

**Bluestreak — Privacy-Aware User
Segmentation for Online Advertisement
using Logistic Regression**

Bluestreak — Privacy-Aware User Segmentation for Online Advertisement using Logistic Regression

Heye Vöcking

A thesis submitted to the
Faculty of Electrical Engineering and Computer Science
of the
Technical University of Berlin
in partial fulfillment of the requirements for the degree
Master of Computer Science

Berlin, Germany
Tuesday 30th March, 2021



Main supervisor:

Prof. Dr. habil. Odej Kao, Technical University of Berlin

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Berlin, den 30.3.2021

Zusammenfassung

Das gesteigerte Bewusstsein für die Privatsphäre des Einzelnen in der digitalen Welt hat einerseits durch das immer häufiger festzustellende Blockieren von Drittanbieter-Cookies und andererseits durch das Inkrafttreten der neuen europäischen Datenschutzgrundverordnung (DSGVO) das Internet im Allgemeinen und die Online-Werbeindustrie im Besonderen inhärent verändert. Unter diesen Bedingungen wird die herkömmliche Vorgehensweise des Trackings über Nutzerprofile zunehmend schwieriger. Diese Thesis stellt einen alternativen Ansatz für die Vorhersage von Alters- und Geschlechtssegmenten eines Nutzers vor: Beim Anwenden der Bluestreak-Methode verbleiben die sensiblen Daten auf dem Gerät des Nutzers und nur die anonymen Segmentvorhersagen werden an den Server zurückgeschickt. Unsere Methode unterscheidet sich von gängigen Ansätzen dadurch, dass die Erfassung der benötigten Daten und die Vorhersage der gewünschten Segmente in den Browser des Nutzers verlagert werden. Dieser Ansatz ist unabhängig von Tracking-Cookies und wahrt so die Privatsphäre des Nutzers. Weiterhin zeigt die durchgeführte Evaluation auf der Basis von realen Daten der Online-Werbeindustrie, dass Bluestreak in der Lage ist, die Genauigkeit der Vorhersage im Vergleich zu einem ausschließlich auf dem User-Agent basierenden Ansatz zu verbessern, ohne die User-Experience im Browser maßgeblich zu beeinflussen.

Abstract

The growing awareness of privacy in the digital world has not only made the blocking of third-party cookies more common but also introduced major regulatory changes through the new European General Data Protection Regulation (GDPR). This regulation has inherently changed the Internet in general and the online advertising industry in particular: under these conditions, the traditional approach of tracking via user profiles is becoming increasingly difficult. In this thesis, an alternative approach for predicting age and gender segments of a user is proposed. With the presented Bluestreak method, the sensitive data remains on the user's device and only the anonymous segment predictions are sent back to the server. It differs from common approaches in that the collection of the required data and the prediction of the desired segments is shifted to the user's browser. This approach is independent of tracking cookies and thus preserves the user's privacy. We conducted an evaluation on a real-world data set and show that it is possible to improve the prediction accuracy for age and gender segments compared to a User-Agent-based approach while only posing a low overhead on user's devices.

Contents

1	Introduction	1
2	State of the Art	5
3	Background	11
3.1	Real-Time Bidding	11
3.2	Logistic Regression	14
3.3	Consumer Segments	16
3.4	Segment Prediction	17
3.5	Server-Side Segment Prediction	17
4	Requirements	21
4.1	Prediction Requirements	21
4.2	Privacy Challenges	22
4.2.1	General Data Protection Regulation	22
4.2.2	Definition of Personal Data	22
4.2.3	Definition of Fingerprinting	23
4.3	Technical Challenges	24
4.3.1	Browser and Device Support	24
4.3.2	Page Loading Speed Impact	25
4.3.3	Size Limitations	26
4.3.4	Processing Limitations	27
4.3.5	Memory Limitations	27
5	Contribution	29
5.1	Logistic Regression Model Development	29
5.1.1	Model Training	32
5.1.2	Hyperparameter Tuning	32
5.2	Library Implementation	35
5.2.1	Selected Features	37
5.2.2	Data Pre-Processing	42

5.2.3	Feature Engineering	42
5.3	Supported Browsers and Devices	43
5.4	Payload Size	44
6	Evaluation	45
6.1	Model Evaluation	45
6.1.1	Results	46
6.1.2	Comparison to the Cookie-Based Approach	50
6.2	Privacy Evaluation	51
6.3	Performance Overhead Case-Study	52
6.3.1	Performance Overhead Results	54
7	Conclusion	59
7.1	Future Work	60

List of Figures

3.1	An overview of how RTB connects an advertiser with a consumer. . .	12
3.2	Workflow chart of an RTB interaction with a consumer, a publisher, an SSP, and a DSP. Server 1, 2, and 3 are just symbolic for different realms that may consist of multiple servers.	13
3.3	Advertising Spending in 2014.	14
3.4	Advertising Spending in 2018.	14
3.5	Total money spent on online advertising in Germany, separated into the non-RTB and the RTB share in million EUR [44, 61].	14
4.1	Summed transfer size of the resources requested by several popular web page resources in a timeseries [25].	27
5.1	Age distribution.	31
5.2	Gender distribution.	31
5.3	Comparison of datapoint counts.	31
5.4	Receiver Operating Characteristic of the Bluestreak model.	33
5.5	Receiver Operating Characteristic of the User-Agent-based model. . .	34
5.6	No removal of features.	35
5.7	Removing whole features.	35
5.8	Removing just "feature=value" combinations.	35
5.9	Score of grid search over C for different strategies of feature removal by $L1$ regularization.	35
5.10	This figure shows (highlighted in teal color) how using the Bluestreak library changes the diagram introduced in Figure 3.2. Through feature collection and segment prediction, consumer segments are determined and sent along with the <i>impression</i> request, where previously the ID used for tracking was sent. Server 1, 2, and 3 are just symbolic for different realms that may consist of multiple servers.	37

5.11 Approach of splitting up the language string to make the information contained in it better processable by the logistic regression model training algorithm.	43
6.1 Age sensitivity.	47
6.2 Gender sensitivity.	47
6.3 Sensitivity results of the Bluestreak approach compared to the Cookie-based and User-Agent-based approaches.	47
6.4 Age specificity.	48
6.5 Gender specificity.	48
6.6 Specificity results of the Bluestreak approach compared to the Cookie-based and User-Agent-based approaches.	48
6.7 Regarding age accuracy, the Bluestreak approach achieves a 4 percentage points higher accuracy than the User-Agent-based approach. . . .	48
6.8 Regarding gender accuracy, the Bluestreak approach achieves a 2 percentage points higher accuracy than the User-Agent-based approach.	48
6.9 Age overall.	49
6.10 Gender overall.	49
6.11 Overall prediction results of the Bluestreak approach compared to the Cookie-based and User-Agent-based approaches.	49
6.12 Cookie-based approach age accuracy compared to the Bluestreak approach and User-Agent-based approach.	50
6.13 Cookie-based approach gender accuracy compared to the Bluestreak approach and User-Agent-based approach.	50
6.14 Users that are uniquely identifiable by different approaches. The lower the percentage the higher the chance of an individual user to be not uniquely identifiable.	52
6.15 Page loading speed impact measured on several news websites without the Bluestreak library, with the Bluestreak library directly being executed, and with a time-delayed execution of the Bluestreak library.	56
6.16 CPU usage, website 1.	57
6.17 CPU usage, website 2.	57
6.18 CPU usage, website 3.	57
6.19 RAM usage, website 1.	57
6.20 RAM usage, website 2.	57
6.21 RAM usage, website 3.	57
6.22 Resource consumption measured using Chrome DevTools on three different websites. CPU usage in milliseconds of scripting time, RAM usage calculated by occupied heap space.	57

List of Tables

3.1	Glossary with key terms used throughout this thesis.	12
4.1	Browsers that need to be supported to be able to handle 95% of the traffic. Each version is the lowest version number that needs to be supported in order to reach this goal.	25
5.1	Age distribution.	31
5.2	Gender distribution.	31
5.3	Total users.	31
5.4	An excerpt of features used in data collection with a short description. A longer description can be found in Section 5.2.1.	38
5.5	Size of the Bluestreak frontend library with and without shims (support for inconsistent browser behavior) and the size of the model with age and gender coefficients.	44
6.1	Per segment comparison of the sensitivity and specificity of the User-Agent-based approach and the Bluestreak approach.	49
6.2	Overall comparison of the sensitivity, specificity, and accuracy of the User-Agent-based approach and the Bluestreak approach.	49
6.3	Median time in milliseconds elapsed until <i>Time to First Paint</i>	55
6.4	Average RAM and CPU usage of the website frame and the Bluestreak frame compared on three different websites.	55
6.5	Average percentual increase of each measured metric.	58

1

Introduction

The Internet is arguably the most impactful change to society in the last 20 years. As with every black swan event, it brought unforeseen consequences, not all of which contributed to the good of its users. One of these is the ever-growing amount of data voluntarily, often unknowingly, provided by consumers¹ and collected by advertisers. Modern hardware and algorithms allow internet companies to track users through the Internet and build detailed user profiles. This enables advertisers to target their ads at consumers that may have an interest in their product and exclude those that may be distracted by this particular ad, which is in the interest of the consumer as well as the advertiser [11, 67]. This impacts the consumer's privacy because of the personal data that is shared with the advertisers. Due to this development, the interest in privacy retaining approaches has risen over the last years [1, 33]. This motivates the development of new methods that protect the consumer's privacy and retain the advertiser's ability to identify potentially interested consumers.

A complex step in predicting consumer segments is determining the consumer's interests: it either involves the collection of hard-facts (consumer-provided information)

¹In this thesis, the term *consumer* is used when talking about an internet user that consumes content without paying for it directly. The production and delivery of this content are financed by the revenue generated by selling online advertising spaces generated on the pages this user visits.

or deriving segments from other data points available about the consumer. Deriving segments from data is non-trivial because the consumer data is often fuzzy or incomplete. Advertisement companies typically can improve their knowledge by building a model about a big group of consumers, allowing them to predict the interests of a specific consumer. This is done by feeding the available data of that specific consumer to the model to generate a prediction for the user segment. Typically machine learning algorithms like logistic regression are used to build such a model.

The recent introduction of the *General Data Protection Regulation* (GDPR) strictly limits which and how data can be used for this. This made analyzing consumer behavior a more challenging task, due to the restrictions on processing personal data.

In the past, the advertising industry was allowed to neglect privacy concerns when doing consumer analysis. For the advertiser, the most important performance indicator is the revenue increase achieved through advertising. Providing relevant ads while maintaining consumer privacy would be a win-win situation, where advertisers reach consumers that want to buy their products, while consumers get content free-of-charge, provide a revenue stream for the content producer, and keep their privacy.

This thesis introduces a novel privacy-aware approach that allows advertisers to reach their target audience under the observation of consumer privacy. This is accomplished by identifying relevant features for age and gender prediction from a real-world data set and training a logistic regression model based on the values of these features. For that, a model-based predictor was developed in JavaScript, which is executed in the consumer's browser. This allows to only send back the prediction results, while the data that is used to perform the prediction stays on consumer's devices. Our privacy-enhancing approach was evaluated and compared against a User-Agent-based approach.

In summary:

- ◇ we identified relevant features for age and gender prediction
- ◇ we built a logistic regression model from the identified features
- ◇ we compared the model performance against the state of the art

- ◇ we performed a case study to evaluate the quality of experience (QoE) when applying the prediction directly on the device

This thesis is structured as followed: *real-time bidding* (RTB), consumer segments, and logistic regression are introduced in Chapter 3. Technical, legal and, privacy requirements are treated in Chapter 4. Chapter 5 describes the tools and techniques that were used to put this idea into action and Chapter 6 presents and analyses the results.

2

State of the Art

This chapter presents the current state of the art and related work that has motivated and informed this thesis. It first introduces a short overview of the origins of Real-Time Bidding, which became the most popular channel for online advertising in recent years. Then a short introduction to research regarding privacy concerns in the advertising realm is given and recent advances in the field of privacy preservation methods in the field are explored. Finally, related work in age and gender segment prediction is presented.

Real-Time Bidding

Real-Time Bidding (RTB) is considered the most significant progress in recent years in online display advertising [68, 69]. It was introduced in 2008 by a company named BlueKai, to enable advertisers to make use of demographic data available about customers [17]. While on TV or billboard ads each person sees the same ad, RTB allows selecting the advertisement for each person individually. BlueKai started with advertisers in the travel industry, such as Kayak and Expedia, who wanted “to advertise to individual consumers”, allowing to target those customers who were thought to be able

to afford the advertised trip [59]. This new user-centric approach, also called personalized advertising [14], enabled the application of new research opportunities such as reinforcement learning and logistic regression.

Reinforcement Learning

Cai et al. [10] proposed a model-based reinforcement learning model to optimize advertising performance. They built a Markov decision process framework for learning the optimal bidding policy and demonstrated that their solution is superior to other state-of-the-art methods. Furthermore, they proposed a neural network model to solve the scalability problem caused by the large real-world data set they used.

Wang et al. [70] used a deep reinforcement learning agent (named LADDER) on a real-world data set provided by the JD.com e-commerce platform. They demonstrated that their approach outperformed JD.com's human expert calibrated real-time bidding policy.

Yang et al. [74] propose a multi-objective reinforcement learning algorithm (named MoTiAC), which computes adaptive weights over time. They show that their method outperforms other state-of-the-art methods on a real-world commercial history data set.

Zhao et al. [77] propose a reinforcement learning framework in the sponsored search realm. Sponsored search differs from regular display advertising in that it also includes a user query with keywords. They deployed their solution on the e-commerce platform Alibaba. Their evaluation includes offline as well as online A/B testing and shows that their method is effective.

Logistic Regression

Szwabe et al. [63] investigated the impact of stochastic gradient descent parameter tuning and performed an experimental offline evaluation of leading logistic regression approaches in Click-Through Rate (CTR) estimation. They conclude that the L2-regularized logistic regression is the state-of-the-art RTB CTR estimation approach.

Privacy

Data analytics and its implications on privacy has become a main concern in online advertising in recent years [18, 56]. Several works that have focused on this topic will be introduced in this section.

Johnson et al. [27] have studied consumer privacy choice enabled by the *AdChoices* program. They found that only 0.23% of American ad impressions arise from opted-out users and report similar rates for users from the European Union and Canada. This shows that the option of opting out through the AdChoices approach is rarely used. Therefore, a development towards a *privacy by default* approach would benefit all consumers and not only those who actively chose to opt-out.

Chaudhry et al. [13] discussed practices of customer data collection and usage within targeted advertising and looked into the implications on ethics behind this method of advertising. They conclude that many of the current data-privacy collection methods are not ethical, but that ethical data collection methods may exist in the future. This means that advances in the field of data-privacy are a relevant area for advances in the industry.

Wieringa et al. [72] argue that data analytics and privacy are not necessarily contradictory. They conclude that data control and responsibility should move to the customer, as this provides superior privacy protection. Therefore, it seems that the approach taken in this thesis goes in the right direction regarding privacy protection.

Privacy Preservation

Qi et al. [51] used *Locality-Sensitive Hashing* (LSH), to perform recommendation decisions in a privacy-preserving manner. They report that using LSH achieves a good tradeoff between recommendation accuracy and efficiency while guaranteeing privacy-preservation.

Osia et al. [45] built a hybrid deep learning architecture, where they split a neural network in a way that one part (the first layers of the neural network) are deployed to the “edge”, on *Internet-of-Things* (IoT) devices, doing the first part of processing and only then send the output of these first layers of the neural network from the IoT

devices back to the cloud to do the remaining processing. While they are splitting up the neural network and send partially processed data back to the cloud, in this thesis, the whole model is sent to the “edge” and only the final output (the predicted consumer segments) are sent back.

Toubiana et al. [65] propose an architecture that, similar to our approach, enables targeting without compromising user privacy by performing behavioral profiling and targeting in the user’s browser. They use a cosine-similarity matrix between tags on the website and words in Google AdsPreferences categories for profiling and targeting.

