VILNIUS GEDIMINAS TECHNICAL UNIVERSITY
FACULTY OF FUNDAMENTAL SCIENCES
DEPARTMENT OF MATHEMATICAL STATISTICS

Margarita Šliachina

REKLAMOS IR SEKIKLIŲ INTERNETINIUOSE PUSLAPIUOSE APTIKIMO METODŲ TYRIMAS
INVESTIGATION OF METHODS FOR TRACKER AND AD DETECTION ON WEB PAGES

Master's degree Thesis

Data Science and Statistics study programme, state code 6211AX009

Statistics study field

Vilnius, 2023

VILNIUS GEDIMINAS TECHNICAL UNIVERSITY
FACULTY OF FUNDAMENTAL SCIENCES
DEPARTMENT OF MATHEMATICAL STATISTICS

APPROVED BY
Head of Department

_____
(Signature)
prof. dr. Kazimieras Padvelskis
(Name, Surname)

_____
(Date)

Margarita Šliachina

REKLAMOS IR SEKIKLIŲ INTERNETINIUOSE PUSLAPIUOSE APTIKIMO METODŲ TYRIMAS
INVESTIGATION OF METHODS FOR TRACKER AND AD DETECTION ON WEB PAGES

Master's degree Thesis

Data Science and Statistics study programme, state code 6211AX009

Statistics study field

Supervisor dr. Viktoras Chadyšas _____ _____
(Title, Name, Surname) (Signature) (Date)

Consultant dr. Vaida Buivydienė _____ _____
(Title, Name, Surname) (Signature) (Date)

Vilnius, 2023

Vilniaus Gedimino technikos universitetas
Fundamentinių mokslų fakultetas
Matematinės statistikos katedra

ISBN | ISSN
Egz. sk. .....
Data .....-.....-.....

Antrosios pakopos studijų **Duomenų mokslo ir statistikos** programos magistro baigiamasis darbas	
Pavadinimas	**Reklamos ir sekiklių internetiniuose puslapiuose aptikimo metodų tyrimas**
Autorius	**Margarita Šliachina**
Vadovas	**Viktoras Chadyšas**

Kalba: anglų

Anotacija

Šiame darbe nagrinėjami metodai skirti reklamos ir sekiklių aptikimui internetiniuose puslapiuose. Tyrimo uždaviniai apėmė literatūros apžvalgą, duomenų rinkimo metodų analizę, modelių apmokymo ir vertinimo metodų analizę. Darbe modeliams apmokyti taikomi trys skirtingi metodai: URL analizė, URL ir HTTP analizė bei AdGraph metodas. Duomenų rinkimui naudoti tokie įrankiai kaip: tarpinis serveris, žiniatinklio tikrinimo programa ir Adblock Plus sąrašai. Modelių efektyvumui vertinti naudotos klasifikavimo uždavinio tikslumo metrikos , tokios kaip: klasifikavimo matrica, mokymosi ir ROC kreivės. Taip pat atlikta slenkančios prognozės analizė, skirta įvertinti modelių efektyvumą ilgalaikėje perspektyvoje. Be to, ištirta kiekvieno modelio požymio įtaka jo efektyvumui ir nustatyti reikšmingiausi. Rezultatai parodė, kad modelis, apmokytas su reikšmingiausiais požymiais, pasiekė geriausių rezultatų 98,7 % testavimo tikslumo.

Baigiamąjį darbą sudaro: įvadas, sekiklių ir reklamos apžvalga, esamų sprendimų apžvalga, tyrimo metodai, eksperimentas ir rezultatai, modelio gerinimas bei išvados. Darbo apimtis 51 puslapis, 22 paveikslai, 7 lentelės ir 16 literatūros šaltinių.

Prasminiai žodžiai: sekiklis, reklama, internetas, mašininis mokymasis, AdGraph, Adblock Plus, kibernetinė sauga

Vilnius Gediminas Technical University
Faculty of Fundamental Sciences
Department of Mathematical Statistics

ISBN | ISSN
Copies No. .....
Date .....-.....-.....

Master Degree Studies **Data Science and Statistics** study programme Master Graduation Thesis

Title	**Investigation of methods for tracker and ad detection on web pages**
Author	**Margarita Šliachina**
Academic supervisor	**Viktoras Chadyšas**

Thesis language: English

Annotation

This thesis explores techniques and tools for detecting trackers and advertisements on the web. The research tasks include conducting a comprehensive literature review, investigating data collection techniques, reviewing model evaluation methods, training machine learning models, evaluating their effectiveness, investigating the impact of different features, and performing a rolling forecast test. The literature review identified three approaches: URL analysis, URL and HTTP data analysis, and AdGraph. Data collection involved using proxies, crawlers, and Adblock Plus lists and constructing graphs from HTML pages for AdGraph. The model evaluation utilized traditional metrics like confusion matrix, learning curves, and ROC curves, with the addition of the rolling forecast test for long-term reliability. Results indicated that the model trained with the most optimal features exhibited superior performance, outperforming other models in accuracy and other metrics. This finding underscores the significance of identifying and utilizing the most relevant features for effective tracker and ad detection.

The structure of this document: introduction, review of online trackers and advertisements, review of already implemented solutions, methods and materials, experiment and results, model improvements, and conclusions. This document consists of 51 pages of text, 22 figures, 7 tables, and 16 sources.

Keywords: web tracking, advertisements, machine learning, cybersecurity, AdGraph, Adblock Plus

Table of Contents

1 Introduction 5
2 Review of online trackers and advertisements 7
2.1 Types of trackers 7
2.1.1 Cookies 7
2.1.2 Super cookies 9
2.1.3 Embedded code 9
2.1.4 Fingerprints 10
2.2 Types of advertisement and connection with trackers 10
3 Review of already implemented solutions 12
3.1 Adblock 12
3.2 AdGraph 13
3.3 URL analysis 15
3.4 HTTP analysis 16
4 Methods and materials 17
4.1 Experiment plan 17
4.2 Model evaluation techniques 18
4.2.1 Important feature detection 19
4.2.2 Rolling forecast 19
4.3 Data collection 21
4.3.1 Data collection using proxy and web crawler 21
4.3.2 Graph-based data collection 23
4.4 Data preprocessing 24
4.5 Data collection for the rolling forecast 26
5 Experiments and results 27
5.1 URL analysis method 27
5.2 HTTP analysis method 30

5.3 AdGraph method 32
5.4 Results of first experiments 35
6 Model improvements 37
6.1 Important feature detection 37
6.2 Model trained with most informative features 39
6.3 Rolling forecast 41
7 Conclusions 43
8 Recommendations 45
9 References 46
Appendix 48

List of Figures

Figure 2.1 Third-party and first-party URLs 8
Figure 3.1. AdGraph workflow 14
Figure 3.2 DTD data preprocessing 15
Figure 4.1 Data collection schema 22
Figure 4.2 Example: delfi.lt HTML graph 23
Figure 4.3 The first dataset 25
Figure 4.4 The second dataset 25
Figure 4.5 Data collected form the graph 26
Figure 5.1 URL method: confusion matrix 28
Figure 5.2 URL method: accuracy learning curves 29
Figure 5.3 URL method: ROC curve 29
Figure 5.4 HTTP method: confusion matrix 31
Figure 5.5 HTTP method: learning curves 31
Figure 5.6 HTTP method: ROC curve 32
Figure 5.7 AdGraph method: confusion matrix 33
Figure 5.8 AdGraph method: accuracy learning curves 34
Figure 5.9 AdFraph method: ROC curve 34
Figure 6.1 Feature correlation matrix 38
Figure 6.2 Random Forest features importance 38
Figure 6.3 Model with informative features: confusion matrix 40
Figure 6.4 Model with informative features: accuracy learning curves 40
Figure 6.5 Model with informative features: ROC curve 41

List of Tables

Table 3.1 AdGraph: structured features 14
Table 3.2 AdGraph: non-structured features 14
Table 5.1 URL analysis method test results 27
Table 5.2 HTTP analysis method results 30
Table 5.3 AdGraph method: results 33
Table 6.1 Model with informative features: results 39
Table 6.2 Rolling forecast results 42

1 Introduction

In recent years, online advertising has become an increasingly prevalent and pervasive aspect of our daily lives. From targeted advertisements on social media to sponsored search results on Google, the Internet is awash with advertising content designed to capture our attention and influence our behaviour. At the same time, the use of tracking technologies such as cookies, beacons, and device fingerprinting has become widespread, allowing advertisers to gather vast amounts of data about our online activities and preferences.

While online advertising and tracking technologies can offer benefits such as personalized content and improved user experiences, they also raise significant concerns about privacy, autonomy, and control. As online advertising becomes more sophisticated and pervasive, users are often unaware of the extent to which their online activities are being tracked and their personal data collected and used for commercial purposes. Moreover, the use of tracking technologies can raise concerns around consent, as users may not always be aware of or fully understand the implications of agreeing to be tracked.

This thesis aims to address these issues by investigating the use of neural networks for detecting tracking technologies and online advertisements. Specifically, developing and testing a machine learning model that can accurately identify if the URL is a tracker on not. By doing so, contributing to a better understanding of the extent and impact of online tracking and advertising, and to inform the development of more effective privacy protections and user controls.

This thesis focuses on detecting tracking technologies on the web using three different techniques: the URL analysis method, HTTP analysis method, and AdGraph. The URL analysis method provides fast detection without analysing the content of the source files, making it easy to implement. However, it is very straightforward and can only analyse the URL, which may not be enough. The HTTP analysis method provides more context about the URL, resulting in potentially more accurate results. However, collecting data for this method is challenging as it requires the implementation of a proxy. Lastly, AdGraph is the most challenging method, as it requires rendering of the whole HTML page. However, it provides a lot of information about the URL, making it potentially the most accurate method. Moreover, in this research all three methods are compared and analysed for the best features selection to develop a fourth method, that would be more robust and accurate. By utilizing these four different techniques, this study aims to provide a comprehensive and accurate analysis of tracking technologies and advertisements on the web.

Research object is the detection of trackers and advertisements on the web using neural networks. This includes exploring and comparing different detection techniques and evaluating the effectiveness.

The goal of the research is to analyse three already developed tracker and adds identification techniques and create a new one, more robust and accurate by finding most informative features.

Research tasks:

1. Conduct a comprehensive literature review to gather information about existing techniques and tools for detecting tracking technologies and advertisements on the web.
2. Conduct research on data collection techniques.
3. Review model evaluation techniques.
4. Train machine learning model for detecting trackers and advertisements on the web based on reviewed literature and evaluating the effectiveness.
5. Investigate the impact of different features, identify the important ones, train an ML model with the important features, and compare the results.
6. Perform the rolling forecast test to evaluate the long-term reliability of each model.

2 Review of online trackers and advertisements

In recent years, the use of tracking technologies and advertisements on the web has become widespread, raising concerns about privacy and security for internet users. As a result, researchers have developed various techniques and tools for detecting and blocking these technologies. This section reviews the existing literature on the detection of tracking technologies and advertisements on the web. Also explores the advantages and disadvantages of these methods, as well as their limitations and potential areas for improvement.

2.1 Types of trackers

A tracker, in the context of the internet, is a type of technology that is used to collect and record data about a user's online activities. This data can include information such as the websites visited, the search terms used, the links clicked, and even the user's location and device information. Trackers are typically implemented by third-party companies, such as advertisers or data brokers, and are embedded in websites and mobile applications through code snippets or scripts. The primary purpose of trackers is to gather information about users' online behaviour in order to deliver targeted advertisements and personalized content. However, the use of trackers has raised concerns about privacy and security, as users may not be aware of what data is being collected about them or how it is being used.

According to M. Kelly (2019), there are four main types of trackers that are commonly used on the web: cookies, super cookies, fingerprints, and embedded code. Each of these four types of trackers has its own strengths and weaknesses, and detecting and blocking them can be a challenging task for users and researchers alike.

