Detecting Potential Subscribers on Twitch: A Text Mining Approach with XGBoost — Discovery Challenge ChAT: CoolStoryBob

Marvin Gärtner[0000−0003−1651−1326], Andreas Theissler[0000−0003−0746−0424], and Marc Fernandes

Aalen University of Applied Sciences, 73430 Aalen, Germany

Abstract. In this paper we describe our approach to solve the text classification problem of the *Chat Analytics for Twitch (ChAT)* discovery challenge of *ECML-PKDD 2020*. The task was to predict the subscription status of Twitch users for a given channel based on their comments posted within the Twitch chat. Users have the opportunity to support channels in the form of monthly subscriptions, giving them exclusive subscriber-only features in return. Half of the earnings from subscriptions are received by the streamers themselves, with the other half going to Twitch. Thus, there is a monetary motivation for Twitch and the streamers to acquire more subscriptions. The motivation of this research is to detect potential subscribers by predicting a user's subscription status using a trained ML model. These users can then be targeted with marketing campaigns. For our solution we use *BOW* and *TF-IDF* vectors as text features as well as additional extracted numerical features. We applied downsampling to the majority class and used XGBoost as the binary classifier. On the organizers' evaluation set our submission achieved an F_1-score of 0.2647 on the class of subscribers (random baseline: 0.0741) and reached second place among all submissions.

Keywords: Twitch.tv · Chat Analytics · Text Mining · Natural Language Processing · XGBoost · ECML-PKDD Discovery Challenge

1 Introduction

With the growth of the video game industry, a new variation of online entertainment has developed. So-called online video game streaming allows private individuals and professional e-sports athletes to stream their video gameplay while others watch them play [11]. Among a variety of different streaming platforms, Twitch.tv is by far the most popular one [4]. In 2020, Twitch had more than 2.4 million average monthly viewers and over 160,000 active channels[2]. Twitch offers streamers with a certain number of monthly viewers to join their

[2] https://twitchtracker.com/statistics, accessed June 27th 2020

partnership-program enabling them to become full-time professional streamers. Besides running ads, Twitch partners can also offer their viewers to subscribe to their channel. Subscribers pay a monthly fee with half of the earnings going to Twitch. Consequently, there is a monetary motivation for Twitch and for the streamers to encourage viewers to subscribe. A subscription offers several advantages for the viewers including, but not limited to, watching the stream without advertisements, extended communication channels and subscriber-only features [7]. The most frequently used communication channel on Twitch is the integrated Twitch chat *(TC)*, allowing viewers to communicate directly with the streamer and with other viewers [13]. Here again, subscribers have the advantage to use extended *TC* functions such as special username highlighting and the possibility of sending special subscriber emotes [7].

In this paper we present our submission to the *Chat Analytics for Twitch (ChAT)* discovery challenge of *ECML-PKDD 2020*. The challenge's task was to predict whether a Twitch user has subscribed to a channel, by applying machine learning *(ML)* methods on the comments posted in the *TC*. Detecting the subscription status of users based on their chat data could help to identify potential subscribers to a channel. These results could then be used for targeted advertisement. In our submission we used *TF-IDF* and *BOW* vectors for the textual features, as well as additionally extracted numerical features. We tested different ML models, among which XGBoost [3] produced the best result with an F_1-score of 0.2647 on the organizers' evaluation set denoted as E and 0.3229 on our test sets T that were randomly sampled from the provided data. Our submission reached second place among all submissions.

