Flowr

Enhancing Dynamic Marketing Audience Creation

C.J.H Bilstra
S. Koppers
F.E.C. List
R. Safarpour Erfani

Technische Universiteit Delft

TU Delft

Flowr

Enhancing Dynamic Marketing Audience Creation

by

C.J.H Bilstra
S. Koppers
F.E.C. List
R. Safarpour Erfani

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended publicly on Tuesday July 4, 2017 at 16:00 PM.

Project duration:	April 24, 2017 – June 27, 2017	
Thesis committee:	Dr. N. Schenk,	Annalect (client)
	Dr. ir. C. Lofi,	TU Delft, supervisor
	Dr. ir. O. Visser,	TU Delft, coordinator
	Dr. ir. H. Wang,	TU Delft, coordinator

This thesis is public.

TU Delft

Preface

This report has been written in the months of May and June of 2017 during a 10-week project at the Technical University of Delft. This project was executed as part of our bachelor thesis for the bachelor of Computer Science and Engineering, with the goal of creating and end-to-end product from scratch. We would like to thank our supervisors Christoph Lofi (TU Delft) and Niels Schenk (Annalect) for helping us throughout the project.

C.J.H Bilstra
S. Koppers
F.E.C. List
R. Safarpour Erfani
Delft, July 6, 2017

Summary

Omnicom Media Group (OMG) is a company heavily involved in marketing and advertising. Our client is Annalect, a solutions provider that helps the marketers of OMG to make data actionable. OMG has processed cookie data to help their marketers set up advertisement campaigns. They buy this cookie data from a 3rd party. They also manage, however, a vast amount of cookie data themselves, which is currently partly unused. In order for the marketers to use this data, Annalect needs to process and prepare database views for them. To let the marketers, who have no database knowledge, be able to manipulate these views, they create dashboards with 3rd party software called Tableau. Annalect wants us to create an application in which they can set up these dashboards for the marketers so they can manipulate the cookie data and use it for their advertisement campaigns.

As a result we have created a web application which supports the workflow of the marketer. After some setup by the people from Annalect, a marketer can sign into our application, choose a dashboard, start working with the cookie data and send the manipulated data off to create an advertisement campaign. All this is done without leaving our application. The data the marketers manipulate in our application is just a small snippet of the complete data set. Since the complete data set contains much more data, it needs to be processed by server clusters paid for by Annalect. This processing is done at night, in order to cut the cost of running the server cluster. After the processing, the result has to be sent to Google DoubleClick Campaign Manager. Furthermore, the ability to use machine learning algorithms was requested by Annalect. This has been implemented through a generic pipeline, which supports multiple machine learning models. A model based on gradient boosting is included as a proof of concept.

In order to evaluate the application some tests were done. Different aspects need different tests. Firstly, a usability test was performed with the end users to test the User Interface. Secondly, unit tests were made where it was applicable. Lastly, the machine learning model was evaluated using the recall precision method and K-fold cross validation method.

The application has some aspects which have ethical interest. Managing vast amounts of cookie data needs to be done discretely as personal information can be derived from such data. Having cookie data leak can cause damage to individuals who supplied this data. Besides that, the right to explanation law coming into effect next year will force companies to explain why their computer models made certain decisions or classifications. This has implications for machine learning models used by our application. And finally, the users of the program have to be aware that they are using sensitive data about individuals, which they have to act upon accordingly.

Contents

1 Introduction

Advertisements play a significant role on the Internet. Almost every free to access website has them and they are essential in keeping these websites freely available. Web advertisements are quite different from classical advertisements such as in a paper or on TV. In classical media, advertisers know about the general target audience of the surrounding media and, based on that, decide when and where they would want to show an advertisement. An example of this would be a toys shop playing advertisements on a kids TV channel or a car manufacturer displaying an ad about their new car in an autosports magazine. With web advertisements, advertisers actually get significantly more information about each individual and are thus able to target ads more precisely to specific individuals instead of a general target audience. Cookies can track a browser by giving it a unique ID and keep track of what websites the user visits and the ads the user has been served. This is exactly what Google is doing with their DoubleClick [5, 6] cookies. This information alone isn't worth much, but keep track of it for long enough with a great number of cookies and you can start to see patterns within all the cookie data. DoubleClick uses these patterns to assign consumers to segments. DoubleClick segments can be things like football fans, car lovers, dog owners, etc. With Google AdSense, advertisers can use these segments and a bunch of other data linked to a cookie ID to determine to who they want to display ads.

One of these advertisers is Omnicom Media Group, a company that is heavily involved in the production and marketing of advertisement campaigns. Part of this group is a company called Annalect. The role of Annalect within Omnicom Media Group is to provide solutions and help marketers make data actionable. Annalect's services and technology cut through the fragmented marketing ecosystem and put the right data at the fingertips of marketing teams.

This report will first provide a problem description and analysis for a problem Annalect encountered. Next we will describe the process we followed during this project. After that, the implementation of our solution will be given followed by an evaluation of the final product. The ethics involved in this project are examined and a discussion is made about the deviations from the plan and recommendations for the future. Lastly a glossary is given, which explains all the terms used in this report.

The application that was build in this project has been given the name Flowr and will henceforth be called so in this report.

2

Problem Description

This chapter introduces the client company and describes the problem our application will solve.

2.1. Company description

This project will be carried out in assignment of Annalect, the technical department of Omnicom Media Group (Omnicom), a marketing and advertisement firm. One of the services Omnicom provides to their clients is buying advertisement spaces on websites. Advertisement spaces are being bought on websites and when some individual visits that website, an advertisement is shown to that individual. The goal is to have a recipient of a marketing message perform a desired action. This is called a conversion. Desired actions in this context can be a whole lot of different things. For example, having the recipient click on the marketing message or subscribe to a mailing list are two common desired actions. Omnicom runs an advertisement campaign for a client in which this process is applied. For example, a client gives orders to promote a new car model. Whilst very inefficient, it is possible to put this advertisement on a web page and show it to all its visitors. A 10-year old boy playing games on the internet will never be able to buy a new car though. The task of Omnicom is thus to select a group of suitable individuals for a certain advertisement campaign, this is called targeting. This group of people is called an audience. Every campaign has a different audience, selected by professionals: the display consultants. They currently use their knowledge and third party data and software to come up with a suitable audience for a certain campaign. Sometimes, data analysts are needed in order to construct a good audience. Data analysts have a good understanding of how they should search for specific characteristics in data and can help the display consultants by narrowing down the search space for an audience. They do this by making a query on the data, specific to the characteristics desired by the display consultants. By doing this, they create dashboards that reflect the data in the database. The display consultants can then further filter the data using the created dashboards to get the audience they deem fit. This process is called filtering.

2.2. Problem

There is a total of three problems that this project is going to solve for the client. Each problem will be discussed in its own paragraph.

The data currently in use by the display consultants to construct their audience is bought from a third party, which is very costly. This third party delivers the data with their software from which the display consultant is able to create the desired audience. However, Omnicom also manages a vast amount of cookie data. This cookie data is currently underused. The data that Omnicom manages is prepared by their data analysts, who are now able to structure the data and write database queries that will return database views as a result, with the aim to narrow down the search-space of the data. The problem here is that the data analysts and display consultants still work separately. The display consultants cannot yet use the views the data analysts provide to select their audiences. If they would work together this could save a lot of money as third party data does not need to be bought anymore, this leads to the next problem.

