



UNIVERSITY OF AMSTERDAM

BACHELOR THESIS

YouTube's bias towards conspiracy content: an analysis of the YouTube algorithm

June 1, 2021

Student:
Roan Schellingerhout

Supervisor:
Dr. Maarten Marx

Abstract

YouTube is the second largest website on the internet. It brings in over 34.6 billion page views each month, most of which consist of videos being watched. 70% of those videos are found through YouTube's recommendation algorithm. Unfortunately, this recommender system, like most, is sensitive to filter bubbles. In recent years, conspiracy content has been gaining popularity on the website, partially because of the way in which the algorithm handles it. To gain a better understanding of how the algorithm deals with conspiracies, an experiment was setup wherein brand new YouTube accounts were made to watch conspiracy content adhering to different watch strategies. After watching fifteen conspiracy videos, all accounts ended up with significantly more conspiracy recommendations than the baseline. Accounts that watched conspiracy videos that were recommended to them by YouTube ended up in filter bubbles especially quickly. While accounts ended up in a filter bubble after watching between five and seven videos, getting out of a filter bubble took considerably longer. This suggests that YouTube's algorithm is indeed susceptible to the creation of filter bubbles of conspiracy content.

Keywords: Recommender systems, Conspiracy theories, Filter bubbles, YouTube, Machine learning

1 Introduction

- ¹ YouTube attracts an average of 34.6 billion page views per month, making it
- ² the world's largest video-sharing website and the second largest website on the
- ³ entire internet (Neufeld, 2021). The overwhelming majority of those page views

4 come from users watching videos, 70% of which are recommended to users by
5 YouTube's algorithm (Cooper, 2020). All types of content get produced and
6 consumed on the website. However, conspiracy content has been booming on
7 YouTube (Donzelli et al., 2018). Alt-right (or far-right) and conspiracy chan-
8 nels are starting to grow their audiences, which could have many negative con-
9 sequences for society at large. For example, the number of people who are
10 distrustful of science is increasing, a development in which conspiracy content
11 on YouTube plays a role. Whenever this increased distrust relates to impor-
12 tant topics, such as believing in the efficacy of vaccines, it can create genuine
13 dangers to the public. As it turns out, more than half of the American popula-
14 tion has doubts about - or is definitely against - taking the COVID-19 vaccine
15 (Rosenbaum, 2021).

16 To better understand how YouTube's algorithm (and recommender
17 algorithms in general) allow(s) conspiracy content to thrive, this research will
18 investigate how quickly the algorithm develops a preference for conspiracy video;
19 in other words: how many videos a user needs to watch before they get sent
20 *down the rabbit hole*.

21 This research is based on the assumption that YouTube's recom-
22 mender system is susceptible to the creation of filter bubbles. This concept,
23 coined by Pariser (2011), has been studied in-depth on many social media web-
24 sites, YouTube included. While results vary slightly, the common finding is that
25 YouTube recommendations do indeed lead to filter bubbles and that extremist
26 and conspiracy content is more likely to do so (Bryant, 2020; O'Callaghan et al.,
27 2013; Ledwich and Zaitsev, 2019). This effect can lead to the radicalization of
28 impressionable users, with deleterious consequences. However, an important
29 factor herein is how quickly a user's recommendations turn into a bubble. If
30 the user has enough time to be exposed to other types of content, they might
31 stray away from the more extreme, preventing them from adopting a potentially
32 harmful view (Bozdag and Van Den Hoven, 2015). That is why this research
33 looks at how *quickly* a user could end up in a filter bubble on YouTube. To
34 do so, brand-new accounts will be made to watch YouTube videos according
35 to different watch strategies; after each video watched, the recommendations of
36 the user will be labeled as being either conspiracy content or regular content
37 by a machine learning classifier, to determine whether or not the user is in a
38 bubble.

1.1 Research question

39 For this research the following research question has been formulated: *What*
40 *is the impact of different watch strategies on the number of conspiracy videos*
41 *that have to be watched until a user's YouTube recommendations start preferring*
42 *conspiracy content?* In this scenario, 'preferring' will be defined as the situation
43 in which the amount of conspiracy videos present in the recommendations is
44 significantly higher than that of the baseline.

45 To assist in answering the research question, the following sub-questions
46 will be answered:

- 47 • How do different watch strategies on YouTube influence the type of con-
48 spiracy content that is recommended to a user?
- 49 • How long does it take for YouTube recommendations to stop preferring
50 conspiracy videos, once they have started doing so?
- 51 • What type of classifier is suitable for labeling conspiracy videos on YouTube?

2 Theoretical Framework

2.1 Filter bubbles on social media

52 Whenever the user of a website finds themselves in their own information uni-
53 verse, in which the content and recommendations play into the user's preexisting
54 opinions and beliefs, they are in a filter bubble (Pariser, 2011). Users are by
55 themselves in such bubbles and each bubble is unique. Different bubbles can
56 have overlap, but each bubble is precisely tuned to an individual. In traditional
57 media, a user makes a conscious *choice* what types of opinions they want to
58 hear, for example by choosing to watch a broadcaster with a specific political
59 opinion. Online this decision is implicit: based on the user's behavior, their
60 content is filtered automatically by an algorithm, without explicit consent.

2.2 Filter bubbles on YouTube

61 Previous research has found that YouTube's recommendation algorithm runs
62 the risk of creating filter bubbles. Roth et al. (2020) came to this conclusion
63 after they analysed YouTube recommendations based on content. YouTube has
64 two distinct types of recommendations: recommendations based on the user's
65 viewing behavior and recommendations based on the content of the current
66 video a user is watching. In their research, Roth et al. focused predominantly
67 on recommendations based on content. They found that such recommendations
68 could quickly lead to a decrease in information diversity (read: filter bubbles)
69 and that this decrease happened sooner for videos with a lot of views; the
70 more views a video had, the less diverse its related recommendations. They
71 speculate that this can be explained by the fact that YouTube tends to store
72 more information about videos with a high view count, allowing the algorithm to
73 give better recommendations for such videos. They also predict that, whenever
74 the algorithm has more information about a user to its disposal, it can combine
75 said information with the information it has about a certain video, which could
76 lead to an even stronger limitation of recommendations. According to Ledwich
77 and Zaitsev (2019), a user's viewing behavior is responsible for approximately
78 70% of their recommendations; this behavior could therefore play a big role in
79 the creation of filter bubbles on YouTube.