2 Related Work

Twitch has received considerable attention over the past years. In [1], Barbieri et al. analyze the problem of emote prediction, i.e. the task of predicting which emote the user is more likely to use based on a collection of chatroom messages, as well as the trolling detection problem, i.e. recognizing a certain set of emotes commonly used in troll messages. They use a bidirectional long short-term memory network (LSTM) which is compared to a bag-of-words baseline and a logistic classifier based on word embedding, where LSTM outperformed the other two baselines [1]. Kobs et al. present sentiment analysis based on the Twitch exclusive emotes [10], which can be used by users in the chat function. With the help of a created emote dictionary, they attach the sentiments to the initially unlabeled chat data set to obtain a labeled data set. This is then used as input for a convolutional neural network. Their results show the suitability of emotes as indicators for sentiment analysis [10]. Poché at al. investigate comments of a similar community in [14]. They analyze user comments on YouTube coding tutorials to support content creators to effectively understand the needs and concerns of their viewers. They use naive bayes and support vector machines (SVM) to classify comments. Their findings can help to deliver higher quality content and increase the number of subscribers. The discussed papers show dif-

ferent valuable text mining approaches applied on problems like user-targeted content and sentiment analysis. Despite extensive research in the field of text mining, the relationship between chat messages and channel subscription status of users in social media platforms has not been widely investigated. In particular, Twitch itself has not gained much attention regarding Natural Language Processing research [1]. With this paper, based on the problem setting defined by the discovery challenge [9], we aim to contribute to fill this gap.

3 Discovery Challenge: Chat Analytics for Twitch

3.1 Data Set

The data for the *ChAT* discovery challenge was provided by the organizers from the Universities of Würzburg, Leipzig, and Weimar [9], who obtained the data via the Twitch API. The data set has more than 400 million Twitch comments from English channels captured in January 2020 and is composed of 29 million unique channel-user combinations with 7.9 million different users U_i and 140,000 channels C_j. The overall data volume is 38 GB. The two-class data set is fully labelled, with channel-user combinations being labelled as *subscribed* or *not subscribed*. The class distribution has an imbalance with a skew towards non-subscription — only 8% of the channel-user combinations hold subscriptions.

3.2 Task Description

The task was to predict whether a user is subscribed to a given channel based on the user's chat messages on Twitch [9]. The organizers handed out the aforementioned data set approx. 12 weeks prior to the submission deadline. In summary, the task was to solve an imbalanced binary classification problem requiring text analytics and supervised ML models. Evaluation took place by a evaluation set E prepared by the organizers, which is composed of 90,000 unseen channel-user combinations, with 50% of the users in the evaluation set E being present in the training data with their contributions to other channels. The evaluation of the submitted model was done with *TIRA*, an online platform presented by Potthast et al. in [15]. Due to the class imbalance, the F_1-score on the class of subscribed users was used as the evaluation criterion.

3.3 Applicability of Results

While the prediction of the subscription itself is an interesting and challenging problem, we view the applicability as highly useful outside this challenge taking the following consideration: In the evaluation set of the discovery challenge the subscription status is obviously not available but is rather the variable to be predicted. However, one can reasonably assume that Twitch as well as the streamers know if a user U_i has subscribed to a Twitch channel C_j. Yet, a ML model to predict the subscription status for $U_i \times C_j$ is highly valuable. When

a model can classify a user's subscription status based chat messages, it can be assumed that the model has managed to extract some sort of knowledge of how to tell if a user is subscribed based on his/her behavior in the chat. Let $C \times U$ be the channel-user combinations, i.e. the chat messages of users in the different channels. In the subset of one channel C_j, let Y_{sub} be the set of subscribers and $\hat{Y}_{sub}$ be the set of users classified as subscribers by some ML model. Since in practice Y_{sub} is known, the ML models developed in this challenge can be used to detect *potential subscribers* P. We refer to potential subscribers as users that act like subscribers but have not subscribed and might hence be interested in a subscription, which could be supported e.g. by special offers. So outside the frame of this discovery challenge, *potential subscribers* P can be determined with the models developed in this challenge by:

$$P = \hat{Y}_{sub} \setminus Y_{sub} \tag{1}$$

Note, that at first glance it seems counter-intuitive to consider the set of misclassifications. However due to the availability of the true class labels, this problem setting differs from traditional classification.