2.1.1 Cookies

According to McQuade (2018) cookies are small text files that are stored on a user's device. It is the most common type of tracker used on the web to track user behaviour and preferences. When a user visits a website, the website can send a small text file called a cookie to the user's device. This cookie can contain information such as the user's login status, preferences, and browsing history on that website. The next time the user visits the same website, the website can read the cookie to retrieve the stored information and use it to personalize the user's experience or display targeted advertisements. However, cookies can also be used to track users across different websites that use the same advertising network. By

setting a unique identifier in the cookie, the advertising network can track the user's activity across different websites and deliver targeted advertisements based on their browsing history.

One way that cookies can be used to track users is through third-party cookies. Third-party cookies are cookies that are set by a domain other than the one the user is visiting. For example, if a user visits Website A and Website A includes content from Website B, Website B can set a third-party cookie on the user's device. This third-party cookie can be used to track the user's activity across both Website A and Website B, even if the user has not directly interacted with Website B. This type of tracking can be particularly concerning for users who value their privacy, as they may not be aware of which third-party domains are setting cookies on their device or how their data is being used. In response to these concerns, some web browsers have implemented measures to restrict or block third-party cookies by default. The Figure 2.1 shows, how cookies look like on *delfi.lt* webpage. Cookies in green rectangular are first party, others are third-party.

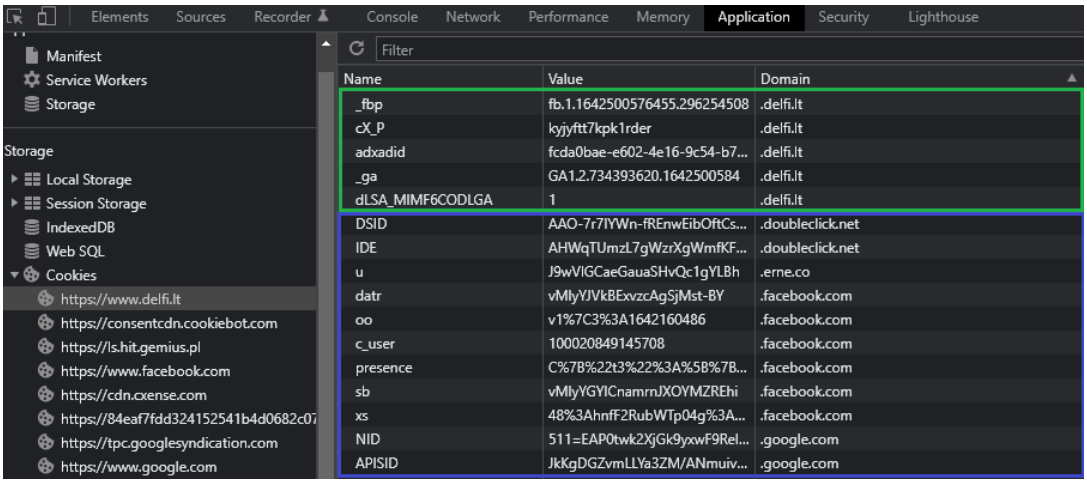

Figure 2.1 Third-party and first-party URLs

Third-party and first-party cookies can also differ in their expiry dates, which can affect how long a user's data is stored and how long they can be tracked. First-party cookies are set by the domain that the user is visiting, and their expiry date is determined by the website's settings. For example, a first-party cookie may be set to expire when the user closes their web browser or after a certain amount of time has elapsed. This can help to limit the amount of data that is stored on the user's device and reduce the risk of data breaches or unauthorized access.

In contrast, third-party cookies are set by domains other than the one the user is visiting, and their expiry date is typically set to a longer period of time. This is because third-party cookies are often used for advertising and tracking purposes, and longer expiry dates can help to ensure that user data is available for future use. However, this can also increase the risk of data breaches or unauthorized access, as the user's data is stored on a third-party domain that

may have weaker security measures than the user's own device. Some web browsers and privacy tools allow users to block or limit the use of third-party cookies, which can help to reduce the amount of data that is stored and improve user privacy.

To sum up, cookies are a common type of tracker used on the web to store user data and track behaviour. They can be divided into two types: first-party cookies, which are set by the domain that the user is visiting, and third-party cookies, which are set by domains other than the one the user is visiting. Cookies can be used to store information such as login status, preferences, and browsing history, which can be used to personalize the user's experience or display targeted advertisements. However, they can also be used to track users across different websites that use the same advertising network. Third-party cookies are often used for advertising and tracking purposes, and their longer expiry dates can increase the risk of data breaches or unauthorized access. Some web browsers and privacy tools allow users to limit or block the use of cookies, which can help to improve user privacy.

2.1.2 Super cookies

Super cookies are a type of tracking technology that are more persistent and difficult to delete than regular cookies. Also known as Flash cookies or Local Shared Objects (LSOs), super cookies are stored in a different location than regular cookies and can be used to track users across multiple websites. Super cookies can also be used to restore regular cookies that have been deleted, making them a powerful tracking tool for advertisers.

Unfortunately, detecting super cookies in the research may not be possible due to their unique storage. Since the research is focused on detecting trackers using specific techniques, it may not be able to identify super cookies that are stored outside of the browser's traditional cookie storage location. However, it is still important to acknowledge the potential privacy risks posed by super cookies.

2.1.3 Embedded code

According to Ivanov S. (2023) embedded code trackers are another type of tracking technology used by advertisers to collect user data. They are small snippets of code that are embedded in websites and can be used to track user activity across multiple sites. This can include collecting data on what pages a user visits, what links they click on, and even what items they purchase online. While embedded code trackers can be useful for advertisers to create targeted ads, they can also be dangerous if they fall into the wrong hands. Hackers can

use these trackers to collect sensitive user information such as login credentials or credit card details, which can then be used for malicious purposes.

It is important for users to be aware of embedded code trackers and take steps to protect their privacy online. This can include using browser extensions or ad blockers to block tracking scripts, regularly clearing browser data, and being cautious when entering sensitive information online.

2.1.4 Fingerprints

According to Krishnamurthy (2018) fingerprinting is a technique used by advertisers to track users without the use of cookies or other traditional tracking methods. Instead, fingerprints rely on collecting data from a user's browser and device, such as the operating system, screen resolution, and installed fonts. This data is then used to create a unique "fingerprint" that can be used to track the user across different websites and devices. Because fingerprinting is not reliant on cookies, it can be much harder for users to detect and block.

Fingerprinting has become increasingly popular in recent years, as it allows advertisers to track users even when they have disabled cookies or are using private browsing modes. However, it has also been criticized for its potential to compromise user privacy and security. Because fingerprints can be used to uniquely identify individual users, they can be used for targeted advertising and even tracking by governments or other malicious actors.

In the research, fingerprinting will not be used as a method of tracking detection. This is due to the fact that fingerprinting is often harder to detect and block than other tracking methods, as it relies on collecting data from a user's browser and device rather than relying on explicit tracking cookies or code. As such, it can be much harder for the research to reliably detect and track fingerprints across different websites. For these reasons, the research will focus on other tracking methods such as cookies and embedded code, which are more commonly used and easier to detect.

2.2 Types of advertisement and connection with trackers

Online advertising is a pervasive aspect of the modern web, with many websites relying on advertising revenue to stay afloat. There are several types of online advertisements, including display ads, text ads, video ads, and pop-up ads, among others. While these ads can sometimes be useful or informative to users, they can also be intrusive and annoying, detracting from the user experience and potentially even harming the user's online privacy and security.

For example, some ads may contain malware or other malicious code, which can infect a user's device or compromise their data.

However, it is worth noting that not all online advertisements are created equal, and there are some types of ads that may be less intrusive or more targeted to a user's interests. For example, targeted ads may be based on a user's browsing history or search queries and may be more likely to be relevant to the user's interests. Similarly, native ads may be designed to blend in with a website's content, rather than standing out as an obvious ad. Ultimately, the impact of online advertising on the user experience will depend on a variety of factors, including the type of ad, the website displaying the ad, and the user's personal preferences (Kumar, 2021).

Learning about trackers and advertisements and how they works makes it obvious that these are tightly connected, as advertisers often use trackers to monitor user behaviour and deliver targeted ads. By tracking a user's browsing history, searches, and clicks, trackers can build a profile of the user's interests and preferences, which advertisers can use to deliver more relevant ads. In fact, many online ad platforms such as Google AdWords and Facebook Ads rely heavily on tracking technology to deliver personalized ads to users. This close relationship between ads and trackers means that users are often subjected to a barrage of ads that seem eerily relevant to their recent online activity. While targeted advertising can be effective for advertisers, it can also feel invasive and creepy for users who may not realize just how closely their online behaviour is being monitored.

3 Review of already implemented solutions

The "Review of already implemented solutions" section of the thesis aims to provide an overview of the existing techniques and methods that have been used to address the issues related to online tracking and advertising. This section will examine various approaches that have been proposed in academic research as well as commercial solutions that are currently available. By analysing these existing solutions, we can identify their strengths and weaknesses and gain insights into potential areas for improvement. This section will serve as a foundation for the development of the proposed methodology in the subsequent sections of the thesis.

3.1 Adblock

According to Ablock official page, Adblock is a software that can be used to block online advertisements and trackers on websites. It is a browser extension that is installed on the user's browser, such as Chrome, Firefox, or Safari. Adblock works by examining the code of each webpage as it loads, and identifying any elements that match its database of known advertisements. Once it identifies an ad, Adblock prevents it from being displayed on the page.

Adblock can be highly effective in reducing the number of ads that are displayed to the user, as it can block all kinds of ads, including pop-ups, banners, and video ads. Additionally, Adblock can help to speed up web browsing by reducing the amount of data that needs to be downloaded, as ads can often be large and take longer to load than the actual content of the webpage. However, Adblock is not always perfect, and sometimes it can fail to block certain ads or interfere with the layout of a webpage.

Machine learning (ML) has emerged as a promising solution for addressing the challenges associated with online advertising and tracking. Unlike Adblock, which relies on pre-determined filter lists to block ads, ML algorithms can adapt to new and evolving advertising tactics, enabling them to detect and block unwanted content more effectively. Moreover, ML can be used to analyse the behaviour of both users and advertisers, providing insights into how and why online advertising and tracking take place. This allows for the development of more comprehensive and effective solutions that can address the root causes of the problem, rather than just the symptoms. Overall, ML has the potential to provide a more robust and sophisticated approach to online advertising and tracking, making it a more attractive solution than traditional ad-blocking methods (Jia et al., 2019).

In conclusion, while Adblock has been a popular solution for dealing with online advertising and tracking, it has limitations in terms of its ability to distinguish between

legitimate and harmful content. On the other hand, machine learning-based solutions have shown promise in being able to accurately detect and filter out unwanted content while allowing safe and relevant ads to pass through. Moreover, ML algorithms can continuously learn and adapt to new forms of advertising and tracking, providing a more robust and dynamic solution. Therefore, it is reasonable to conclude that machine learning-based solutions have the potential to be a more effective and efficient approach to addressing the issues of online advertising and tracking.

3.2 AdGraph

AdGraph is an automated system that uses machine learning techniques to detect and classify online advertising (Umar Iqbal et al., 2019). It was developed by researchers at the University of California, Berkeley, and is designed to provide a more accurate and comprehensive analysis of online advertising than traditional methods. AdGraph works by analysing the structure of web pages to identify and classify advertising content. It uses machine learning algorithms to identify patterns in the page structure that are indicative of advertising, such as the use of certain HTML tags or the presence of specific text strings.

One of the main advantages of AdGraph is its ability to detect advertising that is not easily identifiable using traditional methods. For example, AdGraph can identify advertising that is embedded in images or videos, or advertising that is displayed using JavaScript. AdGraph can also distinguish between diverse types of advertising, such as display ads, sponsored search results, and pop-ups. This level of granularity allows researchers to gain a more detailed understanding of the advertising ecosystem and how it impacts user behaviour.

The research paper "AdGraph: A Machine Learning Approach to Automatic and Effective Adblocking" (Khattak et al., 2019) presents a machine learning approach to adblocking. The technique uses graph-based features to classify web elements as either ads or non-ads. The authors evaluated AdGraph on a dataset of 25,000 web pages and achieved an overall accuracy of 98.6%.