80 Once a YouTube user has entered a filter bubble, it can be difficult
81 to escape it. The most common way to help users get out of a filter bubble is
82 by exposing them to content covering viewpoints other than their own (Bozdag

83 and Van Den Hoven, 2015). However, for YouTube, this could form an issue.
84 YouTube makes its money by displaying advertisements to a user. The longer
85 a user stays on the website, the more profit YouTube can make. As a result,
86 YouTube's algorithm prefers recommending videos that are likely to generate a
87 lot of watch time (Maack, 2019). As it turns out, controversial content (such as
88 conspiracies) tends to have a higher audience retention: people keep watching
89 controversial content for longer (Birch, 2019). Whenever content is surprising
90 (which conspiracy theories often are), it is more likely to capture and keep a
91 user's attention. Thus, by showing the user more diverse content, the algorithm
92 would actively hinder its own goal. Because of this design, filter bubbles are
93 commonplace on YouTube.

2.3 Conspiracy content on YouTube

94 YouTube has limited rules with regards to the spread of conspiracy videos
95 (YouTube, 2021). As long as the content does not directly incite violence or
96 endangers the public health (e.g. misinformation about the COVID-19 virus),
97 objectively incorrect ideas are allowed to be shared on YouTube. As a result,
98 YouTube is a home to multiple conspiracy communities. Conspiracy theories
99 such as 'the earth is flat and the government is hiding it from us', 'the world
100 will end soon and only followers of this specific religion will be spared', and 'the
101 world is ruled by cannibalistic, satanic pedophiles' (better known as QAnon)
102 gather millions of views on the platform (Paolillo, 2018; Miller, 2021). Though
103 such videos could be considered harmful to society, they are not suppressed by
104 YouTube. Whenever a user shows interest in this type of content, they will
105 be recommended similar videos, even when YouTube is aware of their harmful
106 nature (Ledwich and Zaitsev, 2019; Maack, 2019).

2.4 The YouTube algorithm

107 The YouTube algorithm tries to recommend videos based on the expected watch
108 time they will generate, rather than the probability of a user clicking on them
109 (Covington et al., 2016). This decision was made in order to decrease the
110 likelihood of misleading videos (also known as *clickbait*) being recommended.
111 However, gathering feedback about videos through their watch time can cause
112 a lot of noise, making it difficult to measure user satisfaction. As it turns
113 out, even when a users enjoy a certain video, they are unlikely to watch it
114 completely. On average, users watch around 50-60% of a video before they
115 switch it off (Park et al., 2016). Though, videos that are well-structured, or
116 especially interesting, can improve this percentage up till 70-80%, where nearly
117 half of the viewers actually finish the video in its entirety (Lang, 2018). After a
118 video has been watched, there is a 41.6% chance that the user decides to watch
119 a recommended video. Which recommendation the user will choose, follows a
120 Zipf-distribution ($\alpha = 0.78$) with regards to the position of the video in the list
121 of recommendations (Zhou et al., 2010).

122 All in all, previous research has found that YouTube's algorithm is
123 sensitive to filter bubbles and that it has a tendency to recommend conspiracy
124 content. It is also speculated that the algorithm makes decisions based on the
125 user's viewing behavior, which it combines with the content of videos. In order to
126 keep the user on the website as long as possible, which is profitable for YouTube,
127 the algorithm prefers recommending videos that it suspects the user will watch
128 for a longer period of time, even when they may contain harmful content. Based
129 on this information, further research can be done on the origination of filter
130 bubbles and the spread of conspiracy content on YouTube. For example, little is
131 known about how quickly a user's recommendations adapt to a user's behavior,
132 even though this is a critical aspect when it comes to the creation of so-called
133 *rabbit holes*. Furthermore, no research has been done into the way different
134 types of videos (recommendations in different locations, random videos, etc.)
135 influence YouTube's algorithm. For example, whenever a user primarily watches
136 recommended videos, the algorithm could see this as implicit positive feedback,
137 which could cause a snowball-effect.

3 Methodology

3.1 Watching conspiracy videos

138 In order to determine how different watch strategies affect the YouTube algo-
139 rithm, a python script was created to automatically log into a Google account
140 and proceed to watch YouTube videos. The script was made using Selenium
141 WebDriver: a suite of tools used for browser automation.

142 In the following sections, each aspect of the initial experiment will be
143 explained. Firstly, an explanation will be given about how to log into Google
144 accounts using a python bot. Then, the watch strategies as mentioned in the
145 research question will be defined, followed by a description of their video watch-
146 ing behavior. Afterwards, some restrictions about the videos being watched will
147 be mentioned. Additionally the actual script allowing the bots to be run will be
148 described. Lastly, the precise form of the output of the script will be explained.

3.1.1 Google login

149 Due to Google's strict policy regarding automation within their ecosystem, many
150 obstacles are put into place to prevent users from logging into a Google account
151 using automated software such as a selenium script. To circumvent this restric-
152 tion, two steps had to be taken. Firstly, the selenium WebDriver had to be
153 accompanied by the selenium-stealth package, which removes metadata about
154 the current browser, so that it is less obvious that a WebDriver is being used.
155 Additionally, because this metadata was removed, the Google login service was
156 unable to check what browser the client was using. This results in a warning
157 to the user that their current browser may be insecure, which prohibits them
158 from logging in. To avoid this warning, the Google account needs to have been

159 created within a WebDriver, such as Google's ChromeDriver or Mozilla's Gecko-
160 Driver. Therefore, all twenty accounts were manually created in ChromeDriver.
161 Since Google accounts require a phone number verification upon creation, six
162 free (prepaid) SIM cards were ordered from various providers in order to create
163 the accounts. Each SIM card could create two to three accounts before it was
164 blocked due to being used too many times.

3.1.2 The watch strategies

165 After all accounts had been created, they were subdivided into four distinct
166 watch strategies, making for a total of five accounts per strategy.

- 167 **1. Random videos (baseline)** The first watch strategy is the simplest one.
168 The bots following it will watch random, non-conspiracy videos from a
169 dataset. This watch strategy is used as the baseline to compare the other
170 three strategies to.
- 171 **2. Random conspiracies** The second strategy is similar to the first: the ad-
172 hering bots watch random conspiracy videos from a dataset.
- 173 **3. Video recommendations** The second-to-last strategy starts off in the same
174 way as strategy 2: it chooses a random conspiracy video from a dataset
175 to watch. However, it then watches the four most similar videos in the
176 dataset (based on cosine similarity) in order to allow the algorithm to
177 get a feel for the user's interests. After watching those five initial videos,
178 it starts looking at the recommended videos displayed next to the cur-
179 rent video and chooses the recommendation that is most likely to be a
180 conspiracy video (out of the first twenty recommendations). These rec-
181 ommendations consist of a combination of recommendations based on the
182 content of the current video and the personalized recommendations of the
183 user. By using this strategy, bots are likely to go *down the rabbit hole* and
184 eventually end up in a filter bubble.
- 185 **4. Homepage recommendations** Finally, the last strategy is similar to the
186 previous one, though with one alteration: rather than choosing a recom-
187 mended conspiracy video from the list of recommendations next to the
188 current video, it will choose a recommended video from the YouTube
189 homepage of the account (again out of the first twenty recommendations).
190 Compared to the third strategy, this will lead to the user watching more
191 personalized recommendations rather than content-based recommenda-
192 tions, possibly speeding up the creation and/or increasing the strength of
193 the filter bubble.

