.01 ± 0.23	46.80 ± 0.19
3	24000.00 ± 0.00	54.03 ± 0.22	67.29 ± 0.31	59.93 ± 0.21
4	10500.00 ± 0.00	65.97 ± 0.50	81.43 ± 0.46	72.89 ± 0.40

For the final global test we tested the model with the extra features in order to compare it to the normal evenly random selection for each class, we decided do

sub divide each class as we said on the subsection 5.1.2. So in this case we tested the model with the extra features presentend in the the subsection **??** in order to compare it to the normal evenly random selection for each class. In the Table 5.20 we show the results for training with this sub class balancing method, and testing with randomly selected tweets as we did before. We can see that for the first class the performance was 87.14%, which is a 2.4% improvement, as for the class 2 the improvement was 2.93%. Regarding the class 3 the performance basically stayed the same, with a decrement of 0.01% and for the last class the improvement was 2.64%. If we look at these improvements and compare them to the ones in the Twitter Streaming API dataset, the improvements of that dataset using this sub class sampling method were more significant.

Table 5.20: Results for testing with tweets using the extra features with sub class balancing

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	90.68 ± 0.03	83.86 ± 0.09	87.14 ± 0.04
2	54000.00 ± 0.00	43.94 ± 0.10	57.29 ± 0.24	49.73 ± 0.14
3	24000.00 ± 0.00	62.15 ± 0.33	56.00 ± 0.19	58.92 ± 0.22
4	10500.00 ± 0.00	69.80 ± 0.29	82.29 ± 0.12	75.53 ± 0.21

Tests conclusions From the first 2 Tables (Table 5.18 and Table 5.19) we observed that adding these extra features improved the performance of the model, especially in the classes 3 and 4. This is logical since these extra features add extra information to the model. As for the application of sub class balancing the improvements were very noticeable, one of main reason is probably that when we select samples from the different sub classes we select more tweets that are on the limit ranges of the classes, so for example selecting randomly 300k tweets for the class 3, most of them will be contained in the 10-20 interval, and way less in the 90-100.

5.3.2 Tests for the number of retweets of a user last k tweets

The following 10 tables contain the results of the experiments for these groups of tests.
Regarding the Table 5.21 we presented the results when testing the model adding the number of retweets that a user got from his last tweet. We can see than when comparing these results with the previous Table 5.20 that there was slightly improvement for the classes 1, 2 and 3, that was 0.07%, 0.69% and 0.36% respectively. As for the class 4 the performance decreased 0.07%.

Table 5.21: Results for testing using the number of retweets of the last tweet for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	90.96 ± 0.05	83.86 ± 0.12	87.27 ± 0.06
2	54000.00 ± 0.00	44.21 ± 0.12	58.67 ± 0.20	50.42 ± 0.11
3	24000.00 ± 0.00	62.97 ± 0.41	56.00 ± 0.19	59.28 ± 0.19
4	10500.00 ± 0.00	70.75 ± 0.42	80.85 ± 0.45	75.46 ± 0.41

In this Table 5.22 we presented the results when testing the model adding the number of retweets that a user got from his last 2 tweets. We can see than when comparing these results with Tables 5.20 and 5.21 there was an improvement for the classes 1, 2 and 3 over these two Tables. As for the class 4 the performance decreased approximately 0.14% to the same class on Table 5.20 and 0.07% when comparing to the Table 5.22.

Table 5.22: Results for testing using the number of retweets of the last 2 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	91.03 ± 0.06	84.15 ± 0.17	87.45 ± 0.08
2	54000.00 ± 0.00	44.61 ± 0.15	59.14 ± 0.37	50.85 ± 0.18
3	24000.00 ± 0.00	63.63 ± 0.44	56.00 ± 0.40	59.57 ± 0.39
4	10500.00 ± 0.00	71.32 ± 0.51	79.96 ± 0.31	75.39 ± 0.40

In this Table 5.23 we presented the results when testing the model adding the number of retweets that a user got from his last 3 tweets. We can see again than when comparing these results with Tables 5.20, 5.21 , 5.22 there was a slight improvement for all of the classes, so a this moment training the model with the number of retweets of the 3 last tweets of a user is outperforms all of the other tests.

Table 5.23: Results for testing using the number of retweets of the last 3 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	91.07 ± 0.04	84.29 ± 0.08	87.55 ± 0.04
2	54000.00 ± 0.00	44.76 ± 0.08	59.32 ± 0.19	51.02 ± 0.09
3	24000.00 ± 0.00	64.13 ± 0.13	56.15 ± 0.23	59.87 ± 0.18
4	10500.00 ± 0.00	72.04 ± 0.18	80.02 ± 0.24	75.82 ± 0.19

In the Table 5.24 we can see the results when using the retweets of the last 4 tweets for each user. Comparing these results with the ones of the Table 5.23, where we used the last 3 tweets of a user, here we got better results of each one of the classes, representing an improvement of 0.09% for class 1, 0.25% for the class 2, 0.26% for the class 3 and 0.17% for the class 4.

Table 5.24: Results for testing using the number of retweets of the last 4 tweets for each user

Class	**Number of tweets**	**Precision**	**Recall**	**F-score**
1	211500.00 ± 0.00	91.11 ± 0.05	84.43 ± 0.09	87.64 ± 0.04
2	54000.00 ± 0.00	45.01 ± 0.09	59.55 ± 0.22	51.27 ± 0.09
3	24000.00 ± 0.00	64.52 ± 0.22	56.30 ± 0.20	60.13 ± 0.20
4	10500.00 ± 0.00	72.28 ± 0.23	80.10 ± 0.16	75.99 ± 0.16

In the Table 5.25 we can see the results when using the retweets of the last 5 tweets for each user. Comparing these results with the ones of the Table 5.24, where we used the last 4 tweets of a user, here we got better results of each one of the classes, representing an improvement of 0.08% for class 1, 0.24% for the class 2, 0.2% for the class 3 and 0.14% for the class 4.