Backes et al. [5] built an online behavioral advertising system (named ObliviAd) that preserves user privacy through encryption, such that the broker side does not get any personal information of the user profiles. Their method still allows selecting advertisements that fit the user profile and therefore does not impose a penalty on ad revenue.

Reznichenko et al. [57] deployed a prototype of a fully functional privacy-preserving behavioral targeting system. Their userbase had a size of about 13K opted-in users and they reported their Click-Through Rates to be comparable with those for Google display ads.

Federated Learning

While the approach presented in this thesis is based on a data set that is used for training the model once at the beginning, a technique called federated learning was created in recent years [8], which allows training the model continuously. While federated learning is not privacy-protecting by design, methods that allow keeping the model up-to-date while preserving the consumer’s privacy have been discussed in other works:

Ammad-ud-din et al. [3] built a personalized recommendation system using a master machine learning model which is distributed to user clients, where locally stored data is used for inference and model updates which are then sent to update the master model. The updated model is then redistributed. This way, user privacy is preserved because user data never leaves the client.

Bonawitz et al. [6] discussed secure aggregation protocols, which allow multiple parties to collaboratively compute several private values to a sum, without revealing

these private values using deep neural networks in a Federated Learning model.

McMahan et al. [36] learned differentially private recurrent language models, that add user-level privacy protection to the federated averaging algorithm, that makes “large step” updates from user-level data.

Age and Gender Prediction

Al-Zuabi et al. [2] proposed a method to predict a users’ age and gender, based on behavior information sourced from mobile phone data. They used a data set of 18,000 users and achieved an 85.6% accuracy in terms of user gender prediction and 65.5% of user age prediction. The predictions are intended to be used for marketing campaigns. Similar to this thesis, the age and gender prediction was intended to be used for marketing purposes and the size of the data set is also comparable. However, instead of using data collected from the device, the data was based on mobile phone data. With the approach of Al-Zuabi et al., the achieved accuracy was higher than the accuracy achieved with the approach discussed in this thesis.

In contrast to this thesis, which uses logistic regression, Nguyen et al. [43] used linear regression to predict an author’s age from their blog posts, telephone conversations, and online forum posts. They achieved absolute discrepancies between 4.1 and 6.8 years. As this thesis is discussing prediction for age segments that span 10 years, it is difficult to compare their results to ours directly, however, their results show that a regression algorithm is suitable for an age prediction task.

3

Background

This chapter introduces the applied methods, and the online advertising business model focusing on Real-Time Bidding. It explains consumer segments, introduces the logistic regression algorithm, and combines them to perform segment prediction. Then it describes the state-of-the-art approaches to predict consumer segments. Table 3.1 is a glossary of terms used in this thesis.

3.1 Real-Time Bidding

In *Real-Time Bidding* (RTB), the content of online advertising banners, are auctioned to bidders on an advertising exchange. It is called an impression when an advertising - short ad - is shown to a user [39]. RTB is the interface between the Demand-Side, the advertisers advertising a product, and the Sell-Side, the publishers generating revenue through displaying these ads to consumers, see 3.1. The *Supply-Side Platform* (SSP) is an RTB server which connects to several *Demand-Side Platform* (DSP) as RTB clients [52, 73]. A DSP distributes creatives, provided by advertisers paying to reach consumers to boost their brand or to induce conversions by incentivizing the DSP [15, 32]. The advertiser orders the DSP to run an advertising campaign and provides a budget to

Term	Description
Publisher	Websites, apps, or content creators in general, publishing media to consumers
Consumer	A user, consuming content, viewing ads and buying advertised products.
Advertiser	Company selling products and advertising.
Creative	An ad that is displayed on a publishers website, e.g. a banner-ad
Impression	Event of the consumer's device loading and displaying the creative
Click	Event of the consumer clicking on the displayed creative
Conversion	Event of the consumer taking the advertised action (e.g. buy, signup, etc.)
Client device	Device the consumer uses to browse the web
Pixel	Invisible content in publisher's and advertiser's site to track consumers
SSP	Supply Side Platform
DSP	Demand Side Platform
Segment	A grouping based on age (e.g. "20-29") or gender ("F" or "M")
Hard-fact	Information originating from the consumer, e.g. during a sign-up process

Table 3.1: Glossary with key terms used throughout this thesis.

reach the desired count of impressions, clicks, or conversions.

The DSP buys impressions on the SSP, where a first-price auction, the highest bidder pays the price of its bid or second-price auction, the highest bidder pays the price of the second-highest bid, is performed [75].

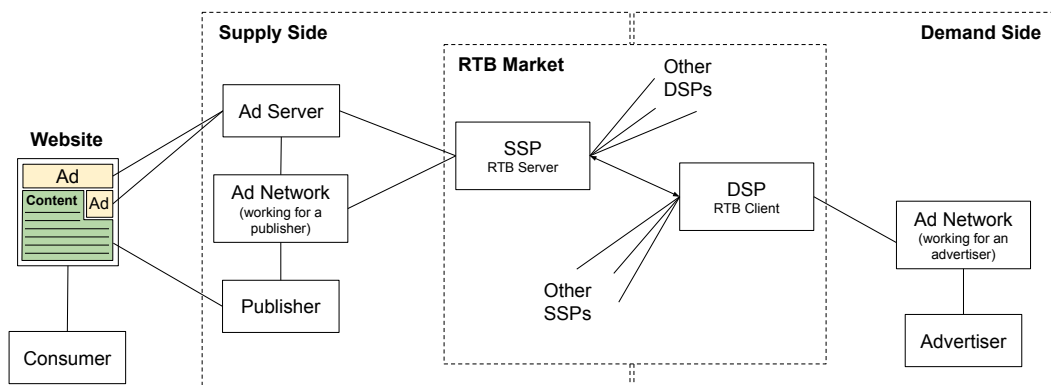


Figure 3.1: An overview of how RTB connects an advertiser with a consumer.

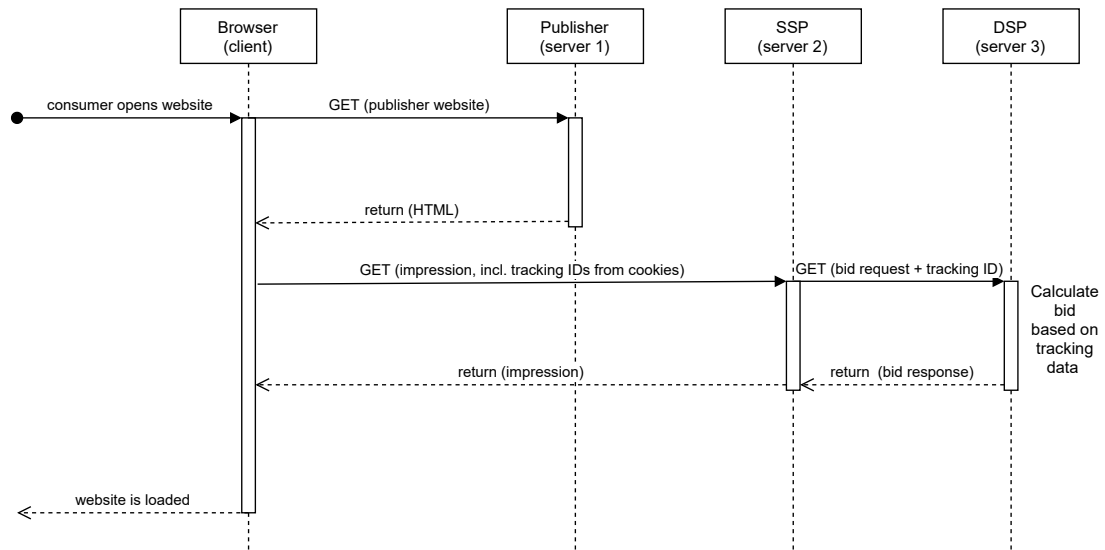


Figure 3.2: Workflow chart of an RTB interaction with a consumer, a publisher, an SSP, and a DSP. Server 1, 2, and 3 are just symbolic for different realms that may consist of multiple servers.

The RTB market has been growing steadily in recent years from 0.35 Billion EUR in 2014 to 1.58 Billion EUR in 2018 in Germany alone¹. RTB increased its ad spend market share from 11.57% in 2014 to 61.39% in 2018 (see Figure 3.5²) and is, therefore, the main channel for distributing online advertising [44, 61].

As the term *real-time* suggests, the auction is happening while the consumer is waiting for the website to load. As seen in Figure 3.2 the typical workflow in an RTB system starts with the consumer opening a website. This site can, for example, be the website of a publisher, such as an online news site. The HTML returned by the publisher's server contains a small piece of JavaScript that triggers a GET request to an SSP, the ad exchange. The ad exchange then sends a bid request, e.g. an HTTP-Request, to all connected bidders to request their bid for this particular impression. It then awaits a response containing the bid price with a timeout of usually 100 milliseconds. The highest bidder will then win the auction. The response to the client then contains the impression of the winning bidder. The user's browser then renders this impression. From the user's perspective, this all happens while the page is loading

¹The amount for 2018 is a projection, as presented in [61]

²Data for total money spent on RTB advertising in Germany is originally in USD, but was converted to EUR with a conversion rate of 1 EUR = 1.2455 USD for 2014 and 1 EUR = 1.1482 USD for 2018

Advertising Spending in Million EUR

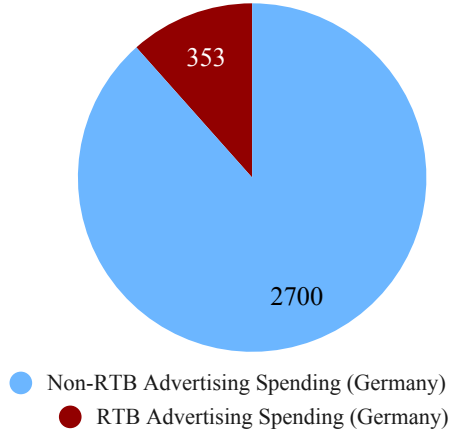


Figure 3.3: Advertising Spending in 2014.

Advertising Spending in Million EUR

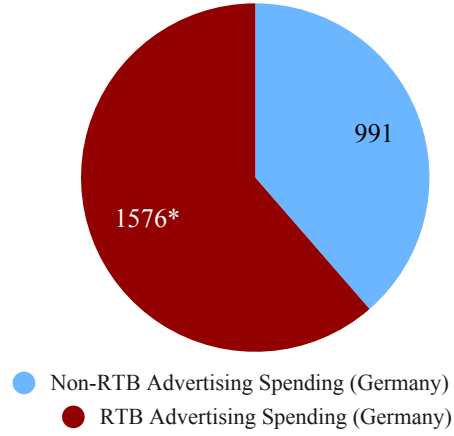


Figure 3.4: Advertising Spending in 2018.

Figure 3.5: Total money spent on online advertising in Germany, separated into the non-RTB and the RTB share in million EUR [44, 61].

[40].

When the RTB client receives a bid request, the DSP identifies the most valuable campaign for this consumer, because the chance of winning increases with the price. To determine the price it is willing to bid, the DSP needs to predict the probability of a consumer performing the desired action (impression, click, or conversion) [12, 23, 58, 76]. These can be expressed as consumer segments, described in Section 3.3. If segments based on hard-facts are unavailable, they can be predicted, described in Section 3.4.

3.2 Logistic Regression

Logistic regression is considered state-of-the-art in the field of real-time bidding [63]. A logistic regression model can be used to estimate the probability that a user i is part of a segment or not, which is indicated as $y_i = 1$ and $y_i = 0$, respectively [24]. Each user has a feature vector $x_i \in \mathbb{R}^p$, which contains known information about this user, such as the device model and browser. The conditional probability is then given by

$$P(y_i = 1|x_i) = \frac{1}{1 + \exp\{-(\beta_0 + \sum_{j=1}^p \beta_j x_{i,j})\}} \quad (3.1)$$

with β_0, \dots, β_p as parameters that can be estimated based on observed data [28].

We assume a linear relationship between the predictor variables and the log-odds (denoted ℓ) of the event that $y_i = 1$:

$$\ell = \beta_0 + \sum_{j=1}^p \beta_j x_{i,j}. \quad (3.2)$$

When all β_j are fixed, it is possible to compute either the probability that $y_i = 1$ or the log-odds that $y_i = 1$ for a given observation.

As an example, a vector with two features is assumed: $(x_{i,1}, x_{i,2})$. The trained logistic model is then used to estimate the probability that $y_i = 1$ given the feature vector. The event $y_i = 1$ can, for example, represent a consumer's gender to be female. The feature $x_{i,1}$ represents the amount of RAM the consumer's device is equipped with, normalized in the range $[0, 1]$ and $x_{i,2}$ is a binary value, which is 1 if the consumer is using an Android phone and 0 otherwise. The following values for β_0, β_1 , and β_2 are assumed:

- ◇ $\beta_0 = 0.5$, this is the y-intercept, the log-odds of the event that $y_i = 1$ when $x_{i,1} = x_{i,2} = 0$. Therefore, the odds of the event $y_i = 1$ are 1-to-2, or $\frac{1}{(1+1^{0.5})} = \frac{1}{3}$.
- ◇ $\beta_1 = -0.25$, this means a higher amount of RAM decreases the log-odds.
- ◇ $\beta_2 = 2$, this means having an Android phone increases the log-odds by 2.

The number of parameters p is often very large in practice which would lead to overfitting [62] and thus, $L1$ regularization is applied, which shrinks the parameter space. This has the additional advantage, in comparison to the often used $L2$ norm, that the resulting number of parameters is lower [42], which yields a smaller model. This is very favorable for this thesis as the model in our approach is sent to the user's browser.

L1 Regularization

L1 Regularization, also called *Lasso*, an acronym for “least absolute shrinkage and selection operator”, is a method used in regression analysis to perform variable selection and regularization. It improves prediction accuracy and interpretability of the logistic model by changing the model fitting process selecting only a subset of the provided covariates for use in the final model [42].

It allows adjusting the regularization strength with a parameter λ or C [55]. The best value for this parameter is not trivially chosen, so a form of grid-search with cross-validation has to be used, to probe which value delivers the best results. How this was done in this thesis is described in Section 5.1.2.

3.3 Consumer Segments

The consumer segments treated in this thesis are a form of market segmentation, called demographic segmentation. Market segmentation separates a heterogeneous market into homogenous segments [9]. Demographic segmentation is based on demographic profiling, which considers parameters such as age, gender, income, educational attainment, etc., to categorize the consumers into segments. Among these segments, age and gender are the most common targeting option [71]. This assumes that consumers in the same segment have similar purchasing patterns translating into similar product or brand preferences.

By combining several segments, it is possible to target a narrow real group through few carefully chosen segments [52].

As advertisers design campaigns for a certain demographic profile, the ability to target such segments directly benefits the advertiser. Conversely, fewer consumers in irrelevant segments will see this ad. As a result, the advertiser spends less money while untargeted consumers are not distracted by an irrelevant ad [16].

3.4 Segment Prediction

If no hard-fact data about a consumer is available one can resort to predicting demographic segments for a consumer individually. This is done by building a model from data of other known consumers and using it to predict segments for an unknown consumer [71].

Machine learning algorithms such as logistic regression can be used to predict the probability of a binary event, like a consumer belonging to a particular segment or not. Deciding on a consumer's gender is straightforward as we have a binary classification case³. It gets however more difficult when it comes to predicting which age segment a consumer belongs to.

Because we are doing the prediction for each segment independently, the classifier may return *True* for more than one of the five age segments. Instead of using the binary outcome, we can however use the confidence value and rank the results for each age segment prediction. This allows us to pick the segment with the highest confidence.

3.5 Server-Side Segment Prediction

Historically, in contrast to the approach proposed in this thesis, segment prediction was focused on doing the prediction process on the server instead of on the consumer's device. In the following, two widely applied approaches for such predictions are discussed in more detail.

Server-Side Segment Prediction with Cookies

In the past, tracking cookies were placed on consumer's devices to record their journey through the internet. This thesis calls this the "*Cookie-based approach*". This uses third-party cookies on partner sites. Cookies are small pieces of data stored in the user's browser, separated by domain, to hold stateful data between browsing sessions.