4 Model Description

4.1 Pre-processing Pipeline

As aforementioned, the data is highly imbalanced, which can cause classifiers to optimize towards the majority class. As a result, these classifiers tend to predict the majority class very accurately, but fail to predict the minority class [5]. To address this, resampling methods were used. Undersampling, where the majority class is randomly sampled to have the number of samples of the minority class, was experimentally found to be the best in our setting. SMOTE [2] was tested but achieved slightly lower results.

In order to reduce noise in the train set, the following pre-processing steps (see Fig. 1) were applied to the text data before training the classifiers [6]: lower case conversion, replacement of emojis[3] and emoticons[4] by a text representation, removal of the most common colloquial terms, stop words removal using NLTK's stop words list[5], removal of the most and least frequently used words, lemmatization using WordNet's Lemmatizer[6], replacement of letters that occur consecutively more than twice (e.g. goood → good), and removal of words that only contain one character.

4.2 Feature Engineering

For the numerical representation of the chat messages, bag-of-words *(BOW)* and term frequency-inverse document frequency *(TF-IDF)* were evaluated. The

[3] https://pypi.org/project/emoji/
[4] https://github.com/NeelShah18/emot/blob/master/emot/emo_unicode.py
[5] https://www.nltk.org/nltk_data/
[6] https://www.nltk.org/howto/wordnet.html

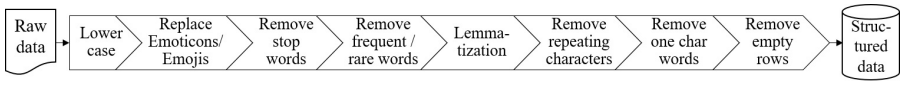

Fig. 1: Pre-processing of the raw user messages

BOW considers the term frequency *TF(w,d)* assuming that a more frequent word w in a document d is more representative for the document [8, 12]. For the representation of the game titles we decided to use the *BOW* model, since the context is irrelevant here. Furthermore, we assume that the more often a game occurs in the text the more likely it represents a respective class.

An alternative is *TF-IDF* which considers the relevance of words by multiplying *TF* with the inverse document frequency *(IDF)*. In *TF-IDF*, a word with lower frequency is associated with higher importance and vice versa [12]. From the most frequent words in Fig. 2, it can be seen that the text data does not differ significantly between subscribers and non-subscribers. From this we conclude that words with a lower frequency are more likely to represent the class than those with a higher frequency, which is why we chose *TF-IDF* vectors for the numerical representation of the chat data.

Fig. 2: Most frequent words of subscribers (left) and non-subscribers (right)

As one key feature we identified the number of subscriber emotes used. Twitch emotes are images or animations that can be posted by users in the chat and provide a quick and wordless form of expression [10]. A certain number of emotes are available for all users, whereas the majority of emotes are exclusive subscriber emotes which can only be used by subscribing to a channel [7] or by having received special gifts which is more likely when frequently watching a channel. During data analysis we observed that the subscriber emotes in the comments are shown as text beginning with a lower-case letter followed by an upper-case letter. This common syntax was used to extract the frequency of emotes. The public emotes[7], which are available to all users, were excluded since their frequency was found to be no indicator for a channel subscription. In addition, we extracted further numerical features from the user messages, as can be seen in Table 1. The correlation of the features with the subscription status was determined with the point biserial correlation coefficient $r_{pbs} = \frac{m_{sub} - m_{notsub}}{s} \sqrt{\frac{n_{sub} * n_{notsub}}{n^2}}$

[7] https://twitchemotes.com/

with $r_{pbs} = [0, 1]$, where m_i and n_i refer to the mean values and number of instances of the two classes and s to the standard deviation.

Table 1: Numerical features extracted for each channel-user combination and their correlation with the subscription status

feature	correlation	feature	correlation
number of words	0.088	number of emojis	0.011
number of distinct emotes	0.168	number of emotes	0.067
number of words in upper case	0.040	number of single chars	0.081
average word length	0.040	number of words with numerical values	0.024
number of stop words	0.084	number of words starting with !, #, @	0.063
number of emoticons	0.030	sentiment using TextBlob[8] (-1...+1)	0.051
number of distinct games	0.169	number of messages	0.110

In order to combine all features into one feature space, we used scikit-learn's column transformer[9], a pipeline that takes input of different data types, then performs the desired transformations and finally combines all features into one feature space, ready to be used as input by a ML algorithm. After creating the input feature space, we used another pipeline that takes the transformed features as input, scales them and finally trains our model. Fig. 3 illustrates the process of feature transformation and model training.