Table 5.25: Results for testing using the number of retweets of the last 5 tweets for each user

Class	**Number of tweets**	**Precision**	**Recall**	**F-score**
1	211500.00 ± 0.00	91.16 ± 0.06	84.53 ± 0.09	87.72 ± 0.04
2	54000.00 ± 0.00	45.26 ± 0.10	59.77 ± 0.27	51.51 ± 0.12
3	24000.00 ± 0.00	64.78 ± 0.12	56.45 ± 0.24	60.33 ± 0.15
4	10500.00 ± 0.00	72.31 ± 0.11	80.37 ± 0.20	76.13 ± 0.12

In the Table 5.26 we can see the results when using the retweets of the last 6 tweets for each user. Comparing these results with the ones of the Table 5.25, where we used the last 5 tweets of a user, here we got better results for the first 3 classes, and the class 4 stayed the same. There was an improvement of 0.07% for class 1, 0.19% for the class 2 and 0.1% for the class 3.

Table 5.26: Results for testing using the number of retweets of the last 6 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	91.21 ± 0.04	84.61 ± 0.06	87.79 ± 0.03
2	54000.00 ± 0.00	45.45 ± 0.14	59.93 ± 0.24	51.70 ± 0.16
3	24000.00 ± 0.00	64.92 ± 0.24	56.52 ± 0.25	60.43 ± 0.23
4	10500.00 ± 0.00	72.27 ± 0.23	80.44 ± 0.19	76.13 ± 0.18

In the Table 5.27 we can see the results when using the retweets of the last 7 tweets for each user. Comparing these results with the ones of the Table 5.26, where we used the last 6 tweets of a user, here we got better results of each one of the classes, representing an improvement of 0.05% for class 1, 0.1% for the class 2, 0.23% for the class 3 and 0.12% for the class 4.

Table 5.27: Results for testing using the number of retweets of the last 7 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	91.22 ± 0.08	84.71 ± 0.10	87.84 ± 0.05
2	54000.00 ± 0.00	45.61 ± 0.18	59.93 ± 0.35	51.80 ± 0.22
3	24000.00 ± 0.00	65.10 ± 0.21	56.80 ± 0.29	60.66 ± 0.20
4	10500.00 ± 0.00	72.36 ± 0.21	80.58 ± 0.34	76.25 ± 0.24

In the Table 5.28 we can see the results when using the retweets of the last 8 tweets for each user. So far for the classes 1 and 2 here we got the better results with 87.89% and 51.86% of performance. But for the classes 3 an 4 there was decrement of 0.06% and 0.09% comparing tho the Table 5.27 where we used the number of retweets of the user's last 7 tweets.
Comparing these results with the ones of the Table 5.27, where we used the last 7 tweets of a user, the only class that improved was the class 3, by 0.06%, the remaining classes had a slight decrease of performance.

Table 5.28: Results for testing using the number of retweets of the last 8 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	91.24 ± 0.06	84.78 ± 0.09	87.89 ± 0.03
2	54000.00 ± 0.00	45.72 ± 0.16	59.92 ± 0.37	51.86 ± 0.22
3	24000.00 ± 0.00	64.91 ± 0.14	56.77 ± 0.39	60.57 ± 0.28
4	10500.00 ± 0.00	72.25 ± 0.34	80.56 ± 0.23	76.18 ± 0.26

In the Table 5.29 we can see the results when using the retweets of the last 8 tweets for each user. So far for the class 2 here we got the better results with 51.93% of performance, that represents a small improvement of 0.07% comparing to the results when the last 8 tweets are used (Table 5.28). For the remaining 3 classes the performances dropped slightly, 0.02% for the class 1, 0.06% for the class 3 and 0.19% for the class 4.

Table 5.29: Results for testing using the number of retweets of the last 9 tweets for each user

Class	**Number of tweets**	**Precision**	**Recall**	**F-score**
1	211500.00 ± 0.00	91.29 ± 0.07	84.70 ± 0.05	87.87 ± 0.03
2	54000.00 ± 0.00	45.74 ± 0.16	60.07 ± 0.31	51.93 ± 0.21
3	24000.00 ± 0.00	64.76 ± 0.12	56.78 ± 0.36	60.51 ± 0.25
4	10500.00 ± 0.00	71.80 ± 0.43	80.70 ± 0.33	75.99 ± 0.35

In the Table 5.30 we presented the results when testing the model adding the number of retweets that a user got from his last 10 tweets. We can see again as we use more of the last users past retweets the performance of the class 2 slightly increases, 0.03% in this case over the performance showed for the same class in the Table 5.29. For the class 1 the performance is also the best one that we so far got which is a little improvement over the results when using the retweets of last 8 tweets of a user, showed in the Table 5.28. For the class 3 the performance dropped 0.06%, and for the class 4 it stayed the same when comparing to the Table 5.29 where we have the results when using the number of retweets for the last 9 tweets of a user.

Table 5.30: Results for testing using the number of retweets of the last 10 tweets for each user

Class	**Number of tweets**	**Precision**	**Recall**	**F-score**
1	211500.00 ± 0.00	91.28 ± 0.04	84.82 ± 0.09	87.93 ± 0.06
2	54000.00 ± 0.00	45.84 ± 0.13	59.97 ± 0.14	51.96 ± 0.08
3	24000.00 ± 0.00	64.85 ± 0.41	56.62 ± 0.47	60.45 ± 0.39
4	10500.00 ± 0.00	71.63 ± 0.46	80.92 ± 0.40	75.99 ± 0.31

Tests conclusions Observing these results that we showed above we can say that using more of the user's previous numbers of retweets of his tweets improves the performance of the model, but mainly for the first 3 classes. The performance for the classes 1 and 2 is higher when using more of the number of retweets of each of the users last k tweets. For the class 3 the performances increases until

the number of retweets of each one of the last 7 tweets of a user. Finally regarding the class 4, we observed that it's the most inconstant one, since the performances stars to drop as we use more of the number of retweets of the user k last tweets, but when it reaches the k values of 3 it starts to increase and then when the k is 8 it starts decreasing again.