³In this work we are only considering the female and male gender, as the hard fact data we are using for training does not contain any non-binary gender datapoints

This cookie is sent to the server when a website is opened. This allows to identify a previously seen user and enables tracking by recording a user's history [53]. A first-party cookie, also called "same-site" cookie, is only sent if the domain of the cookie matches the domain in the browser's address bar. A third-party cookie, also called "cross-site" cookie, can be sent even if the domain does not match the one displayed in the address bar. Therefore, advertisers have been using third-party cookies on partner websites to perform cross-site tracking. This led to an increasing share of consumers disabling third-party cookies in recent years, and several browsers such as Brave [7], Firefox [38], and Apple [26] have started to block third-party cookies by default as well. This caused the count of tracked consumer journeys to decline [30]. In combination with the GDPR becoming effective, a different approach of predicting the consumer's segments based solely on the current request was developed. This is described in Section 3.5.

Server-Side Segment Prediction without Cookies

To migrate away from the Cookie-based approach, age and gender prediction based on the "User-Agent" HTTP-Header field was considered. This thesis calls this the "*User-Agent-based approach*". The User-Agent field-value is a string containing a list of product identifiers with optional comments identifying the browser and its subproducts by their name and version [21]. An example of a User-Agent string is the following:

```
Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:86.0) Gecko/20100101 Firefox/86.0
```

It is sent along with an HTTP request and is commonly used for content negotiation where the server selects which content to respond with, according to the products and version numbers listed in the User-Agent string. This encoded information can be parsed and interpreted to determine the operating system, the device model, the browser version, etc. With a set of hard-facts of age and gender information for known customers, the User-Agent strings sent by their browsers can be collected on each website visit and be collected into a data set. This data set can then be used to train a model, which is able to predict the segment an unknown consumer most probably belongs to [4]. The information provided in the User-Agent is limited but still carries some

information [19].

4

Requirements

In this chapter the requirements that define the frame of this work are discussed. In particular this relates to the prediction performance of the logistic regression model described in Section 4.1, privacy considerations, presented in Section 4.2, and technical challenges, discussed in Section 4.3.

4.1 Prediction Requirements

For a new approach to be considered a valuable contribution it has to deliver additional value over other simpler approaches. Such a solution is for instance the User-Agent-based approach introduced in Section 3.5. As it deals with data available on the server, it does not need to be executed on the client's device, which makes its application simpler.

The Cookie-based approach, introduced in Section 3.5 servers as another benchmark. Its application is more complex because it does not preserve the consumer's privacy and therefore requires consent under the General Data Protection Regulation as discussed in Section 4.2.

The developed approach should have comparable accuracy to the widely applied

state-of-the-art approaches such as the aforementioned User-Agent-based and Cookie-based approaches.

Since the User-Agent-based approach is using less data for the prediction, it is expected to have lower accuracy than the developed approach, while the Cookie-based approach has more data available and is therefore expected to have higher accuracy than the developed approach.

4.2 Privacy Challenges

One of the main goals of this thesis is to provide a solution that protects the consumer's privacy. This section gives an overview of the different challenges and requirements that have to be faced when dealing with consumer privacy on the internet.

4.2.1 General Data Protection Regulation

The General Data Protection Regulation (GDPR) was implemented on May 25 2018. It regulates data protection and privacy for data processing and transfer of personal data within, as well as outside the European Union (EU) and the European Economic Area (EEA). Its main goal is to give each individual control over their personal data. [49]

Therefore, the common method of targeting consumers by tracking them now requires the consumer's consent and would for example need a method, that keeps all personal data on the consumer's device. If no consumer consent is given, the personal data of the consumer can not be sent away from the device.

4.2.2 Definition of Personal Data

The EU law has its own definition of "personal data". An excerpt on this definition from Article 4(1) of the GDPR [49]: "'personal data' means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one, who can be identified, directly or indirectly, in particular by reference to an identifier [...]".

The following is a list of examples of “personal data” provided by the European Commission, which includes:

- ◇ Name and surname
- ◇ Home address
- ◇ Email address
- ◇ Identification card number
- ◇ Location data
- ◇ IP addresses
- ◇ A Cookie ID
- ◇ The advertising identifier on a mobile phone
- ◇ Data held by a doctor or hospital including symbols that uniquely identify a person

This results in the situation that the conventional tracking approaches such as the one used by the Cookie-based approach explained in Section 3.5 can not be legally conducted without the consumer’s consent, because “A Cookie ID” and “The advertising identifier on a mobile phone” would be needed but are both can be assumed to be personal data.

However, a prediction of an age and gender segment does not allow to uniquely identify a user at a later point in time and is therefore not assumed to be personal data.

4.2.3 Definition of Fingerprinting

A way to avoid using cookies but still being able to track consumers on their journey through the internet is possible by creating a fingerprint. This is done by deterministically generating a string based on several settings and values found on the consumer’s device and sending this string back to the server. Such a fingerprinting function has the feature that it is robust in a way that it always produces the same fingerprint on the

same browser on the same device. While at the same time it is very sensitive such that it creates a different fingerprint when on a different browser and device [31].

It essentially enables tracking even if the consumer clears the cookies or has them completely disabled. This allows tracking without cookie support from a technical perspective, however, it does not allow it from a legal perspective, as the fingerprint can be used to identify the consumer and is therefore assumed to be “personal data”.

4.3 Technical Challenges

This section discusses the technical difficulties. In traditional approaches, the data is collected locally on the client and sent to the server. To predict the segments on the client, the model has to be transferred to the consumer device. This impacts the page loading speed due to increased bandwidth by the model download and resource usage caused by the prediction process, discussed in Sections 4.3.2, 4.3.3, and 4.3.4. As the browser and device landscape is very broad, supporting all browser and device combinations is another challenge. This will be discussed first.

4.3.1 Browser and Device Support

The device landscape consists of desktop computers with a count of operating systems and mobile phones from many manufacturers running several operating systems with several browsers having a wide range of different versions. Even though browser manufacturers are adhering to standards, browser behavior is inconsistent, this is also called Cross Browser Inconsistency (XBI) [46]. Even within versions of the same browser behavior inconsistencies can be found. As it would be infeasible to support 100% of all combinations in use we set the goal to support 95% of them. This browser usage data was gathered from real-world RTB traffic. By analyzing the browser usage distribution, we identified the minimum required version for each of the popular browsers to reach this goal. Based on these insights, tests with the required browser and version combinations were conducted using the Lambdatest platform ¹, to ensure that

¹<https://www.lambdatest.com/>

Browser Name	Min. Version
Amazon Silk	69.0.3497
Android	4.3.1
Chrome Mobile iOS	71.0.3557
Chrome Mobile WebView	69.0.3497
Chrome Mobile	69.0.3497
Chrome	15.0.874
Chromium	15.0.874
Edge	15.15063.0
Firefox iOS	14.0.0
Firefox Mobile	68.0.0
Firefox	4.0.1
IE	10.0.0
Mobile Safari UI/WKWebView	8.0.0
Mobile Safari	8.0.0
Opera	20.0.1387
Safari	5.1.7
Samsung Internet	10.1.00.27 (like Chromium 71)
Yandex Browser	15.12.1

Table 4.1: Browsers that need to be supported to be able to handle 95% of the traffic. Each version is the lowest version number that needs to be supported in order to reach this goal.

the Bluestreak library is working correctly on the required version ranges. The list of browsers with their lowest required version to be supported can be seen in Table 4.1.

4.3.2 Page Loading Speed Impact

The time a page takes to deliver the content the consumer is requesting has a direct impact on the user experience [48]. Therefore, client-side segment prediction mustn't add an overhead on processing power that has a noticeable negative impact on the user experience. Measuring this is complicated by the lack of a single metric to determine page loading time. In performance measuring tools like Lighthouse, four metrics are usually considered [22]. These four metrics are listed below in chronological order:

Time to First Byte (TTFB) The time between sending the HTTP-Request header and the first byte received from the webserver.

First Contentful Paint (FCP) The time at which a display element is displayed in the browser for the first time.

First Meaningful Paint (FMP) The time at which the user perceives the website to be loaded.

Time to Interactive (TTI) The time at which the website has been rendered and is ready for user input.

For the consumer perception of page loading time it does not matter whether the time is spent downloading the content, executing the scripts, or rendering the content. Hence, all steps have to be considered in a page loading speed measurement. During a page load, the consumer first perceives progress after the First Meaningful Paint (FMP).

Since the display of advertisement banners is not required for a web page to be functional, it is not relevant for the script to be fully executed before the FMP point is reached. It is only relevant how much the execution of the script slows down the rendering of the page when executed alongside the normal page load.

For this reason, resources may be loaded the background. This is done by triggering them through the “onLoad” event, which is fired after an object defined in the website’s HTML-code is loaded. Such an object can for example be the the HTML *body* elemnt.

4.3.3 Size Limitations

The median page weight, the byte count of all resources loaded by a website, has been rising in the past 10 years as shown in Figure 4.1. On the examined pages the median byte count increased from about 0.52 MiB (Desktop) and 0.14 MiB (Mobile) in 2011 to about 2.06 MiB (Desktop) 1.89 MiB (Mobile) in the year 2020. This shows that the difference in page-weight between desktop and mobile pages has mostly vanished. That means the limitations regarding bandwidth use are not as strict anymore and it is not necessary to build a separate version for a mobile device, but rather build a model that performs well on mobile and desktop devices.

However, especially for mobile connections the number of requests, can have a significant impact on the page loading times, as each new request costs time, even if the connection provides high bandwidth.

UX research shows [64] that slow page loads cause the consumer to bounce (abort the request), which is a bad consumer experience. This leads to a lower chance of the consumer using the page in the future, lost revenue for the publisher, and loss of a potential customer for the advertiser.

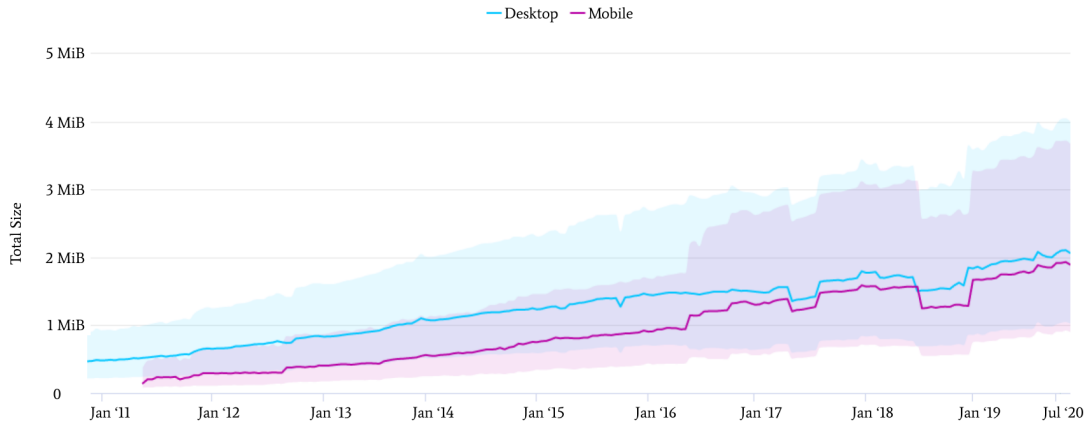


Figure 4.1: Summed transfer size of the resources requested by several popular web page resources in a timeseries [25].

4.3.4 Processing Limitations

A high CPU usage causes the page to load slower and gives a less fluent experience [20]. On a mobile device, increased CPU load contributes to faster battery drain.

Potential factors are parsing of the code and the model, as well as collecting features that cause many CPU cycles. Therefore, the CPU usage was determined by measuring the time spent executing the JavaScript code measured in milliseconds.

4.3.5 Memory Limitations

Similar to the CPU requirements described in Section 6.3.1 limitations apply to RAM usage as well. The code is executed in the browser's JavaScript engine, which provides a heap that each script uses to allocate objects [50]. Low-end and last generation mobile devices have little available memory which increases the risk of impacting the user experience by exhausting this resource [54].

Requirements Summary

Based on the aforementioned limitations we developed the following requirements for our contribution: the overhead induced by executing the Bluestreak library should not be unreasonable compared to the website's resource usage. The perceived loading time of the page should not be exaggerated, which means in detail that, the library size should be kept small in comparison to the page size, the time executing the library should not add a considerable overhead, and the heap space footprint should be within reasonable limits compared to the size of the website. We aim at an overhead less than the resource consumption by the website itself.

5

Contribution

The name “Bluestreak” was chosen because of the bluestreak cleaner wrasse. A small blue fish with a black stripe on the side that lives in a mutualistic relationship with larger fishes [60]. If the consumer is seen as such a big fish, our library can be seen as the bluestreak cleaner wrasse, built to live in a mutualistic relationship with the consumer.

This chapter discusses the implementation work done for this thesis. Based on the requirements mentioned in Chapter 4, we implemented the Bluestreak library and web server written in TypeScript discussed in Section 5.2, and the code for training the logistic regression model written in Python discussed in Section 5.1.

5.1 Logistic Regression Model Development

The training of the model was done in Python using the *Pandas* [34, 35] and the *scikit-learn* [47] library. For the training a real-world data set was used, which is described in the following:

Data Quality Assessment

In general, a larger data set is better for training than a smaller data set. However, quantity does not equal quality. Therefore, we determine the data quality not just based on the number of unique data points, but also evaluate their distributions among the segments.

Another important aspect is the origin of the data. While training models with synthetic data is possible, the data set we are using was provided by an RTB advertising company. It contains real-world data that was collected over a period of 3 days from one website. It contains 13,489 data points, each data point is made up of the features and the hard-fact data for either age, gender, or both of an individual. The data was provided by a German real-time bidding demand-side platform operating mostly in Europe with a focus on the German-speaking market.

Age and Gender Distribution

According to the 13,489 hard-fact data points, 6,414 are in the “Female” segment and 6,389 are in the “Male” segment, as can be seen in Figure 5.3. For the remaining 686 data points, no gender hard-fact information was available. Therefore, the distribution of users in the gender segments is quite even as can be seen in Figure 5.2. However, regarding age segments, the distribution is not as balanced. With 3,966 datapoints in the “20-29”, 4,224 in the “30-39”, 2,040 in the “40-49”, 1,493 in the “50-59”, and 1027 in the “60+” segments, the younger aged segments are much more prominent than the older aged segments, which can also be seen in Figure 5.1. Of all data points, 739 had no hard-fact data for the age segment.

In summary, the data set used for model training provides a diversified sample from all segments. However, as the age segments “20-29” and “30-39” are overrepresented compared to the age segments “40-49”, “50-59”, and “60+”, it is necessary to either remove data points from the overrepresented segments or perform oversampling on the underrepresented segments to avoid a bias towards the overrepresent segments during model training. To avoid removing valuable information from the data set, the latter approach of oversampling is taken for training the model.

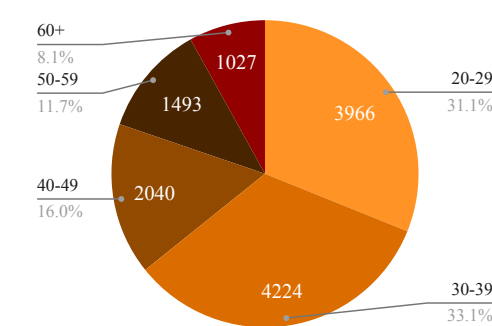


Figure 5.1: Age distribution.

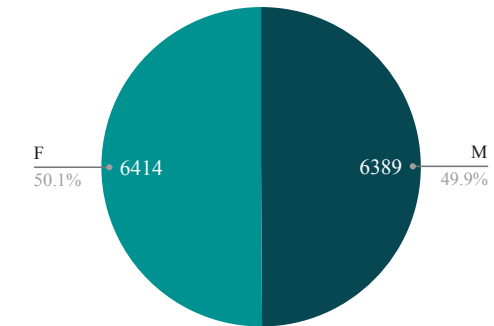


Figure 5.2: Gender distribution.