The 3.1 figure shows how AdGraph works. In general, it gets the DOM tree from the HTML of the webpage and JavaScript code using Blink (Google Chrome engine) and VB (JavaScript engine) technologies. When the content is extracted, it creates a graph, that represents all the connections between HTML elements and JavaScript code. These connections are used to extract features for the model training. When features are extracted, it uses filter lists (such as Adblock) to label the dataset. As a result, classification model is trained on collected data and able to predict if a resource is an advertisement or not.

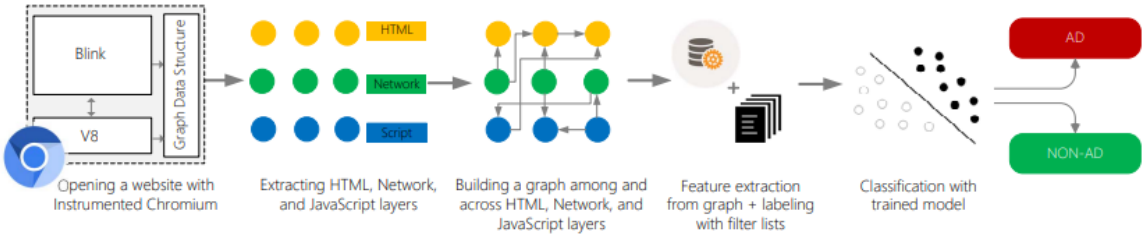

Figure 3.1. AdGraph workflow

Tables 3.1 and 3.2 represent example of features extracted by AdGraph. There are two types of features: structured and non-structured. Structured features represent the connections between HTML nodes and general information about created graph, while non-structured features represent information about triggered URL address.

Table 3.1 AdGraph: structured features

Structured features	
Features	Example
Size of a graph, number of references	30, 2 references
Number of connections	5 connections
Number of siblings and their types	1, <title>
Number of parents and their attributes	1, <img>
URL depth in DOM tree	3
Number of children and their attributes	0

Table 3.2 AdGraph: non-structured features

Non-structured features	
URL	Example (https://ads.google.com/home/)
Are there any ad related words?	Yes
URL parameters	None
Domain	google.com
Subdomain	ads
URL length	28
Is it third party?	No
How many semicolons?	0

To sum up, AdGraph is a new technique, and there is still much to be explored in terms of its potential and limitations. By conducting further research on AdGraph and its

effectiveness, this thesis contributes to the development and refinement of this technique. However, it is also important to consider the challenges and limitations of AdGraph. As mentioned, collecting the necessary data for AdGraph can be challenging, as it requires rendering of the entire webpage. In this thesis, I aim to explore the feasibility of using machine learning algorithms to detect and block online advertisements in a more efficient and less intrusive manner than current solutions like AdGraph. I am attempting to mock AdGraph and reduce the number of features it uses to improve the training of the model and make data collection less challenging and time consuming. By doing so, I hope to create a more practical and effective alternative to traditional ad-blocking solutions that will better meet the needs and expectations of modern users.

3.3 URL analysis

Another approach of detecting ads and trackers is Deep Tracking Detector (DTD) proposed by *Ismael Castell-Uroz et al.* The paper proposes a novel approach to detect web tracking by analysing URLs. The proposed approach uses a deep learning model called Deep Tracking Detector to learn the patterns of web tracking URLs. The DTD model is trained on a large dataset of known web tracking URLs and can detect and classify URLs as either tracking or non-tracking with 98.72% accuracy.

In this research, URLs are analysed based on their domain names and the presence of specific substrings that are indicative of tracking activities. The authors use regular expressions to define these substrings and develop a set of rules to identify them. The features extracted from the URLs are then used to train a deep neural network to classify URLs as either trackers or non-trackers. The network is designed to learn the patterns in the URLs and use them to accurately classify new URLs. The figure 3.2 shows, how DTD preprocesses the URL.

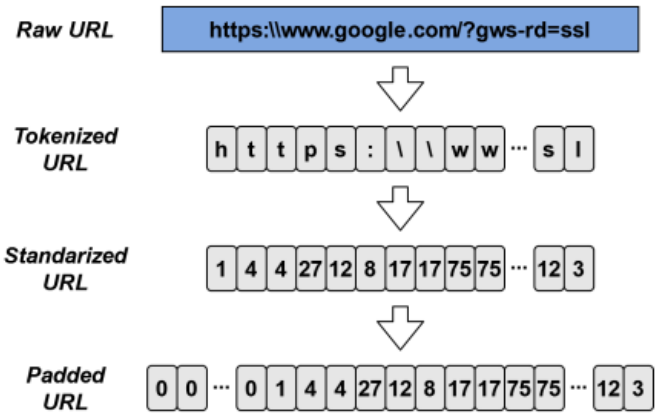

Figure 3.2 DTD data preprocessing

The advantages of the DTD method are its high accuracy in identifying tracking behaviour, its ability to detect both known and unknown trackers, and its use of machine learning techniques which can adapt to new tracking methods. Additionally, DTD does not require a priori knowledge of the trackers, making it more versatile and scalable than other methods that rely on predetermined lists of trackers.

However, there are also some potential disadvantages to using the DTD method. One limitation is that it requires a large amount of training data to achieve high accuracy, which can be difficult and time-consuming to collect. Another potential issue is the possibility of false positives or false negatives, where the DTD system may identify benign tracking behaviour as malicious or fail to identify certain types of trackers. Finally, the DTD method may require significant computational resources, making it challenging to implement on resource-limited devices or networks.

Overall, the paper presents a promising approach to detect web tracking by analysing URLs using deep learning techniques. The proposed approach has the potential to improve privacy and security in online browsing by helping users to identify and block tracking URLs.

3.4 HTTP analysis

The last method, represented in this thesis is based on paper "On Analyzing Third-party Tracking via Machine Learning." written by Alfonso Guarino et al. in 2020. The paper presents a study that uses machine learning to analyse third-party tracking on the web. The authors collected a large dataset of HTTP requests and responses from a variety of websites and used this data to train a machine learning model to identify third-party tracking requests.

Authors of the paper used a feature set that consists of both static and dynamic features. The static features include information extracted from the page's HTML source code, such as the number of third-party domains, the number of cookies, and the size of the page. The dynamic features are collected by running a headless browser to render the web page and capture network traffic. These features include the number of HTTP requests and responses, the number of redirects, the size of HTTP requests and responses, and the content type of resources. Additionally, the paper used a domain-based feature that counts the number of times a third-party domain appears on multiple websites, which can indicate its popularity and potential for tracking across the web.

Method proposed in the paper achieved a high accuracy of over 98% in identifying whether a third-party is tracking or not, and it was also able to identify the specific tracking domains and types of tracking behaviour. The authors also found that tracking behaviour is

prevalent on the web, with over 70% of the analysed pages containing some form of third-party tracking. Overall, the results suggest that machine learning techniques can be a powerful tool for understanding and detecting third-party tracking on the web.

As authors of the paper mentioned, the method has both advantages and disadvantages. One advantage is that it is able to accurately identify and classify third-party tracking behaviour on websites, providing insight into potentially malicious tracking activities. Additionally, the machine learning approach allows for efficient analysis of large datasets, making it a useful tool for researchers studying web tracking.

However, there are also some disadvantages. Collecting the necessary data to train the machine learning models can be time-consuming and require significant resources. Additionally, the models may not be able to keep up with new and evolving tracking techniques, meaning they may become less accurate over time. Another potential disadvantage is that the models may produce false positives or negatives, leading to incorrect identifications and classifications of tracking behaviour.

Overall, while the machine learning approach offers several advantages for analysing third-party tracking, it is important to consider the potential limitations and drawbacks of the method as well.

4 Methods and materials

This section describes the methods and materials used in a study to detect and analyse web tracking. The overall goal of the research is to investigate the effectiveness of machine learning techniques for detecting and mitigating web tracking. In the following subsections, provide a detailed overview of used methods and materials, including data collection and preprocessing, feature selection and extraction, model training and evaluation. Overall, this section provides a comprehensive account of the procedures and resources that were used to carry out the study.

4.1 Experiment plan

The idea behind the experiment is to train three machine learning models using the approaches that have been previously described. These models will be evaluated and compared to each other. The most informative features will be extracted from the models and used to create a new model. The aim is to develop a model that performs better than the existing ones.

The steps of the experiment:

1. Collection of a database using proxy.
2. Extract features from the collected data.

3. Train three models using different feature set, evaluate them and compare.
4. Find the most impactful features.
5. Using extracted most impactful features, train the model, evaluate and compare with other models.
6. Collect data for the rolling forecast and extract features.
7. Evaluate all four models using rolling forecast method.

4.2 Model evaluation techniques

Model evaluation is a crucial aspect of machine learning, which involves the assessment of the performance of a model in making predictions on new and unseen data. There are several techniques for evaluating machine learning models that are used in this experiment: the confusion matrix, k-fold cross-validation, learning curves, and ROC curves.

The confusion matrix is a useful tool for evaluating the performance of a classification model. It provides a summary of the number of correct and incorrect predictions made by the model and is particularly useful when dealing with imbalanced datasets. The confusion matrix consists of four components: true positives, true negatives, false positives, and false negatives. True positives refer to the number of positive cases that were correctly identified by the model, while true negatives refer to the number of negative cases that were correctly identified. False positives are the number of negative cases that were incorrectly classified as positive, while false negatives are the number of positive cases that were incorrectly classified as negative.

K-fold cross-validation is another commonly used method for evaluating machine learning models. It involves dividing the dataset into k subsets, or folds, and using k-1 folds for training the model and the remaining fold for testing. The process is repeated k times, with each fold used once for testing. K-fold cross-validation helps to reduce the risk of overfitting and provides a more reliable estimate of model performance than traditional train-test splits.

Learning curves are plots that show the performance of a model on both the training and validation datasets as the number of training examples increases. They can be used to identify whether a model is underfitting or overfitting and can help to determine the optimal amount of data needed to train the model. Learning curves can also be used to compare the performance of different models and to identify the impact of hyperparameter tuning on model performance.

ROC (Receiver Operating Characteristic) curves are plots that show the trade-off between the true positive rate and the false positive rate of a binary classification model. They are particularly useful for evaluating models that are designed to classify imbalanced datasets,

where the number of positive cases is significantly lower than the number of negative cases. ROC curves help to identify the optimal threshold for classifying positive and negative cases and can be used to compare the performance of different models.

In conclusion, there are several techniques for evaluating machine learning models, each with its own advantages and limitations. By using a combination of these methods, it is possible to obtain a more comprehensive assessment of model performance and make more informed decisions about model selection and optimization.

4.2.1 Important feature detection

One important aspect of building an effective machine learning model is identifying the most important features that contribute to the model's performance. One commonly used technique for feature selection is correlation matrix analysis, which measures the strength and direction of the linear relationship between each feature and the target variable. Features with high correlation coefficients are likely to be important predictors and can be selected for the model.

Another approach is using decision trees, which can identify the most discriminative features by recursively splitting the data based on the feature that maximizes the information gain. Random forest models can also be used for feature selection by measuring the importance of each feature based on how much it contributes to the reduction in the Gini impurity index. These feature selection techniques can help to reduce the dimensionality of the data and improve the accuracy and interpretability of the model, Marco Cerliani (2019).

In this thesis, both correlation matrix and tree-based approaches will be used to identify the most important features for the machine learning model. The correlation matrix method will help to identify the degree of association between different features, while the tree-based approach will help to identify the most informative features based on their contribution to the overall model performance. By combining both techniques, a more comprehensive understanding of the relationship between features and their impact on the model performance can be obtained, leading to better model optimization and selection of the most important features.

4.2.2 Rolling forecast

Rolling forecast is a time-series forecasting method that involves the creation of a model to predict future values based on historical data. The rolling forecast approach involves

splitting the dataset into training and testing sets, where the training set is used to develop the model, and the testing set is used to evaluate the model's performance. Instead of using a fixed testing set, the rolling forecast approach uses a moving window to continually update the testing set with new data points. This means that the model is retrained with new data points and the prediction is updated continuously, which is particularly useful for applications where the underlying data is constantly changing, Hyndman, R. J. (2018).