5.4 Results Discussion

One of the things that we wanted to address in this work was the introduction of sentiment analysis and trending topics. In both of the tests that we had performed regarding the usage of these features we showed that the performance of the model was better, especially in the classes 3 and 4. So one thing that we concluded from here, is that as Zhang et al.[7] thought, indeed these features have a positive impact.

5.4.1 Twitter Streaming API

The second part that we have studied was how the prediction for different classes was affected by different text features. When analyzing the performance of the model regarding these different text features that we had tested, we showed that there are features that are more related to different classes. By that we mean for example shorter texts can be related to tweets that have no retweets, or tweets that have more retweets (class 4). While the tweets that have more text they tend to be less related to the tweets from classes 1 and 4, and more related to the classes 2 and 3 where the performance of the model increases with the length of the tweets. Knowing that usually the main reason that dictates the final popularity of the tweets is the number of followers, we think that users with more followers benefit from writing short texts, while for the users with a few amount of followers tend to have less retweets if they have short text messages. As for the users that belong to a middle level of popularity in terms of followers, they benefit when writing texts with more information in it.
In the case of Hashtags we can say that tweets that do not have any Hashtags are more easily identified when they wont get any retweets. These tweets from the class 1 can also be more related to the tweets that have at least one URL, meaning that regarding other features that we have used, when there is at least one URL the model performs better in respect of the class 1. In the case when the tweet has one or multiple Hashtags the classes 2 and 3 are the ones which have improvements in the performance, while the class 4 gets worse. This fact can seem strange since the presence of Hashtags is associated with popularity. But we must also remember that the number of followers plays one of the greatest roles when predicting popularity.
We also noticed that when the tweets contain one or more than 1 Hashtag, the performance of the model is higher for the classes 2 and 3 comparing when we do not use any Hashtag in the tweet. The opposite happens for the classes 1 and 4 where the performances is higher when we do not use any Hashtag in the tweet. So in this case we can think that popular users do not need Hashtags to have more retweets. We can relate this to the performance of the model when we did not

use any Hashtag regarding the class 4, which was better when testing with tweets that have at least one Hashtag, while middle class users benefit from including Hashtag(s) in their tweets.
As for the replies, in the test that we have made, we left them out and the performance of the model was better for the class 4, which can be related that tweets without replies are more easily a indicative of the most popular class, and for the tweets without retweets the performances is worse, which in this case can indicate that tweets with replies are more linked with this class 1.

5.4.2 Twitter Search API

When analyzing the results of the Twitter Search API dataset, we have noticed that, mainly the performance of the model increased for the first 3 classes when using more number of retweets of the last k tweets of a user. The same thing did not happen in the class 4. The reason that we found is that the number of retweets for less popular users tend to be more similar across the last tweets of a user. In the other hand, for some of the more popular tweets, they might have been tweeted by users which generally got less retweets than the ones that we got for that tweet.

Chapter 6

Conclusions & Future Work

In this dissertation we built a Retweet Predictive Model, this model predicts the popularity level of a tweet. This popularity was divided in the following ranges, tweets with 0 retweets (class 1), tweets with 1-10 retweets (class 2), tweets with 10-100 retweets (class 3) and tweets with 100+ retweets (class 4). We used 2 main groups of features (4.5.3) that were extracted from the Twitter APIs. The first group involved the features related to the user, such as number of followers and followees. The second group had the features related to the content of the tweet, such as the number of Hashtags, the length of the message and the sentiment scores.

The main experiments of our work involved a dataset in which the tweets were collected with the Twitter Streaming API. These tweets represent in general the information that flows in the Twitter's network. In these experiments we showed how the prediction for distinct classes is influenced by different text features (Hashtags, URLs, replies and text length) that we have studied. Knowing for which features the model has better results for different classes, we can use these features in order to build a message that has higher probability of being in the most popular classes.

In respect to the second dataset, the data was collected with the Twitter Search API, which we used to gather up to 200 tweets for each user. We used these tweets to analyze the influence of the number of retweets of the last k tweets of a user. This resulted in a general improvement of the model's performance for the first 3 classes (0 retweets, 1-10 retweets, 10-100 retweets, 100+ retweets) with the increase of the k value. The same did not happened for the class 4, since it is less likely to a user to have his last k tweets belonging to that class.

For both of our datasets we have also studied how the combination of our 2 extra features (sentiment scores and trending topics) affected the performance of the model, which resulted in a improvement especially for the classes 3 and 4.

Possible extensions of this work can be such as, analyzing the performance of the

model for different user profiles (i.e. profiles users with a high amount of followers, users with few followers and many favorites), with a possibility of combining it with the text features that we have tested. This will help the different types of users to have their personalized probabilities of getting a certain range of retweets regarding the values of the features that they introduce in their text message. For a final remark, analyzing the tests that we have performed with the ones that we suggested, namely sentiment analysis and trend features suing other different ML algorithms. This might help even more to understand the pattern of the number of retweets across the different characteristics.

Bibliography

[1] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. *What is twitter, a social network or a news media?*, 2010.

[2] Osman Hegazy, Omar S. Soliman and Mustafa Abdul Salam. *Machine Learning Model for Stock Market Prediction* , 2013.

[3] Lei Shi, Neeraj Agarwal, Ankur Agrawal, Rahul Garg and Jacob Spoelstra. *Predicting US Primary Elections with Twitter* , 2012.

[4] Andranik Tumasjan, Timm O. Sprenger, Philipp G. Sandner, Isabell M. Welpe. *Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment.* , 2010.

[5] Konstantin Tretyakov. *Machine Learning Techniques in Spam Filtering* , 2004.

[6] Sasa Petrovic, Miles Osborne, and Victor Lavrenko. *Rt to win! Predicting Message Propagation in Twitter* , 2011.