If Omnicom wants to use the data they manage, they will need to have data analysts process the data in order for it to be of any use for the display consultants. The display consultants do not have the proper knowledge to process the unprocessed data themselves, which is why data analysts are needed. As the cooperation between the display consultants and data analysts is non-existent at this moment, a new workflow needs to be set up in order for the two groups to work together. The data analysts need to have the ability to create database views and have those visualised in an interface. Display consultants can then further filter the data provided by the visualisation. There should be multiple of those interfaces based on different database views as created by the data analysts. This workflow improves the communication between both the display consultants and analysts which leads to improved quality of either party their work.

Even with a new workflow in place it is still costly to have both data analysts and display consultants do their labour intensive work. Therefore, a modern technique like machine learning algorithms could come to aid in the process of constructing audiences. Such a machine learning algorithm could do the selection of audiences itself, based on a system that can predict the conversion of an individual. When these models perform adequate, they could add to the quality of the created audiences and theoretically even replace the work of display consultants.

The machine learning requirement is more of an extra and a whole different project in and of itself. The main problem to deal with, is that there are three components that need to be brought together in order for the display consultants to effectively work with the underused cookie data. These components are the cookie data itself, a way of filtering the cookie data and to then send it to Google. The problem solution will cover how we achieved to have all these components in one place.

2.3. Problem solution

A web application has been built that supports the new workflow as stated in the problem description. The reason for a web application, is that the program has to be easily accessible to a lot of users. As almost any device has access to a web browser these days, users can use the program on any preferred device. Support for small (mobile) screens is not implemented, as the current users will only use big screens while working with the program. This can easily be adjusted if necessary in the future. Another reason why a web application is very suitable is that creating a user interface is fairly easy. Also, third party software we are using in the program, such as tableau, also offers functionality to be embedded into websites. For more information on technical choices, please refer to the research report in Appendix B.

This application will support all the needs of both the data analysts and display consultants. It will function as a tool that connects the work of both parties. The data analysts will first provide a database view on the cookie data Omnicom manages and then create a dashboard in a software package called Tableau. Tableau is a data analysis platform in which a user can make dashboards that show specific details about provided data, see section 4.1.5. The dashboard is a user interface with different components that display different aspects of the provided data (the amount of advertisements shown on a specific website, for example). It then offers the possibility to let the user filter the data to their liking. This filtering narrows down the amount of entries in a resulting set from the database view, thus making it a more specialised set that is supposed to fit the real world audience best. The whole process of creating a dashboard and adding the correct components to the dashboard that can filter the database view will be done by the data analysts. The dashboard created by the data analysts will use the data present in the database view, which the data analysts also construct. This dashboard is going to be integrated into the application enabling the display consultants to filter the provided database view further. Our application will capture those filters and run those captured filters over the respective database view when the display consultants are content with the filters they added over the data. The results will be scheduled to be sent to Google systems at a configurable interval decided by the display consultants.

A diagram displaying the solution can be found in figure 2.1. The goal is for the application to provide

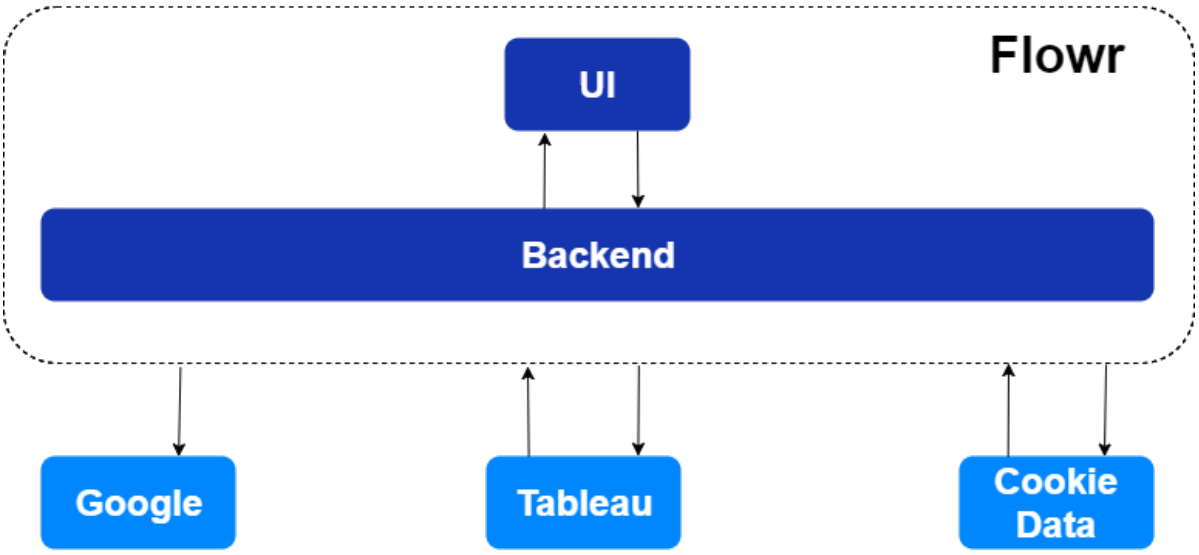


Figure 2.1: Problem our application is going to solve

an environment in which all components work together.

Our application supports, besides the above described workflow, pluggable machine learning models. A data analyst is able to provide a machine learning model trained to predict a conversion. This functionality can improve the quality of the audiences created. Those models can in theory outperform the display consultants. Our application will provide the pipeline necessary to properly execute machine learning models and contain at least one machine learning model we build ourselves as a proof of concept. It is, however, possible for a data analyst to add more machine learning models to the application.

3

Process

In this chapter the process the team followed will be discussed. This concerns topics like weekly meetings, planning and development tools used. Although plans have been made in the first phase of the project, plans can change. So beside the actual plans, we will also discuss what went wrong, how we adapted to the unforeseen changes and reflect on what happened.

3.1. Development Methodology

For the development methodology, we decided to use SCRUM. SCRUM is an agile development framework consisting of iterative periods of time in which tasks are assigned and finished, called sprints. Usually sprints are biweekly, but because of the short duration of the project we decided on sprints of 1 week. We had a scrum-meeting every Monday and discuss what has been done and still needs to be done. From this information we created tasks for the week and construct a sprint around that.

Reflection Although SCRUM is great for keeping track of tasks that still need to be done and help give an overview of the project, this only works when sprints and the backlog are actively updated. This is something we stopped doing very early in the project. A reason for this could be that the application had very distinct parts (discussed in the next chapter), and each team member had a good overview of the part(s) they were working on. This meant keeping SCRUM updated was not as important anymore. We did, however, still have weekly sprint meetings in which we discussed what to do for the week and what had been done in the past week. These meetings did help a lot in working on the right tasks and keeping the client company up-to-date.

3.2. Development Tools

In this section the tools we have chosen to work with will be discussed.

3.2.1. Version Control & SCRUM

To make sure all members can easily share their work amongst each other, version control is needed. As Annalect already uses the Bitbucket & JIRA stack, it was the most convenient choice for us, as we could then make use of the board they use themselves.

3.2.2. Deployment

As mentioned in the previous chapter, the solution is a web application, so we needed to run a server on our local machines.To ensure that everything works on all machines, Docker[4] was used to deploy our code to a virtual environment. The best explanation of Docker we found was in an article on Linux.com: ”Docker is a tool that can package an application and its dependencies in a virtual container that can run on any Linux server. This helps enable flexibility and portability on where the application can run, whether on premise, public cloud, private cloud, bare metal, etc.”[22]

3.3. Communication

Every Monday we worked at Annalect in Amstelveen and every Wednesday and Thursday was spent at the Technical University Delft, usually in a reserved project or meeting room. On the other days and weekends, work was performed at home. Between sessions we would communicate either through WhatsApp with each other or through HipChat to also include our clients. HipChat is a teamchat application and is also part of the Atlassian stack together with Bitbucket and JIRA. Annalect already uses HipChat for communication within the company so it was easy for us to join the chat.

