University of Amsterdam

Bachelor Thesis

Conspiracy videos on YouTube

April 14, 2021

Student:
Roan Schellingerhout
(12399779)

Supervisor:
Dr. Maarten Marx

1 Introduction

To do.

1.1 Research question

This research will focus on the following research question: *What is the impact of different watch strategies on the number of conspiracy videos that have to be watched until a user's YouTube-recommendations start preferring conspiracy content?* In this scenario, 'preferring' will be defined as the situation in which the amount of conspiracy videos present in the recommendations is significantly higher than that of the baseline.

In order to answer the research question, three sub-questions will have to be answered. These questions are the following:

- How do different watch strategies on YouTube influence the type of content that is recommended to a user?
- How long does it take for a YouTube recommendations to stop preferring conspiracy videos, once they have started doing so?
- What type of classifier performs the best when it comes to labeling conspiracy videos on YouTube?

2 Theoretical Framework

2.1 Filter bubbles on social media

Wanneer een gebruiker van een website zich bevindt in een eigen informatie-'universum', waarin de content en aanbevelingen inspelen op diens bestaande

meningen en overtuigingen, zit deze in een filterbubbel (Pariser, 2011). Gebruikers zijn alleen in zulke bubbels; iedereen heeft een eigen, unieke bubbel. Er kan overlap zijn tussen de bubbels van verschillende personen, maar elke bubbel is precies aangepast op het individu. In traditionele media kan een persoon ervoor *kiezen* welk type meningen deze wil aanhoren, door bijvoorbeeld naar een specifieke nieuwszender te kijken. Online is deze keuze niet expliciet, op basis van het gedrag van de gebruiker wordt deze automatisch, zonder toestemming, een bepaald filter voorgelegd.

2.2 Filter bubbles on YouTube

Uit voorgaand onderzoek is gebleken dat het YouTube-algoritme dat verantwoordelijk is voor het geven van aanbevelingen gevoelig is voor het ontstaan van dergelijke filterbubbels. Roth et al. (2020) zijn tot deze conclusie gekomen nadat zij YouTube-aanbevelingen op basis van content hebben geanalyseerd. Op YouTube zijn er twee soorten aanbevelingen: aanbevelingen gebaseerd op het kijkgedrag van de gebruiker en aanbevelingen gebaseerd op de content van de huidige video. In hun onderzoek hebben Roth et al. de focus gelegd op aanbevelingen op basis van content. Hieruit bleek dat dergelijke aanbevelingen snel konden leiden tot lage diversiteit (oftewel: filterbubbels) en dat deze bubbels sneller optraden bij video's met meer weergaven; des te meer weergaven een video had, des te minder divers de gerelateerde aanbevelingen. Zij speculeren dat dit verklaard kan worden door het feit dat YouTube meer informatie opslaat over video's met veel weergaven, waardoor het algoritme betere aanbevelingen kan geven. Ook voorspellen zij dat, naarmate het algoritme meer informatie verkrijgt over de gebruiker, het deze informatie kan combineren met de informatie over video's, wat zou leiden tot een nog sterkere beperking van de aanbevelingen. Volgens Ledwich and Zaitsev (2019) is het kijkgedrag van de gebruiker verantwoordelijk voor ongeveer 70% van de aanbevelingen; kijkgedrag zou daarom veel invloed kunnen hebben op het ontstaan van filterbubbels op YouTube.

2.3 Conspiracy content on YouTube

YouTube heeft beperkte regels tegen het verspreiden van complottheorieën (YouTube, 2021). Zolang de content niet direct aanzet tot geweld of de volksgezondheid in gevaar brengt (denk hierbij aan misinformatie over het COVID-19-virus), mogen ook objectief onjuiste ideeën worden verspreid via YouTube. Dit zorgt ervoor dat er op YouTube meerdere complotgemeenschappen huisvesten. Complottheorieën als 'de aarde is plat en de overheid probeert dat te verstoppen', 'de wereld zal binnenkort vergaan en alleen aanhangers van een bepaalde religie zullen overleven', en 'de wereld wordt geregeerd door kannibalistische, satanische pedofielen' (beter bekend als QAnon), worden op de website door miljoenen mensen bekeken (Paolillo, 2018; Miller, 2021). Hoewel dergelijke video's schadelijk kunnen zijn voor de maatschappij, worden deze door YouTube niet onderdrukt. Als een gebruiker interesse toont in zulke video's, zal de gebruiker

soortgelijke video's aanbevolen krijgen, ook als YouTube ervan op de hoogte is dat het mogelijk schadelijke content is (Ledwich and Zaitsev, 2019).

