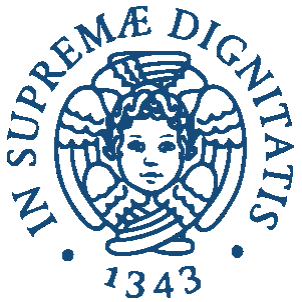


Master Degree in Computer Science
Curriculum Artificial Intelligence

Data Mining
A.Y. 2022/2023

Tweets analysis report

GROUP 11

Alessandro Capurso (638273)
Alessandro Dipalma (626428)
Martina Melero Cavallo (639305)

1 Data understanding and cleaning

1.1 Preliminary data understanding

First of all a preliminary analysis of the data has been performed. Starting from the users, a high concentration of null values can be easily noticed in the `statuses_count` field (399 records). Users are divided in two macro categories, 'bot' and 'genuine user', and the distribution of them is balanced. In particular, there are 6116 bots and 5392 genuine users. In figure 1 the column `statuses_count` has been plotted using a boxplot and an histogram.

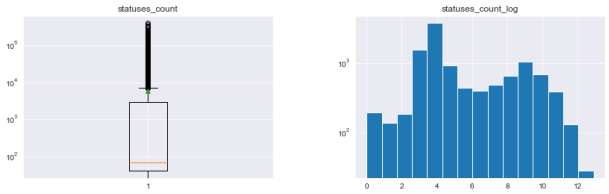


Figure 1: Plots of `statuses_count` in logarithmic scale.

From these plots it can be inferred that there is a large number of outliers that need to be evaluated. Computing the median and the mean of the column, which are 5883 and 68 respectively, it is also possible to infer that there are huge values that lead to a discrepancy between the two statistical indicators. In fact, the maximum value is 399555.

The other interesting attribute of the users is `lang`, which does not have any null value, but printing the unique values of the column is clear that there are similar values like **en-gb**/**en-GB** and **zh-tw**/**zh-TW**. In addition to the misspelled values, there were also two wrong classes of values named "*Select Language...*" and "*xx-lc*" related to three users in total.
The last interesting user attribute to analyze is `user_subscription` (prev `created_at`). It has a lot of outliers. Analyzing the min and the max values, respectively '2012-01-24' and '2020-04-21', both values result plausible.
As final user analysis a correlation matrix has been computed. From the correlation matrix, it is evident that there are no correlations between fields but it could be due to the noisy data.

For what concerns the tweets data, a similar analysis has been performed. As first step, an evaluation of the null values all over the columns has been performed. This analysis showed that all columns, except for `publication_date` (prev `created_at`), contain several null values. After that, all the 'counter' columns have been taken into account and for each of them an outlier analysis has been performed.

By analyzing the boxplots it was possible to notice that the distributions of all the columns are flattened towards the zero values. In fact, also the histograms indicate a high concentration of zeros w.r.t. other values.

The last considered attribute is `publication_date`. From the plots in figure 2 it is possible to notice that there are three different distributions. The oldest date in the column is "1953-04-17" and the most recent one is "2040-04-27"; considering that the creation of Twitter dates back to 2006-03-21, this could mean that one or more distributions are noise.

As done previously for the users, the correlation matrix was also calculated for tweets and also here there are no high correlations between values.

The last check that has been performed is a comparison between the `user_id` in the tweets table and `id` in the user table. This check showed that there are 210778 different `user_id`s

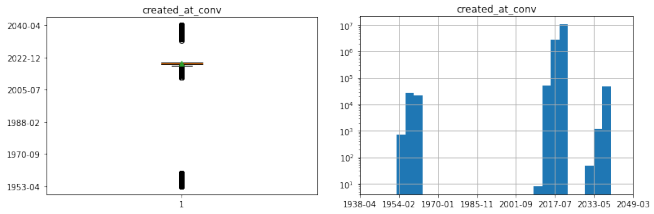


Figure 2: Plots of `publication_date` in logarithmic scale.

that do not have a match in the user table.

1.2 Data cleaning

The data cleaning phase is focused on the correctness of the data.
As first step, all the records with both `user_id` and `text` which are null have been removed because they cannot be used to analyze user behaviour, nor to perform any kind of topic analysis over the text; moreover, the records corresponding to this criteria had a lot of null values in the other columns. Next, an analysis of duplicates on tweets and users records has been performed. Once the dataset was duplicates-free, the analysis and the cleaning of the users and tweets attributes have been performed.
The cleaning task is intended as the substitution of all the missing/wrong values via statistical techniques. In this last phase the median value has been used due to the presence of noise.

1.2.1 Duplicates removal

For the users analysis two users are considered duplicated if and only if they have the same `id`. Fortunately the dataset does not contain duplicated users w.r.t. the this definition.

The duplicates removal process for tweets, instead, was a more complex task. As first step, all exact duplicates have been removed from the dataset (almost 2 millions). Next, a quick check on the dataset revealed the presence of a considerable amount of null values spread among all the attributes. So, a more sophisticated duplicate retrieval process was needed: different subsets of attributes were considered to define different kinds of duplication. For each subset a specific substitution or deletion strategy was applied. These subsets do not include the tweet `id`, since the attribute has no associated semantic that could serve as error correction criteria.

`user_id`, `text`, `publication_date`: About 10% of the data consisted of duplicates under this criteria. Multiple records with the same value of these three attributes are certainly duplicates since we assume that it is not possible that a user publishes more tweets at the same second. For the other tweet attributes, null values are spread among the different copies of the same record, so the copies are merged into a single one keeping the non null value of the counters. In this step 1047351 records were dropped.

`text`, `publication_date`: It is certainly possible that two different users post the same text at the same time, but duplicates w.r.t. these two attributes deserve a check because it could be the case that one of the copies is identical, but with noisy `user_id`. A `user_id` is considered noisy if it is null or contains some alphabetical character. The duplicates removal proceeded as in the first point, merging the records removing the largest possible amount of nulls over all the attributes. In this step 323404 records were dropped.

`user_id`, `text`: The duplicates according to this criterion are more than a million, which

is a considerable amount and is worth further investigation. It's surely possible that a user, especially if it is a bot, tweets many times the same text. What has been checked is that among these duplicates all the dates are valid, which is, in a range that goes from the Twitter foundation up to September 2022. If the `publication_date` is valid, we keep the tweet. Otherwise, the duplicate is considered a noisy duplicate and is removed. In this step 97602 records were dropped.

`users_id`, `publication_date`: Based on the previously made assumptions, it is not possible that a user publishes two different tweets at the same time. Among these duplicates, about half of the records had null `text`, which likely is symptom of them being noisy duplicates. The duplicates removal proceeded with the "merge" procedure as in the first point. Only duplicates with different and non-null `text` were kept. In this step 371323 were dropped.

From now on it is assumed that in the dataset there are no duplicates.

1.2.2 User attributes cleaning

All the user attributes have been explored and only two of them needed a cleaning phase. The attributes are:

- `statuses_count`: Since there were some null values in the column, an estimate of the missing values was needed. The estimate has been performed using the median value of the other users. After that, an analysis about outliers has been performed but all the values seemed to be reasonable.

- `lang`: For the language field there were some values that were clearly misspelled. These values have been replaced as "en-gb" to "en-GB" and "zh-tw" to "zh-TW". Regarding the wrong values, to understand the impact of the possible changes, the number of tweets contained in the tweets file have been taken into account. The accounts seem to be active and have a lot of related tweets. The attribute is categorical so this value could be replaced by the mode which is the "en" language. Since there are only three users, it was possible to double-check the validity of the choice, so the relative tweets have been analyzed. The check confirmed that "en" was the right choice.

1.2.3 Tweet attributes cleaning

Unlike for the user attributes cleaning, all the columns of the tweets were noisy. For the `id` attribute, there were only two missing values. Since the column was not meaningful for the analysis phase, the column has been dropped.

In the `user_id` column there were a lot of missing values (non-numerical ids) and over 100k user ids did not have any correspondence with the users file. Nevertheless, these records have been kept because the other attributes could be useful for future analyses.

In order to exclude all the possible noisy values of the "counter" attributes, a preliminary analysis was required. Due to the shape of the distribution it was not possible to apply a direct cut of the data points over a specific quantile. To solve this problem a different type of thresholds definition was performed. The thresholds were defined by progressively cutting off the data points over a certain value and observing the boxplots until a dense tail was observed.

In the `retweet_count` column some huge values have been noticed. The threshold used for this attribute was $5e5$. All the missing values and the values over the defined threshold were replaced by the median value of that column w.r.t. the same user. If the user doesn't have any other tweet in the dataset the median w.r.t. its class (bot/genuine) is used. The

`reply_count` and `favorite_count` column were handled like the `retweet_count` one. The only difference was in the threshold value: for `reply_count` the threshold defined was $2e6$ and for `favorite_count` the threshold was $5e5$.

Analyzing the `publication_date` column, it was observed the some dates predated Twitter's foundation and others were after 2022. All these values have been substituted using the column's user-wise median. An additional check was performed to verify that no users have tweets before the `subscription_date`, and no inconsistent `publication_date` was found.