194 Strategy 2 was not added in order to study the formation of filter bubbles, since
195 it was unlikely that a filter bubble would come from this strategy. Considering
196 the video choices would be all over the place, it would be difficult for the algo-
197 rithm to determine the specific interest of the user. However, comparing how

198 the algorithm responds to conspiracy content in general as opposed to regular
199 content might still yield interesting results.

200 For strategy 3 and 4, the likelihood of a recommendation being a
201 conspiracy video was estimated by a neural network using the title, descrip-
202 tion, transcript, channel description, and channel keywords of the specific video.
203 Though this is more information than a regular user would have to their dis-
204 posal, it has to be kept in mind that humans are able to interpret the thumbnail
205 of the recommendations, can have foreknowledge about the channel uploading
206 the video or the subject being mentioned in the title, etc. Thus, the choice was
207 made to allow the neural network to consider the transcript and look at general
208 information about the uploader to balance the scales.

209 Each strategy was executed by five different accounts in order to de-
210 crease the probability of a random streak of videos altering the result. The
211 individual accounts watched a total of fifteen videos as described by their watch
212 strategy for a total of three hundred videos watched by the script.

213 Additionally, to simulate real-world user behavior, the average watch
214 time for the videos was normally distributed with a mean of 55% and a standard
215 deviation of 25% (Park et al., 2016; Lang, 2018). The watch time was not be
216 able to exceed a value of 100 or subceed a value of 0, as a video cannot be
217 watched for more than 100% or less than 0%. In the same vein, the clicking
218 behavior of users was simulated as accurately as possible. Whenever none of
219 the recommendations were predicted to be conspiracy videos, the probability of
220 a user clicking on a video at position k within a given list of recommendations
221 (its click-through rate: CTR), was determined using the following formula:

$$CTR(k; N, \alpha) = \frac{1/k^\alpha}{\sum_{n=1}^N (1/n^\alpha)} \quad (1)$$

222 Wherein N is the total number of recommendations and α is the
223 distribution's exponent value ($\alpha = 0.78$) (Zhou et al., 2010). Using this formula,
224 when considering the first twenty recommendations, the first recommendation
225 will have a click-through rate of approximately 20.6%, after which the CTR
226 quickly decreases, until a probability of 1.9% at the twentieth recommendation.

3.1.3 Running the bots

227 After the accounts were logged in, they started watching YouTube videos ac-
228 cording to their watch strategy. However, some restrictions were put into place
229 to make sure the bots did not take too long (considering three hundred videos
230 had to be watched in total, some limitations had to apply). For example, the
231 bots were not allowed to watch videos over an hour long, nor were they allowed
232 to watch live streams, as those could theoretically go on infinitely. Addition-
233 ally, the random videos at the start of the third and fourth strategy were first
234 manually inspected to make sure the bots would not start the experiment by
235 watching a falsely flagged conspiracy video. Considering the way in which the
236 dataset was created, it is possible that some videos that are flagged as conspiracy
237 videos are, in reality, normal videos. This could happen whenever a conspiracy

channel uploads a regular video for once (e.g. a holiday video, promotion of some product, etc.). These *false positives* are far and few between, however, having one of them be selected as the first video for strategy three or four had to be prevented, as that could have greatly altered the final results. With these restrictions in mind, the following script was created and run for all twenty bots, keeping track of the videos they watched and the homepage recommendations they had after each video:

Algorithm 1: Watch YouTube videos according to a watch strategy

Data: User information and a video dataset

Result: The watched videos and homepage recommendations of the user

```
1 for twenty bots do
2   initialize WebDriver;
3   log into Google account;
4   for fifteen videos do
5     if there is a recommendation to be watched then
6       | go to the link;
7     else
8       | pick a random video to watch based on usertype;
9       | determine how long it will get watched;
10      | go to the link;
11      get video metadata and store for overview of watched videos;
12      watch video for given amount of time;
13      if usertype == 3 then
14        | pick recommendation next to current video to watch next;
15        | determine watch time for found recommendation;
16      go to YouTube homepage;
17      store current recommendations for overview;
18      if usertype == 4 then
19        | pick homepage recommendation to watch next;
20        | determine watch time for found recommendation;
21 return watched videos and homepage recommendations;
```

Running the script for all twenty bots resulted in two different datasets: the first containing the videos watched by the bots and the second containing the homepage recommendations for all bots, after each number of videos watched. To determine the influence of the watch strategies on the algorithm, the recommendations were labeled as being either conspiracy or non-conspiracy videos by the classifier. By then grouping all recommendations by their watch strategy and

Example DataFrame

Bot	N	Video id	Views	Likes	Dislikes	Length	Title	Description	Transcript	Channel desc	Keywords	Conspiracy
1	5	ZV0jYdJWZ1k	18832	2623	12	137	Biden's Secretary of Treasury Is SUPER SUS	Biden's Secretary of Treasury Is SUPER SUS! GET THE GA...	personally i do think there...	I love Israel and Goo...	Sinatra_Says Entert...	False
1	5	JG-W7QKozMc	99802	3113	41	286	Robin Hoodwinked	Subscribe to my Pa...	hey guys i'm following this wh...	This channel is to help younge...	"Gonzalo Lira" "How to..."	False
1	5	IyUNUdNOBQ4	34625	1076	38	167	Uncovering Aliens "Bright..."	Filmed in North Myrtle Beach....	there are more UFO sightings...	Researcher and stud...	Art Science Astronomy Her...	True

Figure 1: An example of the experiment's output

the number of videos watched before them (e.g. the recommendations after the third video watched by all bots with strategy one), it was possible to calculate aggregates about general statistics of the recommendations, such as view count and video duration, and the percentage of conspiracy videos present amongst them. This lead to four groups with fifteen entries of different statistics (one for each video watched). In order to find out whether any of the differences between the groups were significant at any point, a number of independent sample t-tests were performed.