2.4 The YouTube algorithm

Het YouTube-algoritme probeert aanbevelingen te doen op basis van de verwachtte kijktijd (watch time) en niet op basis van de kans dat een gebruiker op een video klikt (Covington et al., 2016). Dit is gedaan om misleidende video's (beter bekend als 'clickbait') een lagere kans te geven om aanbevolen te worden. Maar, het verkrijgen van feedback over een video op basis van kijktijd heeft te kampen met veel ruis, waardoor het lastig is om tevredenheid te meten. Zo blijkt dat, ook wanneer een persoon een video interessant vindt, deze de video vaak niet helemaal afkijkt. Gemiddeld kijken personen ongeveer 50-60% van een video voordat zij hiervan wegklikken (Park et al., 2016). Echter, video's die goed gestructureerd, of erg interessant zijn, kunnen dit percentage omhoog halen tot 70-80%, waarin ongeveer de helft van de kijkers de video zelfs volledig afkijkt (Lang, 2018). Nadat een video is afgekeken, is er een 41.6% kans dat de gebruiker een aanbevolen video zal bekijken. Welke video dit wordt, volgt een Zipf's verdeling ($\alpha = 0.78$) op basis van de positie in de lijst van aanbevelingen (Zhou et al., 2010).

In voorgaand onderzoek is er dus vernomen dat filterbubbels zich voordoen op YouTube en dat het algoritme geneigd is om complot-content aan te bevelen. Ook wordt er gespeculeerd dat het algoritme beslissingen maakt op basis van het kijkgedrag van gebruikers en dit koppelt aan de content van video's. Om de gebruiker zo lang mogelijk op de website te houden, wat winstgevend is voor YouTube, probeert het algoritme video's aan te bevelen die de gebruiker waarschijnlijk lang zal kijken, welke soms schadelijke content bevatten. Op basis van deze informatie kan er dus verder onderzoek worden gedaan naar het ontstaan van filterbubbels en het verspreiden van complot-content op YouTube. Zo is er nog weinig bekend over hoe snel de aanbevelingen van een gebruiker zich aanpassen, terwijl dit erg belangrijk kan zijn in het ontstaan van zogeheten 'rabbit holes'. Ook is er nog geen onderzoek gedaan naar hoe het soort video invloed heeft op het algoritme. Wanneer een gebruiker veel aanbevelingen kijkt, zou dit mogelijk gezien kunnen worden als impliciete positieve feedback, waardoor er een sneeuwbaleffect kan ontstaan.

3 Methodology

3.1 Watching conspiracy videos

3.2 Leaving the filter bubble

3.3 Machine learning

3.3.1 Data gathering

To answer the research question, it is necessary to determine which YouTube videos can be considered conspiracy videos. Considering the large amount of videos getting recommended, determining each video manually is simply not possible. There are two possible ways to solve this problem. Firstly, there is a dataset which contains nearly 7000 YouTube channels that have been manually labeled based on their politcal view - almost 2500 of which were labeled as conspiracy channels (Ledwich and Zaitsev, 2019); whenever a video is made by one such channel, it can be considered a conspiracy video. However, due to the enormous amount of existing YouTube channels, the odds of a video being uploaded by a channel that is not present in this dataset are very large. For those videos, a supervised machine learning classifier was used. To optimize performance, five different classifiers have been trained and compared: k-nearest neighbors, support-vector machine, neural network, logistic regression, and ridge regression.

In order to train these machine learning algorithms, a training dataset was created. To get a labeled dataset of conspiracy and non-conspiracy videos, use was made of the aforementioned channel dataset made by Ledwich and Zaitsev (2019). For each channel in that dataset, the title, description, and transcript of the ten most recently uploaded videos were downloaded using YouTube's API. Videos uploaded by a conspiracy channel were then labeled as conspiracy videos, and videos uploaded by a channel from a different category were labeled as normal videos. Additionally, the channel description and channel keywords (which are used for targeted advertising on YouTube) were added to each video. The final dataset contains 65.683 unique YouTube videos, 22.156 of which are considered as conspiracy videos.