Finally the last three columns `num_hashtags`, `num_urls` and `num_mentions` have been fixed. For the number of hashtags the max number that is possible to compose in a limited dimension of characters (140 before 2017 and 280 after [1]) has been taken into account as threshold for the noisy data detection. After that all the missing and noisy values have been fixed. The same procedure has been applied to `num_mentions` and `num_urls`.

1.3 Data understanding post

Now that the data is clean, it is possible to perform a more in depth data understanding analysis on the different attributes.

First, the correlations among the attributes of the tweets and of the users were analyzed. Even after the process of data cleaning, no relevant correlation is present neither in the tweets table nor the users one, so all attributes can be kept without having any redundancy.

Different analyses were performed on the **users** table, first of all the scatter matrix of all numerical attributes was plotted in figure 3. The only pair of attributes is composed of `statuses_count` and `user_subscription_in_secs` differentiated between bots and genuine users; very few bots have higher values of number of posted statuses, mostly the non-bot users have greater values.

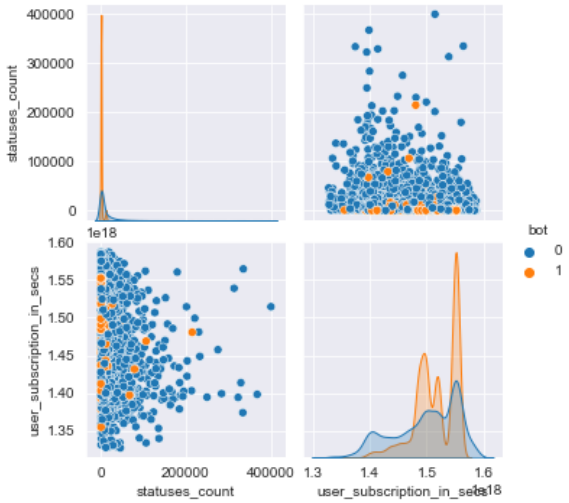


Figure 3: Users scatter matrix.

Another analysis was focused on checking how the language of the user changes varying the date of creation of the user profile. From figure 4 it is possible to see that:

- Almost all profiles from users who are bots were created between 2014 and 2019;
- All users who have their languages set to either "it" or "zh-cn" are bots;

- All users with "ar", "el", "da", "pl", "en-AU", "fil", "sv" and "zh-TW" are genuine users.

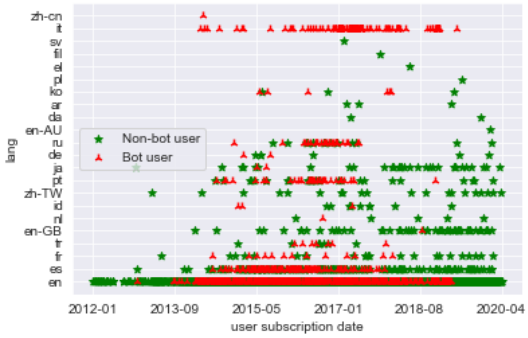
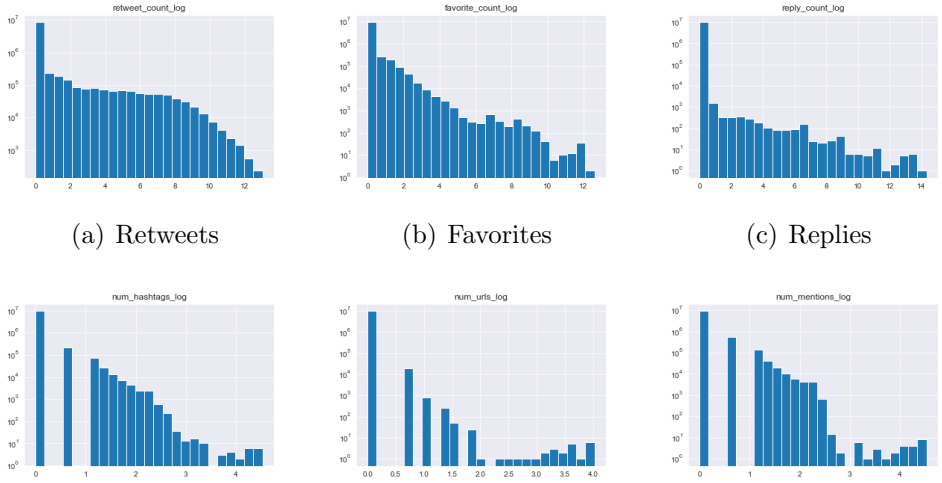


Figure 4: Scatter plot of users' profile creation date and language

After the cleaning phase, for each attribute of the **tweets** table, the histograms in figure 5 were analysed. Even after the process of data cleaning, the distributions remain skewed, with a majority of values tending towards zero.



(a) Retweets (b) Favorites (c) Replies

(d) Num_hashtags (e) Num_urls (f) Num_mentions

Figure 5: Histograms of tweet attributes. The values are in logarithmic scale.

In order to have a clearer view of the data, a merge between the two datasets has been performed to analyze the attributes of the tweets w.r.t. the "bot" class. This made it possible to study the behavior of the users who are bots and the ones which are not. The merge has been performed excluding tweets published by users not present in the users table.

- **retweet_count:** For the most part, genuine users appear to have higher values of retweet count, which makes sense since they are more likely to be shared by other users, unlike tweets coming from bots which could often be spam and thus not be noticed by the Twitter community (see figure 6).
- **favorite_count** and **reply_count:** When considering the favorites and replies, there is not much of a difference between bots and genuine users (see figure 7).

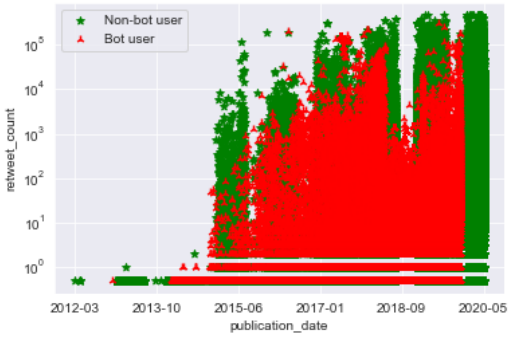
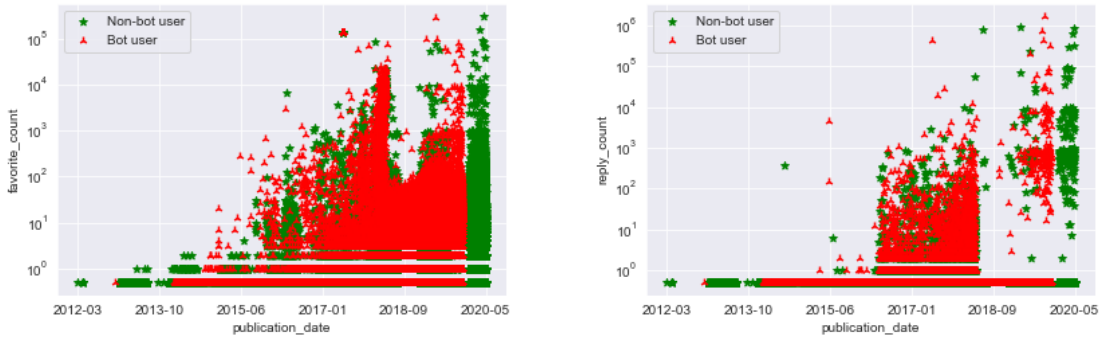


Figure 6: Scatter plot of number of retweets w.r.t. date of creation of the tweet

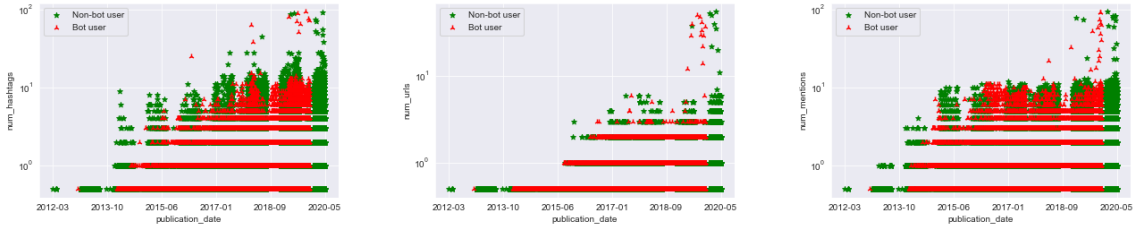


(a) Scatter plot of number of favorites w.r.t. date of creation of the tweet

(b) Scatter plot of number of replies w.r.t. date of creation of the tweet

Figure 7: Scatter plots of attributes w.r.t. the date of creation of a tweet.

- **num_hashtags**, **num_urls** and **num_mentions:** For these attributes the trend is similar between bots and genuine users as well (see figure 8). Some patterns can be noticed which repeat over time.