Table 3.1: Original Roadmap

Week	Summary	Expected work	Deliverables	Report (chapter)
1 & 2	Research	Write both the project plan and research report	Project plan & research report	-
3	Setup platform	Software and frameworks should be running on all developer systems. Start on the user interface.	Basic user interface and working communication with user database. Scheduler on the server should be working.	Introduction and problem description
4	Development of the platform	User interface and link with external systems tested and finished.	Working user interface.	Implementation
5	Functional program	Data scientists and display consultants are satisfied with the program as is.	Program which works for the users.	Process
6	Switching to machine learning	Finishing touch on the program, start with machine learning with proof of concept.	Final release of the main program excluding machine learning functionality.	Evaluation & start with machine learning chapter
7	Machine learning	Provide working models, performance not yet important.	Test set, validation set, training set for the machine learning.	Backup week for other chapters
8	Machine learning	Improve the performance of the machine learning algorithms. Prove usability.	Machine learning models	Conclusion & discussion
9	Finishing	Assure smooth integration of machine learning with the rest of the program, ensure modularity. Assess safety of the software	Final release of the main program including machine learning functionality.	Final checks.

3.4. Testing

A big part of our application is the user interface which means there is a lot of manual testing. This implies we have walked through our application countless times to make sure the buttons do what they are supposed to do. In addition to this near the end of the project a usability test was performed on the actual end-users of the product.

There are areas, however, where unit testing is more valuable. Ensuring the query is built correctly is of utmost importance and is something that can easily be tested in isolation. So for this component Unit Tests have been build. The tests including results are more deeply covered in chapter 5.

3.5. Project Plan

In the research phase of the project a project plan was made. In this plan a roadmap was constructed which contains, per week, what would be worked on and what would be delivered. A copy of this roadmap can be found in table 3.1.

Reflection The project plan certainly served us well in reminding us of what we had to do. However, actually keeping up with the planning was a bit difficult. We fell behind on our planning in the third week. In this week, we had to start building our application, but we still did not really know what to actually build. In the research phase we were trying to figure out what Annalect wanted from us. So we would sit down with them, have a talk, ask questions and from that information construct a diagram of possible workflows of the application. However, each time we came up with a general idea of the application, when discussing it with Annalect, it turned out to be not exactly what they were looking for. Following this we would get some new information, asked more questions and started thinking of ways to incorporate their requirements into an application. This process took us 3 weeks, which lead to us being a week behind very early in the project.

We were, however, able to catch up in the following weeks and after week 6 we were back on schedule again. After a discussion with both our supervisors, we shifted our focus. From there we spent more time on the report and decided to not implement a machine learning model ourselves as this would take too much time. Instead, we started using a library for machine learning. We also extended the work on the user interface. Week 9 was still going as planned and both the report and the code were delivered at the end of week 9. We will potentially tweak some of the machine learning parts in week 10, but we will not discuss this as this report is already handed in by then.

4

Implementation

This chapter explains the details about how the problem solution from section 2.3 was implemented. This is done by giving an overview of the software architecture and discussing the data flow. Some third party software was used and it will be explained why and how it was used. Finally, the implementation of the machine learning is discussed.

4.1. Architecture

For an overview of the architecture, please have a look at figure 4.1. The program starts with the user interface, where the users of the program can put, edit and use everything they need. The Flowr database will be used to store user information. The user can then either use Tableau for manual control or choose to make use of a machine learning algorithm, which will bypass Tableau. On the server, everything will be prepared and sent to the scheduler. The scheduler is responsible for running the chosen method on all data present at Annalect. This data is maintained in the Annalect database. When the database has finished running the query, the resulting audience will be sent to the DoubleClick Campaign Manager (DCM). DCM is a Google service to which you can send an audience to whom your advertisements will be shown. More information about DCM can be found in section 4.1.6. In the following sections, the systems present in the architecture will be discussed in more detail.

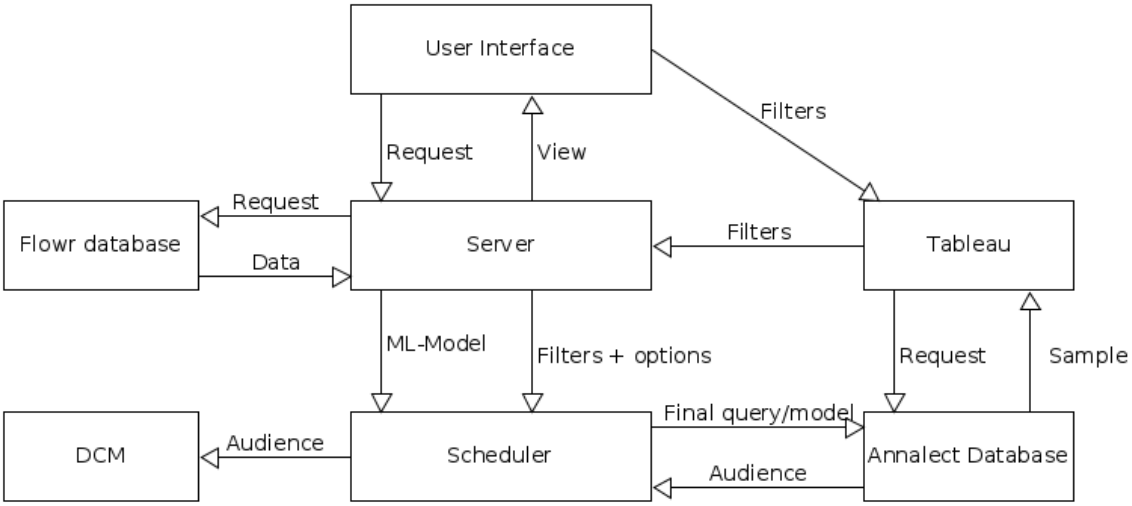


Figure 4.1: Overview of the software architecture

4.1.1. User interface

As Flowr is a web-based application, HTML, CSS and JavaScript are used for the user interface. The user interface is responsible for allowing input into the program. For the different users of the program, different functionality is necessary. Admins must have the access to user management, so a page for user management is only accessible to them. Admins can furthermore see all pages, also those which are only intended for specific roles. Individual users are only able to adjust their own credentials, such as their username and password. Data analysts can access a pages where they are able to add new dashboards or machine learning models to Flowr. These dashboards or models are then used by the display consultants. Data analysts can also adjust all dashboards they created themselves. For display consultants there is a view where they can see the dashboards data analysts have given them access to from where they can visualise those in Tableau.

For all input, forms are created with input validation. Using these forms, it is easy to create the objects needed for the database. Jinja, an HTML templating framework [7], is used to create HTML pages easily and in a modular way. For CSS, the materialize CSS framework [11] is used to ensure good-looking elements out of the box. Jquery [8] is used to make the JavaScript code less complex.

4.1.2. Server

The server functions as the backbone of the application. It fulfils different tasks that ensure correct functioning of the application as a whole. The server is in contact with the database and handles all the insertions, edits and deletions of data as requested by the client. Furthermore, the server makes sure that every user is only able to do what his or her role allows within the structure of the system, see also section 4.1.3. Besides that the server takes care of the routing of the application. This means that all the requests from a client to the server are handled correctly and result in the correct rendering of the views that are sent back to the client.