In rolling forecast, the window size refers to the number of data points used in the testing set. For example, if the window size is set to 30, the first 30 data points are used as the initial training set, and the 31st data point is used as the first test set. The model is then trained on the first 30 data points and used to make a prediction for the 31st data point. The testing set is then updated with the 32nd data point, and the model is retrained using the new training set, including the 31st data point, and used to make a prediction for the 32nd data point. This process is repeated for the remainder of the dataset.

Rolling forecast is particularly useful for detecting trackers and ads, as the situation is constantly changing. By using a rolling forecast approach, the model can be continuously updated with new data, which can help to identify new patterns and trends. This is particularly important in the world of online advertising, where new ads and trackers are constantly being created and deployed. By using a rolling forecast approach, the model can adapt to these changes and provide more accurate predictions over time. Furthermore, rolling forecast can help to identify any changes in user behaviour, which can be used to improve ad targeting and optimize the overall user experience.

4.3 Data collection

The first step in any machine learning project is to collect and preprocess the data. In this study, the data collection process involves the collection of website traffic data to be used to train and evaluate the performance of the models. The data will be collected using web crawling techniques and proxy that are designed to collect data. This process involves sending HTTP requests to the target websites and extracting the relevant information from the response. Since the focus of this study is on web tracking and advertisement detection, the collected data will be mainly composed of web page URLs, features related to web tracking and advertisement.

4.3.1 Data collection using proxy and web crawler

For data collection, a separate project was created using the Golang programming language, consisting of three key parts. Firstly, a proxy was created to intercept HTTP traffic between the web browser and server. The proxy was implemented using the "net/http" Golang library and intercepts network traffic by listening on a specified port, forwarding all incoming traffic to the server, and returning responses to the browser.

The second part of the project involved a web crawler to collect data from the top 10,000 web pages ranked by Alexa. The crawler was developed using the "chromedp" Golang library and mocks Google Chrome browser.

Lastly, Adblock Plus lists were used to label the collected data as ads or trackers. The "adblock-rust" open-source project available on GitHub was used to implement Adblock Plus filtering rules. This multi-part data collection approach aimed to ensure that the dataset was comprehensive and representative of real-world web browsing behaviour.

In this research, the data collection process is visualized in Figure 5.1. The process starts by running the proxy, which intercepts all requests and responses made by the crawler. Then, the crawler loads the top 10,000 web pages from the Alexa ranking list. As the pages are loading, the proxy collects all the information related to requests and responses, such as URL, headers, cookies, and parameters. This information is scraped and stored for further analysis. Next, the collected data is passed to the Adblock Plus library, which uses its rule-based system to identify and label each URL as an ad, tracker, or regular URL. The loop continues until all

the web pages are loaded, and the data collection process is completed. Finally, the collected information is stored in a text file for further analysis.

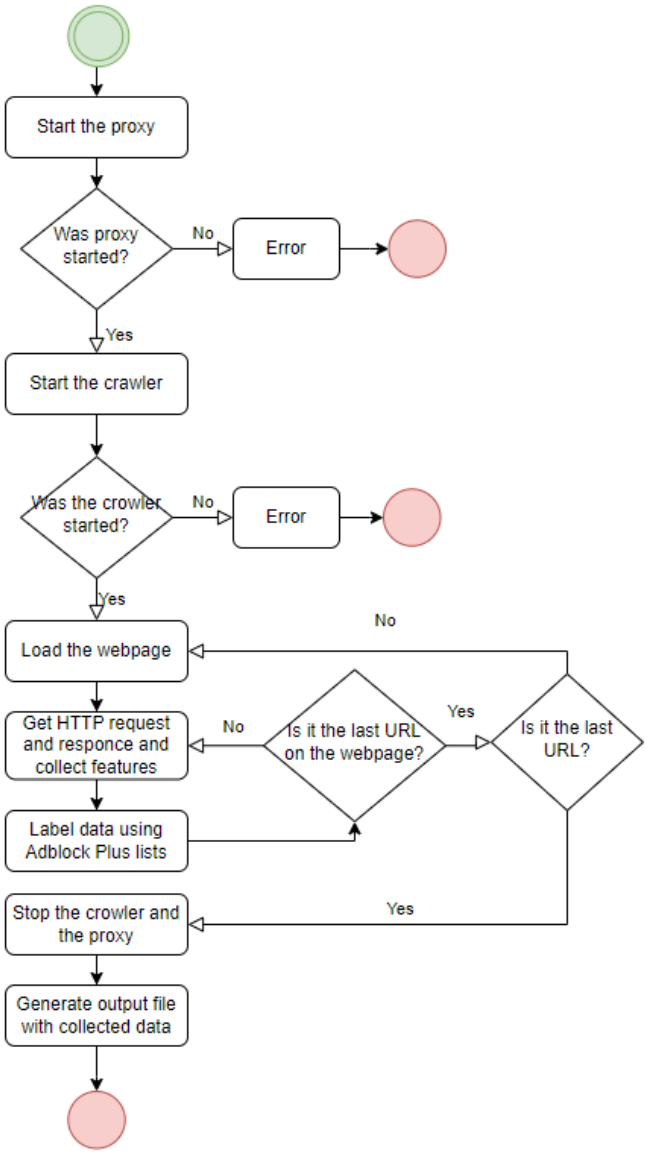

Figure 4.1 Data collection schema

The data collected includes several features that are commonly used in tracking and ad detection research. These features include the triggered URL, the domain of the visited webpage, the length of the URL, the number of sent and received cookies, the average size of the cookie files, the type of the request, the number of parameters in the URL, parameter values, whether the URL is a third-party, and the cookie expiration dates. These features were chosen because they have been previously shown to be informative in identifying tracking and ads in previously reviewed research.

4.3.2 Graph-based data collection

The second part of the data collection process involves the construction of an HTML graph. As AdGraph method uses the entire HTML page to extract features, the graph needs to be created from the Document Object Model (DOM) tree. To accomplish this, R programming language and the "igraph" library are used to construct the graph, while the "XML" library is utilized for HTML preprocessing. The HTML is loaded and converted into a tree structure, from which all relevant information is extracted. This includes details about the graph, such as its size and number of vertices, as well as information about each URL present in the graph. This information includes the name of the HTML node, the location of the URL, the depth of the node in the graph, the number of siblings the node has, and the types of sibling and parent nodes, among other details. All this information is then saved in a text file for further processing.

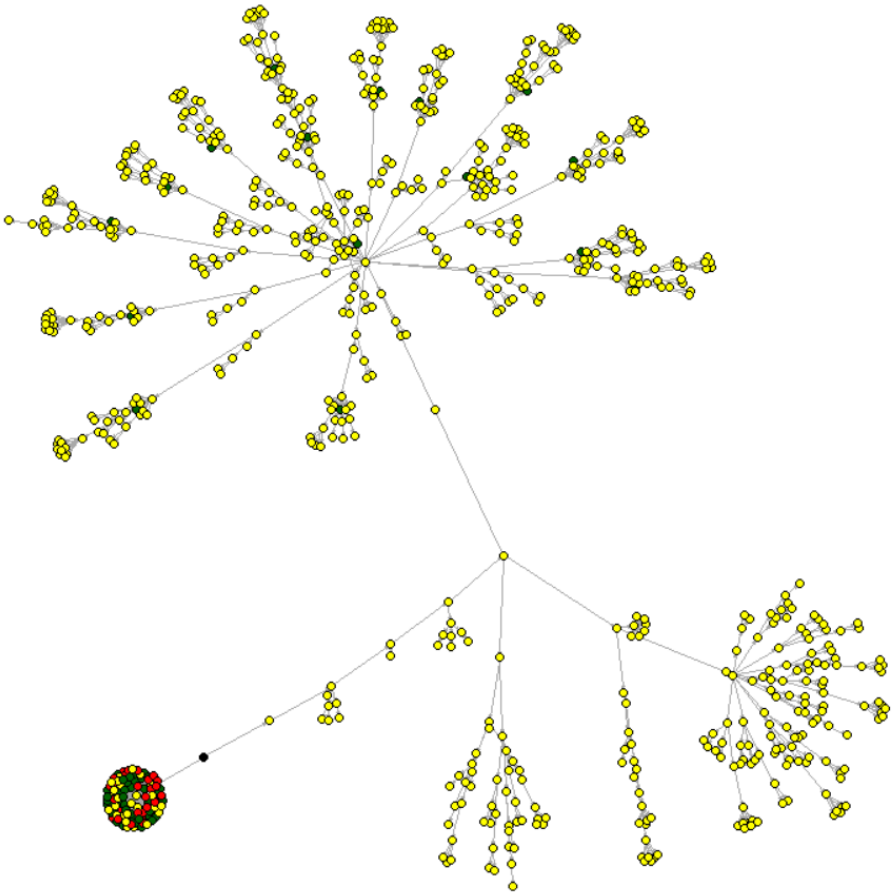

Figure 4.2 Example: delfi.lt HTML graph

Figure 4.2 shows an example of a graph constructed from the HTML page of *defi.lt*. The black vertex represents the root of the graph. The yellow vertices are HTML nodes, which

represent the different elements on the page, such as text, images, and links. The green vertices represent vertex with URLs that were classified as clean, while the red represent URLs that were classified as trackers or ads. Graphs like that are constructed for every HTML in the dataset and used to collect features for the experiment.

It is important to note, that collecting data from the web can be a challenging task due to the dynamic nature of the internet. Webpages can fail to load or load only partially due to various reasons such as server issues, network connectivity problems, and web content changes. This can result in missing or incomplete data, which can significantly impact the analysis and accuracy of the model. Additionally, some websites may have different structures, use various languages and frameworks, and have different security protocols that can make data collection even more complex. Furthermore, some websites may also have measures in place to detect and block web crawlers, which can further complicate the process of data collection. Therefore, it is essential to ensure that the data collected is complete, accurate, and representative to achieve reliable results from the model.

4.4 Data preprocessing

Data preprocessing is an essential step in machine learning projects that can significantly impact the performance of the model. It involves cleaning, transforming, and organizing raw data into a suitable format for analysis. The quality of the data determines the accuracy and reliability of the insights generated by the model. Data preprocessing aims to eliminate inconsistencies, errors, and anomalies in the data that can affect the performance of the model. It involves several techniques, such as data cleaning, feature selection, normalization, and data reduction. The objective of this subsection is to present the data preprocessing techniques used in this study to prepare the collected data for analysis.

After the data collection, the next step is data preprocessing, which is a crucial step in the data analysis process. In this thesis, Python programming language and its libraries were used to preprocess the collected data. The data preprocessing step involved cleaning and preparing the data for further analysis. Firstly, empty fields were removed from the dataset, and strings that exceeded a certain length were truncated. Next, the URLs, their types, and attributes were vectorized. These techniques were used to convert categorical variables into numerical vectors, which can be easily analysed by machine learning algorithms.

In the end, depending on the model type, three different datasets were created. The first dataset contains information about the loaded URL. Figure 5.2 shows an example of collected data.

	Domain	URL	Label
9	youtube.com	https://www.youtube.com/s/_/ytmainappweb/_/ss/...	CLEAN
10	youtube.com	https://www.youtube.com/s/_/ytmainappweb/_/ss/...	CLEAN
11	youtube.com	https://www.youtube.com/s/_/ytmainappweb/_/ss/...	CLEAN
12	linkedin.com	https://static.licdn.com/aero-v1/sc/h/5grujlzk...	CLEAN
13	linkedin.com	https://static.licdn.com/aero-v1/sc/h/5grujlzk...	CLEAN
...	...	...	...
10908	hatena.ne.jp	https://adservice.google.com/adsid/integrator....	NOTCLEAN
10909	hatena.ne.jp	https://adservice.google.lt/adsid/integrator.j...	NOTCLEAN
10918	theglobeandmail.com	https://s.go-mpulse.net/boomerang/JQ423-JN76F-...	NOTCLEAN
10920	theglobeandmail.com	https://adservice.google.lt/adsid/integrator.j...	NOTCLEAN
10921	theglobeandmail.com	https://adservice.google.com/adsid/integrator....	NOTCLEAN

Figure 4.3 The first dataset

The second dataset contains information about URLs and information extracted from HTTP requests and responses, such as the number of sent and received cookies, the average size of the cookie files, the type of request, the number of parameters in the URL, parameter values, whether it is a third-party URL, and cookie expiration dates. Figure 5.3 shows an example of data collected for the second model.