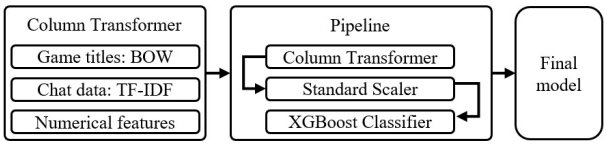

Fig. 3: ML pipeline: feature extraction and training of XGBoost.

4.3 Classification

For classification we evaluated AdaBoost, SVMs, logistic regression, decision trees, and XGBoost. We also experimented with deep neural networks (DNNs), but since it showed that the classifier itself seems not to be the key, but rather the feature representation, we discarded DNNs due to significantly slower training times. We adapted the training process to optimize for F_1-score instead of accuracy. For training 5-fold cross-validation (CV) was used. During CV and on our

[7] https://textblob.readthedocs.io/en/dev/quickstart.html#sentiment-analysis
[9] https://scikit-learn.org/stable/modules/compose.html#column-transformer

test sets XGBoost [3] — a classifier combining the benefits of tree methods and ensembles using gradient-boosted decision trees — achieved the highest F_1-score. In addition to yielding the best results, we found XGBoost to be particularly useful due to the following observations: it yielded robust results over different settings during our experiments, it has a moderate computational complexity compared to DNNs, and as a tree-based method it is scale-invariant which is beneficial while experimenting with features.

5 Experiments and Results

In this section we describe the results on our randomly sampled test sets T and the provisioned evaluation set E. We adopt the baseline provided by the organizers of the challenge: By randomly classifying channel-user combinations as *subscribed*/*not subscribed*, given the respective class distribution of 8% and 92%, the random baseline F_1-score is 0.0741.

Before we initiated the training procedure, we isolated 5 randomly sampled test sets to test our classifier on unseen data. Then we resampled the data and used 5-fold CV to evaluate the generalizability of our model. From the tested classifiers, the best results on our own test sets are achieved using the XGBoost [3] and the described pre-processing and feature extraction steps. This model was submitted and achieved an F_1-score of 0.3229 on our sampled test sets T. On the evaluation set E an F_1-score of 0.2647 (see Table 2) was achieved with 0.4341 of subscribers detected (see Table 3).

Table 2: Results of evaluated models on our own test sets T and for the submitted model on the organizers' evaluation set E.

data set	baseline	**XGBoost**	AdaBoost	SVM	log. regr.	dec. trees
own test sets T	0.0741	**0.3229**	0.3033	0.2171	0.3029	0.2383
evaluation set E	0.0741	**0.2647**	-	-	-	-

Table 3: Confusion matrix of evaluation set E with 90,000 channel-user combinations.

	prediction: *not subscribed*	prediction: *subscribed*
class: *not subscribed*	72363	11440
class: *subscribed*	3507	2690

6 Discussion

In general, all submissions in this challenge had relatively low F_1-scores indicating that the classes are hard to separate based on a user's chat messages. We found that the choice of the classifier itself did not dramatically improve the results, the choice of pre-processing and feature extraction steps was more crucial. The number of distinct emotes was identified as a particularly strong feature. As shown in Fig. 4, the importance of the distinct emotes count exceeds the importance other features significantly. For comparison we re-trained the model without the number of distinct emotes, resulting in a decrease of the F_1-score to 0.3012 on T. Despite the low F_1-scores, following our discussion on the applicability of these models in Section 3, the developed ML models might still prove useful for the acquisition of new subscribers.