4.1.3. User management

It must be known which user is executing some action and for which users a certain action is permitted, as not all users are allowed to perform the same actions. To make this possible, user management is necessary and was implemented using the Flask-Login library. This library enables easy access restriction for user-roles and specifies a good back-end for saving users in the database. All users are able to change their credentials and password. Admins can change their own credentials as well as the credentials of all other users. Because we used a library for handling user management, it was merely a few lines of code to implement.

4.1.4. Databases

There are two databases in use by Flowr, one for the Flowr program itself, and one which is maintained by Annalect. The following paragraphs explain what each database is used for.

Flowr database The Flowr database is used to save user information and all information necessary for dashboards and machine learning models to work. It is a PostgreSQL database, for more information on why PostgreSQL is being used, please refer to the research report in Appendix B.

Annalect database The Annalect database is an Apache Parquet database on Amazon S3 (Simple Storage Service) which stores data as objects within resources called ”buckets” . It contains the parquet files that result from the data transfer files, logs etc. In addition, there is a PostgreSQL database that contains all information necessary for Airflow to create schedules (let us call it definitions database for future references). This database basically contains meta-data for Airflow on where the plugin should retrieve its data from, with what credentials and what table to persist it to. It’s halfway out of the scope of this report which is why we will not go into much more detail.

4.1.5. Tableau

In order to help display consultants visualise and manipulate the cookie data, Tableau [14], a third party software application is used. Tableau is being displayed through an iFrame which is created with the Tableau JavaScript API. Data analysts can create dashboards in Tableau, which generates graphs from cookie data uploaded to the Tableau server. An example of a visualisation can be seen in figure

4.2. As the cookie dataset consist of millions of entries, having Tableau process all of it would take too much time. Instead a sample of the data is uploaded to Tableau. This sample is big enough to still give a good representation of the total data. Display consultants can open a dashboard in Tableau and click on the graphs to filter and create a selection. An example of this would be to only include those with a high click through ratio (CTR) or only those who are part of a certain DoubleClick segment. These segments are especially important as this is the way DoubleClick categorises its cookies into subsets who share a common interest. Consultants can use a Tableau dashboard to quickly filter data and get feedback on their selection. Once the consultant is happy with his or her selection, the results can be downloaded from Tableau. However, because only a sample of the cookie data is used by Tableau, the downloaded data is not representative of the full dataset. The filters that are applied need to be extracted and applied on the full dataset in the Annalect database. As it is not possible to simply extract a query from the Tableau dashboard, we need to build those queries ourselves based on the filters in Tableau. To achieve this, at the creation of a dashboard, a data analyst needs to configure which Tableau filters apply to which columns in the Annalect database. With this information, we can extract the current filters from the Tableau API, save them in the Flowr database and convert them to a query which can be run on the Annalect database. This conversion from filters to a query takes place in the scheduler, which is more thoroughly discussed in section 4.1.7.

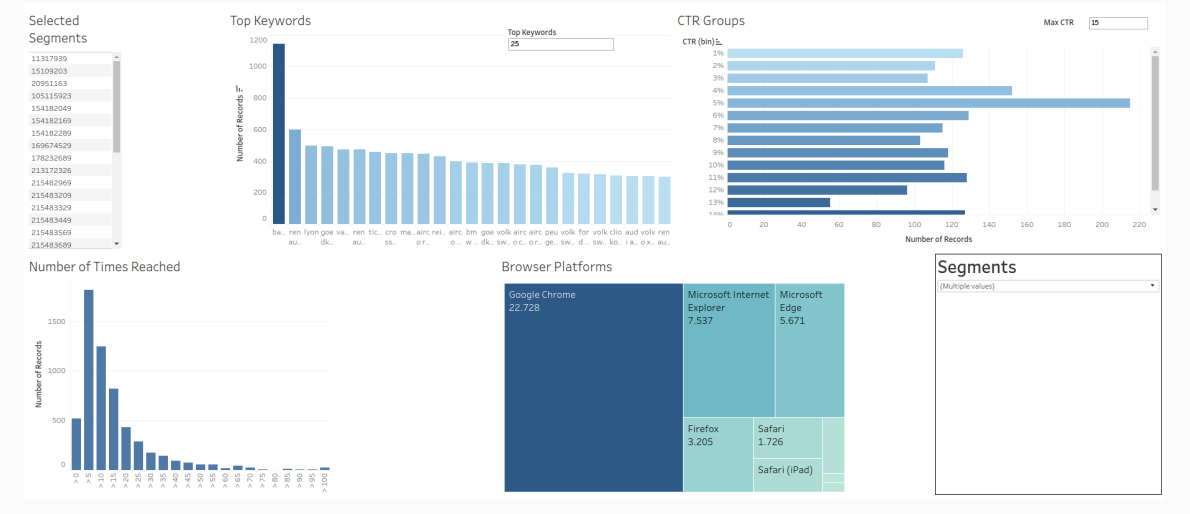


Figure 4.2: Example of a Tableau dashboard

4.1.6. Third party systems

Flowr uses several third party systems to enhance and/or facilitate the flow of the program. These third party systems all have their own purpose and responsibility. All third party systems are necessary for our application to function. This section describes the different systems used and what their function is in the program. For a more detailed explanation of how these systems are incorporated in Flowr please refer to 4.1.7.

Google DoubleClick Campaign Manager Google DoubleClick Campaign Manager (DCM) is as the name suggests, a platform for integrated campaign management. It incorporates the whole process from media-planning to trafficking and reporting where:

- Media-planning: a discipline that dedicates itself to finding the right advertisement to reach a certain target audience
- Trafficking: process of supplying campaign creative materials and tracking links to publishers (websites that display ads)
- Reporting: reporting of the results

We use DCM in two different ways. The first case is collecting data using Data Transfer files. Data Transfer files provide impression, click, rich media, and floodlight activity data in a log level format where:

- Impression: when someone is able to see the advertisement
- Click: when someone clicked the advertisement
- Rich media: advertisement that includes advanced features like video, audio or other elements that encourage viewers to interact.
- Floodlight: an invisible pixel on a webpage that fires a call when loaded, so once someone visits that webpage
- Floodlight activity data: data that logs specific customer actions such as the completion of a purchase or a visit to a page.

Match tables files are used to reduce individual file size while still providing robust data [3]. Please check the glossary for a more detailed explanation of the terms used. The second case is sending an audience to DCM using Airflow. DCM will then use this audience to select to which individuals at some web page certain advertisements should be shown.

Amazon Web Services Athena Amazon Athena is an interactive query service that makes it easy to analyse data in Amazon S3 using standard SQL. Athena is serverless, so there is no infrastructure to manage, and you pay only for the queries that you run [2]. Athena is used to create audiences from the filters created by the data consultants and send it to the DoubleClick Campaign Manager. It is able to do so by running a query that has been created using Flowr and Airflow.

Annalect uses more third party systems to collect data. However, Flowr does not interact with these third party systems and will therefore not be mentioned explicitly.