	URL length	Received cookies	Send cookies	Cookie size	Cookie duration	Parameters number	Received send cookie rate	Graph size	Reference number	Vertex number	...	Father attributes	Father path	URL attributes	URL node depth	Children number	Sibling number	URL	Subdomain	Parameters	Domain
0	32	0.0	0.0	0.0	0	0.0	0.0	30.0	24.0	9053.0	...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, ...	[1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	3	1.0	4	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0, 0, 0, 0, 0, 0, 0, 0]	[0, 0, 0, 0, 121, 111, 117, 116, 117, 98, 101,...
9	145	0.0	3.0	54.0	0	0.0	0.0	30.0	24.0	9053.0	...	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, ...	[1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	3	1.0	4	[104, 116, 116, 112, 115, 58, 47, 47, 119, 119...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0, 0, 0, 0, 0, 0, 0, 0]	[0, 0, 0, 0, 121, 111, 117, 116, 117, 98, 101,...
12	63	0.0	0.0	0.0	0	0.0	0.0	22.0	100.0	411.0	...	[1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, ...	[1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	3	1.0	4	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 115...	[0, 0, 0, 0, 0, 0, 0, 0]	[0, 0, 0, 108, 105, 110, 107, 101, 100, 105, 1...
16	44	0.0	0.0	0.0	0	0.0	0.0	18.0	160.0	1710.0	...	[1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, ...	[1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	3	1.0	3	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0, 0, 0, 0, 0, 0, 0, 0]	[0, 0, 0, 0, 0, 0, 0, 113, 113, 46, 99, ...
17	73	0.0	1.0	2.0	0	0.0	0.0	11.0	159.0	707.0	...	[1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, ...	[1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, ...	3	1.0	4	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[0, 0, 0, 0, 0, 0, 0, 0]	[0, 0, 0, 0, 0, 0, 0, 97, 112, 112, 108, 101, 46,...
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
10947	113	0.0	0.0	0.0	0	0.0	0.0	19.0	149.0	761.0	...	[1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, ...	[1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	[1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, ...	3	1.0	4	[104, 116, 116, 112, 115, 58, 47, 47, 105, 109...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 105...	[0, 0, 0, 0, 0, 0, 0, 0]	[0, 0, 116, 111, 107, 111, 112, 101, 100, 105,...

Figure 4.4 The second dataset

Finally, the third dataset is the largest, containing information about the URL, HTTP requests and responses, and information extracted from the HTML graph, such as the graph size, the number of vertices, and the number of reference links. Figure 5.4 shows an example of data collected for the third model. By creating these three different datasets, it is possible to analyse and compare the effectiveness of different models on different types of data.

	Graph size	Vertex number	Edge number	URL node depth	Children number	Sibling number	URL attributes
9	30.0	9053.0	9052.0	3	1.0	4	[1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
10	30.0	9060.0	9059.0	3	1.0	4	[1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
11	12.0	303.0	302.0	3	1.0	4	[1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
12	22.0	411.0	410.0	3	1.0	4	[1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
13	23.0	4300.0	4298.0	3	1.0	4	[1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
...	...	...	...	...	...	...	...
10908	12.0	1144.0	1143.0	3	1.0	4	[1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
10909	12.0	1144.0	1143.0	3	1.0	4	[1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
10918	21.0	2602.0	2601.0	3	1.0	4	[1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
10920	21.0	2602.0	2601.0	3	1.0	3	[1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
10921	21.0	2602.0	2601.0	3	1.0	3	[1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...

Figure 4.5 Data collected form the graph

To sum up, the data collection was performed through a combination of a Golang project for extracting information from HTTP requests and responses, as well as constructing an HTML graph, followed by data preprocessing using Python. As a result, three datasets were created, each containing different sets of features, depending on the model type. In total, around 1,000 web pages were successfully crawled, which resulted in the datasets containing around 10,000 rows of data. These datasets would later be used for training machine learning models to detect tracking and ads on web pages.

4.5 Data collection for the rolling forecast

The data collection for the rolling forecast involved utilizing a similar technique as the basic data collection, with the addition of downloading the latest Adblock lists to label the data. It is worth noting that the Adblock lists are continually updated. Once the data had been collected from the latest lists, the data used in the previous experiment was extracted to prevent any overlap between the two datasets.

For the rolling forecast, the latest Adblock Plus lists were downloaded and utilized. These lists were obtained on the 2nd of May, 2023. To conduct the evaluation of the models, a total of 100 top webpages were crawled, and from this set, 500 URLs were extracted for further analysis. These selected URLs serve as the dataset for evaluating and comparing the performance of the different models.

The data collected for the rolling forecast is utilized to evaluate the trained models and determine the best one. This process ensures that the most recent information is used for model evaluation, resulting in more accurate predictions.

5 Experiments and results

The following section provides an overview of the first experiments and results obtained in the study. The purpose of the experiments was to evaluate the performance of several machine learning algorithms for ad and tracker classification. The experiments were conducted on three different datasets, which were prepared based on the collected and preprocessed data. The results of the experiments are presented and discussed in detail, including the accuracy, precision and recall of each algorithm. Additionally, the performance of the algorithms was compared using the receiver operating characteristic (ROC) curve and the area under the curve (AUC) metric. Overall, the first experiments and results serve as a foundation for further investigation and optimization of the classification models.

5.1 URL analysis method

The URL-based model was trained using a dataset of 11K URLs. To test the model, 10% of the collected data was used and 20% of the training data was used for validation. A feed-forward neural network was utilized to train the model, consisting of four dense layers flattened with dropout layers. The first three layers employ ReLu activation, while the last layer utilizes sigmoid. The model was trained for a total of ten epochs. Cross-validation was also performed, to make sure, that results are consistent.

The Table 5.1 presents the evaluation results of the model that was trained using only URL data. The model achieved high accuracy with 98% for training and 97% for validation. However, there is a slight overfitting issue, which is indicated by the gap between the training and validation accuracy. On the test dataset, the model achieved an accuracy of 97% and demonstrated high precision and recall rates of 95%. To provide a more detailed analysis of the model's performance, further evaluations were conducted using confusion matrix, ROC curve, and learning curves.

Table 5.1 URL analysis method test results

Metric	Percentage (%)
Train accuracy	98.726
Validation accuracy	97.816
Test accuracy	97.812
Precision	95.324
Recall	95.975

In Figure 5.1, the confusion matrix of the model is depicted, showcasing the model's performance in correctly classifying instances as either clean or trackers. It is observed that out of a total of 636 instances labelled as clean, the model accurately predicted 625 of them. Similarly, out of 278 instances labelled as trackers, the model correctly classified 265 instances.

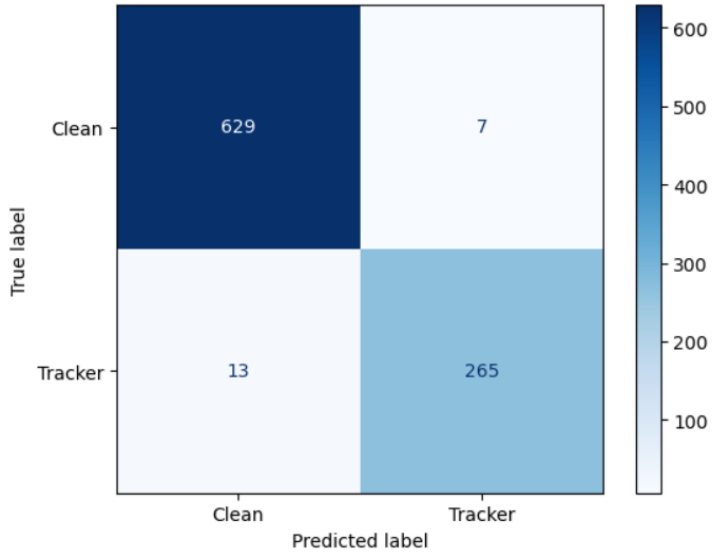

Figure 5.1 URL method: confusion matrix

Figure 5.2 displays the learning curves of the model, depicting the progress of training and validation accuracy as the number of training examples increases. Both curves show a gradual upward trend, indicating that the model is learning and improving over time. The small gap between the training accuracy curve and the validation accuracy curve, which is less than 1 percent, suggests that the model generalizes well to unseen data. This close proximity indicates that the model's performance on the training set is consistent with its performance on the validation set, indicating that overfitting is not a significant concern. Overall, learning curves demonstrate the model's ability to learn effectively and achieve similar levels of accuracy on both the training and validation datasets.

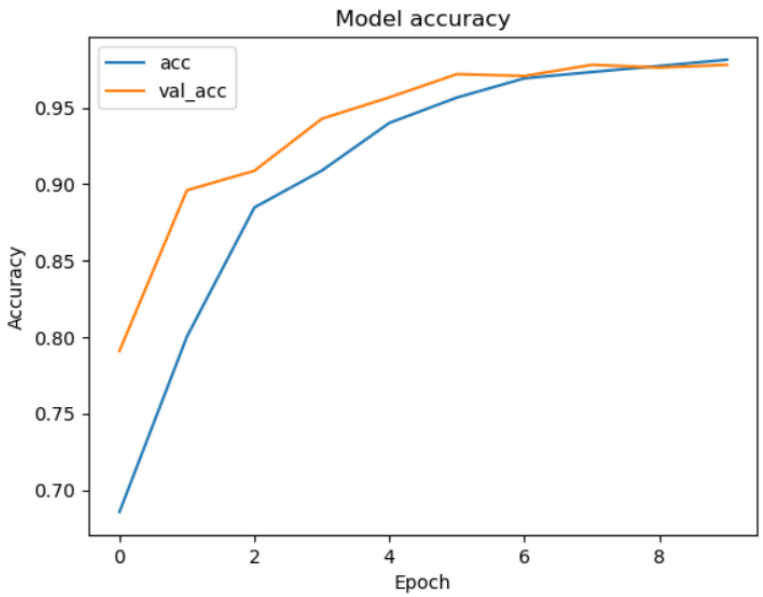

Figure 5.2 URL method: accuracy learning curves

The Figure 5.3 depicts the ROC curve of the model, which shows the relationship between True Positive Rate (TPR) and False Positive Rate (FPR). The plot indicates an AUC of 0.971, indicating reliable performance of the model in terms of distinguishing between clean URLs and trackers. The ROC curve is close to the top left corner, which indicates that the model is doing well in terms of minimizing false positives and false negatives. Overall, the ROC curve shows that the model has good performance in detecting trackers while minimizing false positives.

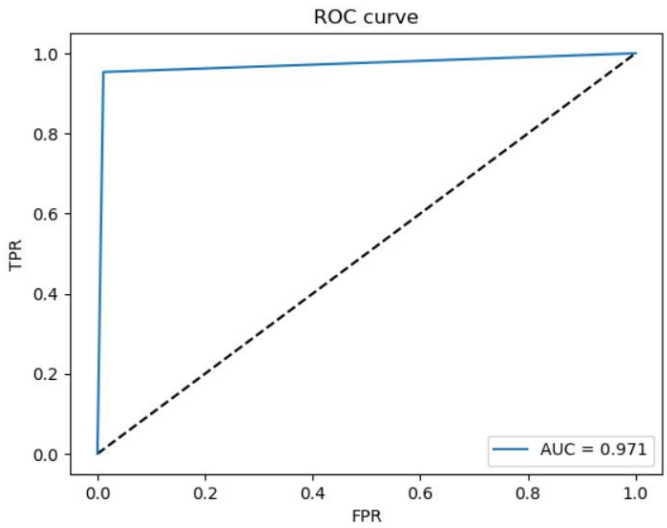

Figure 5.3 URL method: ROC curve

The model trained using only URL data has shown promising results with high accuracy, precision, and recall for both clean URLs and trackers. But the learning curves curve showed that the model's performance improves as the number of epochs increases, so further training can be conducted. Overall, the model that uses only URL data is a promising solution

for detecting tracker URLs as it has shown good results with minimal data collection efforts. With more data, the model's overfitting can be reduced, which would improve its performance even more.