[7] Yang Zhang, Zhiheng Xu and Qing Yang *Predicting Popularity of Messages in Twitter using a Feature-weighted Model* , 2012.

[8] Maximilian Jenders, Gjergji Kasneci, and Felix Naumann. *Analyzing and Predicting Viral Tweets* , 2013.

[9] Marco Bonzanini. *Mastering Social Media Mining with Python* , 2016.

[10] Cortes and V. Vapnik. Support-Vector Networks. *Machine learning*, 20(3):273–297, 1995.

[11] Vairaprakash Gurusamy and Subbu Kannan. *Preprocessing Techniques for Text Mining*, 2014.

[12] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow and Rebecca Passonneau. *Sentiment Analysis of Twitter Data* , 2011.

[13] Arkaitz Zubiaga, Damiano Spina, Raquel Martínez and Víctor Fresno. *Real-Time Classification of Twitter Trends* , 2013.

[14] David Inouye and Jugal K. Kalita. *Comparing Twitter Summarization Algorithms for Multiple Post Summaries* , 2009.

[15] Liangjie Hong and Brian D. Davison. *Empirical Study of Topic Modeling in Twitter* , 2010.

[16] Alan Ritter, Sam Clark, Mausam and Oren Etzioni. *Named Entity Recognition in Tweets: An Experimental Study* , 2011.

[17] Ikuya Yamada, Hideaki Takeda, and Yoshiyasu Takefuji. *Enhancing Named Entity Recognition in Twitter Messages Using Entity Linking* , 2015.

[18] Alexander Pak and Patrick Paroubek. *Twitter as a Corpus for Sentiment Analysis and Opinion Mining* , 2010

[19] Efthymios Kouloumpis, Theresa Wilson and Johanna Moore. *Twitter Sentiment Analysis: The Good the Bad and the OMG!* , 2011.

[20] Bongwon Suh Bongwon Suh, Lichan Hong Lichan Hong, P Pirolli, and Ed H. Chi. *Want to be retweeted? Large Scale Analytics on Factors Impacting Retweet in Twitter Network*, 2010.

[21] Meeyoung Cha, Hamed Haddadi, Fabricio Benevenuto, and Krishna P. Gummadi. *Measuring User Influence in Twitter: The Million Follower Fallacy. In Fourth International* , 2010.

[22] R. Prtzner, A. Garas, and F. Schweitzer *Emotional Divergence Influences Information Spreading in Twitter* , 2012.

[23] Xun ZHAO, Feida ZHU, Weining QIAN, and Aoying ZHOU. *Impact of Multimedia in Sina Weibo: Popularity and Life Span*, 2012.

[24] `https://www.brandwatch.com/blog/44-twitter-stats-2016/`

[25] Qingyuan Zhao, Murat A. Erdogdu, Hera Y. He, Anand Rajaraman and Jure Leskovec. *SEISMIC: A Self-Exciting Point Process Model for Predicting Tweet Popularity* , 2015.

[26] T. R. Zaman, R. Herbrich, J. V. Gael, and D. Stern. *Predicting Information Spreading in Twitter.* , 2010.

[27] Rui Yan, Mirella Lapata and Xiaoming Li. *Tweet Recommendation with Graph Co-Ranking* , 2012.

[28] Ibrahim Uysal and W. Bruce Croft. *User Oriented Tweet Ranking: A Filtering Approach to Microblogs* , 2011.

[29] Kailong Chen, Tianqi Chen, Guoqing Zheng, Ou Jin, Enpeng Yao and Yong Yu. *Collaborative Personalized Tweet Recommendation* , 2012.

[30] `https://www.echosec.net/twitter-api-vs-firehose/`

[31] `https://blog.bufferapp.com/best-time-to-tweet-research`

[32] https://coschedule.com/blog/visual-content-for-social-media/

[33] Kathy Lee, Diana Palsetia, Ramanathan Narayanan, Md. Mostofa Ali Patwary, Ankit Agrawal, and Alok Choudhary. *Twitter Trending Topic Classification* , 2011.

[34] `https://blog.twitter.com/2013/how-videos-go-viral-on-twitter-three-stories-0`

[35] K. Bontcheva, L. Derczynski, A. Funk, M.A. Greenwood, D. Maynard and N. Aswani. 2013. "TwitIE: An Open-Source Information Extraction Pipeline for Microblog Text". *In Proceedings of the International Conference on Recent Advances in Natural Language Processing*, ACL.

`K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz and Y. Singer. 2006. Online passive-aggressive algorithms.` *The Journal of Machine Learning Research* 7:551–585.

[36] https://www.statista.com/statistics/242606/number-of-active-twitter-users-in-selected-countries/

Appendix A

Tables corresponding to the JSON Objects of the Twitter APIs

Table A.1: Table of the main characteristics in a structure of a tweet's JSON Object

key	**value**
id	contains the id of the tweet
text	contains the text that is present in the tweet
in reply to status id	contains the status of which the message was replied
in reply to user id	contains the id of the user from which the message was replied
retweet count	contains the number of retweets of the tweet
favorite count	contains the of favorites of the tweet
lang	contains the language of the tweet
created at	contains the time when the tweet was created
entities	contains the type and information of any extra content like a photo, that are detailed in the table entities
user	contains JSON Object of the user who tweeted the message

Table A.2: Table of the main characteristics in a structure of a JSON Object that contains the entities.

key	**value**
user mentions	contains a list of all the users that were mentioned in the tweet
hashtags	contains a list of hashtags that are in the message
symbols	contains a list of the symbols that are in the message
URLs	contains all the links that are in the message
media	contains the list about the information of the photo, gif or video that may be present in the tweet

Table A.3: Table of the main characteristics in a structure of a retweet's JSON Object.

key	**value**
in reply to status id	contains the status of which the message was replied
id	contains the id of the retweet
in reply to user id	contains the id of the user from which the message was replied
lang	contains the language of the retweet
retweeted status	contains the JSON Object of the original tweet message with the retweet count updated
created at	contains the time when the retweet was created
extended entities	contains the type and information of any extra content like a photo
user	contains JSON Object of the user who retweeted the message