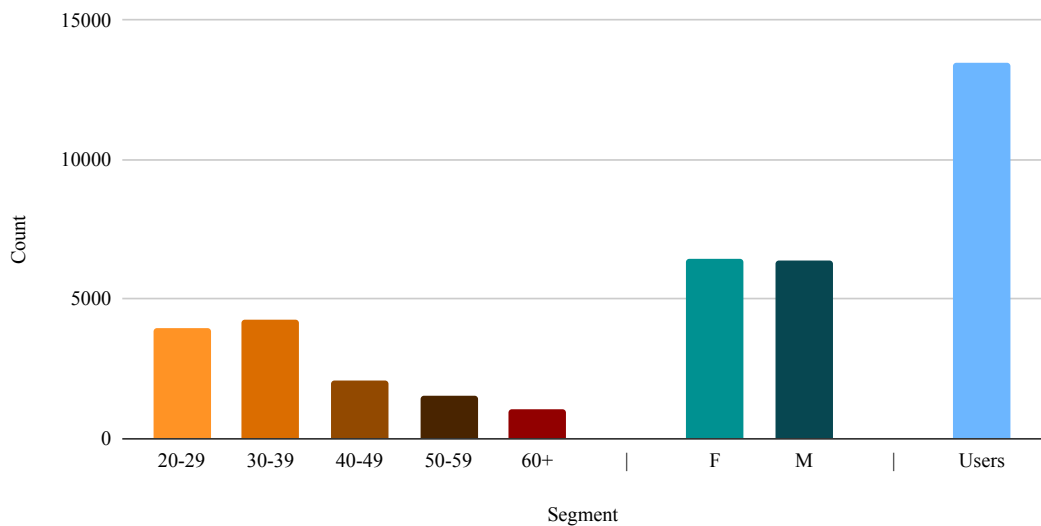


Figure 5.3: Comparison of datapoint counts.

Age Segment	Count
20-29	3,966
30-39	4,224
40-49	2,040
50-59	1,493
60+	1,027

Table 5.1: Age distribution.

Gender Segment	Count
F	6,414
M	6,389

Table 5.2: Gender distribution.

Total	Count
Users	13,489

Table 5.3: Total users.

Features

Each data point has the same 230 features. But in case a browser does not support a certain API or does not offer a certain feature the value is “null”. This is the case for

about 7% of the feature values in the data set. However, even if the value is “null” this very fact may carry important information in combination with other features. One must also consider that not every single feature is relevant and depending on the browser some of these features carry redundant information. Therefore, they are not all used for the prediction. For a detailed discussion on the selected features (those deemed valuable by the $L1$ regularization) see Section 5.2.1.

In summary, it is difficult to represent the quality of the feature values in absolute numbers. However, a good way to evaluate the quality of the information they provide is by training the model and comparing the prediction results against the baseline approaches.

5.1.1 Model Training

The training was performed by first splitting the complete data set using stratified 5-fold cross-validation, such that the training data set has a size of 80% and the testing data set has a size of 20%. This allows comparing the prediction performance of the different models trained on the different folds of the data set [29]. Figure 5.4 shows the *Receiver Operating Characteristic* (ROC) of the Bluestreak models trained on the different folds for gender segment prediction. Figure 5.5 shows the ROC for the User-Agent-based models. The models of the different folds for the Bluestreak model performed an *Area Under Curve* (AUC) of 0.61 to 0.63, while the User-Agent-based models performed with an AUC of 0.57 to 0.59. The AUC for all models is > 0.5 which indicates that the algorithm is suitable for the problem and the data set contains the necessary information to differentiate a consumer in the “Female” segment from a user in the “Male” segment.

5.1.2 Hyperparameter Tuning

Hyperparameters allow controlling the learning process in machine learning. By tuning them the optimal set of hyperparameters is chosen determined by the performance of the models trained with different hyperparameters. Therefore, in the following, we describe the hyperparameter tuning process of the inverse regularization strength and the feature selection we applied when training the model.

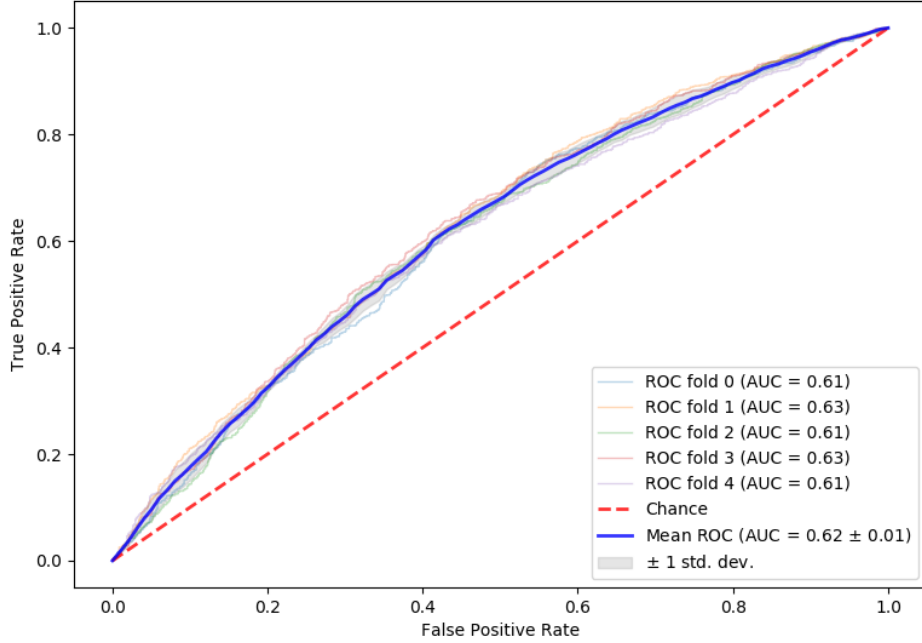


Figure 5.4: Receiver Operating Characteristic of the Bluestreak model.

Inverse Regularization Strength of L1 Regularization

When training a logistic regression model with $L1$ regularization, a parameter called the *Inverse Regularization Strength*, denoted with C must be specified. In order to determine the right value for this parameter, a grid search has been performed, comparing the results of models trained with different values of this parameter against each other. To determine the range of the value for C where the results are the highest, different values for C were at first randomly chosen. Based on the results the range with the best performance was then examined in detail, as can be seen in Figure 5.9.

Feature Selection

Another parameter for this specific use case is the addition or removal of features. As the $L1$ regularization identifies irrelevant features, we can first train a model on all features and afterward do the second round of training by already removing the

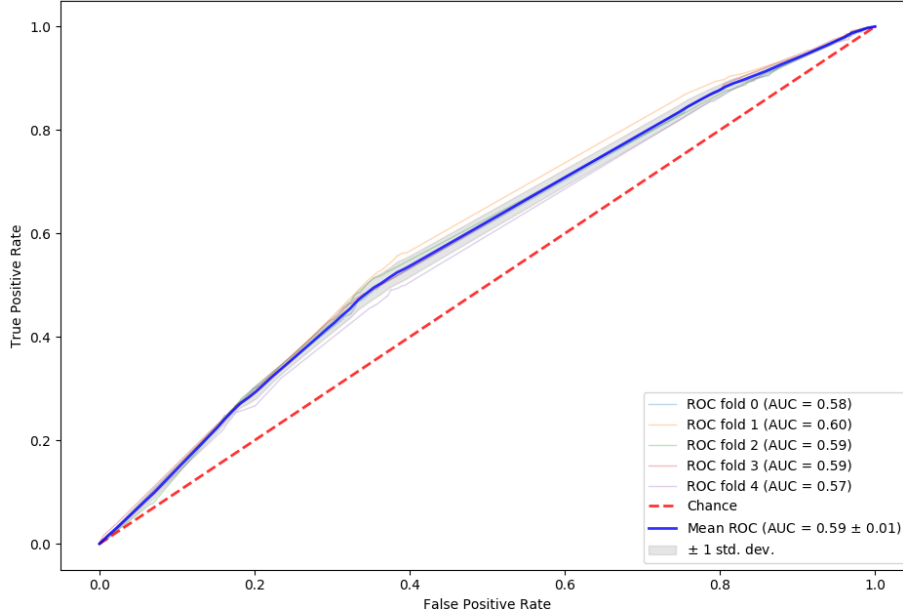


Figure 5.5: Receiver Operating Characteristic of the User-Agent-based model.

features that have been identifiable as irrelevant by the previous run. For this we have two options: either we can remove the whole feature, or we can just remove a certain “feature=value” combinations.

Taking the “language” feature as an example, listing them by the most to the least frequent: “language=de” (12,185 times), “language=en” (8,850 times), . . . , “language=no” (1 times). There is only one data point with the language “no” in the data set, therefore, it is not a valuable feature to determine the belonging of a segment. So instead of including or excluding the “language” feature as a whole, just the languages that have been determined not valuable by the $L1$ regularization can be removed.

As a comparison of these two strategies, the results for the removal of whole features are displayed in Figure 5.7 and the removal of just certain values are displayed in Figure 5.8. This shows that the removal of features whole leads to a higher test score compared to using all features. The removal of selected “feature=value” combinations leads to an even higher score. Therefore the approach of removing “feature=value” combinations has been chosen for model training.

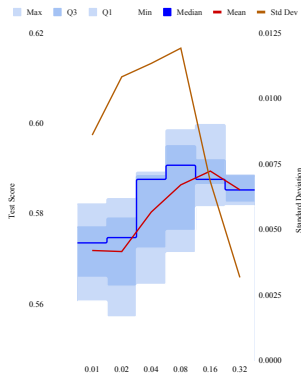


Figure 5.6:
No removal of features.

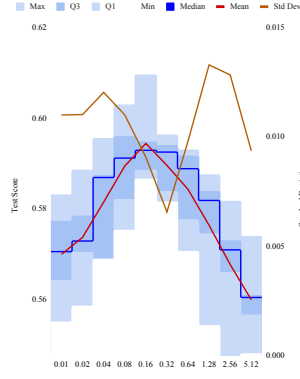


Figure 5.7:
Removing whole features.

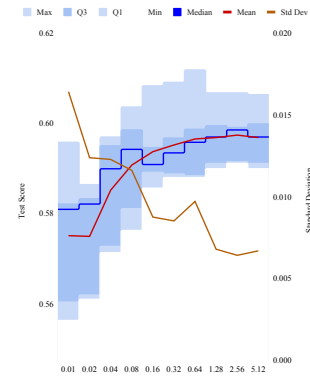


Figure 5.8: Removing just
"feature=value" combinations.

Figure 5.9: Score of grid search over C for different strategies of feature removal by $L1$ regularization.

5.2 Library Implementation

The Bluestreak library is written in TypeScript, which is programmatically converted (transpiled) into JavaScript for execution on the client. The file resulting from the transpilation process of the front-end library will be referred to as the *predictor-script*. It collects data to predict the consumer's segments and sends them back to the back-end. This library is served by a web server, also acting as the aforementioned back-end using the koa web framework for handling web-requests.

Figure 5.10 shows the whole process of a consumer opening the webpage until it is loaded and how the Bluestreak approach integrates into this process.

In order to use the Bluestreak library on a website, a script tag is used for loading the predictor-script. When the browser encounters the script tag it requests the predictor-script from the Bluestreak web server.

Unintuitively for a client-based approach, the data collection starts on the server-side. When the browser requests the predictor-script, the data naturally sent along with the HTTP-Request is pre-processed and sent to the client as part of the model input. The reasoning behind this is to save time and bandwidth. For example, extracting information from the User-Agent is a non-trivial task, that requires sophisticated parsing with a lot of rules. This costs processing time and increases the size of the library

unnecessarily. Therefore, parsing of the User-Agent is done on the server, where a database of previously parsed User-Agents is built for low-latency lookup of common User-Agents. To avoid the overhead of an extra HTTP-Request, the data collected on the server is injected into the library before it is sent as the response to the client.

Once the script execution is started a safety net for catching expected as well as unexpected errors during the script's execution is set up. Initially, this was done for debugging purposes but later on, encountered errors turned out to be another source of detecting certain implicit features for a browser and device combination.

While the features of the browser are collected, the Bluestreak model is downloaded in the background. The model could have been downloaded along with the library, however, doing it in a second request allows for a smaller initial payload and therefore a potentially sooner start of the feature collection as the download of the model happens in the background. Furthermore, during the model download request more features such as download speed and differences between client and server clock can be collected.

This extra request allows to gauge the network speed and allows to estimate the time difference between the clock of the client device and the clock of the server. To gauge the time difference, the strategy of the NTP protocol is used [41].

Once all features are collected and the model has been downloaded, the segment prediction starts. For this, an evaluator of the logistic regression model was implemented. This evaluator sums all coefficients of all collected feature-value combinations and applies the logistic function to calculate the confidence for each segment. It then selects the segment with the highest confidence value from the age and gender segments respectively. Only the predicted segments are then sent along with the impression request. All collected data stays on the device.

This means that all processing of PII-data is transparent and in total control of the client. Because the data is not sent to a server, it can therefore not be used to identify the consumer again at a later point in time. From the perspective of the Bluestreak server, every interaction is with a previously unknown entity as a predicted segment by itself is too coarse to re-identify a previously seen individual.

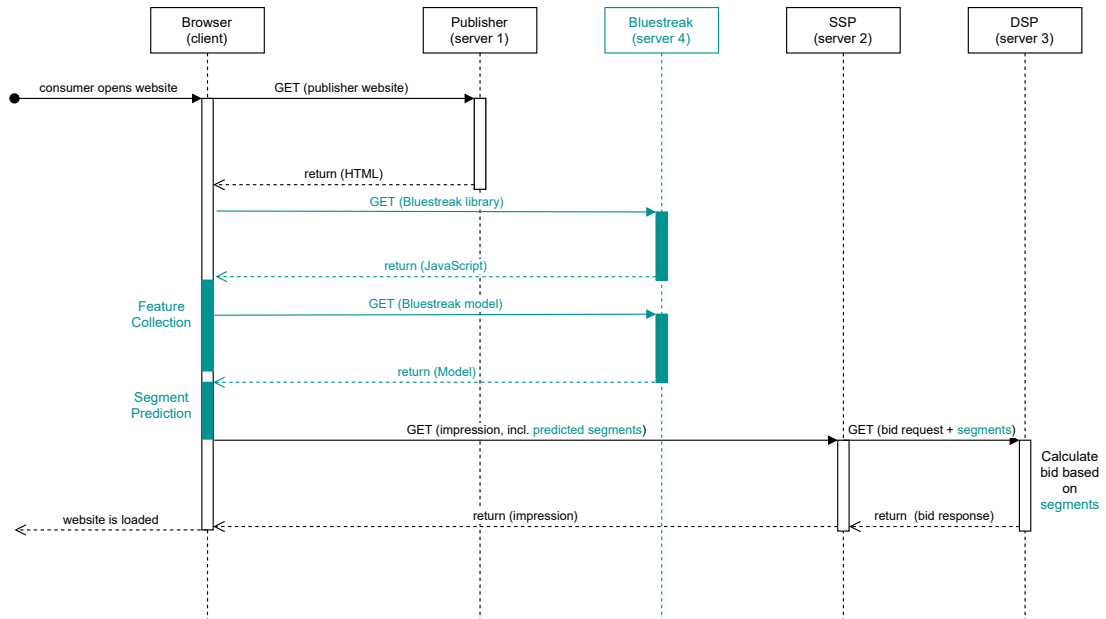


Figure 5.10: This figure shows (highlighted in teal color) how using the Bluestreak library changes the diagram introduced in Figure 3.2. Through feature collection and segment prediction, consumer segments are determined and sent along with the *impression* request, where previously the ID used for tracking was sent. Server 1, 2, and 3 are just symbolic for different realms that may consist of multiple servers.

5.2.1 Selected Features

This section discusses the selected features in detail. Each data point in the data set has 230 features. However, not all of the features are useful for segment prediction, therefore the number of features was reduced by removing those of which all coefficients were set to 0 by the *L1* regularization. A grouped overview of the selected features is given in Table 5.4.