5.2 HTTP analysis method

Another method used in this study was model training with both URL and HTTP features. The validation, training, and testing splits were the same as in the first method, and a feed-forward neural network was used. Table 5.2 presents the results of this method, showing that the training accuracy was 99%, validation accuracy was also almost 98%, and test accuracy was 98%. The precision was 96%, recall was 98%, false positive rate was 5%, and false negative rate was 1%.

Table 5.2 HTTP analysis method results

Metric	Percentage (%)
Train accuracy	99.867
Validation accuracy	97.933
Test accuracy	98.687
Precision	96.751
Recall	98.600

The confusion matrix for the model trained on URL and HTTP features represented in Figure 5.4. shows that the model was able to correctly classify 634 clean URLs out of 637 and correctly classified 268 trackers out of 277. This shows that the model was better at classifying trackers than the previous method which only used URL data. However, the model still made some misclassifications as seen from the false positive rate of 1% and false negative rate of 1%.

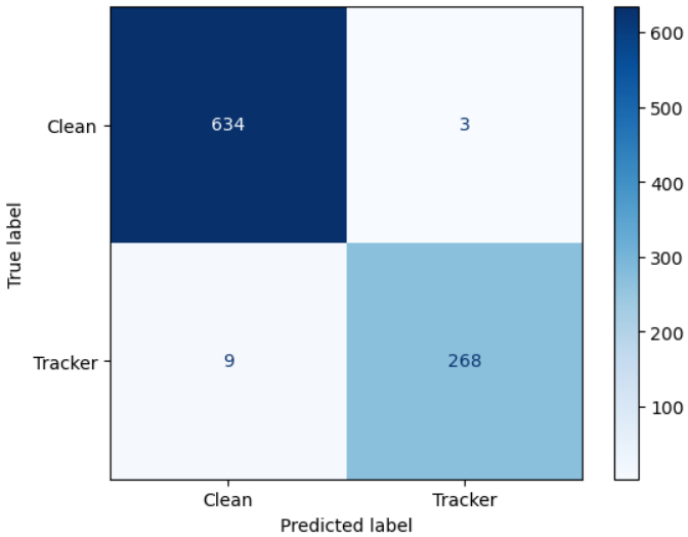

Figure 5.4 HTTP method: confusion matrix

Figure 5.5 showcases learning curves of the model, depicting the progression of training and validation accuracy over multiple epochs. As the number of epochs increases, the training accuracy steadily improves, suggesting that the model is capturing the underlying patterns and becoming more adept at classifying the data. However, there is a slight discrepancy between the training and validation accuracy curves, with the training accuracy consistently higher. This indicates a small degree of overfitting, where the model may be excessively tailored to the training data and not generalizing as well to unseen examples. Despite this minor overfitting, the difference between the two curves remains less than 2 percent, implying that the model still performs reasonably well on the validation set. In conclusion, learning curves demonstrate that the model exhibits a learning capacity, but further measures should be taken to mitigate overfitting and enhance its generalization capabilities.

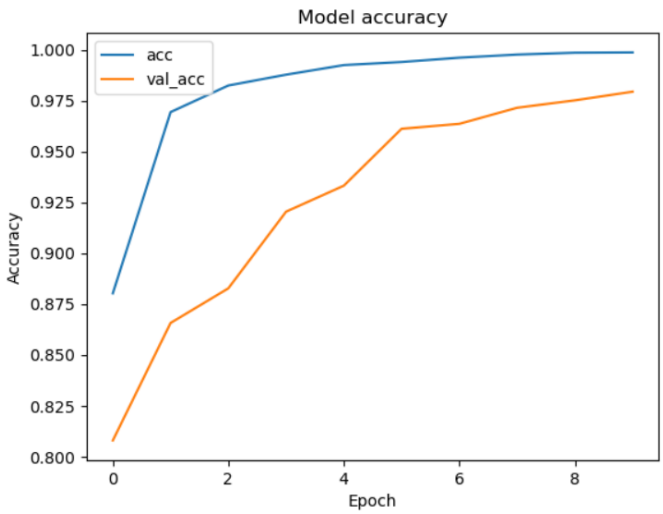

Figure 5.5 HTTP method: learning curves

The ROC curve in Figure 5.6 represents the performance of the model trained using both URL and HTTP information. The AUC plot for this model is 0.981. In comparison, the other models that were trained solely with URL data achieved an AUC curve of 0.97. This comparison indicates that the model incorporating both URL and HTTP features outperforms the models trained with only URL data in terms of classification accuracy and discrimination capability. The higher AUC value for the model suggests that it has a better ability to distinguish between clean URLs and trackers. Consequently, incorporating HTTP features in the training process improves the model's performance and enhances its predictive capabilities. In conclusion, the results demonstrate the value of including HTTP information alongside URL data, leading to a more effective and accurate model for classifying URLs and identifying malicious instances.

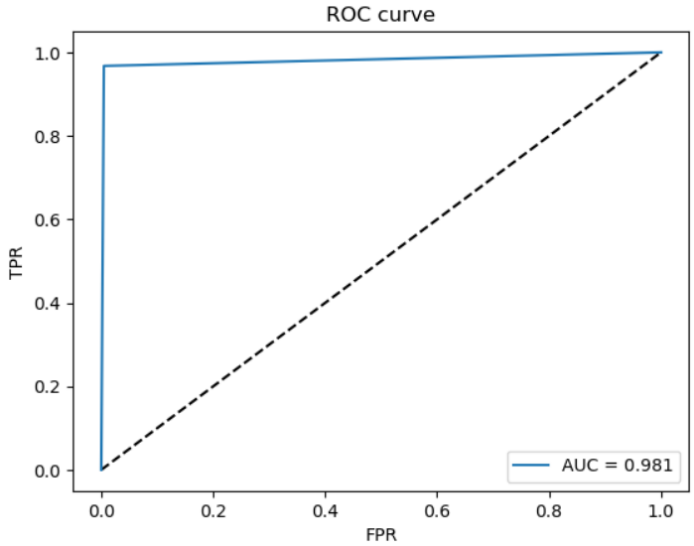

Figure 5.6 HTTP method: ROC curve

In summary, the second method of model training using both URL and HTTP features has shown promising results but also exhibited overfitting. The train accuracy was almost 99%, and the validation and test accuracies were 97% and 97%, respectively. The confusion matrix showed that the model performed better in classifying trackers than clean URLs. Learning curves indicated that more data might be necessary to improve the model's performance and reduce overfitting. The ROC curve showed an AUC of 0.98, which was slightly higher than the first method's AUC of 0.97.

5.3 AdGraph method

The third trained model used AdGraph approach. The model was trained using a feed-forward neural network with 10 epochs, and cross-validation was applied to ensure the stability

of the results. The results presented in Table 5.3 indicate that the model achieved a train accuracy of 99.8%, a validation accuracy of 99%, and a test accuracy of 98%. Furthermore, the model exhibited a precision rate of 96% and a recall rate of 98%.

Table 5.3 AdGraph method: results

Metric	Percentage (%)
Train accuracy	99.920
Validation accuracy	98.241
Test accuracy	98.359
Precision	96.333
Recall	98.229

The confusion matrix in Figure 5.7 provides an overview of the performance of the AdGraph model. The matrix shows that the model was able to correctly classify 612 clean URLs out of 614, and it correctly identified 288 trackers out of 300. This indicates that the model performed well in identifying the different types of URLs, with a high level of accuracy.

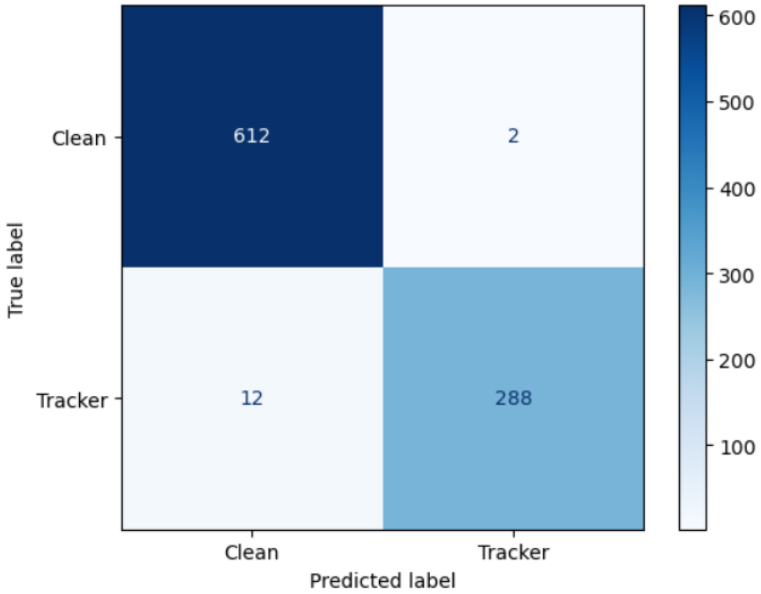

Figure 5.7 AdGraph method: confusion matrix

The accuracy learning curves in Figure 5.8 demonstrates that the AdGraph model was able to learn the classification task efficiently and consistently. Both the training and validation accuracy curves show an upward trend with very little fluctuation, indicating that the model was stable during the training process. Moreover, the gap between the training and validation accuracy curves is almost 2 percent, which suggests that the model was slightly overfitting on

the training data. Therefore, it can be inferred that the AdGraph approach produced a well-generalized model.

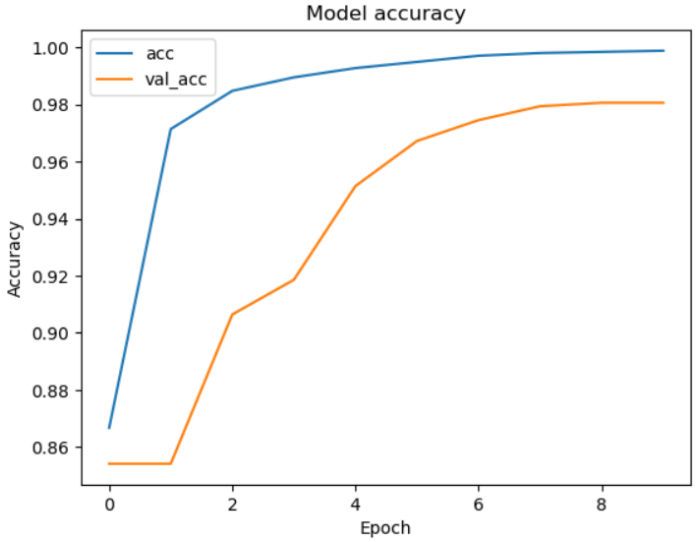

Figure 5.8 AdGraph method: accuracy learning curves

In Figure 5.9, there is a ROC curve that shows the performance of the AdGraph approach model. The AUC plot of the ROC curve is 0.978, indicating that the model has a high discriminatory power in distinguishing between clean and tracker URLs. The high AUC score and the shape of the ROC curve demonstrate that the AdGraph approach model performs well in identifying tracker URLs with high accuracy.

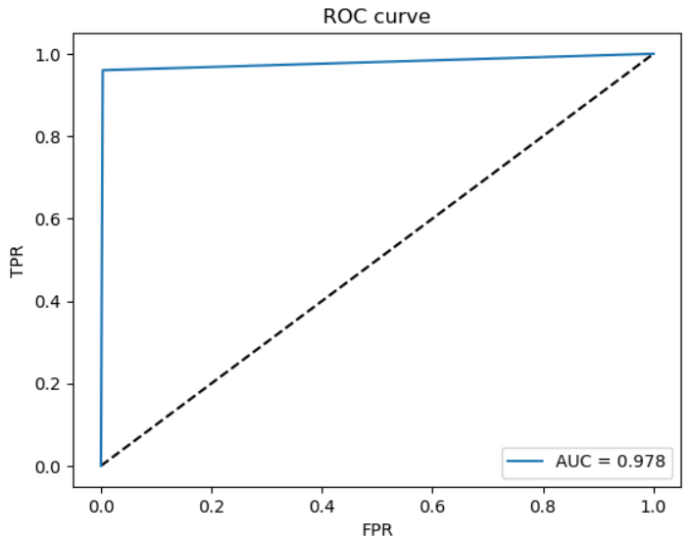

Figure 5.9 AdFraph method: ROC curve

The third model trained with AdGraph approach has shown promising results. It achieved high accuracy rates in both training and testing with a low false positive and false negative rate. The confusion matrix shows that the model correctly classified a high percentage of both clean URLs and trackers. Learning curves indicate that the model was learning stably,

with a small distance between the validation and training accuracy curves. Finally, the ROC curve demonstrated a high AUC plot, indicating that the model has high discriminative ability between clean URLs and trackers. Overall, these results suggest that the AdGraph approach is a viable option for identifying trackers in web browsing.