3.1.4 Outputs of the experiment

To create the initial recommendation dataset, the bots stored the following information about their top twenty recommendations after each video watched:

- the id of the bot that had gotten the recommendations;
- the amount of videos watched before the recommendation was given;
- the URL of the video being recommended;
- the URL of the channel that uploaded the recommendation.

To then allow the classifier to label the videos, and to calculate the aggregates per strategy, additional data was gathered using YouTube's API. For each recommendation, the title, description, transcript, (dis)likes, views, video duration, channel description, and channel keywords were collected. The title, description, transcript, channel description, and channel keywords were collected in order for the classifier to label the videos, as will be further explained in section 3.3. The other features were collected in order to calculate the aggregates used for the aforementioned ANOVAs. An example of the output can be seen in figure 1.

3.2 Leaving the filter bubble

274 To find out how quickly different users can get out of a filter bubble after they
275 have gotten into one, an additional experiment similar to the one described
276 before was set up. This experiment however, was significantly simpler. Rather
277 than having different bots behave differently, all bots behaved the exact same
278 way. After the bots adhering to strategy 2, 3, or 4 had gotten into a filter bubble,
279 this experiment was performed on them in order to see how long it takes for
280 users adhering to different watch strategies to leave a filter bubble once they
281 find themselves in one.

3.2.1 The setup

282 Considering the experiment studies the way in which users leave a filter bubble,
283 only the bots that had actually gotten into a filter bubble were used. This
284 means that the first five bots, which were used as a baseline, were ignored.
285 The remaining bots went through the same starting procedure as they did in
286 the earlier experiment: the WebDriver was initialized and the bots logged into
287 their corresponding accounts. The same restrictions for videos (i.e. maximum
288 watch time) from the other experiment applied. However, rather than the bots
289 watching videos adhering to their original strategy, all bots watched random
290 non-conspiracy videos from the dataset. Once again, after each video, the bots
291 stored their homepage recommendations. Through doing so, it became possible
292 to see how many videos would have to be watched, for each bot, before their
293 recommendations started looking similar to that of the baseline again. Thus,
294 the following script was created:

Algorithm 2: Getting out of a filter bubble

Data: User information and a video dataset

Result: The watched videos and homepage recommendations of the
user

```
1 for fifteen bots do
2   initialize WebDriver;
3   log into Google account;
4   for fifteen videos do
5     pick a random non-conspiracy video to watch;
6     determine how long it will get watched;
7     go to the link;
8     get video metadata and store for overview of watched videos;
9     watch video for given amount of time;
10    go to YouTube homepage;
11    store current recommendations for overview;

12 return watched videos and homepage recommendations;
```

295 After the script had been run, the same steps were taken as in the previous
296 experiment: the title, description, transcript, channel description, and channel
297 keywords of each recommendation on were downloaded, after which the classifier
298 predicted whether or not each recommendations was a conspiracy video. Then,
299 the recommendations were grouped by watch strategy and number of videos
300 watched. In doing so, the results could be used to determine how quickly the
301 recommendations for each watch strategy returned back to normal. By then
302 analysing the results again using independent sample t-tests, the *strength* (or
303 *inescapability*) of the filter bubbles created by different watch strategies could
304 be measured.

3.3 Machine learning

3.3.1 Data gathering

305 To answer the research question, it is necessary to determine which YouTube
306 videos can be considered conspiracy videos. Considering the large amount of
307 videos getting recommended, determining each video manually is simply not
308 possible. There are two possible ways to solve this problem. Firstly, there is a
309 dataset which contains nearly 7000 YouTube channels that have been manually
310 labeled based on their political view - almost 3000 of which were labeled as
311 conspiracy channels (Ledwich and Zaitsev, 2019); whenever a video is made
312 by one such channel, it can be considered a conspiracy video. However, due
313 to the enormous amount of existing YouTube channels, the odds of a video
314 being uploaded by a channel that is not present in this dataset are very large.
315 For those videos, a supervised machine learning classifier was used. To optimize
316 performance, five different classifiers have been trained and compared: k-nearest
317 neighbors, support-vector machine, neural network, logistic regression, and ridge
318 regression.

319 In order to train these machine learning algorithms, a training dataset
320 was created. To get a labeled dataset of conspiracy and non-conspiracy videos,
321 use was made of the aforementioned channel dataset made by Ledwich and
322 Zaitsev (2019). For each channel in that dataset, the title, description, and
323 transcript of the ten most recently uploaded videos were downloaded using
324 YouTube's API. Videos uploaded by a conspiracy channel were then labeled
325 as conspiracy videos, and videos uploaded by a channel from a different cate-
326 gory were labeled as normal videos. Additionally, the channel description and
327 channel keywords (which are used for targeted advertising on YouTube) were
328 added to each video. The final dataset contains 65.683 unique YouTube videos,
329 22.156 of which are considered as conspiracy videos.

3.3.2 Data cleaning

330 However, this dataset was not yet suitable for machine learning, as the data
331 was still messy. Therefore, multiple steps were taken in order to clean the data.
332 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 (a technique where weights are attributed to classes, thereby telling the classifier that getting a prediction correct for a certain, underrepresented class is more important), 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; Sun et al., 2006). As there was plenty of data in the dataset, under-sampling was more convenient than implementing class-weights. After both classes had been balanced, the text for each video had to be translated into English. Since 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 training, test, and validation set, the hyperparameters of each algorithm were tuned to get the optimal performance (Feurer and Hutter, 2019). Performance was measured 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. The classifiers were trained on the training set and the optimal hyperparameters were determined based on the performance of the classifiers on the validation set. By saving these performance measures for every configuration, for every classifier, the optimal configuration for every classifier could be determined. Lastly, the classifiers were equipped with their optimal hyperparameters and then tested for the final time on the test set. By comparing the performance of every optimally configured classifier on the test set, the best-performing classifier could be chosen (Reitermanova, 2010).

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

In the following two sections, the results of the experiments will be given. First, the changes in content will be shown for whenever a user is actively watching conspiracy content. Afterwards, the results for the second experiment are set forth, wherein users are trying to escape a filter bubble after they have ended up in one. The results of both experiments are in part based on the predictions made by the classifier. An overview of the performance of the different classifiers, together with an explanation of why the support-vector machine was chosen to do the predictions, can be read in section 4.3.