Table A.4: Table of the main characteristics in a structure of the JSON Object of a User.

key	**value**
created at	contains the UTC of user account creation
default profile image	contains a boolean that indicates if profile image is the default one
listed count	contains the number of public lists where the user is a member
followers count	contains the number of followers of the user
friends count	contains the number of friends of the user
favourites count	contains the number of tweets favorited by the user
statuses count	contains the number of tweets and retweets of the user
verified	indicates if the user account is verified or not
lang	contains the language of the user
id str	contains the id of the user in a string format
is translator	Indicates if the user is a translator of the twitter community

Table A.5: Number of retweets in different ranges of retweets of the class 3

Range	**Number of tweets**
10-20	344868
20-30	186934
30-40	124465
40-50	91352
50-60	71408
60-70	58097
70-80	48289
80-90	41218
90-100	36329

Table A.6: Number of retweets in different early ranges of retweets of the class 4

Range	**Number of tweets**
100-110	31704
110-120	27724
120-130	24776
130-140	21944
140-150	20061
150-160	18096
160-170	16551
170-180	15223
180-190	13755
190-200	12506

Appendix B

Results for the Class Balancing approach

The following tables show there results for the tests that were performed with the Class Balancing approach for the Twitter Streaming API.

Tests featuring length of the message

Table B.1: Results for testing with tweets with less than 60 characters

Class	Number of tweets	Precision	Recall	F-score
1	231283.10 ± 208.59	90.59 ± 0.05	75.49 ± 0.21	82.35 ± 0.12
2	46490.03 ± 163.02	28.46 ± 0.14	47.51 ± 0.28	35.60 ± 0.14
3	14325.07 ± 104.04	35.72 ± 0.34	45.34 ± 0.46	39.95 ± 0.34
4	7901.80 ± 74.62	53.57 ± 0.48	77.77 ± 0.44	63.44 ± 0.38

Table B.2: Results for testing with tweets between 60 and 100 characters

Class	Number of tweets	Precision	Recall	F-score
1	214280.57 ± 161.55	91.57 ± 0.08	72.89 ± 0.19	81.17 ± 0.10
2	52037.07 ± 136.13	31.58 ± 0.15	49.20 ± 0.33	38.47 ± 0.18
3	22844.30 ± 126.19	37.02 ± 0.30	53.25 ± 0.35	43.67 ± 0.28
4	10838.07 ± 88.53	49.80 ± 0.41	71.21 ± 0.49	58.61 ± 0.33

Table B.3: Results for testing with tweets with more than 100 characters

Class	Number of tweets	Precision	Recall	F-score
1	190662.70 ± 226.12	92.24 ± 0.08	68.27 ± 0.16	78.47 ± 0.10
2	63070.33 ± 201.67	36.55 ± 0.21	53.10 ± 0.26	43.30 ± 0.20
3	33203.67 ± 134.25	39.19 ± 0.27	57.83 ± 0.37	46.72 ± 0.28
4	13063.30 ± 82.16	47.60 ± 0.31	66.59 ± 0.44	55.52 ± 0.32

Tests featuring tweets with no replies

Table B.4: Results for testing with tweets that are no replies

Class	Number of tweets	Precision	Recall	F-score
1	200133.37 ± 250.93	90.93 ± 0.07	65.88 ± 0.15	76.40 ± 0.09
2	59808.37 ± 238.63	32.63 ± 0.18	53.56 ± 0.26	40.55 ± 0.18
3	27544.00 ± 137.97	38.23 ± 0.24	54.01 ± 0.34	44.77 ± 0.23
4	12514.27 ± 108.14	50.34 ± 0.34	72.06 ± 0.42	59.28 ± 0.31

Tests featuring hashtags

Table B.5: Results for testing with tweets without hashtags

Class	Number of tweets	Precision	Recall	F-score
1	220192.53 ± 242.92	91.33 ± 0.05	74.37 ± 0.11	81.98 ± 0.07
2	49847.57 ± 206.18	30.60 ± 0.15	48.15 ± 0.23	37.42 ± 0.14
3	20154.97 ± 136.48	37.26 ± 0.25	52.44 ± 0.32	43.56 ± 0.21
4	9804.93 ± 95.75	51.75 ± 0.38	73.30 ± 0.46	60.67 ± 0.36

Table B.6: Results for testing with tweets with one hashtag

Class	Number of tweets	Precision	Recall	F-score
1	175299.67 ± 109.52	91.22 ± 0.06	62.01 ± 0.20	73.83 ± 0.13
2	69507.27 ± 114.16	37.09 ± 0.10	53.79 ± 0.20	43.91 ± 0.10
3	38808.73 ± 77.86	38.63 ± 0.18	55.76 ± 0.34	45.64 ± 0.19
4	16384.33 ± 59.55	44.64 ± 0.26	65.44 ± 0.48	53.07 ± 0.22

Table B.7: Results for testing with tweets with 1 or more hashtags

Class	Number of tweets	Precision	Recall	F-score
1	184148.63 ± 176.40	91.60 ± 0.06	64.41 ± 0.16	75.63 ± 0.11
2	67959.57 ± 178.20	37.34 ± 0.12	55.72 ± 0.22	44.72 ± 0.13
3	34764.60 ± 150.05	39.17 ± 0.21	56.63 ± 0.41	46.31 ± 0.25
4	13127.20 ± 77.36	45.22 ± 0.36	64.90 ± 0.45	53.30 ± 0.33

Tests featuring URLs

Table B.8: Results for testing with tweets without URLs

Class	Number of tweets	Precision	Recall	F-score
1	204836.93 ± 219.34	90.34 ± 0.07	70.55 ± 0.13	79.23 ± 0.08
2	55794.27 ± 190.15	31.58 ± 0.16	47.61 ± 0.25	37.98 ± 0.17
3	26017.83 ± 121.60	36.83 ± 0.25	51.80 ± 0.31	43.05 ± 0.25
4	13350.97 ± 119.00	50.39 ± 0.34	72.95 ± 0.36	59.60 ± 0.29

Table B.9: Results for testing with tweets with URLs

Class	Number of tweets	Precision	Recall	F-score
1	224815.33 ± 180.79	93.00 ± 0.05	75.51 ± 0.21	83.35 ± 0.12
2	50723.07 ± 163.27	34.09 ± 0.16	55.55 ± 0.32	42.25 ± 0.15
3	18920.37 ± 134.98	40.39 ± 0.36	58.62 ± 0.42	47.83 ± 0.31
4	5541.23 ± 65.79	47.27 ± 0.50	62.69 ± 0.51	53.90 ± 0.38

The following tables show there results for the tests that were performed with the Class Balancing approach for the Twitter Search API.