5.4 Results of first experiments

There were three models trained to classify URLs into either clean or tracker categories. The first model uses only URL data, the second model uses URL and HTTP features, and the third model uses the AdGraph approach.

The first model showed good performance with a high accuracy rate, but its precision rate was low compared to the other models. The model may have overfit the training data, leading to low precision on the test set. The second model used additional features and showed better precision than the first model, indicating that the additional features may have helped the model to distinguish between clean and tracker URLs more effectively. Moreover, the second model showed the best accuracy and recall rates. However, this model still showed some signs of overfitting, as indicated by the distance between validation and training accuracy curves.

The AdGraph model, despite being trained with more features, exhibited lower test accuracy rates than the second model. This could be attributed to the complexity and interdependence of the additional features incorporated in the model. The increased complexity might have led to overfitting, causing the model to perform well on the training data but struggle with generalization to unseen test data. Alternatively, it is possible that the additional features introduced noise or irrelevant information, hindering the model's ability to accurately classify URLs.

In general, all of the trained models performed well in classifying URLs, demonstrating their effectiveness in distinguishing between clean URLs and trackers. However, it is worth noting that overfitting was observed in the second and third models, which were trained using more extensive features. The best overall metrics were achieved with the second model, which incorporated both URL and HTTP data. This indicates that considering additional information beyond just the URL can lead to improved classification accuracy. On the other hand, the first model, which solely utilized URL data, provided a simpler solution but exhibited slightly lower performance compared to the models with more advanced features. As for the AdGraph model, while it showed potential with its additional features, further analysis and refinement are necessary to address any noise or irrelevant information that might have impacted its

performance. Overall, the experiments suggest that incorporating a combination of URL and HTTP features offers promising results, but careful feature analysis and selection are crucial to optimize the model's performance.

6 Model improvements

This section focuses on analysing and optimizing features of the model based on the findings from the previous section. The goal is to identify the best set of features that can improve the model's performance. Additionally, the section involves training the model with most important features and comparing its results with other models. To assess the overall effectiveness of each model, a rolling forecast test is conducted to evaluate their long-term reliability and performance. By examining and comparing these findings, the section aims to determine the superior model for detecting and identifying trackers and ads on the web.

6.1 Important feature detection

In this subsection, the focus is on identifying the most informative features. To accomplish this, the first step is to create a correlation matrix with all parameters. It is necessary to remove all correlations, which will allow for the use of the random forest method to calculate feature importance. By identifying the most informative features, it is possible to create a more accurate and efficient mode. The results of this process will provide insight into which features have the greatest impact on the target variable and will be used to refine the model in subsequent steps.

Figure 6.1 represents the correlation of features, providing insights into the relationships between variables. One interesting correlation observed is between the feature "Vertex number" and "Edge number." This correlation suggests that there may be a connection between the number of vertices and the number of edges in the graph representation of the web page. It is possible that web pages with a higher number of vertices also tend to have a higher number of edges, indicating a more complex network structure.

Another correlation observed is between the features "Cookie size" and "Send cookies." This correlation implies that there is a relationship between the size of the cookies used by the web page and whether the page sends cookies. One explanation is that web pages with larger cookie sizes may have more extensive data storage requirements, leading to the need for sending cookies to track user preferences or maintain session information.

Due to the observed correlation between "send cookies" and "graph edge number," it is decided to remove these features from the analysis. By removing these correlated variables, it helps to eliminate redundancy and potential multicollinearity, ensuring a more accurate and reliable model for further analysis.

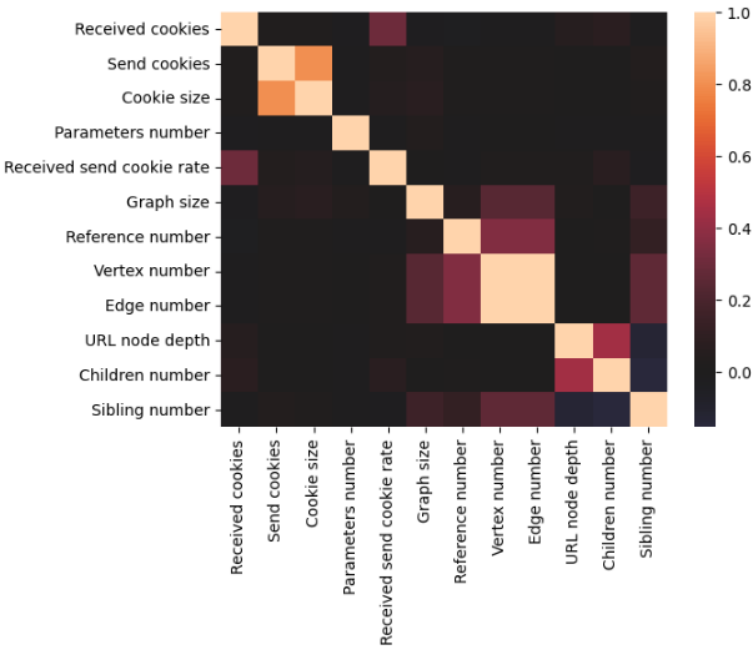

Figure 6.1 Feature correlation matrix

After removing the correlated variables, the random forest model is trained to extract the most important features. The importance of each feature is depicted in the bar chart shown in Figure 6.2. From the chart, it is evident that the most informative features are "URL length," "Source type," and "Third party." These features exhibit higher importance, indicating their strong influence on the model's predictions.

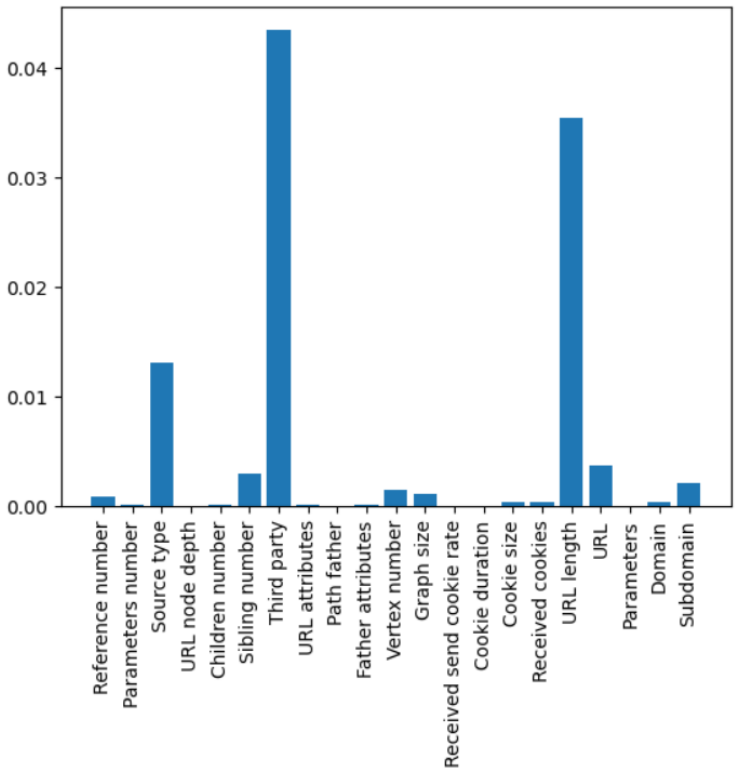

Figure 6.2 Random Forest features importance

On the other hand, several features have a score of zero in terms of importance and are deemed less relevant for model training. These features include URL nodes depth on the graph, path to node's father, father's attributes and URL domain. These features are considered less influential in predicting the target variable and, therefore, will not be utilized in the model training process.

6.2 Model trained with most informative features

The model was trained using the most informative features extracted from the data. Results are shown in Table 6.1. The training accuracy, validation accuracy, and test accuracy are all remarkably high, with each achieving 99 percent accuracy. The precision is also very high at 97 percent, indicating that the model was able to correctly classify almost all the malicious URLs. The recall is almost 99 percent, which suggests that the model was able to identify almost all the malicious URLs in the dataset.

Table 6.1 Model with informative features: results

Metric	Percentage (%)
Train accuracy	99.674
Validation accuracy	99.035
Test accuracy	98.796
Precision	97.455
Recall	98.909

The confusion matrix in Figure 6.3 further shows that the model correctly predicted 635 out of 639 clean files, which is a remarkably high accuracy rate. Additionally, the model correctly predicted 268 out of 275 trackers, which is also a high accuracy rate. These results show that the model is highly effective in distinguishing between trackers and clean URLs.

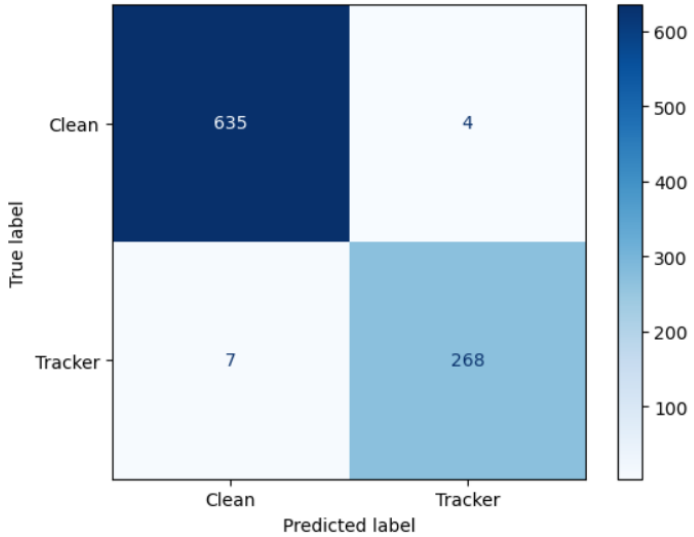

Figure 6.3 Model with informative features: confusion matrix

In Figure 6.4, there are accuracy learning curves of the model trained using extracted features. Both the training accuracy and validation accuracy curves show a slight increase, but there are fluctuations in both curves. The gap between the two curves is small, indicating that the model is not overfitting. However, the fluctuation in the validation accuracy curve indicates that the model may need further fine-tuning. Overall, learning curves suggests that the model is learning effectively.

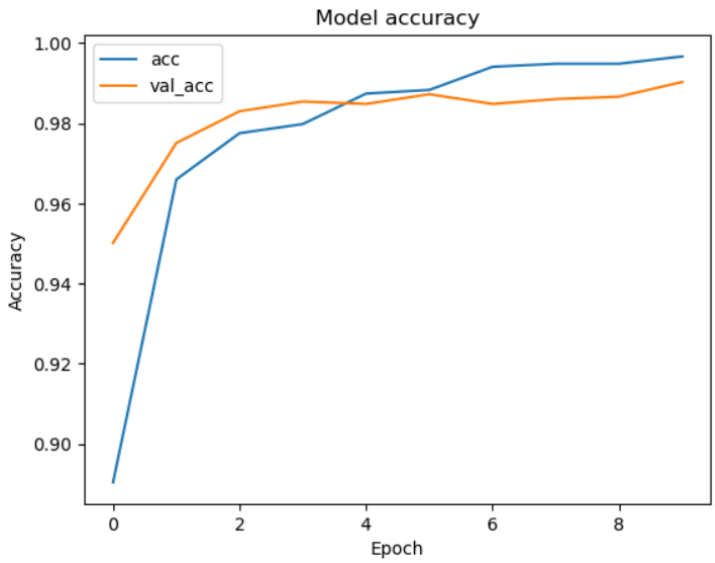

Figure 6.4 Model with informative features: accuracy learning curves

The ROC curve in Figure 6.5 shows the performance of the trained model in terms of true positive rate versus false positive rate. The AUC plot is 0.984, which indicates that the model has high discriminative power in distinguishing between positive and negative samples. The curve closely hugs the top left corner of the plot, which further indicates that the model is highly accurate in its predictions.