(a) Number of hashtags

(b) Number of URLs

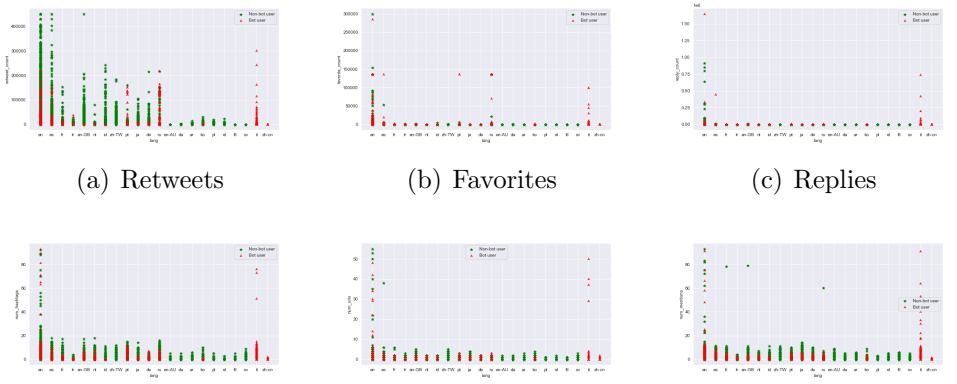
(c) Number of mentions

Figure 8: Scatter plots of attributes w.r.t. the date of creation of a tweet.

Finally, it is noticeable how in the last period of time there are no more tweets by bot users.

From the the plots in figure 9 it is possible to see for each attribute if the users with the highest values are bots or not and which language their profile is set to:

- Users with highest numbers of `retweet_count` are non-bots whose languages are "en", "es" and "en-GB";
- For `favorite_count`, apart from "en", we have a prevalence of bots in languages as "es", "pt" and "ru", but in the last case it has a tie with a genuine user;
- For `reply_count` the highest value belongs to a bot with "en" as language;
- For `num_hashtags`, `num_urls` and `num_mentions` we have mostly users who post in English ("en") but they are pretty varied among bots and non-bots, and some bots who post in Italian ("it");



(a) Retweets (b) Favorites (c) Replies

(d) Number of hashtags (e) Number of URLs (f) Number of mentions

Figure 9: Scatter plots of attributes w.r.t. the language of the user

2 Indicators

After the data understanding and data cleaning phases, the data was ready to extract useful *indicators* from it. These new features are derived from the data already present in the dataset, but they give additional knowledge about some aspects of the users, which can then be exploited to perform insightful analyses.
The extracted indicators have been partly taken from the project description, but also new original ones were formulated. First, in the following list, the indicators that were given in the provided project description:

- Total number of tweets published by each user;
- Total number of tweets published by each user in a given period of time, specifically the tweets have been counted for every year present in the data (i.e. the period of time since the beginning of 2012 up until the end of 2020);
- Total number of tweets;
- Total number of favorites and replies;
- Ratios between the number of tweets and number of favorites;
- Average length of tweets per user;

- Average number of special characters in the tweets per user (all characters that weren't a letter, a digit or whitespace were considered as special characters);

To further describe the dataset, the following global indicators were computed:

- *Total number of retweets* present in the data, to go alongside the total number of favorites and replies computed earlier;
- *Ratios* between:
  1. the number of tweets and number of retweets;
  2. the number of tweets and number of replies;

These ratios show how much retweets and replies there are w.r.t. the total number of tweets of the data.
Among the *original features* derived in this project there are:

- *Mean*, *standard deviation*, *entropy* and the *total sum* for each numerical attribute in the tweets file for each user, the attributes considered were: `retweet_count`, `favorite_count`, `reply_count`, `num_hashtags`, `num_urls`, `num_mentions` and `publication_date`.
  These statistical measures are useful to get a descriptive view of the properties of these attributes by user. In this project's case, the *mean* will retrieve the average of these attributes to get an idea of the behavior of each user, both in the popularity of their tweets, the way they write and the dates they were active in. Instead, the standard deviation is employed to map how inconsistent the user is in their behaviours.
- *Tweeting regularity*, used to check if each user has a certain regularity in the publication (i.e. they post tweets in regular intervals of time). It is defined as:

$$entropy(\{timestamp_i - timestamp_j \mid j = i + 1\}) \tag{1}$$

  where the timestamps are the values in the `publication_date` column, in ascending order.
- *Maximum number of tweets in a day* by user. The number of tweets published is counted for each day, then the maximum count is saved.
- *Densities*. The density for a given attribute was computed as

$$density_{attr} = \frac{total\_amount_{attr}}{user\_activity\_period} \tag{2}$$

  Where `user_activity_period` is defined as the time span from the user subscription (or their first tweet if this information is not available) to their latest tweet.

2.1 Data understanding of indicators

Before moving on to the clustering phase, an analysis on the new extracted features was also be performed. Multiple positive, but also some negative, high correlations were present in the users' indicators correlation matrix. A high positive correlation has to be taken into consideration because it means that the two features represent a very similar aspect of the data, and thus introduce redundancy in the dataset. The threshold to consider two features highly correlated was 0.80 Pearson index.
Following this assumption the features that have been cut off are:

- All features which represented the ratios (`retweet_count_ratio`, `favorite_count_ratio`, etc.) because they are highly correlated with the respective attribute's means;
- All densities, the only one chosen to be kept was `n_tweets_density`, because all the others were correlated among themselves;
- All entropy attributes apart from `reply_count_entropy` and `publication_date_in_secs_entropy` because they were correlated among themselves and with `tweeting_regularity`;
- `reply_count_sum`, highly correlated with the same attribute's standard deviation;
- `num_mentions_sum` because it was correlated with the tweets of the year 2020.

`tweeting_regularity` was kept, despite its high correlation with `n_tweets`. The decision was taken given that the semantic of the two attributes is radically different, and the `tweeting_regularity` was supposed to be a very useful indicator to discriminate bots.

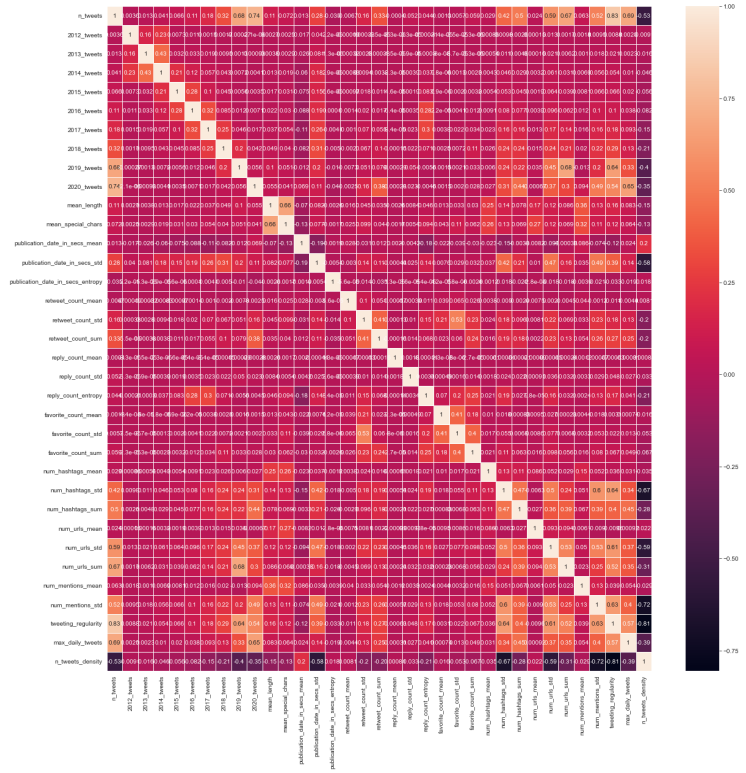
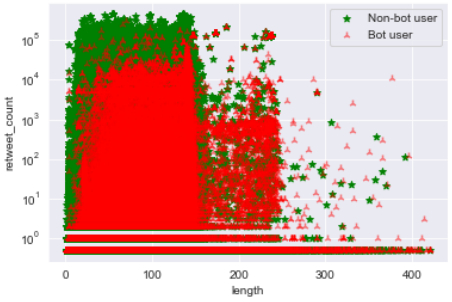


Figure 10: Correlation matrix of indicators after removing redundant columns

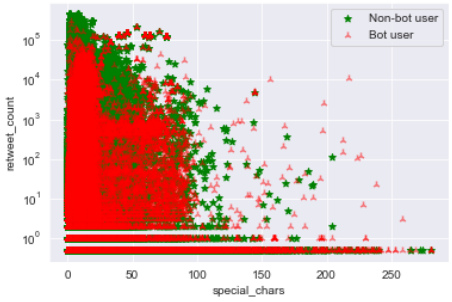
The following observations can be made:

- the length of the tweet w.r.t. the number of retweets, revealing that tweets which have a length shorter than 150 characters have an higher retweet count (see figure 11(a)). Some tweets have a length longer than the maximum number of characters allowed per tweet (the maximum is currently of 280 characters), but it likely is a consequence of the presence of special characters which get formatted erroneously as multiple characters;
- the tweets with more retweets have few special characters in the tweet (see figure 11(b));