Table B.10: Results for testing using the number of retweets of the last tweet for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	92.50 ± 0.02	78.33 ± 0.15	84.83 ± 0.09
2	54000.00 ± 0.00	39.62 ± 0.10	57.60 ± 0.23	46.94 ± 0.11
3	24000.00 ± 0.00	54.14 ± 0.24	67.04 ± 0.21	59.90 ± 0.17
4	10500.00 ± 0.00	66.28 ± 0.27	79.98 ± 0.50	72.49 ± 0.32

Table B.11: Results for testing using the number of retweets of the last 2 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	92.60 ± 0.03	78.60 ± 0.17	85.03 ± 0.10
2	54000.00 ± 0.00	39.97 ± 0.17	58.09 ± 0.20	47.35 ± 0.15
3	24000.00 ± 0.00	54.55 ± 0.12	67.11 ± 0.27	60.18 ± 0.14
4	10500.00 ± 0.00	66.71 ± 0.35	79.16 ± 0.48	72.40 ± 0.29

Table B.12: Results for testing using the number of retweets of the last 3 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	92.68 ± 0.03	78.68 ± 0.10	85.11 ± 0.06
2	54000.00 ± 0.00	40.15 ± 0.09	58.56 ± 0.15	47.64 ± 0.09
3	24000.00 ± 0.00	55.03 ± 0.19	67.06 ± 0.28	60.45 ± 0.18
4	10500.00 ± 0.00	66.73 ± 0.43	79.13 ± 0.43	72.40 ± 0.34

Table B.13: Results for testing using the number of retweets of the last 4 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	92.74 ± 0.04	78.78 ± 0.08	85.19 ± 0.06
2	54000.00 ± 0.00	40.30 ± 0.10	58.72 ± 0.10	47.80 ± 0.10
3	24000.00 ± 0.00	55.21 ± 0.12	67.18 ± 0.27	60.61 ± 0.15
4	10500.00 ± 0.00	66.88 ± 0.43	79.22 ± 0.43	72.52 ± 0.36

Table B.14: Results for testing using the number of retweets of the last 5 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	92.79 ± 0.05	78.86 ± 0.14	85.26 ± 0.09
2	54000.00 ± 0.00	40.47 ± 0.13	58.90 ± 0.12	47.98 ± 0.10
3	24000.00 ± 0.00	55.48 ± 0.19	67.47 ± 0.24	60.89 ± 0.05
4	10500.00 ± 0.00	66.88 ± 0.45	79.47 ± 0.54	72.63 ± 0.41

Table B.15: Results for testing using the number of retweets of the last 6 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	92.84 ± 0.04	78.93 ± 0.08	85.32 ± 0.06
2	54000.00 ± 0.00	40.65 ± 0.08	59.12 ± 0.15	48.18 ± 0.09
3	24000.00 ± 0.00	55.62 ± 0.17	67.53 ± 0.10	61.00 ± 0.08
4	10500.00 ± 0.00	66.82 ± 0.42	79.57 ± 0.46	72.64 ± 0.39

Table B.16: Results for testing using the number of retweets of the last 7 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	92.86 ± 0.05	78.97 ± 0.12	85.35 ± 0.09
2	54000.00 ± 0.00	40.73 ± 0.13	59.17 ± 0.19	48.25 ± 0.13
3	24000.00 ± 0.00	55.66 ± 0.20	67.66 ± 0.20	61.08 ± 0.13
4	10500.00 ± 0.00	66.94 ± 0.42	79.74 ± 0.52	72.78 ± 0.41

Table B.17: Results for testing using the number of retweets of the last 8 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	92.89 ± 0.05	79.06 ± 0.12	85.42 ± 0.09
2	54000.00 ± 0.00	40.83 ± 0.14	59.18 ± 0.21	48.33 ± 0.15
3	24000.00 ± 0.00	55.72 ± 0.21	67.78 ± 0.24	61.16 ± 0.17
4	10500.00 ± 0.00	66.96 ± 0.38	79.85 ± 0.43	72.84 ± 0.38

Table B.18: Results for testing using the number of retweets of the last 9 tweets for each user

Class	Number of tweets	Precision	Recall	F-score
1	211500.00 ± 0.00	92.91 ± 0.07	78.91 ± 0.06	85.34 ± 0.06
2	54000.00 ± 0.00	40.75 ± 0.08	59.24 ± 0.19	48.29 ± 0.11
3	24000.00 ± 0.00	55.62 ± 0.14	67.86 ± 0.21	61.14 ± 0.12
4	10500.00 ± 0.00	66.67 ± 0.41	79.93 ± 0.48	72.70 ± 0.40

Table B.19: Results for testing using the number of retweets of the last 10 tweets for each user

Class	**Number of tweets**	**Precision**	**Recall**	**F-score**
1	211500.00 ± 0.00	92.94 ± 0.07	78.95 ± 0.06	85.38 ± 0.06
2	54000.00 ± 0.00	40.83 ± 0.08	59.33 ± 0.19	48.37 ± 0.11
3	24000.00 ± 0.00	55.67 ± 0.19	67.87 ± 0.27	61.16 ± 0.18
4	10500.00 ± 0.00	66.63 ± 0.36	79.97 ± 0.40	72.69 ± 0.32