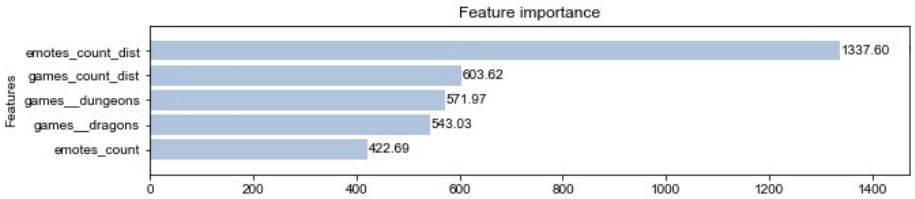

Fig. 4: Top 5 features measured by XGBoost feature importance (gain)

While on our own randomly sampled test sets T we achieved an F_1-score of 0.3229, the results on the evaluation set E were 0.2647, which suggests overfitting. In this case, however, we assume that the lower result is predominantly caused by the composition of the evaluation set E. In E the channels and users were categorized into different activity groups, measured by the amount of chat messages. The lower and upper 25% of the channels and users correspond to an activity of "low" and "high" respectively. The remaining 50% correspond to the category "normal". As a result, there are 9 different activity categories for each channel-user combination (channel=low&user=low, channel=low&user=normal, ...). From each category 10,000 channel-user combinations were sampled yielding the evaluation set with 90,000 instances. While the activity categories were sampled to occur with same frequencies, in the full data set a rather different distribution is present. Table 4 shows a comparison of the distributions of the raw data used to train our model and the evaluation set E.

In addition, the model's F_1-scores for each combination in E is given. On the one hand, in particular for low activities of users and/or channels our model performed poorly. On the other hand, these combinations are quite rare in practice as shown by the percentages for the full data set. For higher channel and/or user activities our model performs significantly better. Naively calculating a weighted average of the categories' F_1-scores weighted by their occurrence in the full data set yields an $F_{1_{avg}}$ of 0.297.

Table 4: Distribution of channel-user activity combinations in full data set and evaluation test set E (l: low, n: normal, h: high) and model performance on each category

channel-user activity	l-l	l-n	l-h	n-l	n-n	n-h	h-l	h-n	h-h
full data set distribution	0.03%	0.49%	7.99%	0.16%	1.87%	29.03%	0.34%	5.22%	54.87%
evaluation set distribution	11.11%	11.11%	11.11%	11.11%	11.11%	11.11%	11.11%	11.11%	11.11%
F_1-score (test set)	0.1565	0.1872	0.1898	0.1567	0.2139	0.2721	0.3546	**0.3913**	0.3206

7 Conclusion and Future Work

In this work we presented a text mining approach to predict the subscription status of Twitch users for a given channel as part of the *ChAT* discovery challenge. We found feature representation to be more important than the actual classifier. As features we encoded the text as *TF-IDF* and *BOW* vectors and additionally extracted numerical features. We experimented with different classifiers, with XGBoost achieving best results with an F_1-score of 0.3229 on our randomly sampled test sets. With an F_1-score of 0.2647 on the evaluation set we reached second place in the challenge. The results show that the problem is challenging and requires more research. Nevertheless, we believe that the submissions' models can be used for the acquisition of new subscribers based on our presented consideration of knowing the subscription status in practice.

In the future, the representation of the chat messages using word embeddings like Word2Vec, as described in [1], could be a promising direction. Also the use of convolutional neural networks as in [10] might be an option for improvement. Applying models on the raw text in order to learn feature representations is another option worth researching.

Acknowledgements

We would like to thank Marc Ebert and Dominik Hahn for implementation support during pre-processing and Julian Theissler for his precious hints during feature engineering based on his domain knowledge on Twitch. Finally, our gratitude goes to the organizers of the discovery challenge Konstantin Kobs, Martin Potthast, Albin Zehe, and Matti Wiegmann [9].