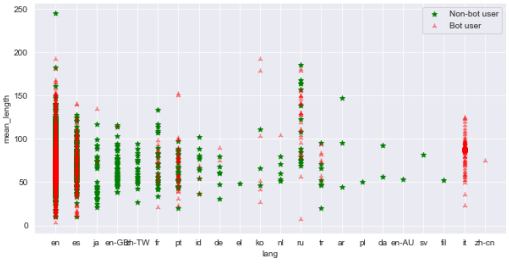
- figure 11(c) shows that the language of the user w.r.t. the average length of the tweet for each user it is possible to see that users with the higher length of tweets have their language set to "en", "ko" and "ru". The ones with a shorter length in average are for example "pl", "el", "fil" and "en-AU";
- from the same plot, it is noticeable how, mostly, users who are and aren't bots behave similarly, only in "pt" and "ko" we have bots who post with a length averagely higher than most other users with that language;
- for the number of special character, instead, the languages with the highest values are "en" and "ko". Less special characters are used instead by users of languages "el", "fil" and "da". In most languages there is no difference between bots and genuine users, but in languages "ja", "pt", "es" and "ko" the tweets with most special characters are from bots (see figure 11(d));
- w.r.t. the language of the user and the total number of tweets published by that user. The languages "el", "pl", "en-AU" and "fil" have only users with a high number of tweets, unlike "zn-ch" bots. Other languages have users with an higher variance, thus covering both the case of having few tweets or, instead, a lot of them (see figure 12(a));
- in figure 12(b) it is possible to see how, as time passed, more and more tweets have been published, maybe because the popularity of Twitter increased .
- The most populated language are English (9973), Italian (906) and Spanish (319).



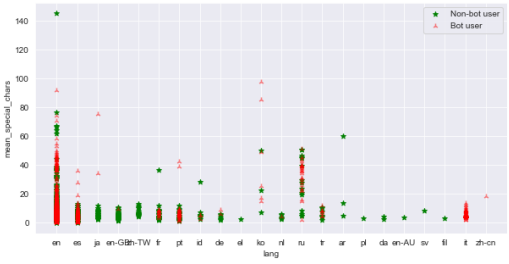
(a) Retweets w.r.t. length (in log-scale)



(b) Retweets w.r.t. special chars (in log-scale)

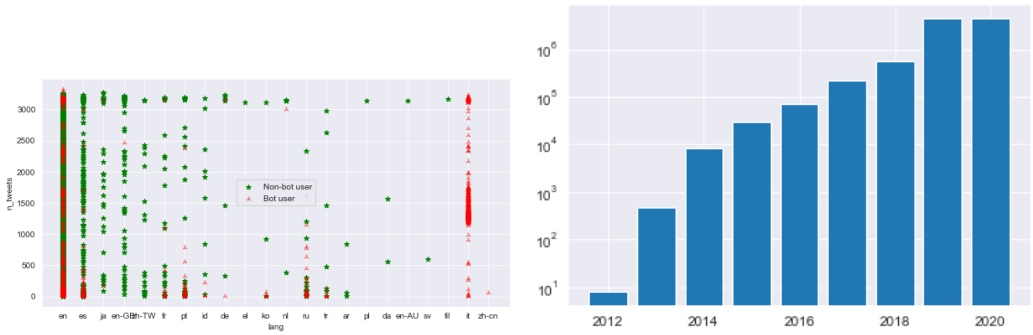


(c) Mean length w.r.t. language



(d) Special chars w.r.t. language

Figure 11: Data understanding plots after the data cleaning process.



(a) Number of tweets w.r.t. language (b) Number of tweets by year

Figure 12: Data understanding plots after the data cleaning process.

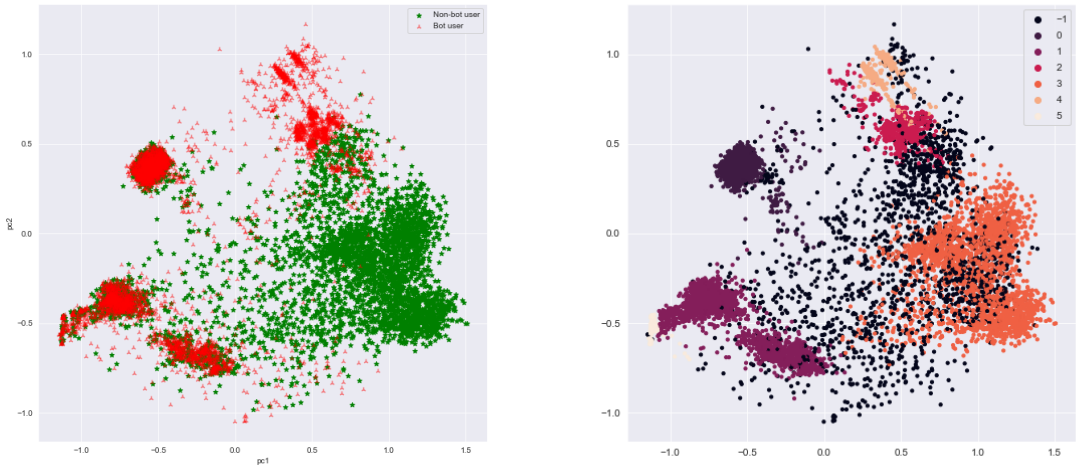
3 Clustering

Once the set of indicators for the users was defined, a clustering analysis was performed to validate the descriptive power of the indicators set along with the other numerical attributes. Different algorithms were tried out. The categorical attributes, which are the language and the bot flag, were used to characterize the clusters.
All the attributes, except `publication_date_in_secs_mean`, `tweeting_regularity`, `mean_length`, `user_subscription_in_secs`, went through a *log transformation*, subsequently all attributes went through *min-max scaling* before performing the clustering. The rescaling was needed in order to bring the data in a more compact space and ease the cluster retrieval and interpretation.
In the following paragraphs the steps for each approach are described. Section 3.5 summarizes the outcomes from all the algorithms. To simplify the discussion, the cluster characterization looks at: time distribution of users tweets (**Time**), how the other users react to users' tweets (**Comm**), and the writing style (**Style**).

3.1 DBSCAN

The DBSCAN algorithm has two fundamentals parameters: the minimum number of points (`min_samples`) that a cluster must possess and the ϵ. Different `min_samples` values were tested; the better results were obtained setting it to $2 \times \#attributes$. The ϵ was decided by studying the curve of the distances based on a 20-nearest-neighbor. The knee of the curve was around 0.6. ϵ values near 0.6 were explored, running DBSCAN and looking at the *silhouette score* and *Davies-Bouldin index*. In the end $\epsilon = 0.575$ was chosen.
With this configuration, 7 clusters were identified, including the noise cluster; figure 13(b) shows a two-dimensional projection of them.
Figure 14(c) shows the average values for each feature for each cluster. It can be observed that the `publication_date_entropy` and `reply_count_entropy` have no discriminative power. In the following the clusters characteristics are summarized.

C-1 (noise) As we can expect from a noise cluster, most of the features have average values. There is a prevalence of genuine users, which makes sense since we expect that a human user has a less schematic behaviour. This cluster is considerably large, about 15% of the users.



(a) PCA with bot labels.

(b) DBSCAN labeling for $\epsilon = 0.575$.

Figure 13: Principal Component Analysis visualization for clustering.

C0 2.5k bots and 1k genuine users. Average `statuses_count` and `n_tweets`.
Time: least active users of 2019, recently subscribed. Mild tweeting regularity and density.
Comm: lowly retweeted, highly replied, lowly liked.
Style: average tweet length, lowest use of hashtags, high use of URLs, average use of mentions.

C1 Very similar to cluster 0.
Time: users are the ones active only in 2018. Low tweeting density and regularity.
Comm: mildly retweeted, lowly replied, lowly liked.
Style: low use of hashtags, high use of URLs, high number of mentions.

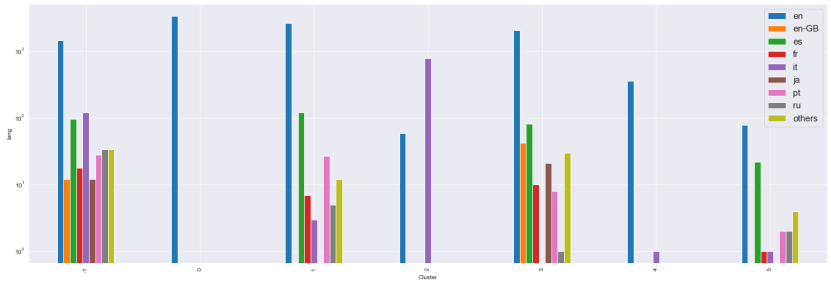
C2 Only 1k bots, 90% of them are Italian. This is the only cluster where the `n_tweets` is not proportional to `statuses_count`.
Time: active in 2019 only. High tweets density. Very high tweeting regularity.
Comm: averagely retweeted, highly replied, highly liked. High tweeting regularity.
Style: longest tweets of any cluster, but Low number of hashtags, URLs and mentions.