Appendix C

Paper published in RECPAD 2017

On the importance of Users's Features in Retweeting

Nelson Oliveira[1]
njoukov@student.dei.uc.pt
Joana Costa[12]
joana.costa@ipleiria.pt
Catarina Silva[12]
catarina@ipleiria.pt
Bernardete Ribeiro[1]
bribeiro@dei.uc.pt

[1] Center for Informatics and Systems
Department of Informatics Engineering
University of Coimbra
Coimbra, Portugal

[2] School of Technology and Management
Polytechnic Institute of Leiria
Leiria, Portugal

Abstract

Nowadays Twitter is one of the most used social networks with over 1.3 billion users. Twitter allows its users to write messages called tweets that can contain up to 140 characters. Retweeting is the key mechanism of information propagation. In this paper, we present a study on the importance of different user's features in predicting the probability of retweeting. The resulting retweet predictive model takes into account different types of tweets, e.g, tweets with hashtags and URLs, among the used classes. Preliminary results show there is a strong relation between specific features, e.g, user's popularity, and the retweet model.

1 Introduction

Twitter is a very popular micro-blogging social network founded in 2006. Twitter plays a role of news media when announcing breaking news [3]. In this social network a user can post a message (tweet), which can be shared (retweeted) by other users. Currently, Twitter has over 1.3 billion accounts, but it is estimated that only 550 million had ever made a tweet. Twitter is widely used by social media, brands and celebrities, which can influence their reputation.

Popularity prediction has been studied by multiple authors [6], [5], [7]. Predicting popularity can allow the users to better structure their tweets in order to understand how they can have higher possibilities of making a tweet that can achieve a higher popularity, which can be useful, for instance, to companies [4] that want to extend their popularity in order o increase their reputation and profits.

Some studies investigated which are the factors that are most important to popularity prediction. Suh *et al*. [2] and Janders *et al*. [5] showed that URLs and hashtags have a positive influence, along with the number of followers (user's popularity), which is considered one of the most influencing features.

Other studies focused on building retweet prediction models. Petrovic *et al*. [6] built a retweet prediction model that tells if a tweet is going to be retweeted or not. Janders *et al*. [5] used Naïve Bayes and Logistic Regression to classify viral tweets and non viral tweets using different tresholds for virality, adding sentiment analysis related features. Hong *et al*. [7] used SVM with the Information Gain of the features to predict the popularity of a tweet using multiple classes.

2 Proposed Approach

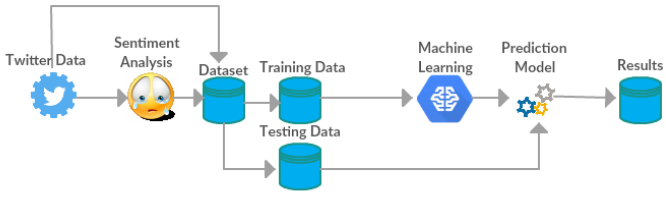

Figure 1: Retweet Prediction Model

The goal of this work is to build a retweet prediction model (Figure 1) and show how different quantity of text features (for example tweets with only 1 hashtag, tweets with less than 60 characters) affect its performance regarding different values of popularity. The text features that we tested were number of hashtags, number of URLs, and length of the text. We used 4 classes as Zhang *et al*. [7] to measure the popularity of a tweet. Class 1 contains tweets that have 0 retweets, class 2 contains tweets that have more than 0 and less than 10 retweets, class 3 contains tweets that have 10 or more retweets and less than 100, and class 4 contains tweets that have 100 or more retweets. Our model uses tweets that were collected with the Twitter Streaming API [1] and performs sentiment analysis of tweets, which were used as training data and testing data. The training data was generated using 2 sampling methods, and fitted to a Random Forests classifier. Finally, we tested the model with different testing sets.

3 Experimental Design

3.1 Dataset description

The dataset used in this work was gathered with the Twitter Streaming API, from 1st to 15th July 2016. The dataset has 12,470,144 (English) tweets. Which are distributed by the 4 classes in the following way: 8,684,496 tweets in class 1, 2,276,806 tweets in class 2, 999,511 tweets in class 3 and 509,331 tweets in class 4. We chronologically divided the dataset into two parts: tweets from the days 1 to 12 were used as training data, and the tweets from the days 12 to 15 as testing data.

3.2 Sampling Methods

For the sampling we used 2 sampling methods to select the training data, the first one we called **Class balancing** and the second one we called **Sub class sampling**.

Class balancing

Based on the unbalanced characteristics of our dataset the first sampling method that we have applied was to select randomly 300k tweets from each one of the classes, in order to have the same number of samples of each class which would not happen with random sampling.

Sub class balancing

In this approach we divided the classes into multiple sub classes, in order to avoid the unbalanced distribution inside each class, as tweets with higher number of retweets are scarcer. This also happens inside of the classes for example if we look at the class 3 which represents the tweets with 10 or more retweets and less than 100, the first interval for which we count the number of tweets within that range (10-20) has almost 10 times more than the range(90-100), so when we randomly select 300k tweets from this class most of the tweets are inside the first interval, which can influence the prediction for the tweets that are near the edges of each class.

3.3 Features

The features that we have used in our study can be divided in two main groups, user features and tweet features. Most of these features can be extracted from the structure of the data directly when it is gathered from the Twitter Streaming API, but we have also added other extra features related to the tweet.

