UNIVERSITÀ DI PISA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Artificial Intelligence and Data Engineering

Body shaming detection on Twitter

Project Documentation

Martina Marino

Roberta Matrella

Academic Year: 2021/2022

Contents

Abstract

Social media offer great communication opportunities that sometimes can be used in an inappropriate way. One of the widely common misuses of them is cyberbullying and in particular body shaming. Twitter enables millions of active users to send and read concise messages (a maximum of 280 characters) on the Internet every day so, it is a good platform to extract and analyze data. This work shows the data mining process that leads to the building of a machine learning model that is able to detect body shaming on Twitter. Later, the presence of concept drift was verified thanks to an online monitoring phase and an adequate solution to this problem was evaluated by testing several solutions.

1 Introduction

"[The internet] has definitely played a significant role in amplifying the voice of people who are perpetuating the stigma and the bias and the shaming," James Zervios, director of communications for the nonprofit Obesity Action Coalition, told Healthline. [1]

Social networks give users the possibility to communicate with all over the world. However, users often do not use these tools correctly, but as a means to spread unsolicited opinions. Being able to express an opinion from behind a screen makes people more free to write negative comments or insults and this leads to the problem of online bullying, named cyberbullying.

One common form of cyberbullying is known as body shaming and consists in making critical comments about the shape or size of someone else's body. In addition to the important repercussions on self-esteem, studies conducted to date have reported a number of problems related to bodyshaming that can encourage the onset of real mental disorders and harmful behaviors. For example, body shaming has been found to reduce body confidence and to influence eating behaviors, school absenteeism, and increases levels of distress and insecurity. [5]

For these reasons, the idea proposed in this project is to analyze online contents with the purpose of identifying the accounts that publish content that are related to body shaming in order to possibly report them.

1.1 Goals

The main goal of this documentation is to describe the steps to realize body shaming detection from Twitter and to develop the **BSblocker** tool that can run the prediction model in real-time in order to obtain a list of the accounts that shame other people about their physical characteristics. Starting from a web-scraped data filtered with some keywords, a preprocessing phase was carried out for having a suitable dataset to classify. Different classifiers were trained and tested in order to better understand if a tweet contains body shaming sentences or not and the best one was selected according to several considerations. Finally, the objective of the online monitoring stage is to find out if the chosen model performs well even in the long-term prediction or if it suffers from concept drift and, once detected the type of concept drift, to alleviate it retraining the model appropriately.

2 Initial Dataset

The social media exploited for this work is Twitter, a microblogging and social networking service where users post content and interact with posts known as "tweets". Unlike other social platforms, Twitter allows to easily access published contents and to make complex queries to obtain filtered data. The dataset has been obtained from Twitter thanks to a web-scraping phase using the snscrape library of Python specifying the following criteria:

- **language**: italian. Has been taken into consideration only the italian tweets;
- **date of posting**: tweets posted from 1st December 2021 to 1st February 2022;
- **specific keywords** contained into the text of the tweets: the keywords are separated by the or-operator and the set of keywords is composed by the most frequently body shaming used words;
- **replies**, **quotes** and **retweet** have been discarded.

Applying these criteria 75,357 tweets have been collected.

With a first cleaning step has been removed URLs, mentions, new lines and multiple spaces in order to simplify the subsequent manual labeling and, after that, tweets posted more than once from the same user or not containing none of keywords have been discarded obtaining a total of 59,583 tweets.

The entire dataset has been divided in:

- **training set**: December 2021 - January 2022
- **test set**: February 2022

2.1 Training set

From the 37,555 cleaned tweets of the period (1st December, 2021 - 31st January, 2022) some random tweets have been manually labeled with two possible class labels: body-shaming for the tweets with negative physical comments, and non-body-shaming for the other tweets. Every tweet has been read, interpreted and correctly class labeled; in this way have been found out:

- 788 tweets with body shaming label
- 788 tweets with non-body shaming label

At the end, has been obtained an initial balanced dataset composed by 1576 tweets as shown in Figure 1 in which label '0' corresponds to the non-body-shaming tweets and label '1' to the body-shaming tweets.

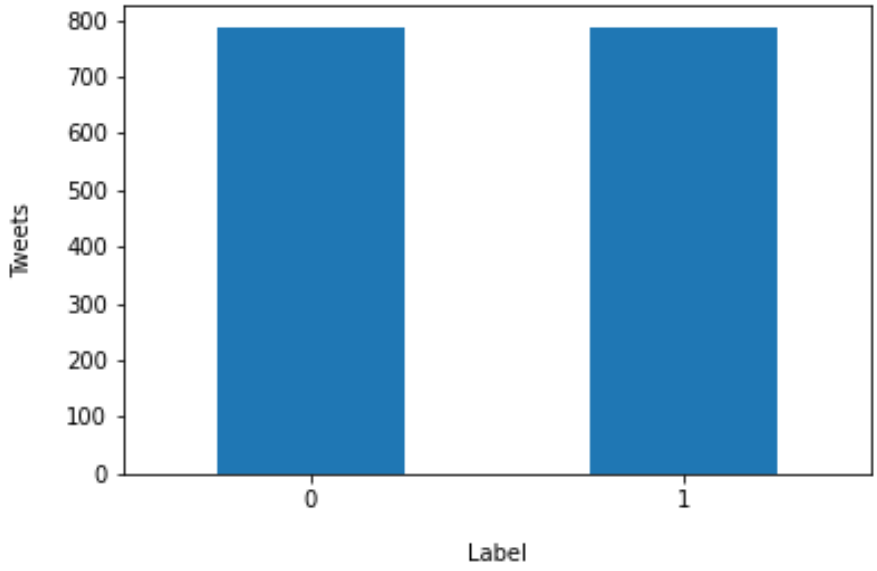

Figure 1: Labeled tweets of the training set

2.2 Preprocessing

After the tweets have been comprehended and correctly labeled some other useless information can be removed. These steps are:

- removing **punctuation**: markers, quotes, brackets, special characters;
- **text reformat**: removing of multiple dots and multiple last letters, lower case of the text;
- **emojis** removal;

At the end of this phase, an Elaboration one follows. The aim of this step is to transform a set of strings (the preprocessed tweets), in a set of numeric vectors ready to be elaborated by the Classification module. These are the steps performed in order to obtain the aimed result:

- **tokenization**: consists of brake the raw text into chunks, denominated tokens. Each token corresponds to a single word. At the end of this step, have been obtained a sequence of words for each tweet;
- **stop-word filtering**: removing of stop-words like articles or prepositions that have no significance for the project analysis purpose;
- **stemming**: is the process of reducing words to their root form;
- remove **miningless words**: all the words with number of characters less or equal than two have been discarded;
- remove **only-digit features**: removal of features composed only by digits

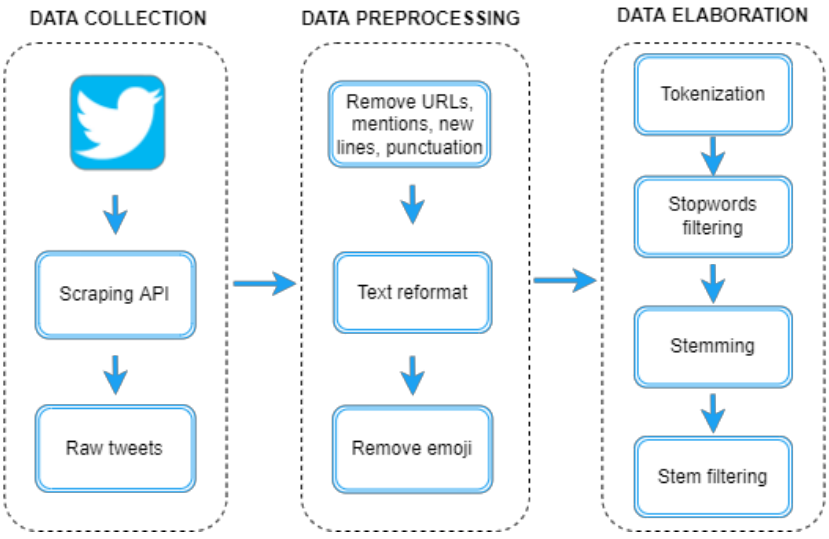

Figure 2: Data fetching and preprocessing

2.3 Feature extraction and selection

The training set has been built starting from the preprocessed tweets, characterized by 788 tweets classified as 'body shaming' and 788 other tweets 'non-body shaming'. Using the *sklearn.feature_extraction* library of Python every model has been integrated as the last step of pipeline composed by the following steps:

- **vectorization** using CountVectorizer() with ngram = 1 the collection of tweets is converted to a matrix of token counts;
- **Tf-Idf**: This weight is a statistical measure used to evaluate how important a word (stem) is to a document (tweet) in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus (data-set);
- **feature selection** using SelectPercentile() with *percentage* parameter equal to 75, a number of features equal to about 3500, with respect to the initial 4669, has been obtained.

For the feature selection phase, different values of percentage have been compared.

	Accuracy	Execution time	Std
ComplementNB	0.722707	2.42 s	0.004399
MultinomialNB	0.721561	1.845 s	0.005267
Logistic Regression	0.720304	4.8848 s	0.003269
SVM	0.717701	11.3515 s	0.004545
Stochastic Gradient	0.699557	1.8741 s	0.004855
Random Forest	0.693331	37.5024 s	0.010199
Gradient Boosting	0.685984	22.5652 s	0.005519
Bagging	0.673037	53.6608 s	0.011945
Ada Boost	0.670812	12.4091 s	0.008493
Decision Tree	0.651520	6.312 s	0.010620
K Nearest	0.574241	3.646 s	0.009663

	Accuracy	Execution time	Std
Logistic Regression	0.722841	6.3229 s	0.002553
ComplementNB	0.720422	4.39 s	0.005183
MultinomialNB	0.719339	4.6538 s	0.005639
SVM	0.718267	24.3714 s	0.005139
Stochastic Gradient	0.695685	3.4955 s	0.006791
Random Forest	0.695615	95.5387 s	0.004888
Gradient Boosting	0.688189	44.3749 s	0.007860
Bagging	0.675505	52.1071 s	0.007506
Ada Boost	0.669984	14.7499 s	0.007770
Decision Tree	0.651143	9.9743 s	0.007162
K Nearest	0.575526	5.684 s	0.009690

Figure 3: Percentage of feature selection compared: 75%, 85%

As shown in the table 3, regardless of the percentage of selected features, the 4 classifiers that perform best are: *Logistic Regression*, *SVM*, *ComplementNB* and *MultinomialNB*. These algorithms have the best accuracy and lowest standard deviation. Considering the execution times all perform well except for SVM that is the slowest.