C3 Only genuine users. Highest `statuses_count`, high `n_tweets`.
Time: active in 2020 only. Medium-high tweeting regularity.
Comm: highly retweeted, highly replied, highly liked.
Style: average-low number of hashtags and URLs, high number of mentions.

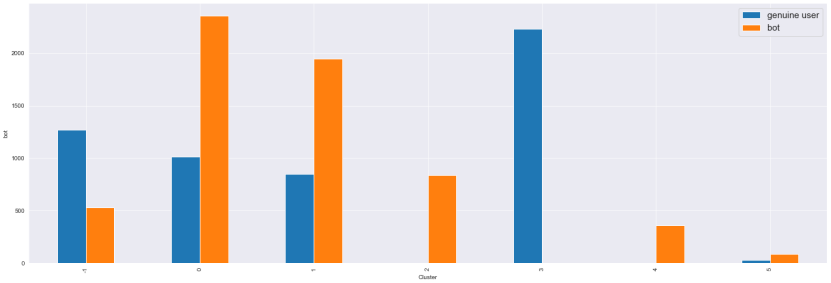
C4 Only bots. Highest `statuses_count` and `n_tweets`.
Time: most active bots in 2019. Highest tweeting regularity.
Comm: lowly retweeted, highly replied, lowly liked.
Style: high use of hashtags, URLs and mentions.

C5 This is the least populated cluster. It contains users with no tweets or a very small amount of them.

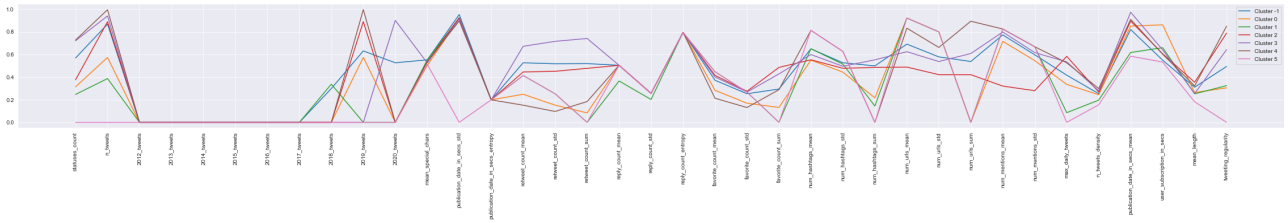
With ϵ=0.55 the retrieved clusters are 9, with the $C3_{\epsilon=0.575}$ that is split in $C3_{\epsilon=0.55}$, $C4_{\epsilon=0.55}$ and $C6_{\epsilon=0.55}$.



(a) characterization by language



(b) characterization by bot label



(c) average features value for each cluster

Figure 14: DBSCAN clusters characterization for the optimal setup.

3.2 K-means

To take advantage of the ability of DBSCAN to identify noisy points, k-means was applied on the users set with and without the noise cluster. The k parameter was decided by looking at the SSE curve. The optimal number showed to be 9 for the whole users set, 7 for the denoised one. The results with the denoised dataset did not produce extraordinary results, hence in the discussion we focus just on the result obtained on the whole dataset.
Overall, both configurations reached a good separation and, as is discussed below, some patterns are definitely visible. Observing the plots in figure 15, for $k = 9$, the clusters have the following characteristics:

C0 Contains only bots, with English and Italian.
Time: highest `statuses_count` and `n_tweets`, active only in 2019, high entropy in the publication date.

Comm: Highly retweeted and replied, lowly liked.
Style: long tweets, low use of special chars, few tweets with a lot of hashtags (high mean, high std, low sum), lot of URLs and mentions. Highest tweeting regularity.

C1 Lowly active users and bots, with low `statuses_count`, low `n_tweets`.
Time: publishing from 2015 to 2018, in a restricted time window (low `publication_date_std` and entropy), few tweets per day, sparse publication (low `tweet_density`), lowest `tweeting_regularity`.
Comm: with a small amount of retweets, reply and favorites.
Style: short tweet length, Average use of hashtags, high use of URLs and mentions.

C2 Contains genuine users only. High `statuses_count` and `n_tweets`.
Time: recently subscribed, active mainly in 2020, a little in 2019. High `tweeting_density` and average `tweeting_regularity`.
Comm: highly replied.
Style: short tweets, little use of special chars and hashtags, large but variable use of mentions.

C3 2k bot, 1k genuine. Low number of `statuses_count` and `n_tweets`.
Time: recently subscribed, active only in 2019, irregular publication schema.
Comm: low retweet, high reply, low favorite.
Style: short tweets, low use of special chars, lowest use of hashtags, largest use of URLs, medium-high use of mentions.

C4 800 bots, 400 users. Low `statuses_count` and `n_tweets`.
Time: active mainly in 2017 and 2018, barely active in 2019 and 2020. Low tweeting density and regularity.
Comm: high retweet, small reply, high favorite mean.
Style: little use of hashtags, average use of URLs and mentions.

C5 Only genuine users. High `statuses_count`, high `n_tweets`.
Time: active in 2019 and 2020, high tweet density, average `tweeting_regularity`.
Comm: highly retweeted and replied, medium to highly liked.
Style: average length tweets, medium-large usage of hashtags and URLs, high use of mentions.

C6 Only genuine users. Highest `statuses_count`, high `n_tweets`.
Time: active in 2020 only, high tweet density medium-high `tweeting_regularity`.
Comm: highly retweeted and replied, medium-highly liked.
Style: medium-low tweets length, few hashtags per tweet, average number of URLs, high usage of mentions

C7 Only 1k bots. 90% are Italian. Average `statuses_count` but high `n_tweets`. Highest `max_daily_tweets`.
Time: active mainly in 2019. Barely active in 2018. High tweeting regularity.
Comm: average retweet, high reply, high favorite.
Style: longest tweets. Low use of hashtags, lowest URLs mean and std, lowest mentions mean and std.

C8 Mostly genuine users. Medium-high `statuses_count` and `n_tweets`.
Time: this cluster seems to group the more active users during the years before 2019 and

the ones subscribed in the first years of the dataset. They have average `tweeting_density` and regularity.
Comm: highly retweeted, highly replied, averagely liked.
Style: average tweets length, medium-large use of hashtags, average use of URLs, high use of mentions.

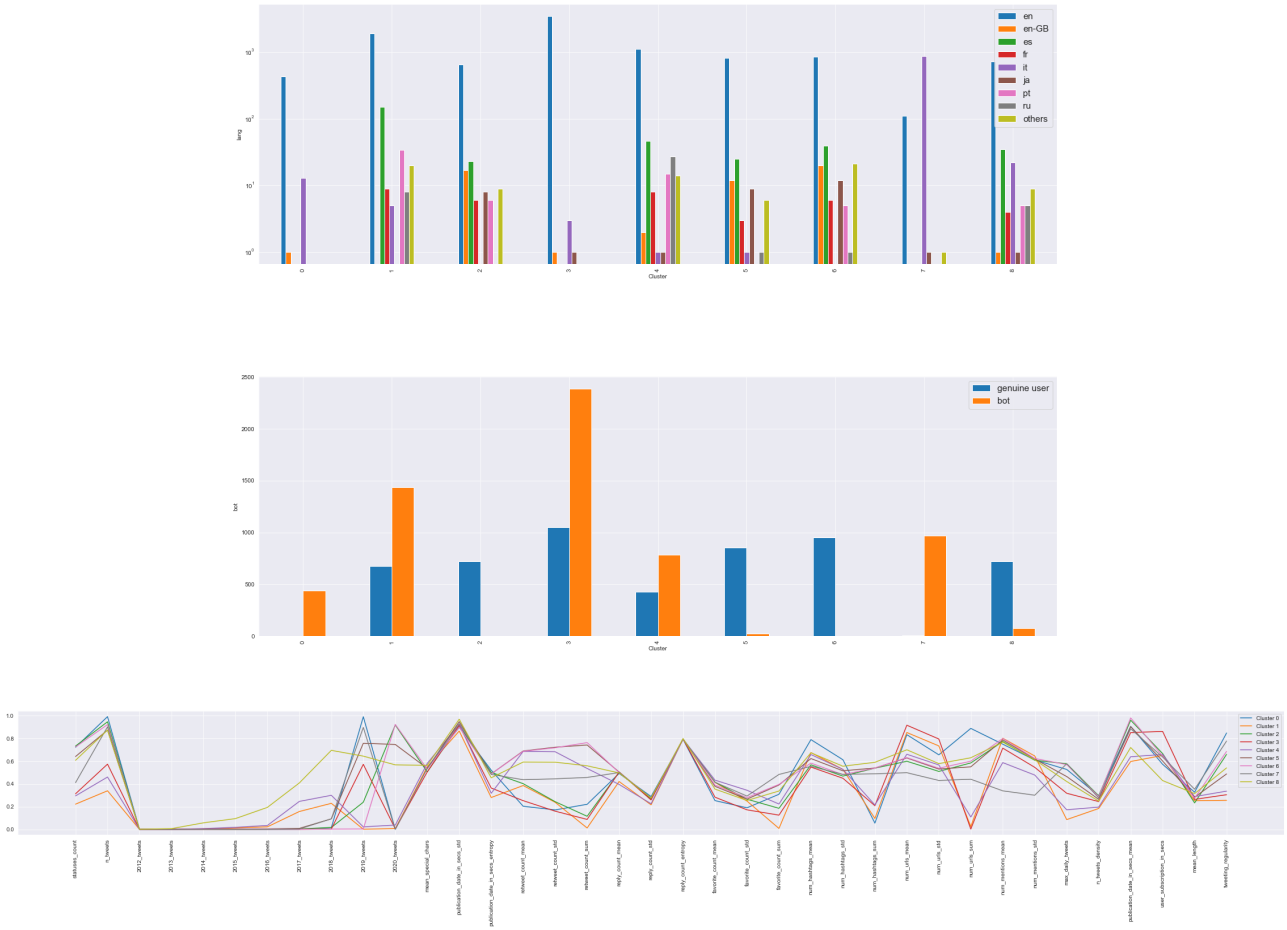
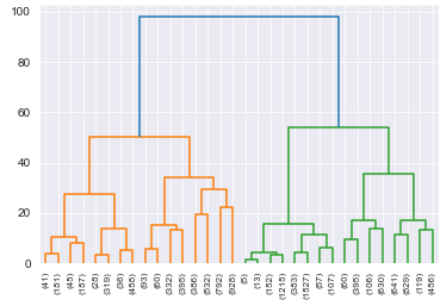