3.3.2 Data cleaning

However, this dataset was not yet suitable for machine learning, as the data was still messy. Therefore, multiple steps were taken in order to clean the data. Firstly, the two classes (conspiracy and non-conspiracy) were balanced, so that the classifier would not develop a bias for non-conspiracy videos. Rather than opting for balancing the two classes through the use of class-weights, the choice was made to under-sample the data in order to equalize both classes (both containing 22.156 videos, for a total of 44.312 videos) (Lemaître et al., 2017). As there was plenty of data in the dataset, under-sampling was more convenient than implementing class-weights: a technique where weights are at-

tributed to classes, thereby telling the classifier that getting a prediction correct for a certain, underrepresented class is more important (Sun et al., 2006). After both classes had been balanced, the text for each video had to be translated into English. As the original dataset by Ledwich and Zaitsev (2019) also contained channels by non-English speakers, these videos had to be automatically translated. Then, a few common cleaning methods were applied: all text was converted to lowercase, after which special characters, such as emojis were removed, whereafter stop words were removed and all words were stemmed using the porter stemmer (Karaa, 2013). Finally, each video was TF-IDF vectorized to allow the classifiers to function.

3.3.3 Performance optimization

After splitting the dataset into a train, test, and validation set, the hyperparameters of each algorithm were tuned to get the optimal performance. Performance was measures using four distinct metrics: the accuracy, which shows the share of correct predictions; the recall, which shows what fraction of truly positive samples were correctly labeled as such; the precision, which shows what part of the positive predictions were correct; and the F1-score, which is the harmonic mean of the recall and precision (Sokolova and Lapalme, 2009). For each classifier, different configurations of hyperparameters (such as the kernel and the penalty-parameter) were systemically tested - each possible combination was tried. By saving the performance measures for every such configuration, for every classifier, each individual classifiers could be optimized. By then comparing the performance of every optimally configured classifier, the best-performing classifier could be chosen.

Additionally, the added value of using a machine learning ensemble was measured. By having each classifier make a prediction for all videos in the dataset, a new dataset was created, wherein the features were the predictions made by the different classifiers. By using all possible combinations of classifiers, and then having different neural networks use those features as input, a machine learning ensemble was created. This ensemble was then optimized in a similar way to the classifiers individually.

4 Results

4.1 The recommended content

4.2 Leaving the filter bubble

4.3 Machine learning

The hyperparameter tuning lead to impressive scores for all classifiers. The best-performing classifier is the support-vector machine making use of the Radial Basis Function (RBF) kernel and a penalty parameter (C-value) of 10. The SVM is closely followed by the neural network using the identity activation function,

Classifier performance

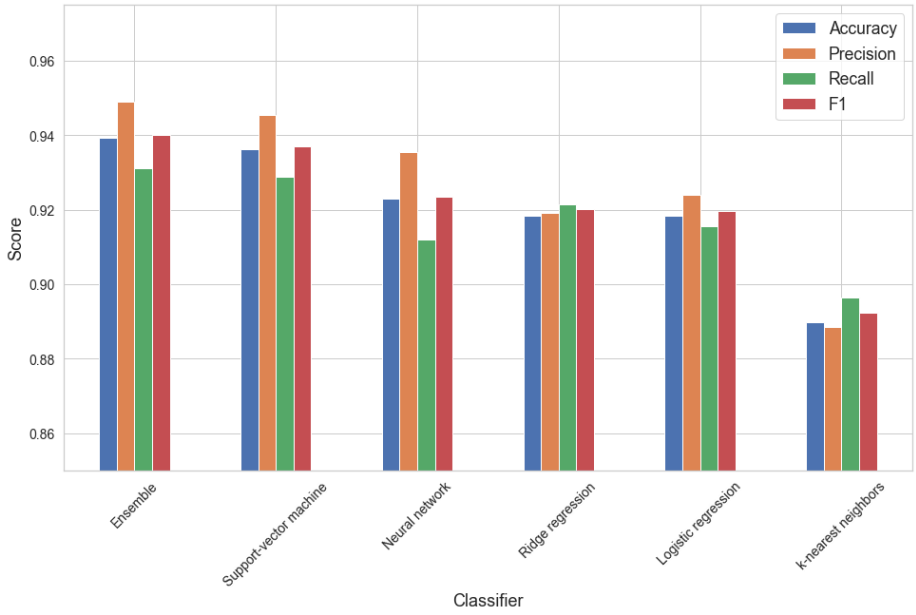

Figure 1: Metrics for each classifier with optimized hyperparameters

with 10 hidden layers of 10 neurons. In third place, there is a two-way tie for accuracy between ridge regression with a sparse-cg solver and penalty (alpha) value of 0.1, and logistic regression with an L2 penalty, a penalty (C) value of 20 and a newton-cg solver. However, ridge regression has a slightly better F1-score, though this difference is neglectable (0.9202 as opposed to 0.9197). The worst-performing classifier is also the simplest of the bunch: the k-nearest neighbors classifier (K=1). Although its performance is still formidable, it does substantially worse than the others. An overview of all metrics for each classifier can be seen in figure 1. The ten best-performing configurations for each classifier can be found in appendix A.