4.1.7. Scheduler

In computing, scheduling is the method by which work specified by some means is assigned to resources that complete the work [23]. The scheduler handles several processes that would otherwise incorporate a lot of management and administration costs, take ETL (Extraction, Transformation and Load) processes for example, where each phase is defined by:

- Extraction: retrieve data from a source
- Transformation: transform data according to rules to conform to database standards and to manufacture usable data
- Load: write the data to a database located somewhere different than the source

If someone was to repeatedly schedule all these processes by hand, one were to keep an agenda at all times, not mentioning human errors that are likely to occur. We are dealing with databases that will be skewed if a process fails and inserts data twice or not inserts certain parts of data at all. Also, as these processes grow, the complexity to manage these processes by hand increases. It is evident that a scheduler is a mandatory piece of software regarding processes that need to be executed daily in an ordered and fail-safe fashion.

Flowr uses an instance of the Airflow Framework [15] as the scheduler. Airflow is used to author workflows as directed acyclic graphs (DAGs) of tasks. The Airflow scheduler executes tasks on an array of workers while following the specified dependencies. The user interface makes it easy to visualise pipelines running in production, monitor progress, and troubleshoot issues when needed [15]. For more information on why Flowr uses Airflow please refer to our research report in appendix B.

Airflow uses a plugin system where developers can extend the airflow framework with their own code using plugins. Annalect has written several plugins to collect data using ETL processes. The plugins programmatically describe how to collect data: Authorise with the external system, download the data using the system's API, extract and transform the data to conform with the Annalect database,

and load it into the Annalect database. This database is required to be updated with new data on a daily basis.

Flowr uses the scheduler in quite a straightforward manner. It actually persists a tuple into the Annalect definitions database which was described in 4.1.4. This tuple contains the filters selected, the dashboard that is used and information for the scheduler e.g.: the schedule interval (amongst some other parameters that are too low-level to mention here). Once a display consultant has selected his or her filters in Tableau based on the dashboard used, he or she then submits this data to be inserted. As airflow continuously polls for new tuples in its database, Airflow immediately discovers this new tuple and schedules it using our self-made AwsAthenaPlugin (Airflow plugin). This plugin translates the filters of a Tableau dashboard to a query which can be executed on the Annalect database. It parses the query on which the dashboard was based and transforms it together with the selected filters into an SQL statement. This query is subsequently sent to AWS Athena which returns an array of DoubleClick cookie ids. These ids are then in turn sent to the DoubleClick Campaign Manager using our self-made GoogleDCMPlugin (Airflow plugin). The whole process can be seen in flow diagram 4.3.

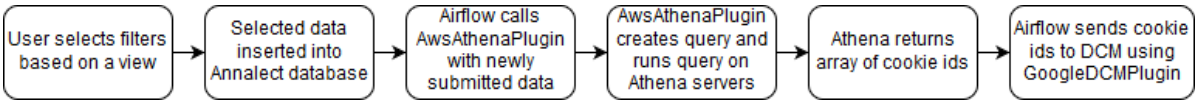


Figure 4.3: Overview of the default flow

As for the machine learning flow, the flow is nearly identical except for the part where Athena is used. Athena is being traded for Spark. The resulting flow can be seen in flow diagram 4.4.

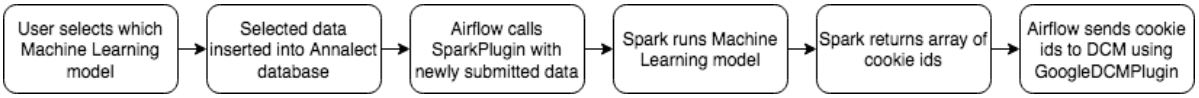


Figure 4.4: Overview of the machine learning flow

4.2. Data flow

This section gives an overview of the data flow in our application, which is visualised in figure 4.5. The life cycle starts with an admin creating accounts for the data analysts and the display consultants who will use the program. The data analyst then creates a dashboard to be used by the display consultants. The display consultant can then log in to the program and will see an overview of dashboards assigned to him or her. A dashboard can be re-used through different campaigns, the current estimate is that 6 to 8 dashboards will be available to each display consultant. The display consultant now has two choices: filter the desired audience manually or use a predefined machine-learning model.

Filtering If the display consultant chooses to filter the dashboard manually, Tableau will be opened and the display consultant can filter until he or she is content with the filtered result. The display consultant also has to choose the options for the scheduler. Flowr then translates the filters to JSON, which the scheduler handles. The scheduler combines the filters with the original query on which the dashboard was created to create the final query. This final query will be run at the intervals selected by the display consultant over the complete data set present in the Annalect database. After the query has finished, the result is an audience which will be sent to DCM. DCM then serves the advertisement of the campaign to the targeted individual.

Machine Learning If a display consultant chooses to use the machine-learning model, he or she can directly select which model to use and the parameters for the scheduler, which will all be passed to the scheduler. The scheduler then runs the machine-learning model at the specified interval to create an audience of a certain size, which is also specified. Once the audience has been created it is sent to DCM. DCM then serves the advertisement of the campaign to the targeted individual.

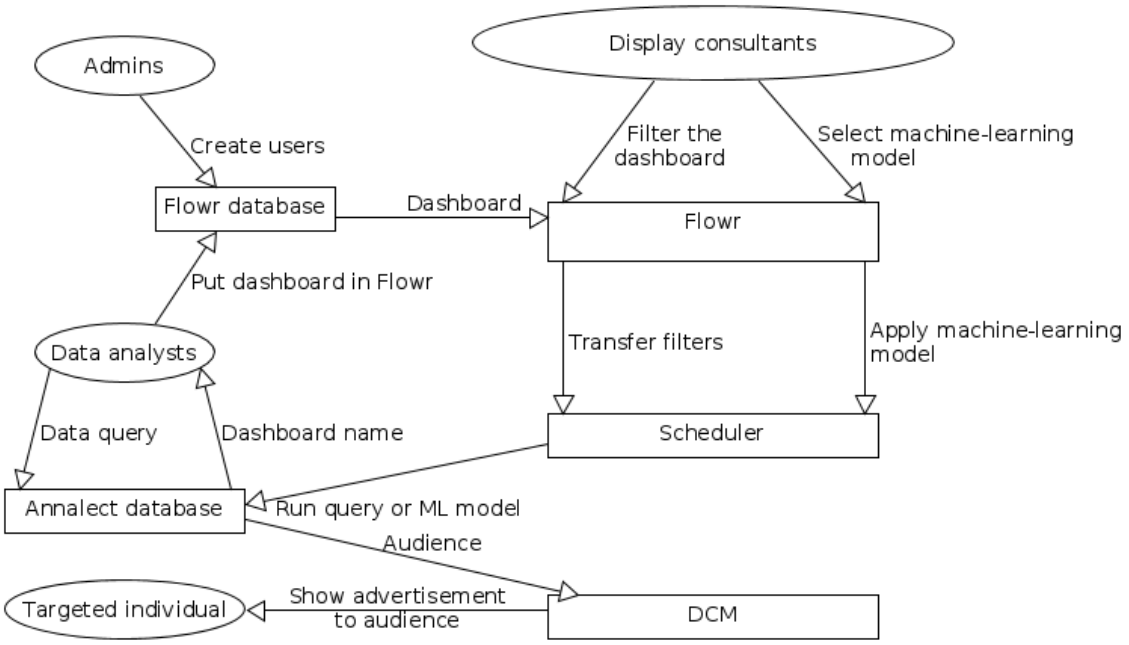


Figure 4.5: Overview of the data flow

4.3. Machine learning

One of the requirements posed by Annalect was this application supporting the use of machine learning to predict conversions. More specific, the machine learning should be able to predict which individual is likely to convert by being shown a certain advertisement. There can be about 200 different types of conversions to predict. The aim is to create a limited amount of models which can together predict the triggering of most of these conversions. Please refer to section 5.4 to see how we evaluate the performance of a model at predicting a conversion. The most important though, is that data analysts should be able to add models themselves to the application, as there is not enough time available to come up with (good) enough models in the duration of the project.