Figure 15: Clusters characterization for the best configuration of k-means ($k = 9$).

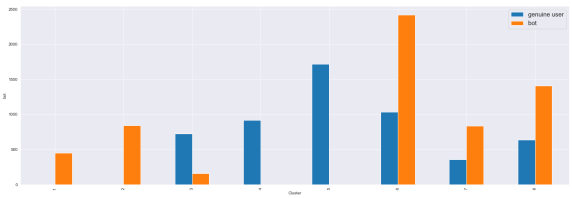
For $k = 8$, the result is quite similar, in fact it is the same but $C2_{k=9}$ and $C6_{k=9}$ are merged into $C4_{k=8}$. For k=10, $C1_{k=9}$ is split in two where a cluster contains the least active users of the set, but proportions of bot/genuine are kept.

3.3 Hierarchical clustering

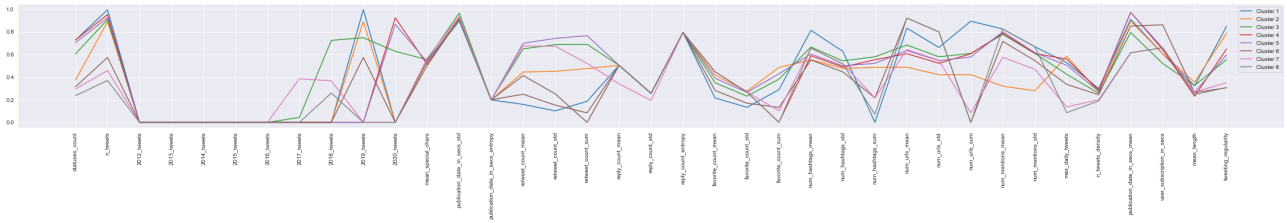
As for k-means, hierarchical agglomerative clustering was run over the noisy and DBSCAN-denoised users set. Again, the result on the denoised are not considerably better than the whole dataset. Different linkage methods and distance metrics were tested. The *Ward* method along with the *euclidean* distance gave better results. Although the hierarchical clustering should be able to automatically find the final number of clusters, the libraries required it as hyperparameter so values similar to the ones determined for the other clustering techniques were explored. Setting k to 8 gave the best results. If k is increased, $C5_{k=8}$ that has only genuine users, is further split.



(a) Dendrogram



(b) Bot characterization



(c) Average value of features for each cluster

Figure 16: Plots for the best configuration of hierarchical clustering.

The clusters characterization is very similar to k-means. The clusters retrieved with hierarchical clustering are not characterized by `publication_date_std` and the per-year tweet counters from 2012 to 2016.
This can be observed from the landscape plot in figure 16(c). The cluster characterization by the bot label shows that the retrieved clusters are purer w.r.t. k-means.

3.4 Other algorithms

X-means, *G-means*, *CURE* and *Expectation Maximization* (EM) were tried out via the `pyclustering` library implementations. Despite x-means and g-means are supposed to find the optimal number of clusters, the k-finding procedure failed to converge. Nevertheless, good results were obtained by fixing k to the optimal value found with k-means, 9.
G-means was particularly good in the separation of the genuine users. In general, w.r.t. k-means, it can be observed a different characterization w.r.t. to `publication_date_entropy` and the retweet count derived features (mean, sum, entropy, standard deviation).
The x-means results were not that good, since it retrieved one big cluster (cluster 6, with 5k users). Looking at the radar plots, the rest of the clusters have similar features to the ones retrieved with k-means, and are almost all pure bots or genuine users. CURE and EM did not produce any acceptable result in any setup. The clustering analysis with these algorithm was also hard to carry out due to the dimensions of the dataset and the available computational resources.

3.5 Clustering summary

Overall, independently from the algorithm used, the retrieved cluster show similar characteristic. Table 1 metrics shows a comparison of the Davis-Bouldin and silhouette scores for the most relevant configurations tried out. The best results were obtained with the k-means clustering,

without denoising the dataset with DBSCAN. Figure 17 shows how the retrieved clusters via the various algorithms are similar. To make this comparison possible, the best configurations having equal number of clusters were picked up. The Adjusted Rand Index (ARI) was used to compare the partitionings. As stated before, k-means and hierarchical produced very similar results, while the difference with DBSCAN is more marked.
The good characterization w.r.t. the bot label shows that the extracted indicators are suitable to discriminate the two users types.

algorithm	params	silhouette	Davies Bouldin score
DBSCAN	$min\_samples = 74, \epsilon$=0.575	0.35	1.65
k-means,	k=9	0.38	1.39
hierarchical	ward, euclidean, k=8	0.37	1.39
x-means	k=9	0.38	1.39
g-means	k=9	0.32	1.38

Table 1: Metrics comparison among different clustering algorithms.

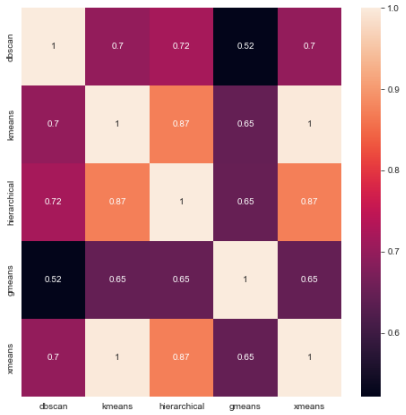


Figure 17: ARI among the best configurations of each clustering algorithm.

4 Classification

The aim of this phase was to predict if a user is a bot or a genuine user. To make it possible a series of binary classification models have been trained and their results compared. For each model a hyper-parameters search has been performed through a grid search. To select the best configuration for that model a validation set was employed performing a *4-Fold Cross Validation*.
The metrics considered for the model selection were *Accuracy*, *Precision*, *Recall* and *F1 score*. Given the good balancing of the dataset, ranking the performance by *accuracy* or *F1* lead to almost identical rankings.

4.1 Data preparation

The data used in this task was a merge between the users dataset and the extracted indicators. The user `id` and `name` have been removed because they were not useful. The language column

was discretized to be able to feed it to the classification models. Finally a data scaling (*MinMax*) was applied in order to improve classification performances and reduce the training time of most models.
Once the data was prepared, a *stratified* split between development set (train and validation) and test set was performed (80%-20%).
The configurations of all the classifiers have been tried out both with and without feature selection. Three approaches were employed: Recursive feature elimination (RFECV), selection of top k features via χ^2 test (with $k = 15$ or $k = 25$) and Model based feature selection (with Random Forest).

4.2 Models

4.2.1 Decision Tree

The hyper-parameters which were explored for the Decision Tree are shown in Table 2.

Parameter	Values
criterion	gini, entropy, log_loss
splitter	random, best
max depth	4, 8, 16, Unbounded
min samples split	2, 4, 8, 16, 32
min samples leaf	1, 2, 4, 8
max features	$\sqrt{\#features}$, $\log_2(\#features)$, Unbounded
max leaf nodes	Unbounded
min impurity decrease	0.0, 1e-2

Table 2: Values of hyper-parameters used in the Decision Tree grid search

The best tree, depicted in figure 18, uses *gini* criterion, $max\ depth = 4$, $min\ samples\ leaf = 4$, $min\ samples\ split = 16$ and *best* as splitter. The other parameters are set to Unbounded. This model was trained selecting the top k=15 features. Interestingly, these features do not include languages, tweet counters for years before 2018, reply and favorite related attributes. Among the selected features, the tree uses the year counters from 2018 to 2020, `statuses_count` and `max_daily_tweets`.