4.1 The recommended content

4.1.1 The types of recommendations

Conspiracy recommendations

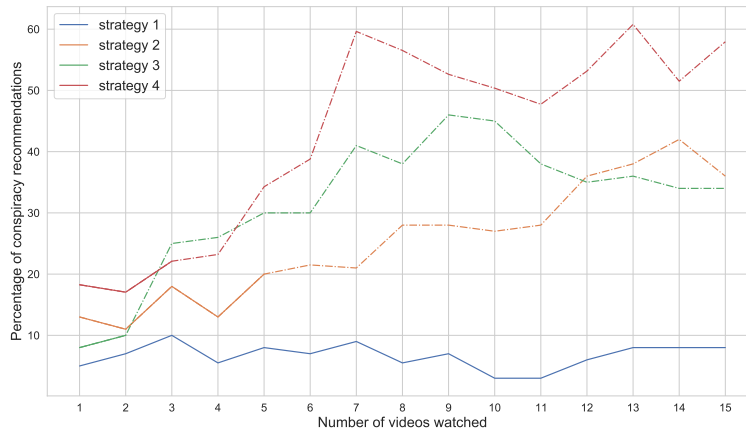


Figure 2: The percentage of conspiracy recommendations after each number of videos watched per strategy. Each measure is based on the top twenty recommendations of the five accounts for the strategy. A dashed line indicates a significant difference compared to the baseline (strategy 1) at $\alpha = 0.05$.

All strategies, excluding the baseline, led to an increased number of conspiracy recommendations on the user's homepage. For all strategies, this increase was significant, meaning filter bubbles were indeed being created. However, the strategies all behaved differently, as can be seen in figure 2.

Strategy 2 took the longest out of all the strategies to get into a filter bubble. Only after having watched six videos did the difference between it and the baseline become significant. After having increased significantly, the

percentage of conspiracy videos being recommended to the users of strategy 2 kept steadily growing, eventually plateauing at about 42%.

Strategy 3 got into a bubble the quickest. After watching only three videos did the difference compared to baseline become significant. However, following this head start, the increase came to a halt rather quickly. After reaching its peak of 45% at nine videos watched, the percentage of conspiracy recommendations started to decline again, eventually settling around 35%. Though most values are slightly higher, strategy 3's course mimics that of strategy 2 until the decline.

Strategy 4 took slightly longer to get into a filter bubble than strategy 3. However, considering that these strategies are identical up until video five and the difference between the two strategies was insignificant ($p > 0.05$), this dissimilarity is negligible. As soon as homepage recommendations started getting watched, the percentage of conspiracy recommendations increased drastically, reaching a peak of around 60%. Similarly to strategy 3, once the number of conspiracy recommendations reached a peak, it was followed by a decrease.

To better understand how the different strategies affected the recommendations provided by the algorithm, a directed network of each strategy was created and analyzed. In this network, an edge between videos indicates the second video being present in the list of homepage recommendations provided after watching the first video. The different networks are displayed in figure 3. It can be seen that strategies 1 and 2 have barely any cross-class edges (strategy 1 has two in total, strategy 2 has zero), while strategies 3 and 4 have a lot of connections between the two separate classes. This stronger connection also becomes apparent when looking the networks' betweenness centrality and clustering coefficient (table 1).

When looking at the conditional probabilities of cross- and within-class edges per network (table 2), it can be seen that regardless of the strategy, regular videos are highly likely to recommend more regular videos. However, the more personalized the strategy, the higher the probability of conspiracy videos recommending more conspiracy videos. This supports the creation of

	Betweenness centrality	Clustering coefficient
Strategy 1	1.597418e-07	0.001592
Strategy 2	0.000000e+00	0.000000
Strategy 3	2.198488e-06	0.009512
Strategy 4	1.136934e-04	0.039018

Table 1: The average betweenness centrality and clustering coefficient of each strategy's network. A node having a higher betweenness centrality indicates that it is present on more shortest paths between other nodes, meaning it is located more centrally within the network. A higher clustering coefficient indicates the network as a whole is more clustered, meaning more ties between triplets are present.

	$P(C R)$	$P(R C)$	$P(C C)$	$P(R R)$
Strategy 1	0.065493	NaN	NaN	0.934507
Strategy 2	NaN	0.750000	0.250000	NaN
Strategy 3	0.185714	0.663043	0.336957	0.814286
Strategy 4	0.105263	0.560673	0.439327	0.894737

Table 2: Conditional probabilities of cross- and within-class edges for each network. Values indicate the probability of an edge going to a class, given it starts in another (in the form: $P(end | start)$).

419 filter bubbles, as users are more likely to be recommended content similar to
 420 that which they already watched. This is also shown in the probability of cross-
 421 class edges for the strategies: the more personalized the strategy, the lower the
 422 probability of cross-class edges being present. This, too, can cause filter bubbles,
 423 as users are less likely to be recommended content that is different from that
 424 which they have already watched.

425 When solely looking at the conspiracy recommendations of each strat-
 426 egy, it once more becomes clear that the recommendations of strategies 3 and
 427 4 are connected more strongly than those of strategies 1 and 2 (appendix 1).
 428 However, these networks are somewhat deceiving, as an edge not being present
 429 does not necessarily indicate there is no link between the two nodes; it could also
 430 simply be that the recommendations coming from that node were not checked
 431 (i.e., the recommendation was not clicked, hence its outgoing recommendations
 432 could not be stored). Yet, even with these missing edges, the clustering coeffi-
 433 cient and betweenness centrality of strategies 3 and 4 end up being higher than
 434 those of strategies 1 and 2, as can be seen in table 3.

	Betweenness centrality	Clustering coefficient
Strategy 1	0.000000	0.000000
Strategy 2	0.000000	0.000000
Strategy 3	0.000012	0.016064
Strategy 4	0.000747	0.058816

Table 3: The average betweenness centrality and clustering coefficient of each strategy's sub-network consisting of only conspiracy recommendations.