4.3.1. Data set

The training of a machine learning algorithm is done with a data set. Which data we used and how it looks is discussed in this section. As stated before, Annalect manages huge amounts of data owned by their clients. This data is stored in parquet files and contains information about historical advertisement campaigns (cookie data). This is data about when an advertisement is shown (impressions), when an advertisement is clicked and by who (clicks) and activities on advertisements (conversions). They are based on user ids, with which it is possible to identify the individuals to whom advertisements were shown. These user ids can also be used to create a new audience. This is also the goal of the machine learning. However, it is difficult for us to make something out of this data, as there are about 200 columns containing information. We have little understanding of which columns are important or the meaning of some of the column names. This is one of the reasons why it is hard for us to come up with a good machine learning model. Data analysts can do this much more efficiently as they have an expert understanding of the data and thus know which features to train the model on. We thus focused on specifying a machine learning pipeline which can be used by data analysts with their own models. Furthermore, we will include a simple machine learning model as a proof of concept for our pipeline.

4.3.2. Apache Spark

”Apache Spark is a fast, in-memory data processing engine with elegant and expressive development APIs to allow data workers to efficiently execute streaming, machine learning or SQL workloads that require fast iterative access to datasets. With Spark running on Apache Hadoop YARN, developers

everywhere can create applications to exploit Spark's power, derive insights, and enrich their data science workloads within a single, shared dataset in Hadoop" [12]. As the machine learning models present in Flowr use large amounts of data, in amounts impossible to be handled by a single computer, scalability becomes an issue. Flowr uses Spark because this way the machine learning pipeline can be executed without having to worry about scaling issues.

4.3.3. Input and Output

To be able to properly use a machine learning pipeline which can be reused, input and output to the model must be specified. The input will be the data as mentioned in section 4.3.1. It depends per model which columns of the data will be used in the pipeline. The model then outputs a score per individual, this score is then used in the pipeline to predict which individuals are most likely to have a certain conversion. The output of the machine learning are thus user id's.

4.3.4. Pipeline

The pipeline is a Spark [13] script which contains the general data flow necessary to use machine learning models. Parameters can be passed to the script, containing for example which data to use and the desired size of the audience. Once the parameters are configured, the script automatically loads all desired data and executes the machine learning model on this data. The outcome for every data entry (individual) is then a number predicted by the model of how likely the individual will be to converge. A set of the best performing individuals, in size equal to the desired audience size, will be selected to be sent to the Doubleclick Campaign Manager. Running this machine learning pipeline is executed in the scheduler. At least one machine learning model will be concluded as a proof of concept that this pipeline works, but most important is the ability to easily adjust and add machine learning models in the future.

4.3.5. Gradient boosting

The algorithm we chose for our proof of concept machine learning model is gradient boosting. This is mainly because our clients at Annalect have good experiences with this type of machine learning algorithm on similar projects and that this type of algorithm has proven itself on websites like Kaggle [9], where gradient boosting works very well on similar projects to Flowr. Implementing such an algorithm ourselves would take a lot of time and experience with machine learning algorithms, which both we do not possess. We thus made the choice, together with our supervisor at the TU Delft, to make use of a machine learning library. This way we can spend our time much more effectively, for example by spending time on finding the right parameters for the machine learning algorithm. The library we will use for this purpose is MLlib[1]. This library is specifically designed for Spark and supports gradient boosting.

5 Evaluation

In this chapter an evaluation of the final product will be given. The most important is that the client has an actual working application, with all the functionality that was agreed upon. This means that each of the problems mentioned in the problem description 2.2 should be solved. On top of that, the software should also be well maintainable. This chapter discusses the feedback of the Software Improvement Group (SIG), a company which performed static analysis on our codebase. Followed by each problem and evaluates whether and to what extent our application solves it.

5.1. Software Improvement Group

There were two evaluations of our codebase by SIG. The first evaluation took place in the 5th week and the second in the 10th week of the project. SIG performs static analysis on the code and provides feedback with possible improvements. On the 6th of June we delivered our first iteration of our codebase to SIG and on the 12th of June we received feedback. The full feedback can be found in appendix C.

In short: A score of 3/5 was given because of low scores on Unit Size, because of a few functions which had gotten too lengthy and Unit Interfacing, as some functions had too many arguments. It was also mentioned we had too few tests and that it is important to at least test important functionality such that changes will not introduce unwanted behaviour.

Improvements In response to the feedback the lengthy functions were split into smaller sized functions. This was especially useful as the functions in question were so lengthy because they actually had functionality for different user roles and so the functions could easily be split up into one for each role.

Regarding the functions with too many parameters. These functions are inside Airflow plugins which are bound to the way Airflow works. It is therefore hard to abstract these functions to reduce the amount of parameters. Airflow is a fairly new framework and lacking a lot of abstraction possibilities. In addition, the whole flow needs to be highly dynamic. Parameters can not be properties as they reside in the database. Once they are retrieved from the database they are inserted into a Python dictionary (which is an object) to be passed to a python callable using Python kwargs (keyword arguments). Python automatically converts the kwargs into parameters, thus keeping code readable and simple. These arguments were convincing enough to not implement this part of the feedback.

5.2. Cookie data

The first problem mentioned in the problem description is about the unused cookie data Omnicom Media Group manages. By enabling a smooth cooperation between data analysts and display consultants, the former can create different dashboards in Tableau. These dashboard reflect the cookie data with each dashboard giving a different perspective on or section of the data. These dashboards can then be used by the display consultants to further filter on and create their audience with. So through this process of cooperation, a workflow is created in which the display consultants can directly use the formerly unused cookie data.

5.3. Workflow improvement

As mentioned in the previous section, it is the cooperation between the display consultants and data analysts that make the solution for the first problem possible. So in order for the two parties to work together without issues, a workflow needs to be set in place that makes cooperation effortless. The application as it currently stands does exactly that. The data analysts can create dashboards in our application and share them with the display consultants. They do need to do some work before hand, such as creating a view for the database and setting up a dashboard in Tableau. However, once this is done, the display consultants can apply their expertise on the different dashboards, filter away and send their results to the DoubleClick Campaign Manager (DCM) without leaving Flowr. So although there is some setup needed by the data analysts, once the dashboards are created, the display consultants can independently use Flowr and make audiences.

5.3.1. Usability test

In order to evaluate the workflow more accurately, a usability test was done. In this test a single user for each role (admin, data analyst, display consultant) was given the assignment to perform tasks in the application. The reason only one user was chosen for each role, is that there are not yet more people which can work in the workflow this program enables. It is expected that this number rises a lot in the future though. While doing their tasks, the users were supposed to think out loud and give feedback on how all procedures went for them. This test was also filmed so that an accurate description could be given in this report.

The first role to be tested was the admin role. The tasks to perform were to log in and create two new accounts. One for a data analyst and one for a display consultant. These account were then going to be used for the next two usability tests. The user found the UI to be intuitive and it was easy for him to create new users. One thing the user noted, was that when trying to log out, he first started looking for the ”log out” button in the top right. As all of our application’s navigation is housed in a sidebar on the left, the log out button was also put there.

The second role to be tested was the data analyst role. The task to perform was to create a dashboard for the display consultant created in the previous test. This was the most difficult task as it requires the most prior knowledge about the dashboard. As the data analyst tried to create a dashboard he ran into some problems. Firstly some of the names of the input fields were not clear enough for him to immediately recognise what had to be filled in. This is easily changed after discussing what names would be the most clear to use.