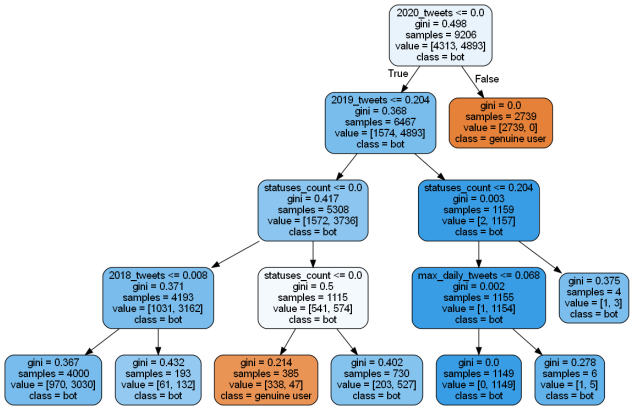


Figure 18: Decision tree generated by selecting the best 15 features via χ^2 test.

4.2.2 SVM

The explored hyper-parameters ranges are listed in table 3.

Parameter	Values
C	1e-2, 1e-1, 1e1, 1e2, 1e3, 1e4
kernel	linear, polynomial, radial basis function
degree	2, 3, 4, 5, 6, 7, 8
gamma	$\frac{1}{\#features \times var(tr)}$, $\frac{1}{\#features}$, 1e-2, 1e-1, 0, 1e1, 1e2

Table 3: Values of hyper-parameters used in the SVM grid search

The best model was the one with $C = 1e4$ and linear kernel. It was found by all approaches, both with and without feature selection, with the same exact performances.

4.2.3 Naïve Bayes

For Naïve Bayes, the best hyper-parameters found are $\alpha = 0.25$ and no prior probability fitting, among the ones explored which are shown in Table 4. α is the Laplace/Lidstone smoothing parameter.

Parameter	Values
α	0.25, 0.5, 0.75, 1, 2, 1e1, 1e2, 1e4
fit prior	True, False

Table 4: Values of hyper-parameters used in the Naïve Bayes grid search

The best combination of parameters has been obtained by first performing RFECV.

4.2.4 Neural Network

For the Neural Network, many preliminary trials were made. The more promising values for the hyper-parameters underwent the final grid search. These values are depicted in Table 5.

Parameter	Values
hidden layer sizes	5, 10, 16, [64,32,16]
activation	logistic, tanh
solver	sgd, Adam
regularization	1e-5, 1e-8
batch size	8, 16, 32
learning rate init	1e-2, 1e-3, 5e-3
learning rate	constant, adaptive
momentum	0.0, 0.2, 0.4, 0.6, 0.8

Table 5: Values of hyper-parameters used in the Neural Network grid search

The best model was found by applying RFECV. It is the one with hidden layer sizes `[10]`, activation `logistic`, solver `Adam`, $regularization = 1e-8$, $batch\ size = 8$, `adaptive` learning rate initialized to $5e-3$ and $momentum = 0.2$.

Parameter	Values
n estimators	30, 50, 70, 90
criterion	gini, entropy, log_loss
max depth	Unbounded, 5, 25, 45, 65, 85
min samples split	2, 4, 16, 64
min samples leaf	2, 4, 16, 64, 100
max features	$\sqrt{\#features}$, $\log_2 \#features$, Unbounded
max leaf nodes	Unbounded, 3, 5, 7, 9
min impurity decrease	0.0, 0.1, 0.3

Table 6: Values of hyper-parameters used in the Random Forest grid search

4.3 Ensemble Models

4.3.1 Random Forest

The hyper-parameters explored for the Random Forest are shown in Table 6. All feature selection approaches reached the same results, the hyper-parameters which were in common are `entropy` as criterion, $max\ features = Unbounded$, $min\ impurity\ decrease = 0.0$, $min\ samples\ split = 4$, while the others vary.

4.3.2 Adaboost

For Adaboost training the best configurations of the SVM and Decision Tree models have been used. The hyper-parameters explored for Adaboost are shown in Table 7. The best set was: `DecisionTree` as classifier, `SAMME` as algorithm, $learning\ rate = 0.1$ and `128` estimators, without applying feature selection.

Parameter	Values
classifier	SVM, DecisionTree
n estimators	4, 8, 16, 32, 64, 128, 256
learning rate	1e-3, 1e-2, 1e-1, 1, 1e1, 1e2
algorithm	SAMME

Table 7: Values of hyper-parameters used in the Adaboost grid search

4.4 Classification Summary

Table 8 shows the best performances of each model on the Test set, it can be seen that the better performing algorithms are Random Forest, Adaboost and Decision Tree. Most of the models were able to reach the exact same performances.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
Decision Tree	85	88	85	84
SVM	81	86	81	81
Naïve Bayes	80	85	80	79
Neural Network	82	87	82	82
Random Forest	86	88	86	85
Adaboost	86	88	86	85

Table 8: Comparison of performances on all models.

5 Timeseries analysis

As last point of this work, the time aspect of the tweets has been considered by focusing on each user as a timeseries of scores. From project description constraints only the tweets published in 2019 and their relative users have been considered. For each day of the year a specific score was computed, the score used is the Success Score defined as:

$$Success\ score = \frac{retweet\_count + reply\_count + favorite\_count}{num\_hashtags + num\_mentions + num\_urls + 0.1} \tag{3}$$

For the users that have more than one tweet, the score is computed once with the sum of each counter. Hence, for each user a timeseries with the score for each day of the year was created. Once the timeseries have been extracted, an analysis was performed in order to find interesting groupings of users.

5.1 Clustering

Different types of clustering have been applied, in particular *shape-based clustering*, *feature-based clustering* and *approximation-based clustering*. Before running clustering a data pre-processing step was performed. Pre-processing consisted in log transforming data and scaling it. Two different methods of scaling timeseries were employed: *MeanVariance* and *MinMax* scaling.

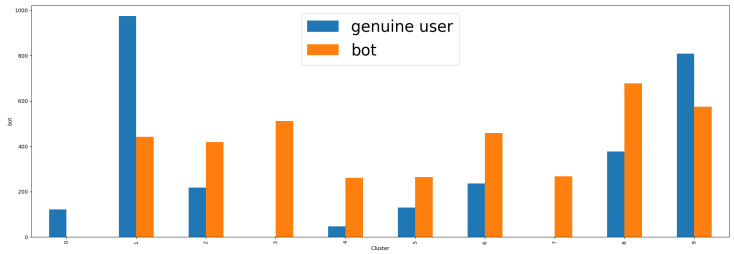
5.1.1 Shape-based clustering

For the shape-based clustering the applied approach was *timeseries Kmeans*. In order to find the best clusters a grid search has been applied with the following parameters 9.

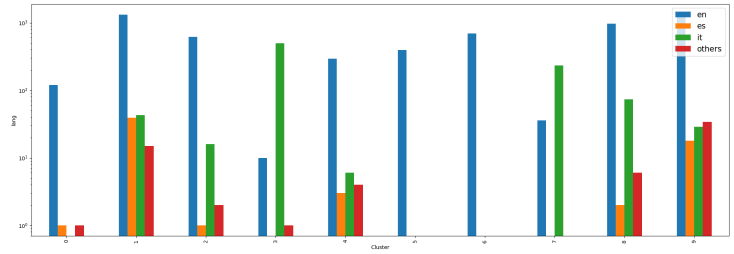
Parameter	Values
scaler	MeanVariance, MinMax
metric	euclidean
n centroids	[2, 3, ..., 25]

Table 9: Values of parameters used in the Shape-based clustering grid search

From each combination three different scores have been extracted to validate the results: SSE, DB score and silhouette. The most performing scaler has been selected by looking at the



(a) characterization by bot



(b) characterization by language

Figure 19: Cluster characterization for shape based clustering with $k = 10$.

SSE score. Instead to find the best k the SSE curve has been plotted finding $k = 10$. The best combination obtained an SSE of 8.01, a silhouette score of 0.13 and a DB score of 1.82. From figure 19(a) it is possible to notice that there are three "pure" clusters. One composed of only genuine users (C0) and two composed of only bots (C3 and C7). C1 contains the highest number of genuine users. In figure 19(b) it is shown that C5 and C6 are composed only by English users, instead C3 and C7 are predominantly composed of Italian bot users.