Changing the feature percentage the accuracy does not change a lot therefore, focusing on *Logistic Regression*, *ComplementNB* and *MultinomialNB*, the standard deviation and the execution times are lower with a value of 75% that allows to lighten the computation. Moreover, in this scenario the best 4 classifiers will be compared with some metrics.

3 Model training and testing

Starting from the features selected in the previous steps, a model prediction phase followed. This has been done, first of all, using the k-fold cross validation technique. It has been applied with a number of folders equal to 10 and repeated more than one time as suggested in [3]. In this work the validation was repeated for a number of iterations equal to 10.

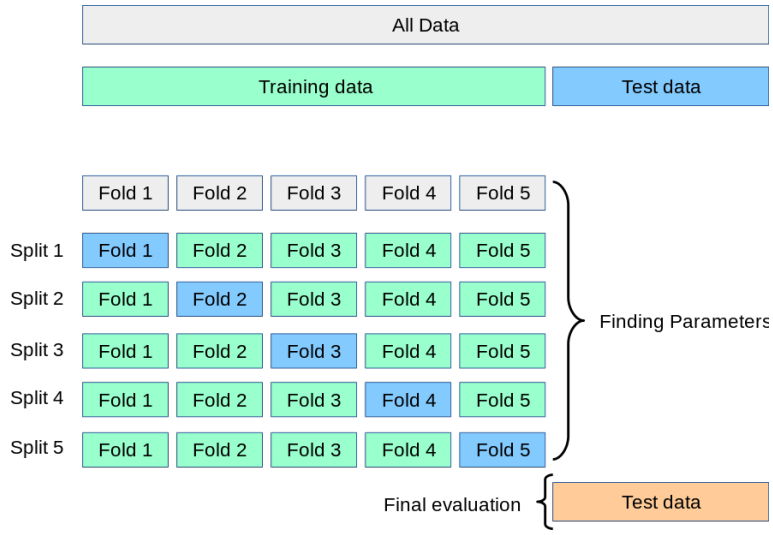

Figure 4: Cross-validation example

The output, composed by ten elements, each one representing the mean of the accuracy of one 10-fold iteration, has been used in order to perform the statistical test.

3.1 Model selection

Some classifiers have been discarded after comparing some parameters shown in 3: *K Nearest*, *Decision Tree* and *Stochastic Gradient* because of their low accuracy compared to the others and *Random Forest*, *Gradient Boosting*, *Bagging* and *Ada Boost* because of their high execution time.
The remaining classifiers have been compared in terms of accuracy (more than 70%), precision, recall and f1-score.

Classifier	Accuracy (%)	Precision (%)		Recall (%)		F1-score (%)	
		0	1	0	1	0	1
Logistic Regression	72.52	71.00	74.28	76.14	68.90	73.48	71.49
SVM	71.76	70.63	73.02	74.49	69.03	72.51	70.97

Table 1: Report comparison *Logistic Regression* and *SVM*

Logistic Regression performs better than *SVM* considering all the metrics (Table 1).

Classifier	Accuracy (%)	Precision (%)		Recall (%)		F1-score (%)	
		0	1	0	1	0	1
ComplementNB	73.03	72.21	73.91	74.87	71.19	73.52	72.52
MultinomialNB	72.39	71.87	72.95	73.60	71.19	72.72	72.06

Table 2: Report comparison of *ComplementNB* and *MultinomialNB*

Comparing the Naive Bayes classifiers, *ComplementNB* is preferred (Table 2).

3.2 Paired t-test

The paired t-test has been performed to compare the four classifiers two-by-two to establish if their performances are statistically different or not starting from a null hypothesis saying that they are the same or, in other words, that the difference in mean error rate between the two is zero. If we can reject this hypothesis, then we can conclude that the difference between the two models is statistically significant, so can be selected the model with the lower error rate (higher accuracy) [4].

The scores obtained are k=10 and the degree of freedom considered are k-1. By computing the t-statistic value and using a significance level $sig = 0.05$ and a confidence limit of $z = 0.025$ (that is half of sig) some considerations can be done comparing t with a value j obtained from the t-distribution table: if $t > j$ or $t < -j$ the null hypothesis is rejected and there is statistically significant difference between the two models, otherwise the hypothesis cannot be rejected and this means that any difference between them is attributed by chance.

Figure 5 shows the result of the paired t-test executed on the 4 different classifiers.

	SVM	MultinomialNB	ComplementNB
Logistic Regression	1.592327	-0.651762	-1.448561
SVM		-2.159805	-3.053071
MultinomialNB			-2.108346

Figure 5: Student T-test result

	Accuracy	Execution time	Std
ComplementNB	0.722707	2.42 s	0.004399
MultinomialNB	0.721561	1.845 s	0.005267
Logistic Regression	0.720304	4.8848 s	0.003269
SVM	0.717701	11.3515 s	0.004545

Figure 6: Performances compared

The pink cells indicate the comparable classifiers while the green ones are related to the non-comparable ones. Reading Table 5 row by row some considerations can be done:

- The difference between *SVM* and *ComplementNB* is statistically significant, but *SVM* has a worse accuracy with respect to *ComplementNB*, so ComplementNB have been chosen;

At the end, *ComplementNB*, *MultinomialNB* and *Logistic Regression* models are taken into consideration and used for the test phase.