Audio

The *Audio* fingerprint, in combination with the *User-Agent*, helps to identify the browser and device version with more precision. It is generated by generating a certain sound which, can be expressed in a float that differs between the browser and device versions.

Feature	Short Description
Audio	An audio fingerprint.
Battery	If the battery is charging or not (if battery API available).
Browser	The browser name and version as in Table 4.1 and WebKit version.
CSS Colors [†]	Default css colors, might differ based on theme of OS.
Devices ^{†,◇}	Result from <i>navigator.mediaDevices.enumerateDevices()</i> .
Errors	Errors encountered during feature collection.
Language/Country [◇]	combination of all language and country information.
OS CPU	Number of CPUs.
Platform	Underlying platform eg: <i>Win32, Linux, iPhone</i> , etc.
Plugins [◇]	Installed plugins eg: <i>Chrome PDF Plugin</i> .
Timezone [◇]	Timezone indicated by city.
Timings	Different time periods measured during collection.
Touch Support	Whether the device has a screen supporting touch input.
User-Agent	A string describing the browser and device.
WebGL	Values, and features like shaders and extensions.
Width & Height [‡]	All values regarding width & height of the screen and the window.

◇ Values encoded using one-hot encoding as explained in Section 5.2.2.

† Multiple values joined together in order of appearance.

‡ Multiple values joined together in alphabetical order.

Table 5.4: An excerpt of features used in data collection with a short description. A longer description can be found in Section 5.2.1.

Battery

The *Battery* information consists of battery charging and discharging speed and the current battery level. Charging speed in combination with the battery level can yield information about battery health as well as the use of fast charging. The discharging speed in combination with the battery level can yield information about the battery health as well as the current power drain.

Browser

Information about the *Browser* is gathered via browser name and family fields, as well as its version and the WebKit version. The browser version also allows gauging if important apps like the browser are kept up to date or not.

Connection

The *Connection* information allows differentiating if a device is used with a broadband connection, via cable or WiFi, or through a mobile connection.

CSS Colors

The *CSS Colors* depend on the browser and its version but are sometimes also customized by a theme selected by the consumer. This can be determined in combination with the browser name and version.

Devices

The detectable *Devices* are determined by a call to *navigator.mediaDevices.enumerateDevices()*. These may be very limited on certain operating systems, but they can also be a good indicator of what device is used by a consumer.

Errors

During execution, all encountered *Errors* are collected, which gives information about missing features or broken APIs as these are consistent on each run. This feature was originally only used for debugging purposes but was deemed useful in more fine-grained browser detection.

Language/Country

Language and *Country* information is collected from several sources, namely all language fields found in the navigator such as: “navigator.language”, “navigator.languages”, “navigator.browserLanguage”, “navigator.systemLanguage”, and “navigator.userLanguage”, as well as the *accept-language* field in the HTTP-header. The “accept-language” field also contains information about the preference factor, which allows an even better judgment of how important a particular language is. It is also possible to get information about the country, as the language identifiers usually also contain country information, such as “en-US” or “en-UK”.

OS CPU

The *OS CPU* information is gathered from the *navigator.oscpu* field. This information also allows improving to identify which device model is used, if this information is unclear based on for example the User-Agent field. It also helps to identify inconsistent information if the browser reports a wrong User-Agent. In that case, this particular segment prediction could be flagged as potentially inconsistent.

OS CPU and Platform

The *Platform* information is gathered from the *navigator.platform* field. This information also helps to narrow down the device model information and uncover inconsistencies.

Timezone

The *Timezone* field does not only provide information about the consumer's time zone but also which city was selected to represent a time zone.

Timings

During feature collection, different timestamps are recorded to determine how long the feature collection took, how long the download of the model took and how big the time difference between the client's clock and the server clock is. This provides insights into the processing speed, connection speed and can be an indication of the device's clock being out of sync relative to the server clock.

Touch Support

The *Touch Support* information is gathered from determining if touch support is available through the *ontouchstart* event and how many touchpoints are supported through *navigator.maxTouchPoints*. This information also helps to narrow down the device model and identify inconsistencies.

User-Agent

The *User-Agent* can be used to determine the browser and device used to a certain degree. But this is not always accurate. Therefore, it helps to combine this information with other features discussed here.

WebGL

The *WebGL* information is gathered from several sources. First of all WebGL support can be identified. If WebGL is supported information about shaders and available extensions can be gathered from *WebGLRenderingContextBase*. Through the use of a canvas, it is possible to identify the GPU vendor and renderer as well as creating a fingerprint to help identify the device model. This information also helps to identify inconsistencies.

Width & Height

There are several *Width & Height* values that may provide information on a device. First of all the regular screen width and height. But it is also possible to determine how much of this screen real-estate is reported to be actually available. Otherwise, the reported size of the outer window and boolean values if the device is in portrait mode and if the browser's "top bar" is present.

Excluded Features

A much larger number of other features have been considered. However, many of these carry redundant information or are non-existent in modern browsers anymore. Some also offer very low information gain while adding a lot of weight to the library and increase the execution time tremendously, for example identifying installed fonts. As those features were excluded by the *L1* Regularization they proved to have either no importance for the data set we are using or are simply not relevant for segment prediction in general. Therefore we are not going into further detail on these features.

Summary

In summary, each feature by itself provides some information, but this information can be leveraged by combining features with each other. They even allow to spot inconsistencies regarding device information but also language information.

How features are combined is discussed in Section 5.2.3.

5.2.2 Data Pre-Processing

All numerical features were scaled to fit in the interval (0, 1) and all categorical features were one-hot encoded, by creating a binary vector, equal in length to the number of features, with a 1 indicating a presence of that feature and a 0 otherwise.

Due to the $L1$ regularization, it was not required to exclude features manually, as they have been regularized down to 0 and could therefore be excluded from the final model, without the need of assumptions on feature relevance.

5.2.3 Feature Engineering

The raw features themselves contain all the information available. However, if interpreting the features manually, this information can be made better understandable for the logistic regression algorithm. Taking the languages as an example, we have several fields mentioned in 5.2.1. Looking just at the “accept-language” header from the HTTP-Header, we might get a string like this: “de-CH,de-DE;q=0.9,de;q=0.8,en-US;q=0.7,en;q=0.6”. This means that the most preferred language is the Austrian version of German. However, the version of German spoken in Germany is in the same preference level. The second highest preference level has the German language in general. If those language options aren’t available the U.S. version of English has the next highest preference, followed by general English as the least preferred version. In this case, the logistic regression algorithm just sees this string and can not interpret it. For example it could not correlate it to the almost similar string of “de-CH;q=0.9,de;q=0.8,en-US;q=0.7,en;q=0.6;da;q=0.5”. To make this more accessible the string can be split at each comma character.

But this still wouldn't be totally correlatable to the almost similar accept-language header of "de-CH,de-DE;q=0.9,en-US;q=0.8,en;q=0.7". As also seen in Figure 5.11 we decided to split up the language string such that all information contained in this string can be interpreted:

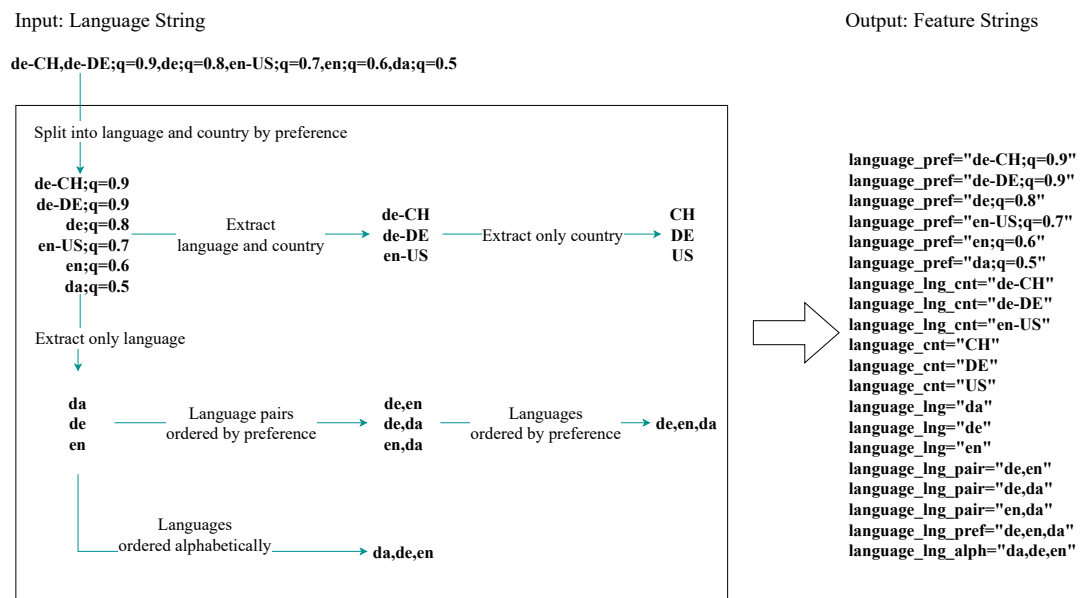


Figure 5.11: Approach of splitting up the language string to make the information contained in it better processable by the logistic regression model training algorithm.

It indicates that the languages "de", "en", and "da" are accepted but "de" is preferred over "en" and "en" over "da" and in general the language combination of the three languages "da,de,en" are accepted. It allows correlating the countries "CH", "DE", and "US". Even though "fr" and "fr-CH" is not accepted, it is still possible to correlate to "CH" due to splitting country and language.

Other language fields, those found in the "navigator" also provide further information and can also be checked for inconsistencies, which would point to a misconfiguration and mean that the data may not be trusted as much.

5.3 Supported Browsers and Devices

As stated in Section 4.3.1, it is required to support about 95% of the encountered traffic. Through the use of shims (es5-shim and es6-shim) and by adding custom implementa-

Name	Size
Library (with shims)	41.9 KiB
Library (without shims)	18.8 KiB
Model	13.7 KiB

Table 5.5: Size of the Bluestreak frontend library with and without shims (support for inconsistent browser behavior) and the size of the model with age and gender coefficients.

tion of functions (such as the “slice” and “reduce” methods of the Float32Array) that are not available in all browsers we were able to support all versions listed in Table 4.1. Furthermore, some features, such as the battery API or the precision API, are not available at all and these values are just left empty. For a specific use case, it should always be considered if the extra overhead produced by including these libraries is really needed or if a higher percentage of traffic may be excluded for a simpler and slimmer library.

5.4 Payload Size

As stated in the requirements, the size of the library, as well as the size of the model, have to be kept as small as possible to minimize the noticeable impact by the consumer. Therefore, two different versions of the library are shown in Table 5.5, where the size difference between the library with shims and polyfills compared to the library without can be seen. Despite the large number of considered features, the size of the model could be kept considerably small due to the aforementioned feature selection.

6

Evaluation

Based on the requirements developed in Chapter 4, we implemented a proof of concept library and also trained a model for user segment prediction in the client’s browser as described in Chapter 5. To evaluate if our contribution fulfills the requirements we conducted the following experiments: At first, the performance of the trained model is evaluated, then the effect on privacy-protection of individuals is analyzed, and finally, the overhead on bandwidth, resource consumption, and overall page loading speed on the client device is discussed.

6.1 Model Evaluation

The logistic regression models for the User-Agent-based and the Bluestreak approach were trained on the real-world data set described in Chapter 5. In the following, their performance is evaluated and compared against each other. As the User-Agent-based approach is only relying on the User-Agent string sent in the HTTP-Header and therefore has less information available, the Bluestreak approach is expected to perform better than the User-Agent-based approach.

Furthermore, we also compared the performance of the aforementioned models

against the performance of a state-of-the-art approach using Cookies for recording user history to predict the consumer segments as introduced in Section 3.5. This Cookie-based approach was provided by an RTB advertising company, therefore it was implemented as part of this work and was trained on a different much larger real-world data-set. It serves as a real-world baseline for comparison. Since the Cookie-based approach is using consumer journey data, which reveals more about the consumer's behavior it has a potentially better performance than the Bluestreak approach.

6.1.1 Results

This evaluation compares the sensitivity, specificity, and accuracy of the user segment prediction of the User-Agent and the Bluestreak approach. This was done by first counting the *True Positives* (TP), *True Negatives* (TN), *False Positives* (FP), and *False Negatives* (FN) and then determining sensitivity, specificity, as well as accuracy and comparing them among the different approaches. For the evaluation, the data set introduced in Chapter 5 was used. Due to the limited size of the data set, it was split using stratified 5-fold cross-validation, such that the training data set has a size of 80% and the evaluation data set has a size of 20%. This allows training 5 different models, where each is evaluated on a different portion of the data set. This allows using the whole data set for evaluation while still keeping a strict partitioning of training and evaluation data points for each model, as described in [29].

Sensitivity Comparison

Sensitivity measures a method's ability to correctly identify subjects belonging to a segment. It is defined as

$$\frac{TP}{TP + TN}$$

and therefore the *True-Positive-Rate* (TPR). In Figure 6.1 the sensitivity of the age predictions and in Figure 6.2 the sensitivity of the gender predictions are compared. The Bluestreak approach is generally performing better or about as good as the User-Agent-based approach for the age segments, except for the “60+” age segment. For the “F” (“Female”) segment, the User-Agent-based approach is performing better than

the Bluestreak approach, however, the Bluestreak approach is performing better in the “M” (“Male”) segment.

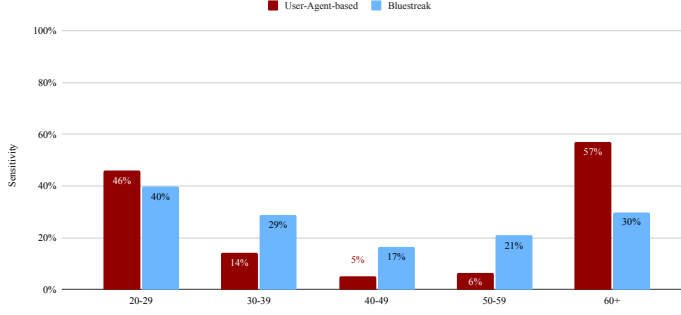


Figure 6.1: Age sensitivity.

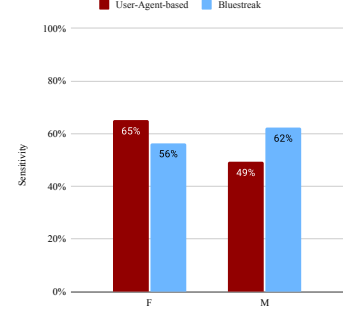


Figure 6.2: Gender sensitivity.

Figure 6.3: Sensitivity results of the Bluestreak approach compared to the Cookie-based and User-Agent-based approaches.

Specificity Comparison

Specificity measures a method’s ability to correctly identify subjects not belonging to a segment. It is defined as

$$\frac{TN}{TN + FP}$$

and therefore equivalent to $1 - \text{False-Positive-Rate}$ (FPR). In Figure 6.4 the specificity of the age predictions and in Figure 6.5 the specificity of the gender predictions are compared. Here the Bluestreak approach is generally performing better or about as good as the User-Agent-based approach. For the “F” segment, the Bluestreak approach is performing better than the User-Agent-based approach, however, the User-Agent-based approach is performing better in the “M” (“Male”) segment.

Accuracy Comparison

Accuracy measures a method’s ability to do both, correctly identify subjects belonging to a segment and those not belonging to a segment. It is defined as

$$\frac{TP}{TP + FN},$$

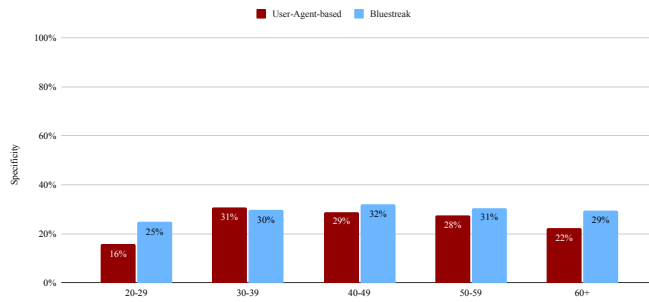


Figure 6.4: Age specificity.