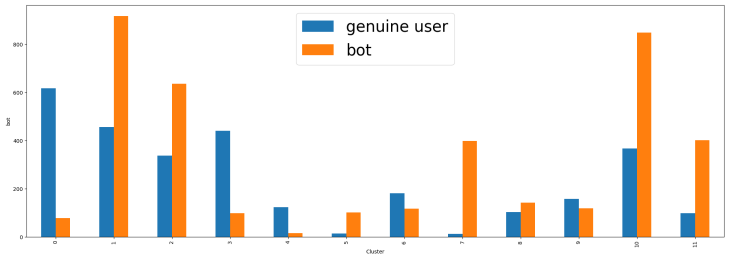
5.1.2 Feature-based clustering

For feature-based clustering *Kmeans* was applied. The extracted features for each timeseries are all statistical descriptors: average, std, variance, median, $10^{th}, 25^{th}, 50^{th}, 75^{th}, 90^{th}$ percentile, inter-quartile range, covariance, skewness, kurtosis and entropy. A grid search was run over the parameters in table 10.

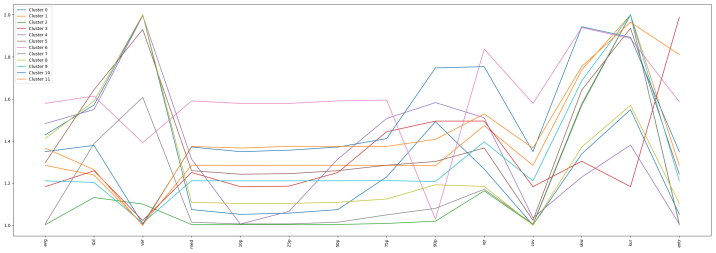
To select the best k the same process applied in the shape-based clustering has been applied and $k = 12$ was selected as the best value. The best combination obtained an SSE of 310.23, a silhouette score of 0.47 and a DB score of 0.83.

Parameter	**Values**
scaler	MeanVariance, MinMax
metrics	euclidean
n centroids	[5,6,...,50]

Table 10: Hyper-parameters space explored for the feature-based clustering.



(a) characterization by bot



(b) Features value for each centroid. Each line represents a centroid.

Figure 20: Cluster characterization for features-based clustering with k=12.

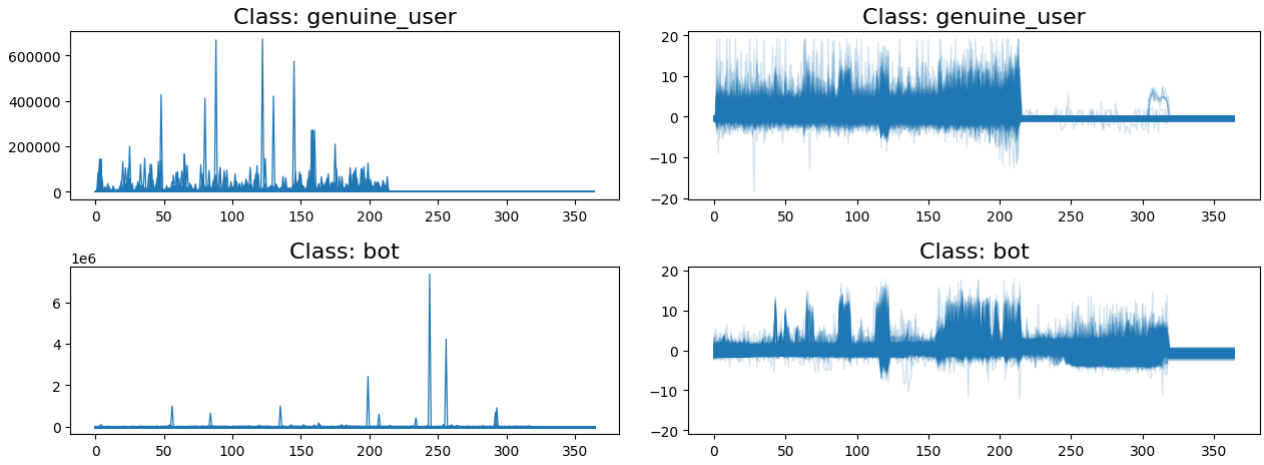
5.1.3 Approximation-based clustering

For approximation-based clustering the applied approach is *timeseries Kmeans*. Clustering has been performed on a piecewise aggregate approximation of the original timeseries. The grid search was run over the parameters in table 11.

Parameter	Values
scaler	MeanVariance, MinMax
n segments	10, 50, 100, 200, 250
n centroids	[2, 3,..., 25]
metric	euclidean

Table 11: Hyper-parameters space explored for Approximation-based clustering.

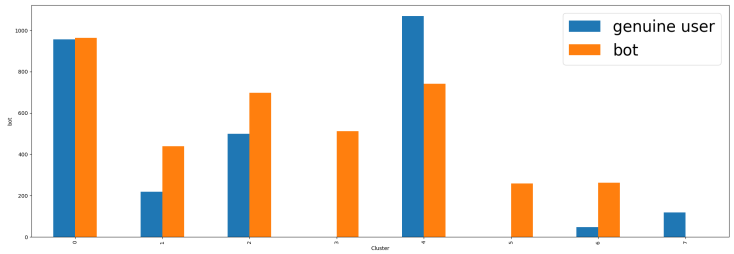
As in the other two approaches, the best scaler has been selected looking at inertia and so was the number of segments. In this case as well, the best k has been picked looking at the SSE curve obtaining k=8. The best configuration reached an inertia of 0.06, a silhouette score of 0.15 and a DB score of 1.71.
Also here (fig. 21(a)) there are "pure" clusters. C7 is composed of only genuine users and C3 and C5 are composed of only bots. C0 is almost composed of half bots and half genuine users. As in the shape-based clustering results in figure 21(b) it is shown that pure bots clusters are predominantly made up of Italian bot users but in C3 there are also users who have languages different from English and Italian.



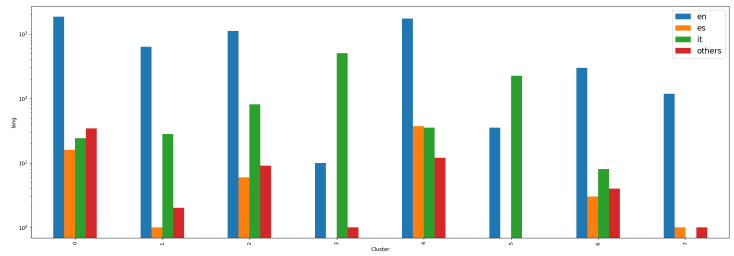
(a) Not scaled

(b) Scaled

Figure 22: Timeseries divided by classes.



(a) characterization by bot



(b) characterization by language

Figure 21: Cluster characterization for Approximation based clustering with k=8.

5.2 Shapelet

A shapelet discovery and extraction was performed to classify each timeseries as belonging either to a bot or a genuine user. An analysis was carried out to better understand the distribution of scores w.r.t. their classes. From figure 22(a) emerges that unlike bots, genuine users have a lot of high peaks of success score even though the highest value is obtained by a bot. Another noticeable aspect is that in the last part of the year the peaks in the genuine users timeseries disappear.

Before proceeding with the classification, it was needed to log transform and scale data as done in the clustering analysis. Then a grid search has been applied with the parameters shown

Parameter	Values
scaler	MeanVariance, MinMax
l	1e-1, 3e-1, 5e-1, 7e-1, 9e-1
r	1
batch size	64, 128, 256, 512
regularization	0, 1e-3, 1e-2, 3e-2, 5e-2, 7e-2, 1e-1
max iter	250
optimizer	sgd, Adam

Table 12: Values of parameters used in the shapelet classification grid search.

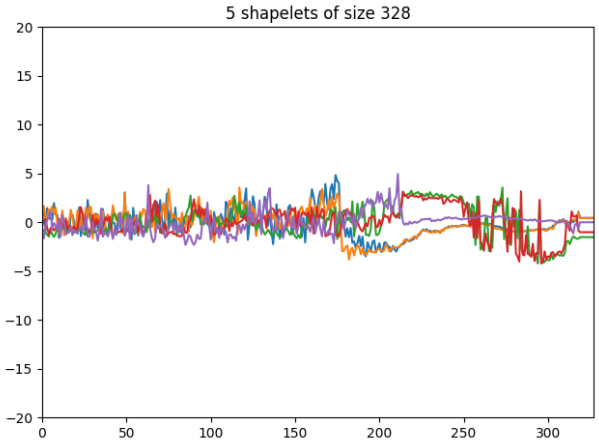


Figure 23: Shapelets of the best performing configuration.

in table 12.

Where `l` is the fraction of the timeseries length which is gonna be employed as base shapelet length and `r` is the number of different shapelet lengths. The metric considered was *binary accuracy*. The best combination found used `Adam` as optimizer, `MeanVariance` as scaler applied on shapelet fraction $l = 0.9$ using no regularization and a batch size of 64. This configuration resulted in a binary accuracy of *83*% on the test set. An interesting aspect is that higher values of the `l` parameter, which means longer shapelets, perform better. Looking at the figure 23 and the figure 22(b), the trend of the shapelets seems to follow the values in the bots timeseries. It is evident that in the first part of the shapelets the success scores values are always smaller than 3 like in the first part of the bots timeseries, while in the second part shapelets and bots timeseries have higher magnitude.

References

[1] Aliza Rosen. Giving you more characters to express yourself.