Noteworthy is the fact that the optimal ensemble actually outperforms the support-vector machine by a slight margin. This ensemble, consisting of the SVM, the neural network, and surprisingly, the k-nearest neighbor classifiers, gets slightly higher scores than the runner-up across the board. The ensemble had a 16-way tie for best-performing parameters, all of which contained at least the SVM, neural network, and k-NN classifiers.

Though the ensemble outperforms the other classifiers, it has a significant drawback: its training time is significantly larger than that of the individual classifiers. Support-vector machines are infamous for their slowness when there is a lot of training data, and neural networks can require a lot of training time whenever the number of neurons gets large (Burges and Schölkopf, 1997;

Kamarthi and Pittner, 1999). Requiring both algorithms to run will therefore require a lot of additional training time. Considering the marginal performance increase, the cost outweighs the benefit. As a result, when taking everything into account, the support-vector machine is the best classifier for labeling conspiracy videos on YouTube.

5 Discussion

To do.

6 Planning

Table 1: Planning

Week	Handelingen	Afgehandeld
28/03-03/04	Hyperparameters optimaliseren	Ja
03/04-10/04	Classifier-ensemble optimaliseren	Ja
11/04-17/04	Google accounts maken, deelvraag 3 maken	
18/04-24/04	Inleiding uitbreiden	
25/04-01/05	Uitvoering experiment, labelen met classifier	
02/05-08/05	Deelvraag 1 schrijven, beginnen deelvraag 2	
09/05-15/05	Deelvraag 2 afschrijven	
16/05-22/05	Beginnen met discussie schrijven	
23/05-29/05	Discussie afschrijven	
30/05-05/06	Abstract schrijven, tekst proof-readen	
06/06-12/06	Laatste aanpassingen en verbeteringen	
13/06-19/06	Inleveren scriptie en verdediging	

References

Burges, C. J. and Schölkopf, B. (1997). Improving the accuracy and speed of support vector machines. *Advances in neural information processing systems*, pages 375–381.

Covington, P., Adams, J., and Sargin, E. (2016). Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pages 191–198.

Kamarthi, S. V. and Pittner, S. (1999). Accelerating neural network training using weight extrapolations. *Neural networks*, 12(9):1285–1299.

Karaa, W. B. A. (2013). A new stemmer to improve information retrieval. *International Journal of Network Security & Its Applications*, 5(4):143.

Ledwich, M. and Zaitsev, A. (2019). Algorithmic extremism: Examining youtube's rabbit hole of radicalization. *arXiv preprint arXiv:1912.11211*.

Lemaître, G., Nogueira, F., and Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *The Journal of Machine Learning Research*, 18(1):559–563.

Miller, D. T. (2021). Characterizing qanon: Analysis of youtube comments presents new conclusions about a popular conservative conspiracy. *First Monday*.

Paolillo, J. C. (2018). The flat earth phenomenon on youtube. *First Monday*.

Pariser, E. (2011). *The filter bubble: What the Internet is hiding from you*. Penguin UK.

Roth, C., Mazières, A., and Menezes, T. (2020). Tubes and bubbles topological confinement of youtube recommendations. *PloS one*, 15(4):e0231703.

Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information processing & management*, 45(4):427–437.

Sun, Y., Kamel, M. S., and Wang, Y. (2006). Boosting for learning multiple classes with imbalanced class distribution. In *Sixth international conference on data mining (ICDM'06)*, pages 592–602. IEEE.

Zhou, R., Khemmarat, S., and Gao, L. (2010). The impact of youtube recommendation system on video views. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 404–410.