User features are the ones related to the author of the tweet. All of these features already have been used in other studies related to popularity

[1] https://dev.twitter.com/streaming/overview

Table 1: Results for F1-Score of the different experimental tests

Test \ Class	1	2	3	4
Randomly selected tweets without sub class balancing	80.80 ± 0.10	39.45 ± 0.15	44.50 ± 0.21	58.60 ± 0.32
Randomly selected tweets	85.19 ± 0.07	45.74 ± 0.26	57.18 ± 0.31	78.18 ± 0.15
Tweets without hashtags	86.36± 0.05	43.22± 0.24	56.41± 0.21	78.93± 0.52
Tweets with 1 hashtag	78.65 ± 0.14	51.55 ± 0.11	58.21 ± 0.12	75.32 ± 0.26
Tweets with 1 or more hashtags	80.23 ± 0.24	**52.12 ± 0.10**	58.61 ± 0.16	75.72 ± 0.11
Tweets without URLs	83.58 ± 0.12	44.46 ± 0.23	56.38 ± 0.12	78.44 ± 0.36
Tweets with 1 or more URLs	**87.72 ± 0.07**	48.50 ± 0.12	**59.38 ± 0.46**	76.57 ± 0.36
Tweets with 0 to 60 characters	87.04 ± 0.08	39.77 ± 0.29	54.51 ± 0.34	**80.34 ± 0.24**
Tweets with 60 to 100 characters	85.36 ± 0.08	44.76 ± 0.16	56.40 ± 0.35	78.01 ± 0.25
Tweets with 100 to 140 characters	82.62 ± 0.09	50.83 ± 0.07	58.81 ± 0.12	77.08 ± 0.22

prediction. We used the following features: number of followers, statuses, favorites, number of times the user was listed, number of days of the account and if the user is verified.

Tweet features are the ones related to the tweet itself. We used the following features: number of hashtags, URLs, mentions, length of the tweets, number of words, is a tweet a reply, hour of the tweet timestamp. Besides these features we also added the number of videos, number of photos and GIFs. To our knowledge no previous studies used these features. We have also used features related to Sentiment Analysis, which were positive sentiment score (from 1 to 5) and negative sentiment score (from -1 to -5). These values were obtained using the framework SentiStrength [2] , which already was used by Janders *et al*. [5]. The last feature that we used here was a binary one, representing if a tweet contains a trend or not.

4 Results and Discussion

In Table 1, we present the experimental results, the used algorithm was Random Forest , with 300 trees in the forest, and the maximum amount of features to consider when looking for the best split was the square root of the number of used features. The remaining parameters were set to its default values [1]. The first line contains the results when we tested the model with class balancing, and the remaining lines contain the results when we tested the model with sub class balancing, for the different test sets. In our experiments we used the 80:20 ratio for training and testing the model, so in this case since we used 300k of tweets for each class, respectively we tested the model with 300k tweets.

Comparing the global results when we used the sub class balancing, we noticed a big improvement in all of the classes especially in the last one, the reason why this happened was because, as we said the distribution inside each class is also unbalanced, meaning that dividing each class into multiple sub classes helps in better selection of the training samples which lead in a performance of the model.

Analyzing the performance of the model regarding the different text features that we had tested, we showed that some features are more related to certain classes. As an example, shorter texts are more related to tweets that have no retweets (class 1) and tweets with more retweets (class 4), while longer tweets tend to be more related to the classes 2 and 3, where the performance of the model increases with the length of the tweets

In the case of hashtags we can say that tweets that do not have any hashtags are more easily identified when they will not get any retweets. These tweets of the class 1 can also be more related to the tweets that have at least one URL, meaning that regarding other features that we have used, when there is at least one URL the model performs better in respect of the class 0.

In the case when the tweet has one or multiple hashtags the classes 2 and 3 are the ones which have improvements in the performance, while the class 4 gets worse. This can seem strange since the presence of hashtags is associated with popularity. But we must also remember that the number of followers (user's popularity) plays one of the greatest roles when predicting the popularity of a tweet. So in this case we can think that popular users do not need hashtags in their tweets to have more retweets. We can relate this to the performance of the model when we did not use any hashtags regarding the class 4 where most of the tweets were made by users that are more popular than the ones contained in the other classes. This performance was better than when testing with tweets that have at least one hashtag. In the classes 2 and 3 we can see that the model has the same behavior for these 2 classes, by that we mean when comparing the performance when testing with different values of the text features, to the performance of the model, the performance values for both of the classes either increases or decreases. This can mean that the number of retweets between the both ranges of these classes are affected in same way regarding to the different text features and their values.

[2] http://sentistrength.wlv.ac.uk/

5 Conclusions

In this work we have presented an analysis on the importance of different factors, as the user's popularity on a retweet prediction model, using 2 sampling techniques. The results of these techniques shows that dividing each class into multiple samples helps to increase the performance of the model. We also showed that the performance of the model is different regarding each class for tweets with different characteristics, which might suggest that there are some characteristics that are more related to a certain class, for example the best performance achieved by the model regarding the class 4 was when we used tweets with a short text (0 to 60 characters).

Possible extensions of this work can be such as, analyzing the performance of the model adding different user model profiles (i.e users with many followers, users with few followers and many favorites). This will help the different types of users to have their personalized probabilities of getting a certain "range" of retweets regarding the values of the features that they introduce in their text message.

References

[1] URL http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

[2] P. Pirolli B. Suh, L. Hong and E. H. Chi. Want to be retweeted? large scale analytics on factors impacting retweet in twitter network. In *Proceedings of the 2nd IEEE International Conference on Social Computing, SOCIALCOM*, pages 177–184, 2010.

[3] H. Park H. Kwak, C. Lee and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web, number 2 in WWW*, pages 591–600, 2010.

[4] Jean Burgess Merja Mahrt Katrin Weller, Axel Bruns and Cornelius Puschmann. In *Twitter and Society*, pages 293–304, 2014.

[5] Gjergji Kasneci Maximilian Jenders and Felix Naumann. Analyzing and predicting viral tweets. In *Association for Computing Machinery*, pages 657–664, 2013.

[6] S. Petrovic, M. Osborne, and V. Lavrenko. Rt to win! predicting message propagation in twitter. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain*, 2011.

[7] Zhiheng Xu Yang Zhang and Qing Yang. Predicting popularity of messages in twitter using a feature-weighted model. In *International Journal of Advanced Intelligence*, 2012.