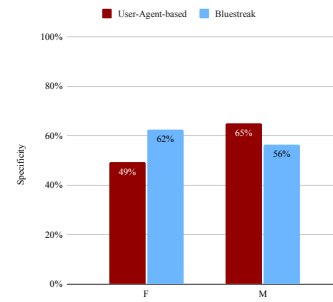


Figure 6.5: Gender specificity.

Figure 6.6: Specificity results of the Bluestreak approach compared to the Cookie-based and User-Agent-based approaches.

and equivalent to the sum of all sensitivity and all specificity segments respectively. In Figure 6.7 the accuracy of age predictions, and in Figure 6.8 the accuracy of gender predictions are compared. Here the Bluestreak approach is performing better than the User-Agent-based approach for both segments.

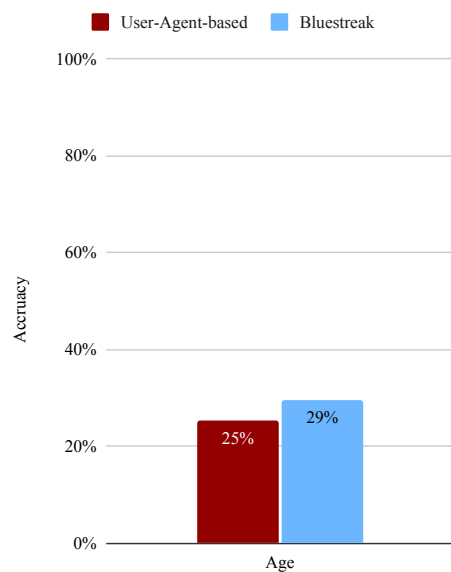


Figure 6.7: Regarding age accuracy, the Bluestreak approach achieves a 4 percentage points higher accuracy than the User-Agent-based approach.

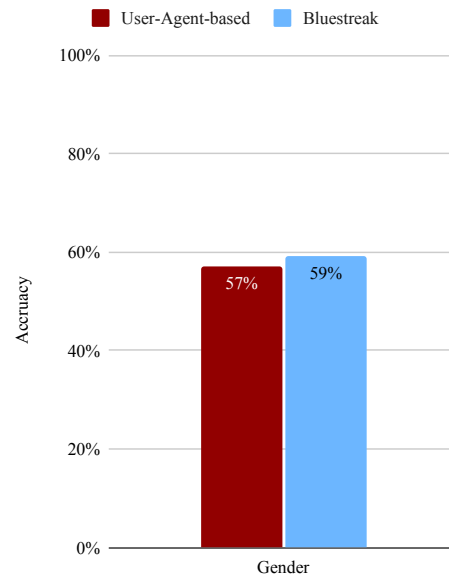


Figure 6.8: Regarding gender accuracy, the Bluestreak approach achieves a 2 percentage points higher accuracy than the User-Agent-based approach.

The overall comparison for the model evaluation can be seen in Figure 6.11. The results show that the Bluestreak approach is performing better than the User-Agent-

Segment	Sensitivity [†]	Specificity [†]	Sensitivity [‡]	Specificity [‡]
20-29	45.91%	15.78%	39.73%	24.81%
30-39	14.17%	30.59%	28.91%	29.72%
40-49	5.24%	28.95%	16.59%	31.90%
50-59	6.29%	27.65%	20.95%	30.58%
60+	57.09%	22.35%	29.90%	29.41%
F	65.06%	49.25%	56.38%	62.29%
M	49.25%	65.06%	62.29%	56.38%

[†] Stands for User-Agent-based approach.

[‡] Stands for Bluestreak approach.

Table 6.1: Per segment comparison of the sensitivity and specificity of the User-Agent-based approach and the Bluestreak approach.

Overall	Sensitivity [†]	Specificity [†]	Accuracy [†]	Sensitivity [‡]	Specificity [‡]	Accuracy [‡]
Age	25.74%	25.06%	25.15%	27.22%	29.29%	29.45%
Gender	57.15%	57.15%	57.17%	59.34%	59.34%	59.33%

[†] Stands for User-Agent-based approach.

[‡] Stands for Bluestreak approach.

Table 6.2: Overall comparison of the sensitivity, specificity, and accuracy of the User-Agent-based approach and the Bluestreak approach.

based approach. Therefore, the requirements stated in Section 4.1 have been met.

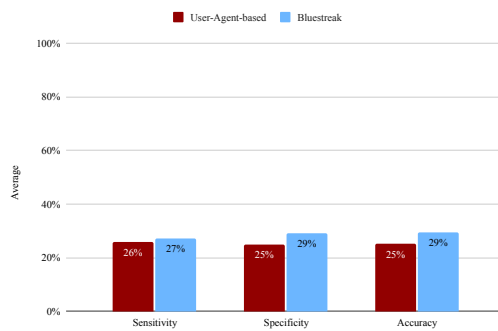


Figure 6.9: Age overall.

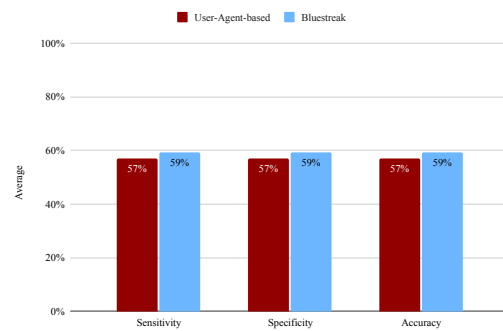


Figure 6.10: Gender overall.

Figure 6.11: Overall prediction results of the Bluestreak approach compared to the Cookie-based and User-Agent-based approaches.

6.1.2 Comparison to the Cookie-Based Approach

Regarding age accuracy, the Cookie-based approach achieves a 19 percentage points higher accuracy than the Bluestreak approach, and a 23 percentage points higher accuracy than the User-Agent-based approach. The Bluestreak approach, however, still outperforms the User-Agent-based method by 4 percentage points, as shown in Figure 6.12. This demonstrates that while Bluestreak does not match the accuracy of cookie-based tracking, it provides a meaningful improvement over relying solely on User-Agent data by leveraging additional on-device signals. In scenarios where tracking via cookies is restricted or not feasible, Bluestreak offers a privacy-preserving alternative with significantly better accuracy than the baseline.

Regarding gender accuracy, the Cookie-based approach achieves a 22 percentage points higher accuracy than the Bluestreak approach, and a 24 percentage points higher accuracy than the User-Agent-based approach, as illustrated in Figure 6.13. Similar to the age prediction results, Bluestreak surpasses the User-Agent-based method by 2 percentage points, again highlighting the added value of privacy-respecting on-device inference in constrained environments.

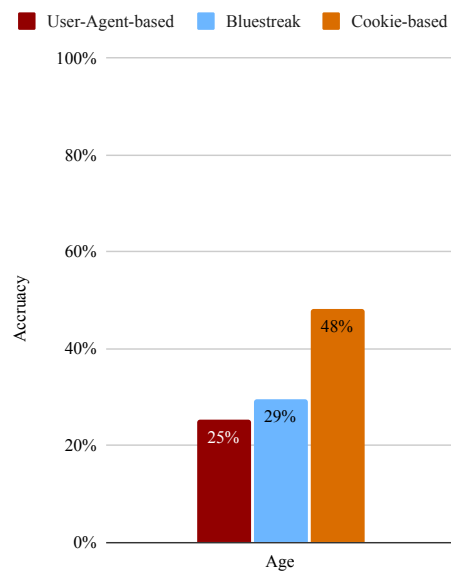


Figure 6.12: Cookie-based approach age accuracy compared to the Bluestreak approach and User-Agent-based approach.

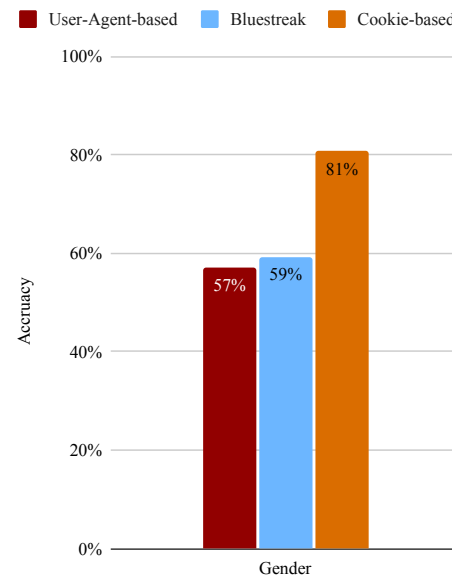


Figure 6.13: Cookie-based approach gender accuracy compared to the Bluestreak approach and User-Agent-based approach.

6.2 Privacy Evaluation

This section compares the gain in consumer privacy of the Bluestreak approach over other approaches.

To determine the gain in privacy we have calculated the uniquely identifiable users in three different scenarios:

Unique Identifier Each individual is identified with a unique identifier like a cookie ID. Therefore, no two different individuals will get the same identifier.

Fingerprint All data not sensitive to change between sessions is gathered on the device and concatenated into a long string. This string is then hashed and used as the fingerprint. This is a common method for building browser fingerprints [66].

Bluestreak Only the predicted age and gender segments are sent back to the server.

Privacy Results

The results for the different scenarios are shown in Figure 6.14. The total number of individuals is 13,489. In the *Unique Identifier* scenario, each user is assigned a unique identifier, like a cookie ID, which makes 100% of the users identifiable.

For the *Fingerprint* scenario a fingerprint is built based on the raw data on the device. In this scenario 11,126 (or 82.48%) users can be uniquely identified with this fingerprint.

In the *Bluestreak* scenario not a single individual could be uniquely identified by just using the predicted segments. Therefore, a predicted age and gender segment could not be used to re-identify an individual and can therefore not be considered personal data, this confirms the requirement discussed in Section 4.2.1. One could argue that, if the prediction assigns one individual an age and gender segment combination not assigned to any other individual, that this individual could be re-identified by the algorithm. However, this is a very unlikely case if the number of individuals is much larger than the number of segments. Therefore, the assumption that two different requests

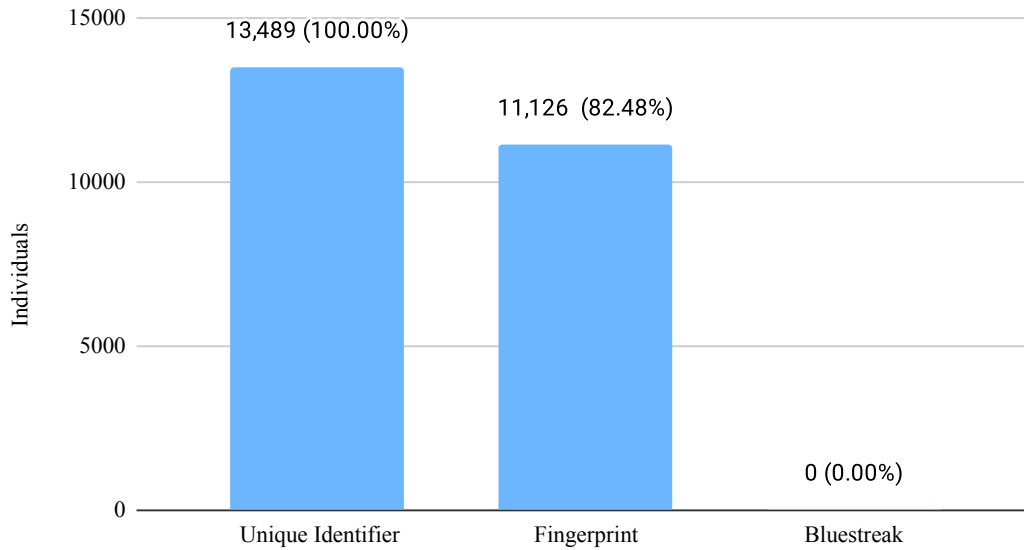


Figure 6.14: Users that are uniquely identifiable by different approaches. The lower the percentage the higher the chance of an individual user to be not uniquely identifiable.

are coming from the same individual, just based on the same predicted age and gender segment, would only apply if there are more segments than individuals.

6.3 Performance Overhead Case-Study

It is not trivial to determine the performance overhead in a single metric, as each device could behave differently, and depending on the scenario a big overhead in one category might have a big impact on one device but a small impact on another. It is also difficult to determine at what point the overhead becomes too big. Especially when considering that each individual values their privacy differently.

Therefore, we conducted a case study to assess the performance measured with and without the Bluestreak library.

Experiment Setup

To test the library in a realistic environment, we decided to conduct the experiment on popular websites that display banner ads next to the content they are publishing. To

inject the library, we used an HTML-*iframe* tag, in the following simply called *iframe*. For this, We created an HTML-file containing the Bluestreak library in a script tag and the website in an iframe. However, this technique is not optimal, as it does not work on every website, therefore we could only test it on such websites that allowed the inclusion via an iframe. In particular, we tested in on a German news website (referred to as Website 1), a German gaming-news website (referred to as Website 2), and a German tech-news website (referred to as Website 3).

The performance was measured using the Chrome DevTools performance profiler [37]. It allows measuring the heap usage and CPU time spent on executing JavaScript. As the library and the tested site itself are running in different iframes, it is possible to record the resource consumption of each separately. Furthermore, it is possible to determine the *Time to First Paint*, which gives us an indication of how much the page loading speed is impacted by executing the Bluestreak library.

Each site was tested in three different scenarios:

direct this scenario measures the impact when the Bluestreak library is executed during page load, therefore the Bluestreak library is included directly in a script tag.

onLoad this scenario measures the impact when the Bluestreak library is executed after page load, therefore the script is injected by a function triggered by the “onLoad” event.

plain this scenario profiles the website without the Bluestreak library.

This allows differentiating the impact on page loading speed between the three scenarios.

For each of the three websites, each of the three scenarios was repeated ten times. In total 90 executions were performed. The milliseconds of CPU time used for scripting was calculated by averaging the CPU time of the executions on each website respectively. The same was done for RAM usage. The experiments were conducted on a Lenovo ThinkPad T470, running Ubuntu 20.4 and Chrome version 87.0.4280.88.

6.3.1 Performance Overhead Results

Several aspects have been evaluated to assess the performance overhead induced by our approach, including CPU usage, RAM usage, script execution time, as well as bandwidth overhead.

As stated in the requirements, the display of advertising banners can be done in the background, after the “onLoad” event, to limit the main page loading speed impact. For this reason, two scenarios have been evaluated to determine the impact on page loading speed:

1. the Bluestreak library is executed directly while the page is loading
2. the Bluestreak library is executed after the “onLoad” event has been fired

Page Loading Speed Impact

The impact on page loading speed is displayed in Figure 6.15. Measured was the “Time to First Paint” in Chrome DevTools. The center lines in the figure represent the medians, box limits indicate the 25th and 75th percentiles, whiskers extend 1.5 times the interquartile range from the 25th and 75th percentiles, and the data points are represented by small black circles. It shows that most of the measurements lie close to each other except for a few outliers. Therefore, the median lines will be used for comparison, as outliers are biasing the average value.

The “direct” scenario has the highest impact on the Time to First Paint, for all three websites the median of the “direct” scenario are the highest. However, when delaying execution to after the “onLoad” event is fired, the median Time to First Paint can be reduced in all three scenarios. Finally, for each of the three sites, the median Time to First Paint is the lowest in the “plain” scenario, because no additional code is executed.

CPU Usage

The Bluestreak library required 528 ms (on Website 1), 543 ms (on Website 2), and 714 ms (on Website 3) of CPU time on average. This is already a substantial part compared to the CPU time used by the tested websites themselves, as shown in Figure 6.22.

An interesting observation can be made in Figure 6.18. The Bluestreak library took 714 ms while in the other figures it took 528 ms and 543 ms. The reason behind this is unclear.