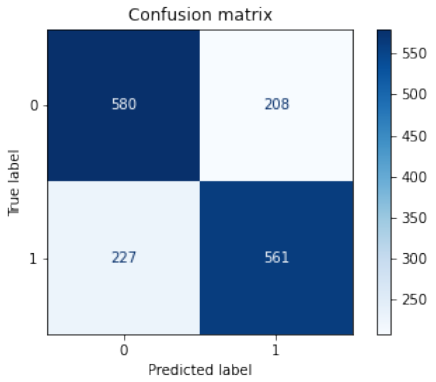

Figure 7: MultinomialNB

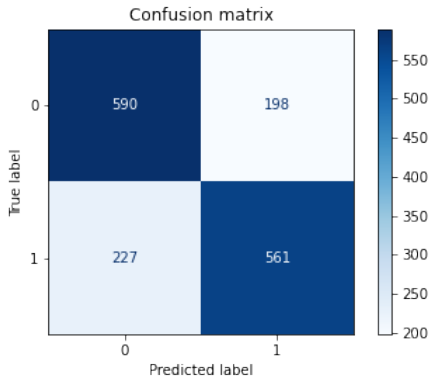

Figure 8: ComplementNB

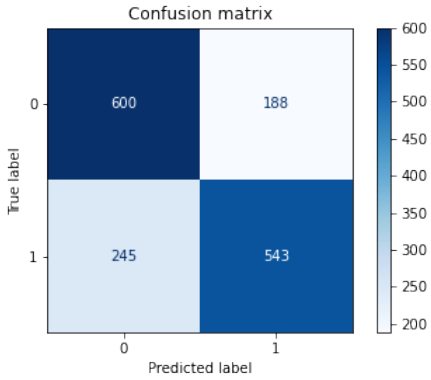

Figure 9: LogisticRegression

3.3 Model testing

After the model building phase, the test phase has been performed. In order to test the performance of the models, 176 tweets from February were randomly and manually labeled, corresponding to a proportion 90:10 with respect to the training set. Testing the models built in the precedent phase, the results shown in Table 3 below have been obtained.

Classifier	Accuracy (%)	Precision (%)		Recall (%)		F1-score (%)	
		0	1	0	1	0	1
ComplementNB	73.29	71.13	75.94	78.40	68.18	74.59	71.85
Logistic Regression	71.59	68.26	76.38	80.68	62.50	73.95	68.75
MultinomialNB	73.29	71.13	75.94	78.40	68.18	74.59	71.85

Table 3: Report comparison of Logistic Regression and ComplementNB

As shown in Table 3, ComplementNB and MultinomialNB performs a little bit better than the training set instead of the Logistic Regression which performs a little worse. ComplementNB and MultinomialNB have the same performance in this phase but ComplementNB have been chosen because of its better performance in the training set phase.

4 Online monitoring

The model just built is now ready to be used to classify the upcoming tweets. Predictive modeling consists in learning a model from historical data and using the model to make predictions on new data where the answer is unknown. For this reason, it is possible that a model built on past data can not be suitable for the analysis of subsequent data.

It is well known from the literature that classification models for analyzing streams of data suffer from **concept drift**. In predictive analytics and machine learning, the concept drift means that the statistical properties of the target variable, which the model is trying to predict, change over time in unexpected ways. This causes problems because predictions may become less accurate as time goes on.

For what concern text mining this problem occurs because people may change the way they express their ideas on social medias according to real world events. The consequence is that the volume of the data collected can be different and the features may change. If these changes are able to be detected, it may be possible to update the learned model to avoid performance loss.

4.1 Identifying peaks

In order to understand if the ComplementNB model, built in the previous phases, suffers from concept drift, some peaks, probably corresponding to some events verified immediately before, have been identified from the overall distribution of the collected tweets in the period following the test (Figure 10).

The goal is to analyze the behaviour of the built model during the days associated with the occur of some events. Indeed, during these days, a change of the terminology used is expected and, for this reason, the corresponding tweets are particularly useful to highlight the concept drift. The objective of this analysis is to adapt the vocabulary used by the model in order to find a way to alleviate the concept drift.

The days highlighted in the diagram in Figure 10 correspond to the peaks chosen for the experiments conducted on the concept drift; **80 tweets** were randomly selected from every peak. Moreover, since no relevant peaks were found in May, 80 tweets were randomly taken from the entire month. The peak corresponding to the 1st of March has been discarded because too close to the test set: in a more central date with respect to the month, as 2022-03-10, the effects of concept drift may be more evident.