The recommendation network of each strategy

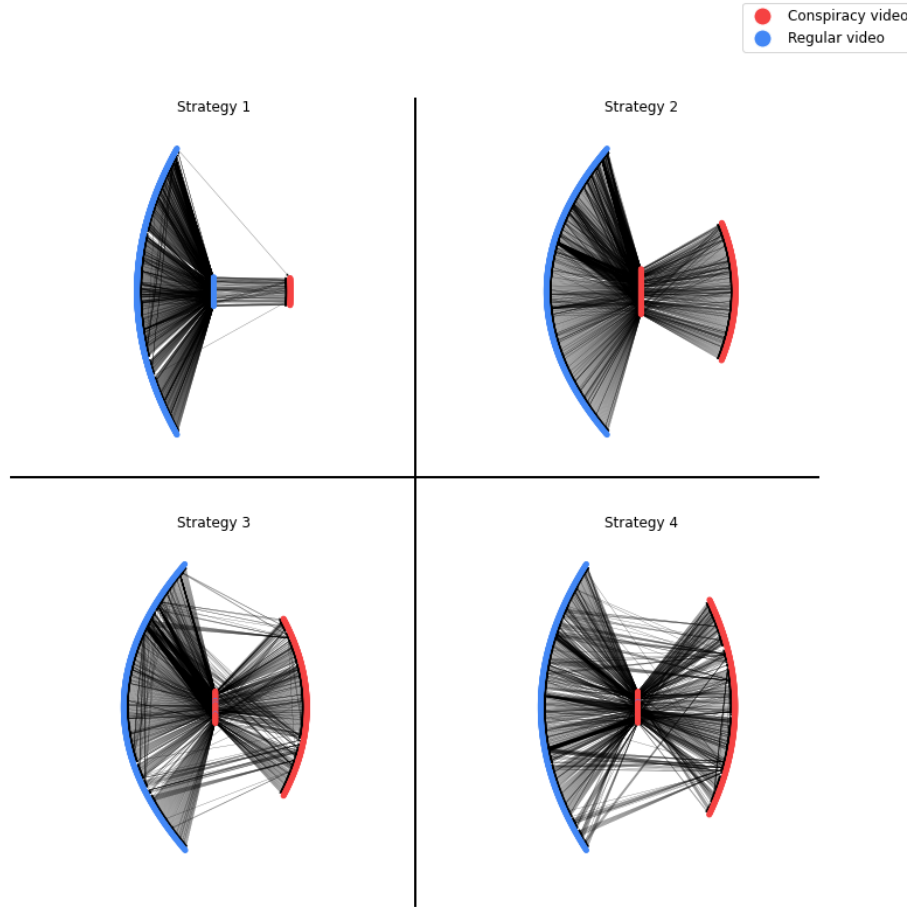


Figure 3: The networks of recommendations of each strategy. Vertices on the left indicate regular (non-conspiracy) videos, vertices in the middle are the videos that were actually watched by the bots, and vertices on the right are conspiracy videos. An edge not being present does not always indicate a link between videos was not present, as not all videos were clicked (and thus not all links could be checked).

Conspiracy recommendations

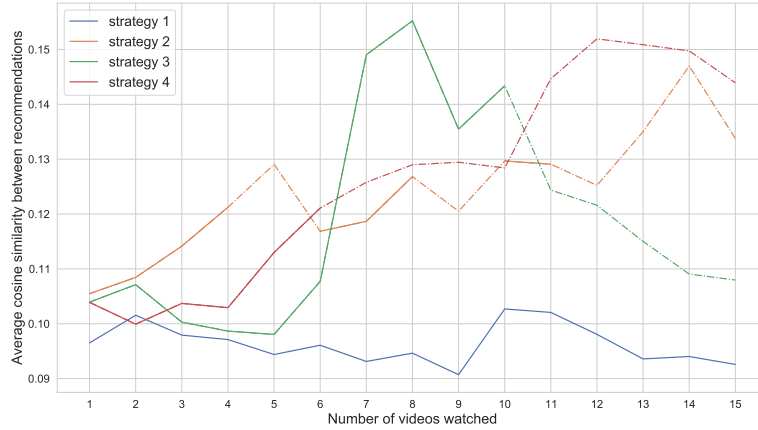


Figure 4: The average cosine similarity of the homepage recommendations per strategy after each number of videos watched. Each measure consists of the first twenty recommendations of the five users for each strategy. A dashed line indicates a significant difference compared to the baseline (strategy 1) at $\alpha = 0.05$.

4.1.2 Characteristics of the recommended content

Every strategy, excluding the baseline, eventually led to a decreased diversity of the recommended content. The similarity of the recommended content increased relatively slowly, however. When compared to the speed with which the recommendations started to prefer conspiracy content, the number of videos that needed to be watched until the average cosine similarity between recommendations became significantly higher than that of the baseline is quite high (figure 4).

Surprisingly, strategy 2 actually was the first strategy for which the difference in content-similarity became significant; considering the bots watched random conspiracy videos, it is surprising that the recommendations became similar at all. However, this only lasted for two videos, after which the difference became insignificant again until video nine. After having watched all fifteen videos, strategy 2 actually had more similar recommendations than strategy 3, which is remarkable, considering this strategy watched random (thus, seemingly unrelated) conspiracy videos.

Strategy 3 took the longest for the difference compared to baseline to become significant. While its similarity skyrocketed after video six, this high value was caused by a single outlier, causing it not to be significant. Eventually, the other values caught up to the outlier to some degree, causing the result to

454 become significant after all. Still, even then is the similarity lower than that of
455 strategy 2.

456 Strategy 4 took slightly longer than strategy 2 to show a significant
457 difference. However, this significance never disappeared afterwards. Strategy
458 4's similarity increased in two separate bursts of quick growth, both followed by
459 a slowdown. After fifteen videos, strategy 4's average similarity barely outper-
460 formed that of strategy 2, although this difference is insignificant ($p > 0.05$).

461 The average popularity of recommended videos dropped quickly for
462 all strategies, including the baseline. For all strategies, the average view count
463 of recommendations started at more than ten million. This number then rapidly
464 decreased, already dropping into the single millions or hundreds of thousands
465 after five videos watched. Thus, YouTube starts by recommending very popular
466 content, as that is most likely to cater to the largest audience. However, after
467 the user has watched a few videos, thereby giving the algorithm a better idea of
468 their interests, the recommendations become more personalized and therefore
469 less generic.

470 There seemed to be no trend in the average duration of the recom-
471 mended videos for any of the strategies. For each strategy, the average length
472 of the recommended videos stayed between three and ten minutes, spiking up
473 or down seemingly at random. Detailed graphs of the average view count and
474 duration of recommendations can be found in appendix 2.

4.2 Leaving the filter bubble

- 475 • The number of conspiracy recommendations drops quickly, but keeps hov-
476 ering a fair amount above baseline;
 - 477 – This means that YouTube quickly adapts to the new content being
478 watched, but still keeps the previous content in mind;
 - 479 – The amount at which it hovers, seems to be proportional to the initial
480 value of the filter bubble: if a user gets recommended more conspiracy
481 content at first, more of that will stay once they stop watching that
482 type of content;
- 483 • The number of videos until the algorithm stops recommending conspiracy
484 content is well over 15;
- 485 • This is interesting, as a user ends up in a bubble after ± 5 videos watched.