RAM Usage

On average the amount of heap space used by the frame of the Bluestreak library was measured to be between 4.14 MiB and 4.36 MiB as shown in Figure 6.22. When the Bluestreak library was not loaded, the heap space occupied by the frame was 1.72 MiB on average. Therefore, it is likely that not all of the measured occupied heap space can be accounted for by the Bluestreak library. This means we will assume 4.36 MiB as an upper bound. Therefore, the heap space occupied by the Bluestreak library is less than the heap space usage of each of the websites on average.

It is also important to note that the frame of the site does not include other iframes loaded by the site such as advertisement banners, etc. This means that the heap space usage incurred by loading the site may be higher overall than what is reported for the website in this work.

Website	plain	onLoad	direct
Website 1	149.25 ms	175.40 ms	292.20 ms
Website 2	124.35 ms	214.20 ms	240.90 ms
Website 3	250.55 ms	305.35 ms	360.60 ms

Table 6.3: Median time in milliseconds elapsed until *Time to First Paint*.

Website	Frame	Heap Space	Scripting Time
1	Website	8.61 MiB	775 ms
	Bluestreak	4.36 MiB	528 ms
2	Website	7.98 MiB	581 ms
	Bluestreak	4.14 MiB	543 ms
3	Website	12.07 MiB	858 ms
	Bluestreak	4.30 MiB	714 ms

Table 6.4: Average RAM and CPU usage of the website frame and the Bluestreak frame compared on three different websites.

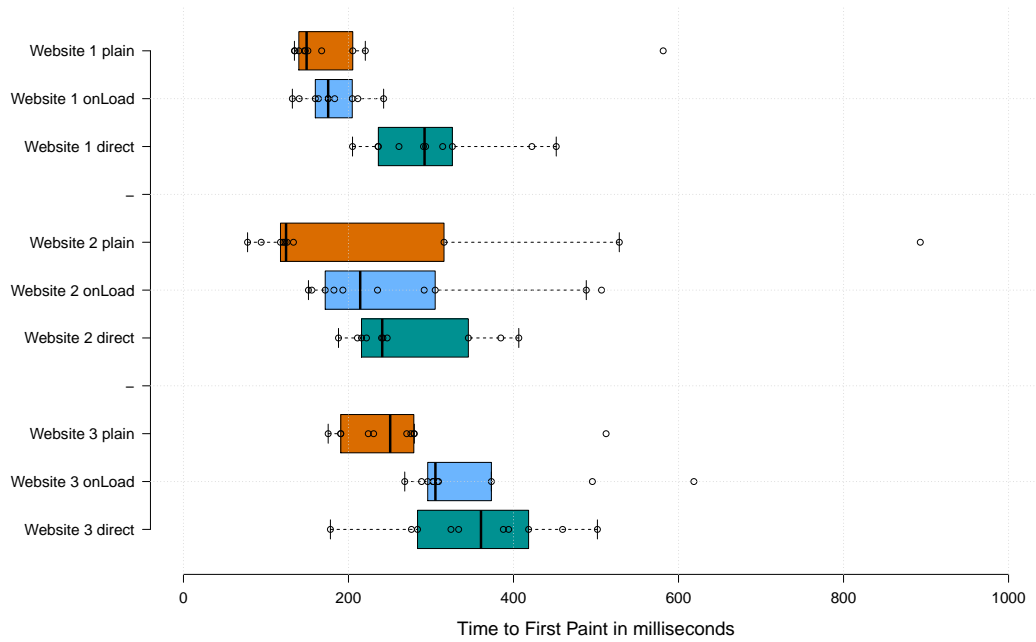


Figure 6.15: Page loading speed impact measured on several news websites without the Bluestreak library, with the Bluestreak library directly being executed, and with a time-delayed execution of the Bluestreak library.

Bandwidth Consumption

While the prediction results that are being sent back to the server are made up of just a few bytes identifying the predicted age segment and gender segment, the majority of bandwidth is consumed by the JavaScript library and the model itself.

As discussed in Section 4.3, the bandwidth requirements of websites have been increasing in recent years. As a baseline, we are considering the median size for a mobile website in the year 2020, of 1.89 MiB [27]. As listed in Table 5.5 the size of the Bluestreak library with shims (with support for older browser versions) is 41.9 KiB and therefore considerably larger than the size of the Bluestreak library without shims of 18.8 KiB. The model itself has a size of 13.7 KiB. Using the more heavy version, the Bluestreak library with shims, and adding the overhead of the Model yields a total

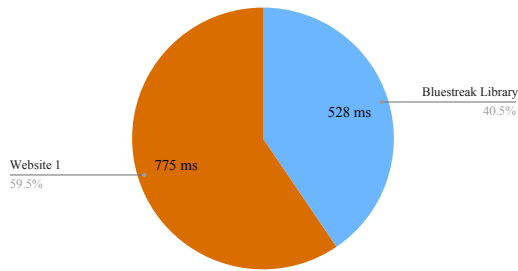


Figure 6.16: CPU usage, website 1.

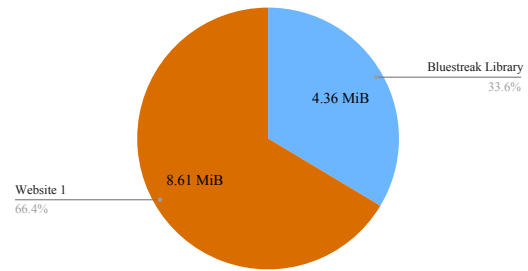


Figure 6.19: RAM usage, website 1.

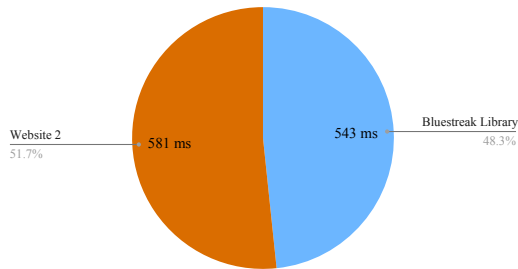


Figure 6.17: CPU usage, website 2.

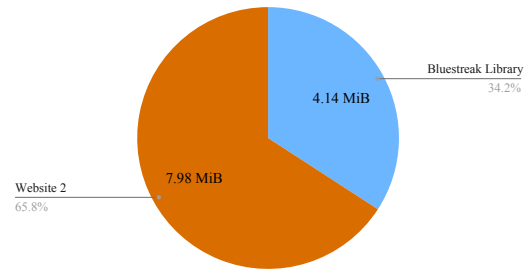


Figure 6.20: RAM usage, website 2.

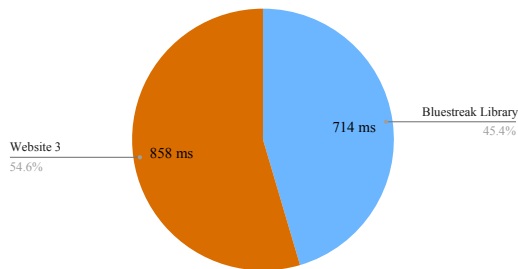


Figure 6.18: CPU usage, website 3.

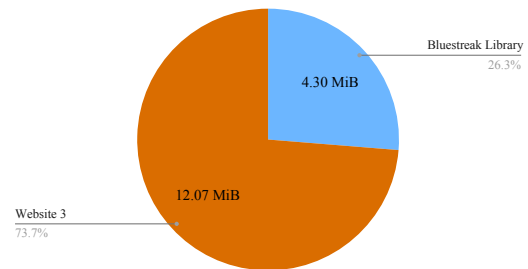


Figure 6.21: RAM usage, website 3.

Figure 6.22: Resource consumption measured using Chrome DevTools on three different websites. CPU usage in milliseconds of scripting time, RAM usage calculated by occupied heap space.

overhead of 55.6 KiB. Compared to a total page weight of 1.89 MiB, the bandwidth overhead is about 2.94%.

Summary

In summary, an overhead on all three measured metrics: loading speed, CPU usage, and RAM usage was measurable as can be seen in Table 6.5. All of the measured numbers are lower than those measured directly on the websites themselves, but they still

represent a non-negligible portion. However, the numbers recorded for each website serve as a lower bound as only the frame of the site itself was taken into consideration, not other frames that may be loaded by the site like advertisement banners, etc. Therefore, the impact on the user experience might not be as severe as it seems from the numbers reported in the table.

Metric	Website 1	Website 2	Website 3
Loading Speed	49.68%	6.80%	35.33%
Scripting Time	68.13%	93.58%	83.27%
Heap Usage	50.59%	51.88%	35.65%

Table 6.5: Average percentual increase of each measured metric.

Furthermore, it is possible to reduce the impact on page loading speed by delaying the execution of the Bluestreak library until after the “onLoad” event has been fired. However, since this delays the calculation of prediction results, it might not be suitable in all cases.

7

Conclusion

In this thesis, we introduced a novel approach for age and gender segment prediction, which utilizes data of the consumer's browser to perform the prediction on the device instead of sending the personal data to a web server, therefore not violating the consumer's privacy. In summary, we gave an introduction to the state of the art in online advertising and the related privacy issues. Furthermore, we introduced logistic regression and its use to predict age and gender segments in the context of Real-Time Bidding (RTB). We specified requirements that should be fulfilled to enable on-device prediction and described how we implemented our approach to overcome the technical challenges posed by these requirements. We conducted experiments and case-studies to evaluate the overhead on the client devices and our results show that the overhead is manageable and the impact on the user experience can be adjusted for the use-case, e.g. by delayed execution or the removal of low impact features.

Furthermore, we compared the accuracy, sensitivity, and specificity of the Bluestreak approach to a User-Agent-based approach. This was done to assess the prediction performance improvement enabled by the increased amount of information present on the consumer's device. In addition, we compared our results to a state-of-the-art approach that uses tracking cookies, which was provided by an RTB advertising company.

The results indicate that it is indeed possible to predict these segments without using cookies or fingerprinting methods while achieving a result that is better than a purely User-Agent-based approach.

In conclusion, we demonstrated that previous privacy-violating practices can be replaced by a privacy-preserving method, while still providing valuable insights to the advertiser. With our approach, each interaction of a consumer looks unique to the RTB advertising company—even if the same consumer had a previous interaction already, they still can not be re-identified by our method. Furthermore, consumers could potentially verify that all personal data stays on their device by reviewing the delivered JavaScript library. Therefore, they do not need to rely on a promise of responsible data processing but rather can verify themselves.

7.1 Future Work

In future work, the Bluestreak library could be extended to give consumers the option to send segments of their choice. This would enable them to request advertisements closer related to their preferences.

With further effort, it would be possible to reduce the size of the library, by implementing imported functions manually and by using a different compiler, like the Google Closure compiler. The size of the model could be decreased further by hashing the names and storing the values of the coefficients with a lower resolution. Also, the library model could be tailored for each request individually, according to the information provided in the User-Agent header.

Bibliography

- [1] Lalit Agarwal, Nisheeth Shrivastava, Sharad Jaiswal, and Saurabh Panjwani. Do not embarrass: Re-examining user concerns for online tracking and advertising. In *SOUPS 2013 - Proceedings of the 9th Symposium on Usable Privacy and Security*, page 1, New York, New York, USA, 2013. ACM Press. URL: <http://dl.acm.org/citation.cfm?doid=2501604.2501612>, doi:10.1145/2501604.2501612.
- [2] Ibrahim Mousa Al-Zuabi, Assef Jafar, and Kadan Aljoumaa. Predicting customer’s gender and age depending on mobile phone data. *Journal of Big Data*, 6(1), 2019. doi:10.1186/s40537-019-0180-9.
- [3] Muhammad Ammad-ud din, Elena Ivannikova, Suleiman A. Khan, Were Oyomno, Qiang Fu, Kuan Eeik Tan, and Adrian Flanagan. Federated Collaborative Filtering for Privacy-Preserving Personalized Recommendation System. *arXiv*, jan 2019. URL: <http://arxiv.org/abs/1901.09888>, arXiv:1901.09888.
- [4] Aaron Andersen. WebAIM: History of the browser user-agent string, 2008. Accessed 2021-02-20. URL: <https://webaim.org/blog/user-agent-string-history/>.
- [5] Michael Backes, Aniket Kate, Matteo Maffei, and Kim Pecina. ObliviAd: Provably secure and practical online behavioral advertising. In *Proceedings - IEEE Symposium on Security and Privacy*, pages 257–271. Institute of Electrical and Electronics Engineers Inc., 2012. doi:10.1109/SP.2012.25.
- [6] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical Secure Aggregation for Federated Learning on User-Held Data. nov 2016. URL: <http://arxiv.org/abs/1611.04482>, arXiv:1611.04482.
- [7] Brave. OK Google, don’t delay real browser privacy until 2022 — Brave Browser, 2020. Accessed 2021-03-13. URL: <https://brave.com/ok-google/>.
- [8] Research Scientists Brendan McMahan and Daniel Ramage. Google AI Blog: Federated Learning: Collaborative Machine Learning without Centralized Training Data, 2018. Accessed 2020-12-06. URL: <https://ai.googleblog.com/2017/04/>

federated-learning-collaborative.html.

- [9] Marija Burinskiene and Vitalija Rudzkiene. Application of logit regression models for the identification of market segments. *Journal of Business Economics and Management*, 8(4):253–258, 2007. URL: www.jbem.lt, doi:10.1080/16111699.2007.9636177.
- [10] Han Cai, Kan Ren, Weinan Zhang, Kleanthis Malialis, Jun Wang, Yong Yu, and Defeng Guo. Real-time bidding by reinforcement learning in display advertising. In *WSDM 2017 - Proceedings of the 10th ACM International Conference on Web Search and Data Mining*, pages 661–670, New York, NY, USA, feb 2017. Association for Computing Machinery, Inc. URL: <https://dl.acm.org/doi/10.1145/3018661.3018702>, arXiv:1701.02490, doi:10.1145/3018661.3018702.
- [11] Farah Chanchary and Sonia Chiasson. User Perceptions of Sharing, Advertising, and Tracking. In *Proceedings of the Eleventh USENIX Conference on Usable Privacy and Security*, SOUPS ’15, pages 53–67, USA, 2015. USENIX Association.
- [12] Olivier Chapelle, Eren Manavoglu, and Romer Rosales. Simple and scalable response prediction for display advertising. *ACM Transactions on Intelligent Systems and Technology*, 5(4), dec 2014. doi:10.1145/2532128.
- [13] Raheel Chaudhry and Paul Berger. Ethics in Data Collection and Advertising. *GPH - International Journal of Business Management* →, 2(07 SE - Articles):1–7, 2019. URL: <http://gphjournal.org/index.php/bm/article/view/240>.
- [14] Qi Chen, Yuqiang Feng, Luning Liu, and Xianyun Tian. ‘Understanding consumers’ reactance of online personalized advertising: A new scheme of rational choice from a perspective of negative effects. *International Journal of Information Management*, 44:53–64, feb 2019. doi:10.1016/j.ijinfomgt.2018.09.001.
- [15] Ye Chen, Pavel Berkhin, Bo Anderson, and Nikhil R. Devanur. Real-time bidding algorithms for performance-based display ad allocation. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1307–1315, New York, New York, USA, 2011. ACM Press. URL: <http://dl.acm.org/citation.cfm?doid=2020408.2020604>, doi:10.1145/2020408.2020604.
- [16] Ye Chen, Dmitry Pavlovy, and John F. Canny. Large-scale behavioral targeting. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 209–217, 2009. doi:10.1145/1557019.1557048.
- [17] Stephanie Clifford. Privacy Groups Urge F.T.C. to Examine Real-Time Ad Auctions - The New York Times, 2010. URL: <https://mediadecoder.blogs.nytimes.com/2010/04/08/privacy-groups-f-t-c-must-examine-real-time-ad-auctions/>.
- [18] Erdong Deng, Huajun Zhang, Peilin Wu, Fei Guo, Zhen Liu, Haojin Zhu, and Zhenfu Cao. Pri-RTB: Privacy-preserving real-time bidding for securing mobile advertisement in ubiquitous computing. *Information Sciences*, 504:354–371, dec 2019. doi:10.1016/j.ins.2019.07.034.