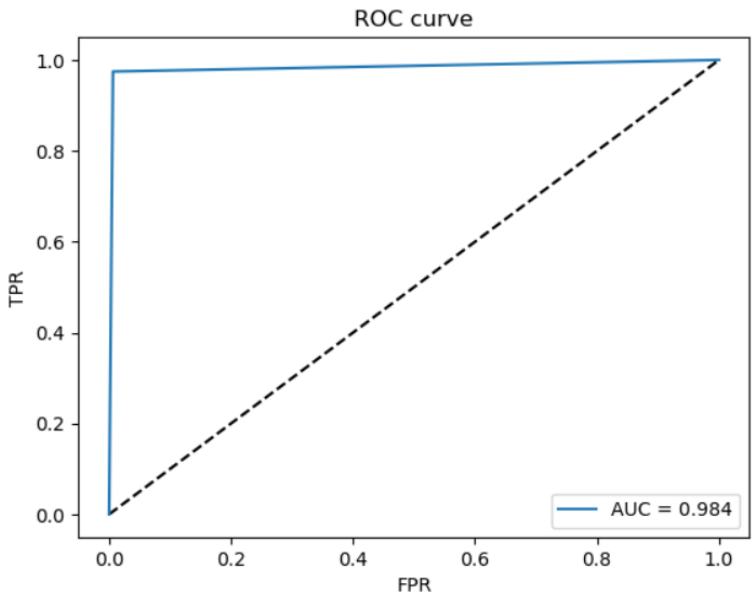

Figure 6.5 Model with informative features: ROC curve

Overall, this model trained with informative parameters performs better than all the other models. It has high accuracy on all three sets - train, validation, and test. The precision and recall are also high, indicating that the model is good at correctly identifying both clean URLs and trackers. The false positive and false negative rates are low, which means that the model is not misclassifying many URLs. Additionally, the learning curves show a steady increase in both train and validation accuracy with little fluctuation, indicating that the model is well-trained and not overfitting.

6.3 Rolling forecast

The last step of the experiment entails conducting a rolling forecast on all the models to determine the best performer in forecasting. This step assesses the predictive capabilities of each model over time and evaluates their performance in real-world scenarios. By applying the models to a rolling window of data, the ability to adapt and make accurate predictions with new data is observed. This analysis provides insights into the long-term performance and reliability of each model, enabling the identification of the most effective one for forecasting purposes.

In the rolling forecast results displayed in Table 6.2, a dataset of approximately 500 URLs was used for analysis. The experiment revealed that the model trained with the most important parameters showcased the best performance across all metrics. It achieved an accuracy of 98.6%, recall of 99%, precision of 88%, false negatives of 0.7%, and false positives of 11%. Notably, the accuracy-wise runner-up was the model trained with both URL and HTTP data,

which attained an accuracy of 98.3%. While its results were similar to the model with important features, it exhibited a significantly higher false positive rate of nearly 16%.

Table 6.2 Rolling forecast results

	URL model	**URL & HTTP model**	**AdGraph model**	**Most important features model**
Test accuracy	95.681%	98.339%	97.009%	98.671%
Precision	88.889%	88.889%	88.889%	88.889%
Recall	99.270%	99.291%	99.281%	99.293%
False Positive Rate	40.740%	15.789%	30.435%	11.111%
False Negative Rate	0.729%	0.729%	0.719%	0.707%

Comparatively, the AdGraph model demonstrated a test accuracy of 97% and a false positive rate of 30%, with other metrics aligning closely with those of the other models. On the other hand, the model trained solely using URL data exhibited poorer performance, with a test accuracy of 95.7% and a notably high false positive rate of almost 41%.

Upon evaluating the models, it can be concluded that the model incorporating the most important features is likely to deliver better long-term performance. Although the results from the other models are also promising, the model trained with the most important features consistently outperforms them in various metrics. This highlights the significance of selecting and incorporating informative parameters for enhanced predictive accuracy. Further refinement and optimization may improve the performance of the other models, but for now, the model with the most important features appears to be the most reliable choice for long-term forecasting.

7 Conclusions

This thesis has explored various techniques and tools for detecting tracking technologies and advertisements on the web. Extensive research was conducted by reviewing a multitude of papers in the field. After thorough analysis, three different approaches were selected for further investigation: URL-based analysis, combined URL and HTTP analysis, and the AdGraph method. These approaches were chosen based on their prominence in the literature and their potential to address the challenges of detecting trackers and advertisements effectively.

To collect data from the web, a combination of techniques was employed. Proxies and crawlers were used to access webpages and gather information about trackers and advertisements. Additionally, Adblock lists were utilized to label the collected data, providing valuable insights for model training and evaluation. Furthermore, in the case of the AdGraph approach, graphs were constructed from HTML pages to extract relevant features for detecting trackers and advertisements.

Model evaluation techniques were extensively reviewed to ensure a comprehensive assessment of the trained machine learning models. Traditional classification task evaluation metrics, such as the confusion matrix, learning curves, and ROC curves, were utilized to analyse and measure the models' performance. These metrics provided valuable insights into the models' accuracy, precision, recall, and overall effectiveness in detecting trackers and advertisements. Additionally, recognizing the importance of continuously monitoring the models' forecasting ability in the research field, a rolling forecast test was incorporated as an additional evaluation metric. This test allowed for the assessment of the models' long-term reliability and their ability to adapt to evolving tracking and advertising techniques, providing valuable insights for real-world applications.

Three distinct models were trained utilizing techniques identified and reviewed in the literature. The evaluation of these models revealed important insights into their performance. The model that utilized both URL and HTTP data demonstrated the highest level of effectiveness in detecting trackers and advertisements, outperforming the other models. I reached 98.6% test accuracy. This highlights the significance of incorporating additional data sources beyond just the URL for improved detection accuracy. Furthermore, the AdGraph model, which utilized a wide range of features, exhibited promising results of 98% test accuracy. However, it became apparent that careful analysis and filtering of these additional features is crucial to ensure they do not introduce noise or negatively impact the model's

performance. These findings emphasize the importance of selecting and utilizing relevant features to achieve optimal results in the detection of trackers and advertisements.

To determine the most important set of features for detecting trackers and advertisements, a thorough investigation was conducted. Features that exhibited correlation were excluded to ensure independence, followed by evaluating feature importance using the Random Forest method. The research identified highly informative features such as "URL length," "Source type," and "URL vectorized." These features significantly influenced the model's performance, achieving high accuracy and efficiency. Additionally, some features had a score of 0 due to data collection errors during web crawling. Addressing these challenges and ensuring accurate data collection remains an ongoing focus. Overall, model trained with the most important features reached the best results: training accuracy 98.8%.

In the last research task, the rolling forecast test was conducted to evaluate the long-term reliability of the trained models. The results revealed that the model trained with the most important features demonstrated the highest level of reliability, achieving a remarkable test accuracy of 98.7%. Additionally, all other metrics of this model were also superior compared to the other models.

Key findings:

- URL method: Initial experiment achieved 98% accuracy but rolling forecast test showed lower reliability with 40% FPR and 95% test accuracy, which makes it the worst performing model.
- URL and HTTP method: Initial experiment yielded the best results with almost 98.7% accuracy. Rolling forecast test showed 15.8% FPR and 98.3% test accuracy, making it the second-best model.
- AdGuard model: Slightly lower results compared to the URL and HTTP method with 98.4% accuracy in the initial experiment and 30% FPR and 98.3% test accuracy in the rolling forecast. The model incorporates more features but was affected by noise in the data.
- Best performance: The model trained with the most important features demonstrated the best performance in both the initial experiment (99% test accuracy) and rolling forecast (98.6% test accuracy and 11.1% FPR), establishing it as the most reliable model.
- Key features: The most important features identified include "URL length," "Source type," and "Third party."

8 Recommendations

Based on the findings of this thesis, the following recommendations can be made:

- Further exploration of advanced feature engineering techniques: While the models trained with the most important features yielded promising results, there is room for further investigation into more advanced feature engineering techniques. Exploring additional features or refining existing ones could potentially enhance the performance of the models.
- Continuous monitoring and updating of models: Given the evolving nature of tracking technologies and advertisements on the web, it is crucial to continuously monitor and update the trained models. Regularly incorporating new data and retraining the models can help ensure their effectiveness in detecting and blocking trackers and ads.
- Evaluation of models on diverse datasets: It is recommended to evaluate the trained models on diverse datasets to assess their generalizability and performance across different contexts. This can involve collecting data from various sources and domains to ensure the models' effectiveness in real-world scenarios.
- Investigation of emerging tracking techniques: As tracking technologies continue to evolve, it is important to stay updated on emerging tracking techniques and adapt the models accordingly. Investigating and understanding new tracking techniques can help improve the models' accuracy and ensure their relevance in detecting the latest tracking methods.

By considering these recommendations, future research, and development in the field of tracker and ad detection can contribute to the advancement of privacy protection and user experience in the online environment.

9 References

M. J. Kelly. (2019). *What is a web tracker?* Retrieved from: https://blog.mozilla.org/en/internet-culture/mozilla-explains/what-is-a-web-tracker/

Surveillance Self-Defense. (2020). *What Is Fingerprinting?* Retrieved from: https://ssd.eff.org/en/module/what-fingerprinting

Cyberclick. (2021). Retrieved from: https://www.cyberclick.net/advertising/types-of-online-advertising

U. Iqbal et al. (2019). *ADGRAPH: A Graph-Based Approach to Ad and Tracker Blocking.* Retrieved from: https://arxiv.org/abs/1805.09155

I. Castell-Uroz et al. (2020). *URL-based Web Tracking Detection Using Deep Learning.* Retrieved from: https://www.researchgate.net/publication/347539022_URL-based_Web_Tracking_Detection_Using_Deep_Learning

A. Guarino et al. (2020). *On Analyzing Third-party Tracking via Machine Learning.* Retrieved from: https://www.scitepress.org/Papers/2020/89720/pdf/index.html

Brave. (2022). *adblock-rust.* Retrieved from: https://github.com/brave/adblock-rust

R. (2022). *R igraph manual pages*. Retrieved from: https://igraph.org/r/doc/

MathWorks. (2023). *Rolling-Window Analysis of Time-Series Models.* Retrieved from: https://www.mathworks.com/help/econ/rolling-window-estimation-of-state-space-models.html

M. Cerliani. (2019). *Feature Importance with Neural Network.* Retrieved from: https://towardsdatascience.com/feature-importance-with-neural-network-346eb6205743

McQuade, S., & Zhao, Q. (2018). *HTTP Cookies - Mozilla.* Retrieved from: https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies

Ivanov, S. (2023). *What Should You Know About Web Trackers?* Retrieved from: https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies

Krishnamurthy, B., & Wills, C. E. (2018). *The fingerprinting arms race: A privacy perspective*. Proceedings of the Internet Measurement Conference, 85-98. https://doi.org/10.1145/3278532.3278550

Kumar, V., & Raju, P. S. (2021). *Online Advertising: A Review and Future Directions.* Journal of Interactive Marketing, 55, 103-118. doi: 10.1016/j.intmar.2021.01.004

Jia, Y., Huang, Z., & Li, X. (2019). *A Survey of Machine Learning for Ad-Blocker.* International Journal of Emerging Technologies in Learning (iJET), 14(06), 53-71.

Adblock. (n.d.). *Adblock.* Retrieved from: https://adblock.com/

Hyndman, R.J. and Athanasopoulos, G. (2021) *Forecasting: principles and practice*, 3rd edition, OTexts: Melbourne, Australia. Retrieved from: https://otexts.com/fpp3/

Appendix

The code used for conducting the experiments can be found at: https://github.com/MarimoM/MBD