4.3 Machine learning

486 The hyperparameter tuning led to impressive scores for all classifiers. When
487 making predictions for the test set, the best-performing classifier is the support-
488 vector machine making use of the Radial Basis Function (RBF) kernel and a
489 penalty parameter (C-value) of 10. The SVM is tied for F1-score with the
490 neural network using the identity activation function, with 10 hidden layers of
491 10 neurons. Ridge regression with a sparse-cg solver and penalty (alpha) value of

Conspiracy recommendations

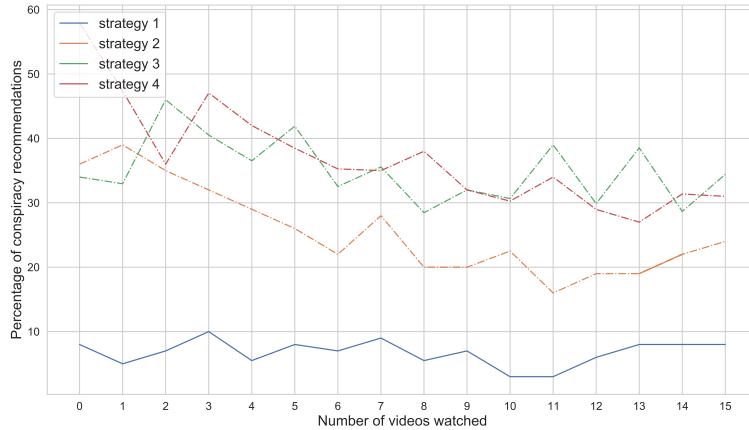


Figure 5: The percentage of conspiracy recommendations after each number of videos watched per strategy while trying to escape a filter bubble. Each measure is based on the top twenty recommendations of the five accounts for the strategy. A dashed line indicates a significant difference compared to the baseline (strategy 1) at $\alpha = 0.05$.

0.1 takes the third place, very closely followed by logistic regression with an L2 penalty, a penalty (C) value of 20 and a newton-cg solver. 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 6. The ten best-performing configurations for each classifier can be found in appendix 3.

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; Kamarathi 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

Classifier performance

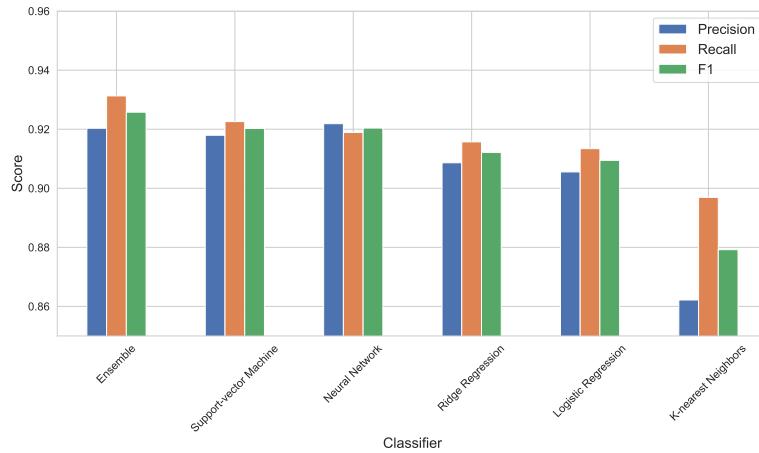


Figure 6: Metrics for each classifier with optimized hyperparameters

513 account, the support-vector machine is the best classifier for labeling conspiracy
 514 videos on YouTube.

515 Additionally, for predicting the likelihood of recommendations being
 516 conspiracy videos in real-time, as was done for strategy 3 and 4, the best-
 517 performing classifier is the neural network. Here, the neural network is preferred
 518 over the support-vector machine, as neural networks are better optimized for
 519 providing probabilities of samples belonging to a certain class (Specht, 1990).

5 Discussion

520 To do.

6 Conclusion

521 To do.

7 Future work

522 To do.

8 Planning

Table 4: 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	Ja
18/04-24/04	Inleiding uitbreiden	Ja
25/04-01/05	Uitvoeren experiment, labelen met classifier	Ja
02/05-08/05	Beginnen deelvraag 1	Ja
09/05-15/05	Deelvraag 1 afschrijven	Ja
16/05-22/05	Beginnen deelvraag 2	Ja
23/05-29/05	Deelvraag 2 afschrijven	
30/05-05/06	Discussie schrijven	
06/06-12/06	Tekst proof-readen, laatste aanpassingen	
13/06-19/06	Inleveren scriptie en verdediging	



References

- Birch, M. K. S. (2019). White rabbit. the logic and proportion of conspiracy theory videos on youtube: a foucauldian discourse analysis. *Malmö universitet*.
- Bozdag, E. and Van Den Hoven, J. (2015). Breaking the filter bubble: democracy and design. *Ethics and information technology*, 17(4):249–265.
- Bryant, L. V. (2020). The youtube algorithm and the alt-right filter bubble. *Open Information Science*, 4(1):85–90.
- 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.
- Cooper, P. (2020). How does the youtube algorithm work? a guide to getting more views.
- 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.
- Donzelli, G., Palomba, G., Federigi, I., Aquino, F., Cioni, L., Verani, M., Carducci, A., and Lopalco, P. (2018). Misinformation on vaccination: A quantitative analysis of youtube videos. *Human vaccines & immunotherapeutics*, 14(7):1654–1659.
- Feurer, M. and Hutter, F. (2019). Hyperparameter optimization. In *Automated Machine Learning*, pages 3–33. Springer, Cham.
- Kamathi, 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.
- Lang, P. (2018). Youtube average view duration - the 50% rule.
- 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.
- Maack, M. M. (2019). ‘youtube recommendations are toxic,’ says dev who worked on the algorithm.
- Miller, D. T. (2021). Characterizing qanon: Analysis of youtube comments presents new conclusions about a popular conservative conspiracy. *First Monday*.