- [19] Peter Eckersley. Browser Versions Carry 10.5 Bits of Identifying Information on Average, 2010. Accessed 2021-02-20. URL: <https://www.eff.org/deeplinks/2010/01/tracking-by-user-agent>.
- [20] J Elliott, A Kor, and OA Omotosho. Energy Consumption in Smartphones: An Investigation of Battery and Energy Consumption of Media Related Applications on Android Smartphones. *International SEEDS conference*, (September), dec 2017. URL: <http://eprints.leedsbeckett.ac.uk/4703/>.
- [21] Roy T. Fielding and Julian F. Reschke. Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. Technical report, 2014. URL: <https://tools.ietf.org/html/rfc7231#section-5.5.3>, arXiv:arXiv:1011.1669v3.
- [22] Qingzhu Gao, Prasenjit Dey, and Parvez Ahammad. Perceived performance of Top Retail web-pages in the wild: Insights from large-scale crowdsourcing of above-the-fold QoE. In *Internet QoE 2017 - Proceedings of the 2017 Workshop on QoE-Based Analysis and Management of Data Communication Networks, Part of SIGCOMM 2017*, volume 17, pages 13–18, New York, NY, USA, aug 2017. Association for Computing Machinery, Inc. URL: <https://dl.acm.org/doi/10.1145/3098603.3098606>, arXiv:1704.01220, doi:10.1145/3098603.3098606.
- [23] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, and Joaquin Quionero Candela. Practical lessons from predicting clicks on ads at Facebook. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2014. doi:10.1145/2648584.2648589.
- [24] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.
- [25] Paul Calvano. Ilya Grigorik, Pat Meenan, Rick Viscomi. Page Weight, 2020. Accessed 2020-12-20. URL: <https://httparchive.org/reports/page-weight>.
- [26] John Wilander. Full Third-Party Cookie Blocking and More — WebKit, 2020. Accessed 2021-03-13. URL: <https://webkit.org/blog/10218/full-third-party-cookie-blocking-and-more/>.
- [27] Garrett A. Johnson, Scott K. Shriver, and Shaoyin Du. Consumer privacy choice in online advertising: Who opts out and at what cost to industry? *Marketing Science*, 39(1):33–51, 2020. doi:10.1287/mksc.2019.1198.
- [28] David G Kleinbaum, K Dietz, M Gail, Mitchel Klein, and Mitchell Klein. *Logistic regression*. Springer, 2002.
- [29] Ron Kohavi and Others. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.
- [30] Martin Koop. *Preventing the Leakage of Privacy Sensitive User Data on the Web*. PhD thesis, Universität Passau, 2021.

- [31] Pierre Laperdrix, Walter Rudametkin, and Benoit Baudry. Beauty and the Beast: Diverting Modern Web Browsers to Build Unique Browser Fingerprints. In *Proceedings - 2016 IEEE Symposium on Security and Privacy, SP 2016*, pages 878–894. Institute of Electrical and Electronics Engineers Inc., aug 2016. doi:10.1109/SP.2016.57.
- [32] Kuang Chih Lee, Burkay Orten, Ali Dasdan, and Wentong Li. Estimating conversion rate in display advertising from past performance data. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 768–776, 2012. doi:10.1145/2339530.2339651.
- [33] Miguel Malheiros, Charlene Jennett, Sneha Patel, Sacha Brostoff, and M. Angela Sasse. Too close for comfort: A study of the effectiveness and acceptability of rich-media personalized advertising. In *Conference on Human Factors in Computing Systems - Proceedings*, pages 579–588, 2012. doi:10.1145/2207676.2207758.
- [34] Wes McKinney. Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, volume 445, pages 51–56. Austin, TX, 2010.
- [35] Wes McKinney. pandas: a foundational Python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, 14, 2011.
- [36] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning Differentially Private Recurrent Language Models. *arXiv*, oct 2017. URL: <http://arxiv.org/abs/1710.06963>, arXiv:1710.06963.
- [37] Prateek Mehta and Prateek Mehta. Introduction to Google Chrome Extensions. In *Creating Google Chrome Extensions*, pages 1–33. Apress, 2016. URL: https://link.springer.com/chapter/10.1007/978-1-4842-1775-7_1, doi:10.1007/978-1-4842-1775-7_1.
- [38] Mozilla. Firefox 69.0, See All New Features, Updates and Fixes, 2019. Accessed 2021-03-13. URL: <https://www.mozilla.org/en-US/firefox/69.0/releasenotes/>.
- [39] S. Muthukrishnan. Ad exchanges: Research issues. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5929 LNCS, pages 1–12, 2009. doi:10.1007/978-3-642-10841-9_1.
- [40] S. Muthukrishnan. AdX: a model for ad exchanges. *ACM SIGecom Exchanges*, 8(2):2–3, dec 2009. URL: <https://dl.acm.org/doi/10.1145/1980522.1980531>, doi:10.1145/1980522.1980531.
- [41] Network Time Foundation. NTP Timestamp Calculations, 2012. Accessed 2021-01-02. URL: [https://www.eecis.udel.edu/\\$sim\\$ills/time.html](https://www.eecis.udel.edu/simills/time.html).
- [42] Andrew Y. Ng. Feature selection, L1 vs. L2 regularization, and rotational invariance. In *Proceedings, Twenty-First International Conference on Machine Learning, ICML 2004*, pages 615–622, New York, New York, USA, 2004. ACM Press. URL: <http://portal.acm.org/citation.cfm?doid=1015330.1015435>, doi:10.1145/1015330.1015435.
- [43] Dong Nguyen, Noah A Smith, and Carolyn Rose. Author age prediction from text using linear

- regression. In *Proceedings of the 5th ACL-HLT workshop on language technology for cultural heritage, social sciences, and humanities*, pages 115–123, 2011.
- [44] Nielsen. Gross online advertising spending in Germany from July 2017 to June 2020 (in million euros), 2020. URL: <https://www.statista.com/statistics/410937/online-advertising-spendings-in-germany/>.
- [45] Seyed Ali Osia, Ali Shahin Shamsabadi, Sina Sajadmanesh, Ali Taheri, Kleomenis Katevas, Hamid R. Rabiee, Nicholas D. Lane, and Hamed Haddadi. A Hybrid Deep Learning Architecture for Privacy-Preserving Mobile Analytics. *IEEE Internet of Things Journal*, 7(5):4505–4518, 2020. arXiv:1703.02952, doi:10.1109/JIOT.2020.2967734.
- [46] Chandra Prakash Patidar and Meena Sharma. An automated approach for cross-browser inconsistency (XBI) detection. In *ACM International Conference Proceeding Series*, volume 21-23-Octo, pages 141–145, New York, NY, USA, oct 2016. Association for Computing Machinery. URL: <https://dl.acm.org/doi/10.1145/2998476.2998496>, doi:10.1145/2998476.2998496.
- [47] F Pedregosa, G Varoquaux, A Gramfort, V Michel, B Thirion, O Grisel, M Blondel, P Prettenhofer, R Weiss, V Dubourg, J Vanderplas, A Passos, D Cournapeau, M Brucher, M Perrot, and E Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [48] Behnam Pourghassemi, Ardalán Amiri Sani, and Aparna Chandramowlishwaran. What-If Analysis of Page Load Time in Web Browsers Using Causal Profiling. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 3(2):1–23, jun 2019. URL: <https://dl.acm.org/doi/10.1145/3341617.3326142>, doi:10.1145/3341617.3326142.
- [49] Publications Office of the European Union. REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL, 2016. URL: <https://eur-lex.europa.eu/eli/reg/2016/679/oj>.
- [50] Ihsan Ayyub Qazi, Zafar Ayyub Qazi, Theophilus A. Benson, Ghulam Murtaza, Ehsan Latif, Abdul Manan, and Abrar Tariq. Mobile web browsing under memory pressure. *Computer Communication Review*, 50(4):35–48, oct 2020. URL: <https://dl.acm.org/doi/10.1145/3431832.3431837>, doi:10.1145/3431832.3431837.
- [51] Lianying Qi, Ruili Wang, Chunhua Hu, Shancang Li, Qiang He, and Xiaolong Xu. Time-aware distributed service recommendation with privacy-preservation. *Information Sciences*, 480:354–364, 2019. doi:10.1016/j.ins.2018.11.030.
- [52] Rui Qin, Yong Yuan, and Fei Yue Wang. Exploring the optimal granularity for market segmentation in RTB advertising via computational experiment approach. *Electronic Commerce Research and Applications*, 24:68–83, jul 2017. doi:10.1016/j.eelerap.2017.07.001.
- [53] Harrison Rainie and Barry Wellman. *Networked: The new social operating system*, volume 419. Mit Press Cambridge, MA, 2012.
- [54] Paruj Ratanaworabhan, Benjamin Livshits, and BG Zorn. JSMeter: Comparing the behav-

- ior of JavaScript benchmarks with real web applications. *Conference on Web applications*, 2010. URL: http://www.usenix.org/event/webapps10/tech/full_papers/Ratanaworabhan.pdf.
- [55] Pradeep Ravikumar, Martin J. Wainwright, and John D. Lafferty. High-dimensional Ising model selection using ℓ_1 -regularized logistic regression. *The Annals of Statistics*, 38(3):1287–1319, jun 2010. doi:10.1214/09-AOS691.
- [56] Ignacio Redondo and Gloria Aznar. To use or not to use ad blockers? The roles of knowledge of ad blockers and attitude toward online advertising. *Telematics and Informatics*, 35(6):1607–1616, sep 2018. doi:10.1016/j.tele.2018.04.008.
- [57] Alexey Reznichenko and Paul Francis. Private-by-design advertising meets the real world. In *Proceedings of the ACM Conference on Computer and Communications Security*, pages 116–128. Association for Computing Machinery, nov 2014. doi:10.1145/2660267.2660305.
- [58] Matthew Richardson, Ewa Dominowska, and Robert Ragno. Predicting clicks: Estimating the click-through rate for new ads. In *16th International World Wide Web Conference, WWW2007*, pages 521–530, 2007. doi:10.1145/1242572.1242643.
- [59] Jeffrey Rosen. "Who Do Online Advertisers Think You Are?", 2012. URL: <https://www.nytimes.com/2012/12/02/magazine/who-do-online-advertisers-think-you-are.html>.
- [60] Marta C. Soares. The Neurobiology of Mutualistic Behavior: The Cleanerfish Swims into the Spotlight. *Frontiers in Behavioral Neuroscience*, 11:191, oct 2017. URL: <http://journal.frontiersin.org/article/10.3389/fnbeh.2017.00191/full>, doi:10.3389/fnbeh.2017.00191.
- [61] Socintel360. Real time bidding digital display advertising spending in selected countries in 2014 and 2018, 2014. URL: <https://www.statista.com/statistics/377658/rtb-digital-display-ad-spend-selected-countries/>.
- [62] Jill C. Stoltzfus. Logistic Regression: A Brief Primer. *Academic Emergency Medicine*, 18(10):1099–1104, oct 2011. URL: <http://doi.wiley.com/10.1111/j.1553-2712.2011.01185.x>, doi:10.1111/j.1553-2712.2011.01185.x.
- [63] Andrzej Szwab, Pawel Misiolek, and Michal Ciesielczyk. Logistic regression setup for RTB CTR estimation. In *ACM International Conference Proceeding Series*, volume Part F1283, pages 61–70, New York, NY, USA, feb 2017. Association for Computing Machinery. URL: <https://dl.acm.org/doi/10.1145/3055635.3056584>, doi:10.1145/3055635.3056584.
- [64] Ashley Tate. Why Marketers Should Care About UX, 2013. Accessed 2020-12-26. URL: https://www.thinkwithgoogle.com/_qs/documents/2698/53dff_Why-Marketers-Should-Care-About-Mobile-Page-Speed-EN.pdf.
- [65] Solon Toubiana, Vincent and Narayanan, Arvind and Boneh, Dan and Nissenbaum, Helen and Barocas. Adnostic: Privacy Preserving Targeted Advertising. 2010.
- [66] Randika Upathilake, Yingkun Li, and Ashraf Matrawy. A classification of web browser fingerprint-

- ing techniques. In *2015 7th International Conference on New Technologies, Mobility and Security - Proceedings of NTMS 2015 Conference and Workshops*. Institute of Electrical and Electronics Engineers Inc., sep 2015. doi:10.1109/NTMS.2015.7266460.
- [67] Blase Ur, Pedro Giovanni Leon, Lorrie Faith Cranor, Richard Shay, and Yang Wang. Smart, useful, scary, creepy: Perceptions of online behavioral advertising. In *SOUPS 2012 - Proceedings of the 8th Symposium on Usable Privacy and Security*, page 1, New York, New York, USA, 2012. ACM Press. URL: <http://dl.acm.org/citation.cfm?doid=2335356.2335362>, doi:10.1145/2335356.2335362.
- [68] Jun Wang and Shuai Yuan. Real-Time Bidding. pages 415–416. Association for Computing Machinery (ACM), feb 2015. doi:10.1145/2684822.2697041.
- [69] Jun Wang, Weinan Zhang, and Shuai Yuan. Display advertising with real-time bidding (RTB) and behavioural targeting. *Foundations and Trends in Information Retrieval*, 11(4-5):297–435, 2017. arXiv:1610.03013, doi:10.1561/15000000049.
- [70] Yu Wang, Jiayi Liu, Yuxiang Liu, Jun Hao, Yang He, Jinghe Hu, Weipeng P. Yan, and Mantian Li. LADDER: A human-level bidding agent for large-scale real-time online auctions, aug 2017. arXiv:1708.05565.
- [71] Musen Wen, Zhen Xia, Deepak Kumar Vasthimal, João Vinagre, Alípio Mário Jorge, Albert Bifet, and Marie Al-Ghossein. Practical Lessons from Predicting New User Demographics for Ad Targeting. Technical report, oct 2019. URL: <http://proceedings.mlr.press/v109/wen19a.html>.
- [72] Jaap Wieringa, P. K. Kannan, Xiao Ma, Thomas Reutterer, Hans Risselada, and Bernd Skiera. Data analytics in a privacy-concerned world. *Journal of Business Research*, (July 2018):0–1, 2019. doi:10.1016/j.jbusres.2019.05.005.
- [73] Wush Chi Hsuan Wu, Mi Yen Yeh, and Ming Syan Chen. Predicting winning price in real time bidding with censored data. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 2015-Augus, pages 1305–1314. Association for Computing Machinery, aug 2015. doi:10.1145/2783258.2783276.
- [74] Chaoqi Yang, Junwei Lu, Xiaofeng Gao, Haishan Liu, Qiong Chen, Gongshen Liu, and Guihai Chen. MoTiAC: Multi-Objective Actor-Critics for Real-Time Bidding. *arXiv*, feb 2020. URL: <http://arxiv.org/abs/2002.07408>, arXiv:2002.07408.
- [75] Shuai Yuan, Jun Wang, and Xiaoxue Zhao. Real-time bidding for online advertising: Measurement and analysis. In *Proceedings of the 7th International Workshop on Data Mining for Online Advertising, ADKDD 2013 - Held in Conjunction with SIGKDD 2013*, pages 1–8, New York, New York, USA, 2013. Association for Computing Machinery. URL: <http://dl.acm.org/citation.cfm?doid=2501040.2501980>, arXiv:1306.6542, doi:10.1145/2501040.2501980.
- [76] Weinan Zhang, Shuai Yuan, Jun Wang, and Xuehua Shen. Real-Time Bidding Benchmarking with iPinYou Dataset. jul 2014. URL: <http://arxiv.org/abs/1407.7073>, arXiv:

1407.7073.

- [77] Jun Zhao, Guang Qiu, Ziyu Guan, Wei Zhao, and Xiaofei He. Deep reinforcement learning for sponsored search real-time bidding. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1021–1030, New York, NY, USA, jul 2018. Association for Computing Machinery. URL: <https://dl.acm.org/doi/10.1145/3219819.3219918>, arXiv:1803.00259, doi:10.1145/3219819.3219918.