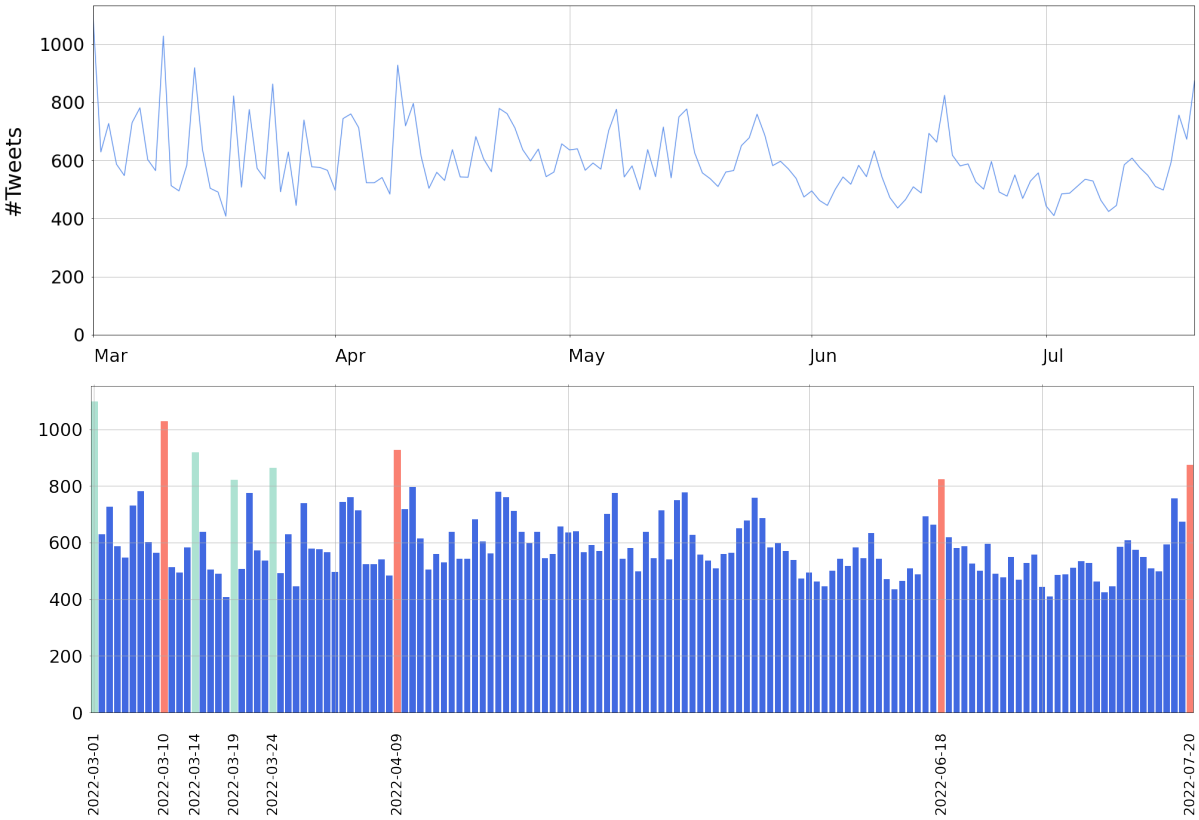

Figure 10: Tweets distribution from 1st March, 2022 to 30th June, 2022

Table 4 describes how many tweets have been collected in every temporal period and the hypothetical events that may have caused an increase in the volume of tweets posted.

Name	Peak/Period	Date/Month	Event	Tweets
Period1	Peak 1	2022-03-10	Tv program GF semifinal	1028
Period2	Peak 2	2022-04-09	Episode of the tv program Amici with Sangiovanni as host	928
Period3	May	2022-05		18959
Period4	Peak 4	2022-06-18	Vanessa Incontrada host of Gigi D'alessio's tv show	824
Period5	Peak 5	2022-07-20	Government crisis	874

Table 4: Peak/period details

4.2 Testing concept drift

As suggested in [2] the 3 different types of concept drift that can be examinated are:

- **static**, in which the model is always the same and it is tested using new tweets;
- **sliding window**, in which the model is retrained adding n new tweets and removing the older n ones;
- **incremental**, that retrains the model adding new labeled tweets also considering all the previous ones.

For each event 80 tweets were randomly selected, manually labeled and used as test set for the **static model** to verify if the concept drift can really be a problem for this work.

Name	Peak	Accuracy (%)	Precision (%)		Recall (%)		F1-score (%)	
			0	1	0	1	0	1
Period1	Peak 1	73.75	74.35	73.17	72.50	75.00	73.41	74.07
Period2	Peak 2	72.50	70.45	75.00	77.50	67.50	73.80	71.05
Period3	May	73.75	72.09	75.67	77.50	70.00	74.69	72.72
Period4	Peak 4	66.25	64.44	68.57	72.50	60.00	68.23	64.00
Period5	Peak 5	61.25	60.97	61.53	62.50	60.00	61.72	60.75

Table 5: Static model

For incremental and sliding approach, the training set have been updated basing on the behaviour of the different approaches.

The **sliding approach** provides a new model for each new chunk of tweets, discarding the oldest and adding the newest, following the window dimension. The window dimension chosen was of 80 tweets. For each peak, the new 80 labeled tweets were inserted into the training set and the same number of the oldest tweets were removed. In this way, the number of tweets considered for the training set was always the same. The model is retrained each time considering the new tweets and the number of features changes during the various peaks.

Name	Peak	Accuracy (%)	Precision (%)		Recall (%)		F1-score (%)	
			0	1	0	1	0	1
Period1	Peak 1	73.75	72.09	75.67	77.50	70.00	74.69	72.72
Period2	Peak 2	76.25	70.58	86.20	90.00	62.50	79.12	72.46
Period3	May	71.25	68.08	75.75	80.00	62.50	73.56	68.49
Period4	Peak 4	71.25	69.76	72.97	75.00	67.50	72.28	70.12
Period5	Peak 5	65.00	64.28	65.78	67.50	62.50	65.85	64.10

Table 6: Sliding model

The **incremental approach** foresees a new retrain of the model each time basing on a training set gradually incremented with the previous examples. The approach adopted is not purely incremental but it is an approach that re-evaluates the model at every new peak. At each peak the training set have been incremented using the tweets of the previous peaks. In this way, at the end, the dimension of the training set goes from 1576 to 2152. With this approach, as consequence of the increasing of the training set dimension, the number of features increases too.