- 559 Neufeld, D. (2021). The 50 most visited websites in the world.
- 560 O’Callaghan, D., Greene, D., Conway, M., Carthy, J., and Cunningham, P.
561 (2013). The extreme right filter bubble. *arXiv preprint arXiv:1308.6149*.
- 562 Paolillo, J. C. (2018). The flat earth phenomenon on youtube. *First Monday*.
- 563 Pariser, E. (2011). *The filter bubble: What the Internet is hiding from you*.
564 Penguin UK.
- 565 Park, M., Naaman, M., and Berger, J. (2016). A data-driven study of view
566 duration on youtube. In *Proceedings of the International AAAI Conference*
567 *on Web and Social Media*, volume 10.
- 568 Reitermanova, Z. (2010). Data splitting. In *WDS*, volume 10, pages 31–36.
- 569 Rosenbaum, L. (2021). Escaping catch-22—overcoming covid vaccine hesitancy.
- 570 Roth, C., Mazières, A., and Menezes, T. (2020). Tubes and bubbles topological
571 confinement of youtube recommendations. *PloS one*, 15(4):e0231703.
- 572 Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance
573 measures for classification tasks. *Information processing & management*,
574 45(4):427–437.
- 575 Specht, D. F. (1990). Probabilistic neural networks. *Neural networks*, 3(1):109–
576 118.
- 577 Sun, Y., Kamel, M. S., and Wang, Y. (2006). Boosting for learning multiple
578 classes with imbalanced class distribution. In *Sixth international conference*
579 *on data mining (ICDM’06)*, pages 592–602. IEEE.
- 580 YouTube (2021). Youtube community guidelines & policies - how youtube works.
- 581 Zhou, R., Khemmarat, S., and Gao, L. (2010). The impact of youtube rec-
582 ommendation system on video views. In *Proceedings of the 10th ACM*
583 *SIGCOMM conference on Internet measurement*, pages 404–410.

Appendices

Conspiracy recommendation networks

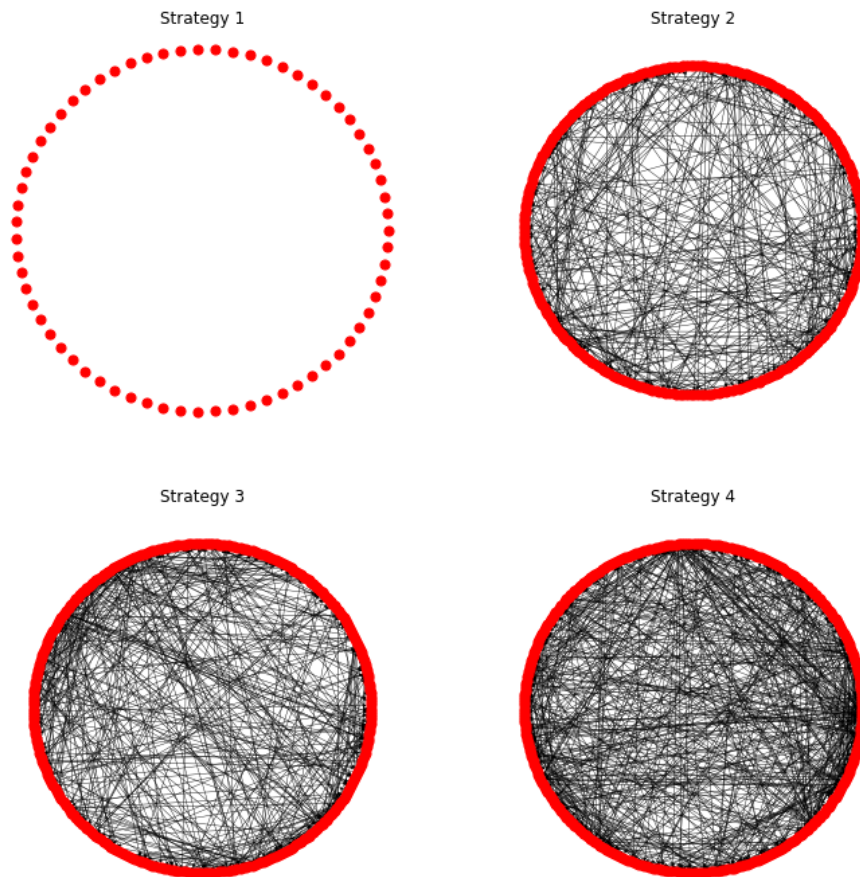


Figure 1: The recommendation networks of each strategy if only conspiracy videos are kept.

Average view counts

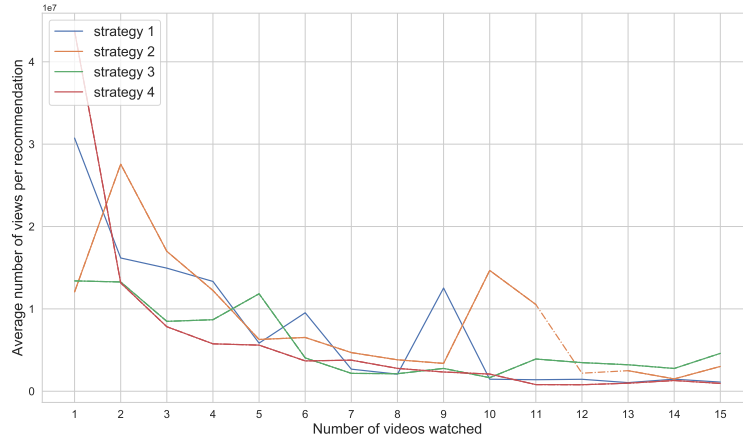


Figure 2: The average number of views per recommendation of each watch strategy. Each measure consists of the first twenty recommendations of the five users for each strategy. A dashed line indicates a significant difference compared to the baseline (strategy 1) at $\alpha = 0.05$.

Average video lengths

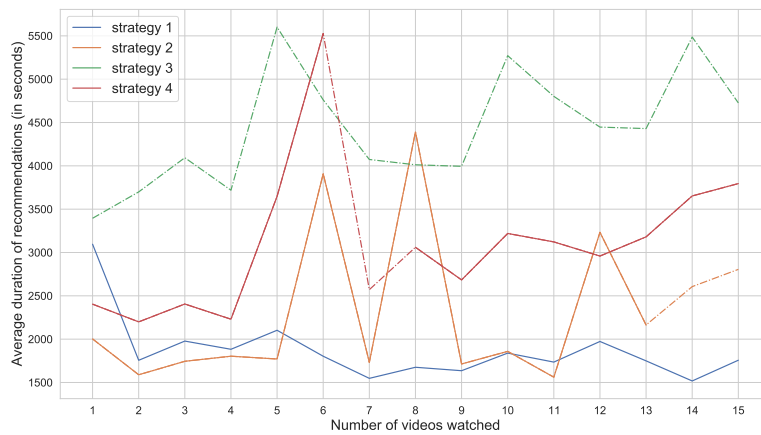


Figure 3: The average duration of the recommendations of each watch strategy in seconds. Each measure consists of the first twenty recommendations of the five users for each strategy. A dashed line indicates a significant difference compared to the baseline (strategy 1) at $\alpha = 0.05$.



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

(a) 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

(b) 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

(c) 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

(a) 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

(b) 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

(c) K-nearest neighbors.

Table 3: Results of the classifiers on the validation set with different hyperparameters. The best score per metric is written in bold.