References

1. Barbieri, F., Espinosa Anke, L., Ballesteros, M., Soler, J., Saggion, H.: Towards the understanding of gaming audiences by modeling twitch emotes. In: Proceedings of the 3rd Workshop on Noisy User-generated Text. pp. 11–20. Association for Computational Linguistics, Stroudsburg, PA, USA (2017). https://doi.org/10.18653/v1/W17-4402
2. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: Synthetic minority over-sampling technique. J. Artif. Int. Res. **16**(1), 321–357 (Jun 2002)

3. Chen, T., Guestrin, C.: XGBoost: A Scalable Tree Boosting System. In: 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 785–794. KDD '16, ACM, New York, NY, USA (2016). https://doi.org/10.1145/2939672.2939785
4. Claypool, M., Farrington, D., Muesch, N.: Measurement-based analysis of the video characteristics of twitch.tv. In: 2015 IEEE Games Entertainment Media Conference (GEM). pp. 1–4. IEEE (14102015 - 16102015). https://doi.org/10.1109/GEM.2015.7377227
5. Estabrooks, A., Jo, T., Japkowicz, N.: A multiple resampling method for learning from imbalanced data sets. Computational Intelligence **20**(1), 18–36 (2004). https://doi.org/10.1111/j.0824-7935.2004.t01-1-00228.x
6. Haddi, E., Liu, X., Shi, Y.: The role of text pre-processing in sentiment analysis. Procedia Computer Science **17**, 26–32 (2013). https://doi.org/10.1016/j.procs.2013.05.005
7. Hamilton, W.A., Garretson, O., Kerne, A.: Streaming on twitch. In: Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14. pp. 1315–1324. ACM Press, New York, New York, USA (2014). https://doi.org/10.1145/2556288.2557048
8. Joachims, T.: A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. In: Proceedings of the Fourteenth International Conference on Machine Learning. p. 143–151. ICML '97, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1997)
9. Kobs, K., Potthast, M., Wiegmann, M., Zehe, A., Stein, B., Hotho, A.: Towards Predicting the Subscription Status of Twitch.tv Users — ECML-PKDD ChAT Discovery Challenge 2020. Proceedings of ECML-PKDD 2020 ChAT Discovery Challenge (2020)
10. Kobs, K., Zehe, A., Bernstetter, A., Chibane, J., Pfister, J., Tritscher, J., Hotho, A.: Emote-Controlled: Obtaining Implicit Viewer Feedback Through Emote-Based Sentiment Analysis on Comments of Popular Twitch.Tv Channels. Trans. Soc. Comput. **3**(2) (Apr 2020). https://doi.org/10.1145/3365523, https://doi.org/10.1145/3365523
11. Nascimento, G., Ribeiro, M., Cerf, L., Cesario, N., Kaytoue, M., Raissi, C., Vasconcelos, T., Meira, W.: Modeling and analyzing the video game live-streaming community. In: 2014 9th Latin American Web Congress. pp. 1–9. IEEE (22102014 - 24102014). https://doi.org/10.1109/LAWeb.2014.9
12. Neto, J.L., Santos, A.D., Kaestner, C.A., Alexandre, N., Santos, D., A, C.A., Alex, K., Freitas, A.A., Parana, C.: Document clustering and text summarization (2000)
13. Pan, R., Bartram, L., Neustaedter, C.: Twitchviz. In: Kaye, J., Druin, A., Lampe, C., Morris, D., Hourcade, J.P. (eds.) Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '16. pp. 1959–1965. ACM Press, New York, New York, USA (2016). https://doi.org/10.1145/2851581.2892427
14. Poche, E., Jha, N., Williams, G., Staten, J., Vesper, M., Mahmoud, A.: Analyzing user comments on youtube coding tutorial videos. In: 2017 IEEE/ACM 25th International Conference on Program Comprehension (ICPC). pp. 196–206. IEEE (2017). https://doi.org/10.1109/ICPC.2017.26
15. Potthast, M., Gollub, T., Wiegmann, M., Stein, B.: Tira integrated research architecture. In: Information Retrieval Evaluation in a Changing World, pp. 123–160 (2019). https://doi.org/10.1007/978-3-030-22948-1_5