Name	Peak	Accuracy (%)	Precision (%)		Recall (%)		F1-score (%)	
			0	1	0	1	0	1
Period1	Peak 1	71.25	68.88	74.28	77.50	65.00	74.94	69.33
Period2	Peak 2	73.75	69.38	80.64	85.00	62.50	76.40	70.42
Period3	May	72.50	70.45	75.00	77.50	67.50	73.80	71.05
Period4	Peak 4	70.00	69.04	71.05	72.50	67.50	70.73	69.23
Period5	Peak 5	66.25	64.44	68.57	72.50	60.00	68.23	64.00

Table 7: Incremental model

As it is possible to observe in the (Figure 11), the more time goes on the more the model designed with the static approach does not perform well. The other two, indeed, perform better during the time. Moreover, it is possible to observe that the static model has decreasing performances during the peaks but performs not so bad in May in which there is no event. The sliding, however, performs better than the incremental during all the peaks except for May and the last peak.

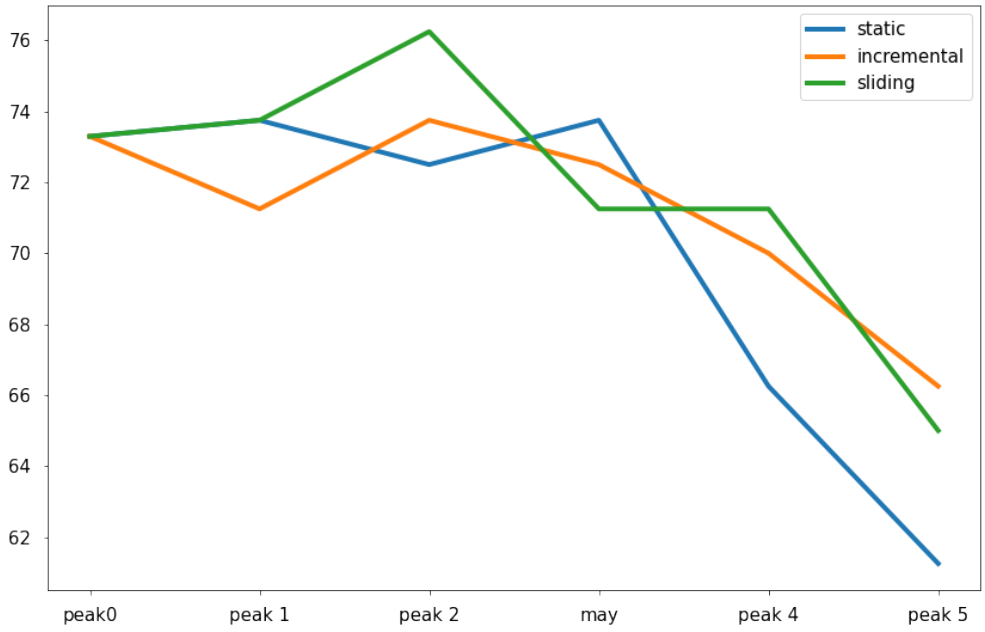

Figure 11: Accuracy for the three approaches

The sliding approach results the one with the best performance except for the last peak. At the same time, the number of features of the incremental model increases quickly. On the contrary, the number of features using the sliding approach are always very low.

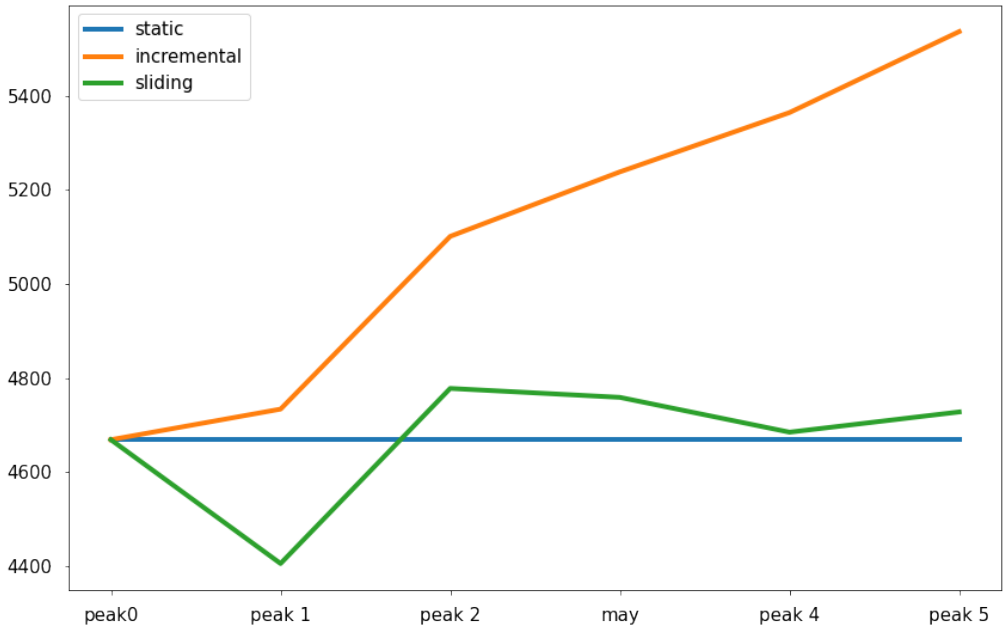

Figure 12: Number of features for each approach

At the end, the sliding approach have been chosen. Indeed, the performance with this approach are mostly the best in the peaks and at the same time the number of features is low. However, in the last peak its performances decrease quickly and, for this reason, there is the possibility that in a long period the incremental approach will be more suitable than the sliding. The decrease in performance could be due to the fact that all the precedent peaks concern television programs events whereas, the last one, concerns a political event. It is possible that, considering an additional time period, the incremental approach will reveal to be the best choice.