Secondly, a harder to fix problem was the confusion of the data analyst when it was needed to link the Tableau filters to the columns in the database. Even though the data analyst in question created the Tableau dashboard himself, he had trouble figuring out what filters to select and link to the database columns. This is because of the way Tableau presents the filters in the JavaScript API. A single filter can occur twice in the dashboard, for example once as filtername and once Action(filtername). If both are present, which is not always the case, both have different purposes. Usually the Action(filtername) form of the filter is the filter of importance, but also this is not always the case. Because of this inconsistency we decided to let the data analyst make the decision of which filter to link. The problem arises when the data analyst has no knowledge about the underlying names of the filters the JavaScript API returns. As this is something that is different for each dashboard and not easy to explain shortly, we decided the best solution to this problem was to train the data analysts in this matter.

The third and final role we tested was the display consultant role. The task to perform was to log in, go to a dashboard, create an audience by filtering in Tableau and sending the selection to DCM. The display consultant found it easy to navigate to the dashboard. He did mention it could be useful to give notes of some kind for the controls of Tableau. The display consultant in question was already familiar with Tableau and knew some shortcuts could come in handy, so a short guide should be helpful. Besides that, the display consultant was able to create an audience and send it to DCM.

5.4. Machine learning

Evaluation of machine learning models bares great importance. It is a good way to assess the quality and ensure proper functioning of the model at hand. There is a wide variety of different methods to evaluate machine learning models. Some methods being specific to certain types of machine learning models and some which are more general and apply to all machine learning models.

A strong method in evaluating a model is the recall precision method [16]. This method has two ratios that evaluate the false positives and the false negatives respectively.

$$precision = \frac{tp}{tp + fp}$$

$$recall = \frac{tp}{tp + fn}$$

Where $tp =$ true positives, $fp =$ false positives and $fn =$ false negatives. It is important for our models to have a recall value close to 1. Having many false negatives in the model means that people are left out of audiences that should have been in there. Which leads to those people not being shown advertisements applicable to their persona. Thus having false negatives leads to missing potential customers. False positives, however, are less important. Having false positives in the audience will result in those people seeing inapplicable advertisements, resulting in no clicks. The resulting loss of money is nothing compared to missing out on potential customers. Therefore, having a precision value close to 1 is not crucial for the performance evaluation of the machine learning model.

One other method that is generally applicable to any model is the K-Fold cross validation method [20]. This method creates K subsets of the complete data set that the model uses to learn. These subsets are composed of two parts, the test set and the validation set. Generally, 80% of the data in the subset consists out of the test set and the remaining 20% of the data is the validation set. The test set will be used to train the model with and the validation set will be used to validate the trained model to assess the capability of the model to classify correctly.

The K-Fold cross validation method creates K of above described subsets where the test set and validation set constantly vary from the others. These subsets are then used to train and validate the model. This method makes the model less prone to overfitting, thus increasing the overall quality of the model its classifying capabilities.

6 Ethics

Ethics in Computer Science is getting more and more attention [18]. Privacy concerns in the computer science field are rising [17] and people want to understand what is going on under the hood of complex systems, such as machine-learning algorithms. The following sections discuss the ethical aspects of the user and password management and the machine learning present in our application. The last section gives a view on how this application will be used and the possible ethical complications that come with it.

6.1. User and password management

As stated before, privacy concerns in the computer science field are rising. Hackers are becoming more and more capable of retrieving information which should be hidden from them. This can be a problem if a hacker is able to retrieve the password of on of our users. The application can then be manipulated. On top of this, accounts on other websites the individual is using the same password for, might now be accessed by the hacker. It is thus our ethical duty to make sure user information is stored and processed as safe as possible.

6.2. Machine learning

Since the right to explanation [19] will be introduced next year, people will have the right to know why a certain decision was made by some algorithm. As it is possible to use machine learning algorithms with our application, there has to be thought about how insight can be given to why some individual was served some advertisement. As stated in section 4.3.4, at first the gradient boosting algorithm will be used as the main machine learning algorithm. This algorithm is a type of decision tree ensembles. Decision tree ensembles are known to be difficult to interpret as they are composed of many generated decision trees. Each decision tree within the ensemble has their own classification and the final classification is done by a majority vote over all the individual classification. These types of algorithms have proven to be very resilient to over fitting and tend to classify with high accuracy. Since there is no free lunch in computer science a decision tree ensemble with high accuracy loses on interpretability. Each decision tree adheres to its own rules which causes the whole ensemble to have a complex and hard to extract set of rules. It is possible to use a rule-extraction method in order to give some insights, as described by Mashayekhi et al. [21]. This way, some information could be given to people who would like to know why they were shown a certain advertisement. When other machine learning algorithms are added to this application, the solution to this problem will change. Having other machine learning algorithms besides the one just discussed leads to the need of additional explanation algorithms.

6.3. Program usage

As Flowr indirectly handles vast amounts of data gathered from internet users, it is crucial that data is handled in a discrete manner. The data should stay within the circles of the application and never leak into the real world, especially after it has been processed by different machine learning algorithms or data analysts and display consultants. Information on individuals can be extracted from the raw and

processed cookie data. Luckily, the data is anonymous as the cookie ids are only linked to a browser. However, if a browser is linked with a Google account, Google can directly link a DoubleClick cookie id to a user. This is out of the scope of Flowr and therefore none of our concern.

7

Discussion

It is evident that during a project, unforeseen problems arise. This chapter describes the deviations from the project plan and, due to the short time span of this project, possible features that were not implemented. We therefore include a recommendation section that describes additions that may possibly improve the product.

7.1. Deviations

During development, we encountered several problems that resulted in changes to choices made in our research report. These changes were quite significant as they changed our dataflow setup. We therefore dedicated a section to explain how we concluded that trading a system for another improved the product.

7.1.1. Amazon Web Services Athena

While the research report describes why we should use Amazon Presto, it was not very practical. Without going into too much detail, Presto required us to start and stop EMR (Elastic Map Reduce) clusters. When an EMR cluster is being started, an ID is created. This ID needs to be retrieved in order to execute a query on Presto as Presto requires this ID. This resulted in more API calls than were necessary. Not only would this extend the time necessary to successfully run a SQL query, it would also induce significantly more code-complexity. Due to the nature of Athena (Software as a Service) a lot of complex operations as starting and stopping clusters is automatically initiated, stripping complexity down merely to the one thing the service should be doing, running a SQL query. Less code and less complexity while maintaining the same functionality were convincing enough to remove Presto and introduce Athena.

7.1.2. Machine learning

During project planning we had a rather different view on what the machine learning part of the project would look like. We assumed that we would program our own model, train that model and hope it would perform somewhat acceptable. During later phases of the project we realised that this was far from feasible to get working in time. We therefore chose to go for an established library that offered us multiple working implementations of machine learning algorithms. We chose to shift the difficulty of the machine learning from building a model ourselves to finding and training an already implemented algorithm. Finding such a model is rather difficult as every data set and wanted outcome differs from each model. Besides that, specifying the input is challenging. We do not know what aspects of the data are important for its classifying performance.

7.1.3. Evaluation of our machine learning model

In section 5.4 machine learning evaluation methods are discussed. We did research on those methods and how they were applicable to the model we were building at that time. Our model, despite our best effort, did not succeed in producing any tangible results. This was due to a lack of time and underestimation of the complexity of translating the data into input that makes sense for the model.