Appendices

A Hyperparameter tuning

Ensemble	Activation	Layers	Neurons	Accuracy	Precision	Recall	F1
svm, nn, knn	logistic	1	20	0.939318	0.948923	0.931204	0.93998
svm, nn, knn	relu	1	20	0.939318	0.948923	0.931204	0.93998
svm, nn, knn	identity	1	1	0.939318	0.948923	0.931204	0.93998
svm, nn, knn	identity	10	1	0.939318	0.948923	0.931204	0.93998
svm, nn, knn	logistic	1	10	0.939318	0.948923	0.931204	0.93998
ridge, svm, nn, knn	tanh	10	1	0.939318	0.948923	0.931204	0.93998
svm, nn, knn	tanh	1	10	0.939318	0.948923	0.931204	0.93998
svm, nn, knn	tanh	1	20	0.939318	0.948923	0.931204	0.93998
svm, nn, logr, knn	identity	1	1	0.939318	0.948923	0.931204	0.93998
svm, nn, logr, knn	identity	1	10	0.939318	0.948923	0.931204	0.93998

Table 2: Ensemble.

Kernel	C	Accuracy	Precision	Recall	F1
rbf	10.0	0.936309	0.945473	0.928747	0.937035
rbf	100.0	0.935557	0.942289	0.930713	0.936465
rbf	1.0	0.925276	0.930163	0.922850	0.926492
poly	10.0	0.916499	0.946017	0.886978	0.915547
poly	100.0	0.915246	0.944940	0.885504	0.914257
linear	1.0	0.913741	0.917944	0.912531	0.915229
poly	1.0	0.909729	0.935065	0.884521	0.909091
linear	10.0	0.904965	0.907882	0.905651	0.906765
linear	100.0	0.898195	0.905830	0.893366	0.899555
rbf	0.1	0.878887	0.878906	0.884521	0.881705

Table 3: Support-vector machine.

Activation	Layers	Neurons	Accuracy	Precision	Recall	F1
identity	10	10	0.923019	0.935484	0.912039	0.923613
identity	25	10	0.921013	0.933031	0.910565	0.921661
relu	10	10	0.919007	0.917561	0.924324	0.920930
identity	10	20	0.916750	0.906056	0.933661	0.919652
relu	10	20	0.915998	0.912221	0.924324	0.918233
tanh	10	10	0.915747	0.914592	0.920885	0.917728
relu	1	1	0.915747	0.931876	0.900737	0.916042
tanh	25	20	0.915496	0.919052	0.914988	0.917016
tanh	10	20	0.914744	0.920178	0.912039	0.916091
logistic	1	1	0.913741	0.925516	0.903686	0.914470

Table 4: Neural network.

Solver	Alpha	Accuracy	Precision	Recall	F1
auto	0.1	0.918506	0.919118	0.921376	0.920245
sparse_cg	0.1	0.918506	0.919118	0.921376	0.920245
sag	0.1	0.918255	0.919902	0.919902	0.919902
auto	1.0	0.917252	0.923497	0.913514	0.918478
sparse_cg	1.0	0.917252	0.923497	0.913514	0.918478
sag	1.0	0.917252	0.923497	0.913514	0.918478
sag	10.0	0.878385	0.893002	0.865356	0.878962
auto	10.0	0.878134	0.892549	0.865356	0.878743
sparse_cg	10.0	0.878134	0.892549	0.865356	0.878743
auto	100.0	0.812437	0.854545	0.762162	0.805714

Table 5: Ridge regression.

Penalty	C	Solver	Accuracy	Precision	Recall	F1
l2	20	newton-cg	0.918506	0.924107	0.915479	0.919773
l2	20	saga	0.918506	0.924107	0.915479	0.919773
l2	20	sag	0.918506	0.924107	0.915479	0.919773
l2	10	sag	0.916249	0.920831	0.914496	0.917653
l2	10	newton-cg	0.916249	0.920831	0.914496	0.917653
l2	10	saga	0.916249	0.920831	0.914496	0.917653
l2	10	lbfgs	0.915747	0.919506	0.914988	0.917241
l2	20	lbfgs	0.914744	0.921432	0.910565	0.915966
none	1	sag	0.913992	0.923848	0.906143	0.914909
none	10	saga	0.913240	0.922461	0.906143	0.914229

Table 6: Logistic regression.

K	Accuracy	Precision	Recall	F1
1	0.889669	0.888456	0.896314	0.892368
3	0.888415	0.882212	0.901720	0.891859
4	0.879137	0.908899	0.848157	0.877478
5	0.873370	0.858482	0.900246	0.878868
6	0.873119	0.882441	0.866830	0.874566
2	0.872618	0.935043	0.806388	0.865963
7	0.868355	0.848891	0.902703	0.874970
8	0.867603	0.867382	0.874201	0.870778
9	0.861585	0.835672	0.907125	0.869934
10	0.859579	0.854397	0.873710	0.863946

Table 7: K-nearest neighbors.