5 Application

The application provides a body-shaming detection service.

5.1 Use cases

This application provides a service of real time body shaming detection. It scrapes tweets from Twitter and, using the model built in the precedent steps, evaluates which user is doing body shaming in the time period selected by the user. The application has been thought as a simple tool that a Twitter moderator can use to detect a shamer user in order to block him. Indeed, automatically, the tweets scraped are saved into a csv file and, if a user has been found more than a specific number of times, he/she is inserted into a blacklist.

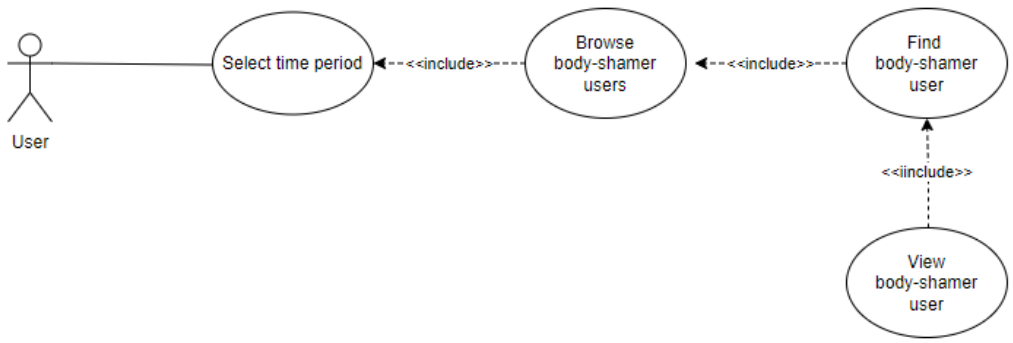

Figure 13: Application use cases

5.2 User Manual

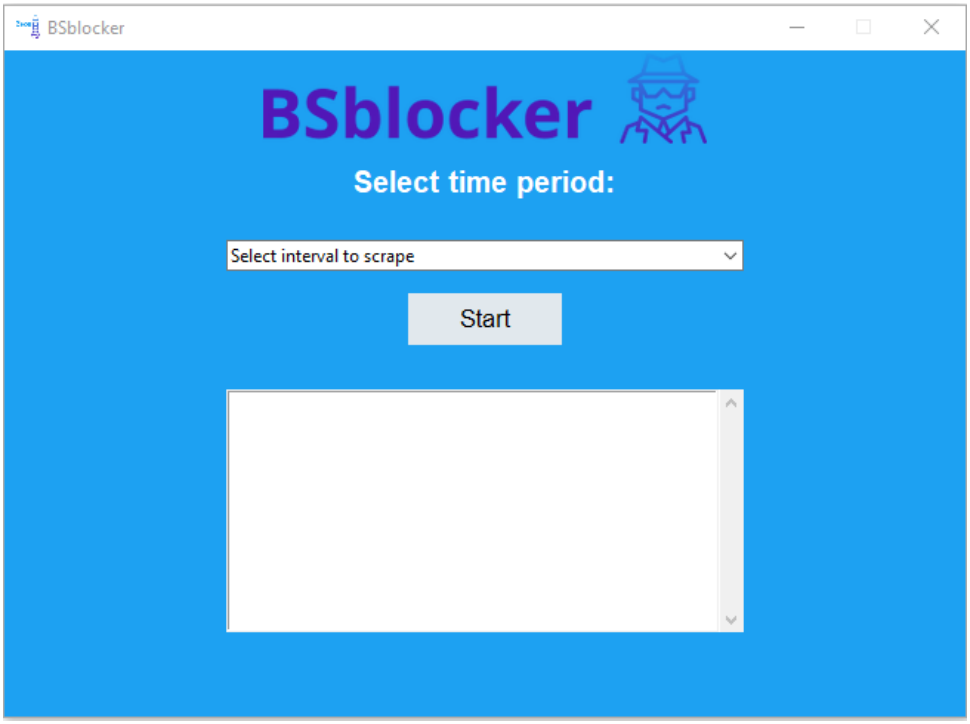

Figure 14: Application home page

The user can select a time period in which the application will scan the tweets in order to find out the users that have been done body-shaming. After selecting the period, it is only necessary to press the 'Start' button to let the process start. When the detection ends, the name of the users that have done body-shaming appears in the scroll bar below.

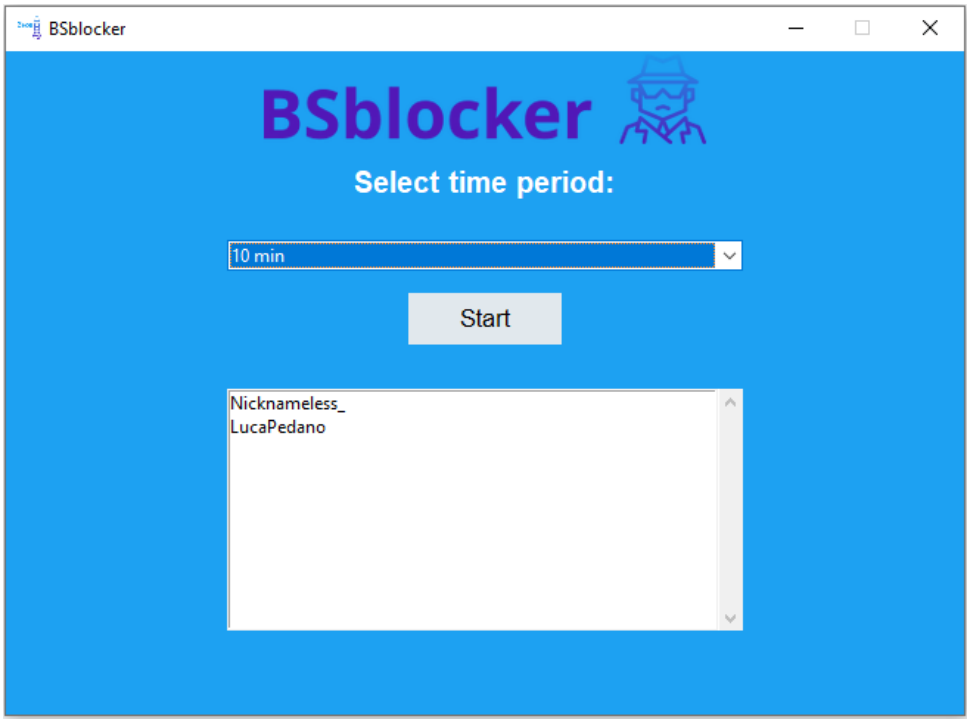

Figure 15: Detection result

When one of the shown users has been detected more than 5 times, it is colored in red.

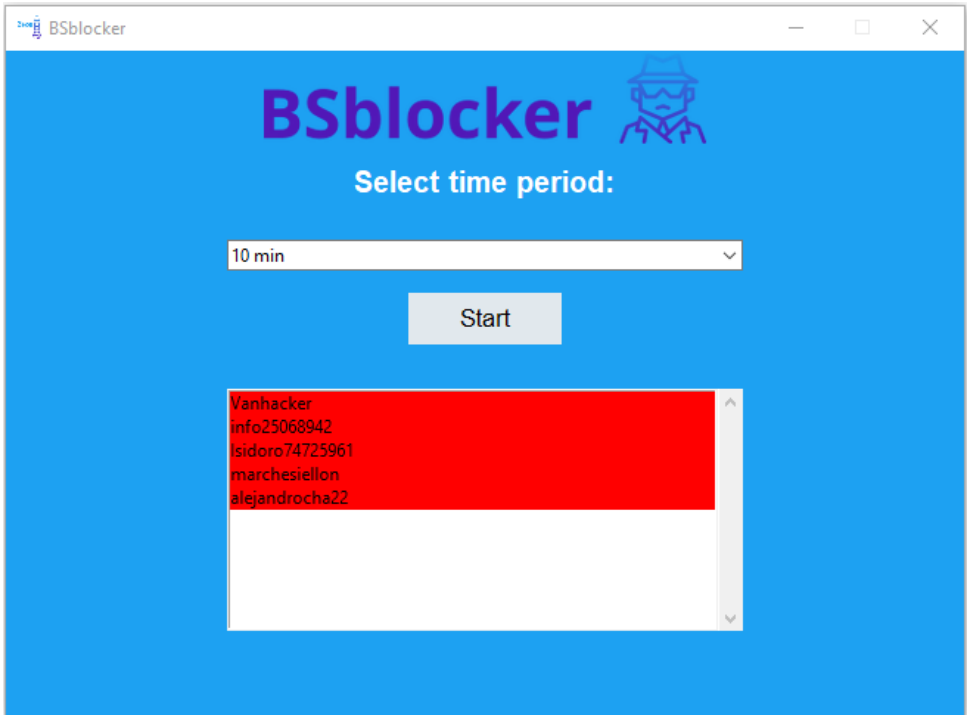

Figure 16: Users in blacklist

5.3 Conclusion

In conclusion, the work described in this documentation has provided satisfying results and a valid model for the body shaming detection. Moreover, an application for this purpose, for the real time body shaming detection, has been developed in order to let a user, more specifically a twitter moderator, detect users doing body shaming.

Surely, the model could be updated and the best approach to cope with concept drift may be re-evaluated, in order to maintain good performances.

To visit the repository click here.

References

[1] Carolyn Abate. Body Shaming in an Age of Social Media. *Healthline*, 16(4):2269–2283, 2015.

[2] Alessandro Bondielli, Giuseppe Cancello Tortora, Pietro Ducange, Armando Macri, Francesco Marcelloni, and Alessandro Renda. Online Monitoring of Stance from Tweets: The case of Green Pass in Italy. In *2022 IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–8, 2022.

[3] Eleonora D'Andrea, Pietro Ducange, Beatrice Lazzerini, and Francesco Marcelloni. Real-Time Detection of Traffic From Twitter Stream Analysis. *IEEE Transactions on Intelligent Transportation Systems*, 16(4):2269–2283, 2015.

[4] Jiawei Han, Micheline Kamber, and Jian Pei. *Data Mining Concept and Techniques*. Morgan Kaufmann, 2012.

[5] C. Schlüter, G. Kraag, and J. Schmidt. Body Shaming: an Exploratory Study on its Definition and Classification. *Int Journal of Bullying Prevention*, 2021.