Both the precision recall and k-fold cross validation methods have therefore not been applied. If our model had produced meaningful results we could have used the precision recall method to create a graph that shows the inverse relation of the precision term and the recall term. This graph would give a clear view on the performance of our machine learning model too which is also interesting from a technical point of view. We decided to keep the section on evaluation methods written in section 5.4, even though we have not been able to apply those evaluation methods. We made this decision, because we did do research on machine learning model evaluation and it would therefore be a shame to let that go to waste.

7.2. Recommendations

As the project time span was fairly short, a working product had to be made as soon as possible. This subsequently meant that it was hard to create robust, extensible code that would comply with all the programming principles. There is still significant room for improvement.

7.2.1. Framework

While the whole product is based on Flask, it is still in its early stages and therefore changing frameworks may still be worthwhile. When the project started, it seemed the product would remain a small application given the time span of the project. However, as the project progressed, it became more clear that the size of the application would eventually outgrow the limitations of Flask. Flask is amazing in small-scale applications but after running into too many problems, it might be good to research and consider switching frameworks. Additionally, Flask is poorly documented and it is therefore hard to conform to Flask standards when learning the Flask framework.

7.2.2. Documentation

During the project, we paid little attention to documentation. The code itself is well commented. However, documentation for the environmental setup, such as setting up Docker, is fairly outdated. In addition, there is zero documentation for end-users on how to actually operate the product which can improve both user-experience as well as user operating times.

7.2.3. Testing

There is still significant room for improvement regarding testing. There are very few tests that test the code. Both Flowr and Airflow need to be thoroughly tested. First of all, the testing structure needs to be set up. As for Airflow: Airflow provides little to no testing possibilities so one has to come up with a solution oneself. DAGs may utilise several Airflow plugin tests and define the flow of the schedule from start to end. Because of this, DAG testing can be seen as integration testing (testing the integration of several parts of airflow working together). Airflow plugin testing can be seen as unit tests as these tests merely test the functionality of the plugin itself, each function in the plugin will be tested individually. Regarding Flowr: Flowr models and controllers can all be tested using unit testing, these parts can together be tested using integration tests. Also to create confidence in the UI, end-to-end tests are highly recommended.

7.2.4. Continuous Integration

To improve developing a workflow, Continuous Integration should be introduced to the product. The product resides in Bitbucket as our git solution. Bitbucket provides Bitbucket Pipelines which brings continuous integration and delivery to the Bitbucket Cloud, empowering teams to build, test, and deploy their code within Bitbucket. Setting up Bitbucket Pipelines to work with amazon web services can significantly improve developing speeds and create confidence in failsafe product builds.

7.2.5. Trusted Authentication for Tableau

As of now the application is still hosted on our local machines. To use the Tableau API, the user needs to be logged in to their Tableau account. Users can log in through the embedded Tableau interface, however, this process does not work optimally. Once the application is deployed to a server, the server address can be added as a trusted IP-address to the Tableau server. Once this is done, the application can get a trusted key by only providing a valid username. With this trusted key, Flowr can communicate

with the Tableau API as if the user were logged in. Enabling this feature can make for a better integration of Tableau within Flowr.

7.2.6. Editing of Scheduled tasks

Once a display consultant has scheduled his or her selection, the resulting query will be performed on the Annalect database and the results sent to DCM for each interval that was given. Currently no end time can be configured, so the query will run indefinitely. Of course this is something that should be configurable. A screen should be created in which a display consultant has an overview of all the scheduled task currently running with the option to pause or delete them.

7.2.7. Dependency Manager

No dependency manager has been used during the course of the project. This is not of great importance, since there are not many dependencies to manage in this project. It would, however, be more efficient to still use a dependency manager when it comes to installing the libraries and keeping them up to date. Bower, for example, is great for just that. Having Bower would make it much more easy to pull in different third party libraries without having to download every library and save it in the local asset folder manually.

7.2.8. Lightweight Directory Access Protocol

The Lightweight Directory Access Protocol (LDAP) is an open, vendor-neutral, industry standard application protocol for accessing and maintaining distributed directory information services over an Internet Protocol (IP) network. Directory services play an important role in developing intranet and Internet applications by allowing the sharing of information about users, systems, networks, services, and applications throughout the network [10]. By using LDAP as our login authentication, users will have instantaneous access to the application once registered within the company its LDAP server. In addition, user rights and permissions are defined within the LDAP server, if the product inherits these same rights and permissions, the application follows the same set of rules defined by the LDAP administrator.

Glossary

audience A group of individuals which is selected to be shown a certain advertisement..

conversion The point at which a recipient of a marketing message performs a desired action. This can for example be clicking on the advertisement, but also staying on the advertisement page for a certain amount of time, subscribing to a mailing list or the ultimate conversion: buying a product..

dashboard An overview of data with specific characteristics, made by data analysts.

data analyst People working for Annalect which have a background in Computer Science. They are able to write queries to the database and create machine learning models. A user of the program..

display consultant People working for Omnicom which are experienced in selecting audiences for marketing campaigns. They will use the dashboards provided by the data analysts to filter the huge amount of data present at Annalect. A user of the program..

DoubleClick Campaign Manager (DCM) A programmable machine that receives input, stores and manipulates data, and provides output in a useful format.

DoubleClick segment A segment in which cookies are grouped by DoubleClick. These segments are things like: football lovers, pet owners, etc..

filtering The process of applying filters on the data in order to narrow down the audience.

tableau A third-party program used for visualising and filtering data.

targeting The process of selecting a suitable audience.

Bibliography

[1] Mllib machine learning library for spark. URL https://spark.apache.org/mllib/.

[2] Amazon athena. URL https://aws.amazon.com/athena/.

[3] Data transfer v2.0. URL https://developers.google.com/doubleclick-advertisers/dtv2/overview.

[4] Docker - build, ship, and run any app, anywhere. URL https://www.docker.com/.

[5] Doubleclick official website, . URL https://www.doubleclickbygoogle.com.

[6] Explanation of what doubleclick cookies are, . URL https://support.google.com/adsense/answer/2839090?hl=en.

[7] Jinja2, the python template engine. URL http://jinja.pocoo.org/.

[8] The write less, do more, javascript library. URL https://jquery.com/.

[9] Kaggle - your home for data science. URL https://www.kaggle.com/.

[10] Lightweight directory access protocol. URL https://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol.

[11] Materializecss, a modern responsive front-end framework based on material design. URL http://materializecss.com/.

[12] What is apache spark, . URL https://hortonworks.com/apache/spark/.

[13] Apache spark - a fast and general engine for large-scale data processing, . URL https://spark.apache.org/.

[14] Tableau software - business intelligence and analytics. URL https://www.tableau.com/.

[15] Airflow. Apache airflow (incubating) documentation. URL https://airflow.incubator.apache.org/.

[16] Michael Buckland and Fredric Gey. The relationship between recall and precision. *Journal of the American society for information science*, 45(1):12, 1994.

[17] Anupam Datta. Privacy through accountability: A computer science perspective. In *International Conference on Distributed Computing and Internet Technology*, pages 43–49. Springer, 2014.

[18] Fernando Diaz and Solon Barocas. Wsdm 2016 workshop on the ethics of online experimentation. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 695–696. ACM, 2016.

[19] Bryce Goodman and Seth Flaxman. European union regulations on algorithmic decision-making and a" right to explanation". *arXiv preprint arXiv:1606.08813*, 2016.

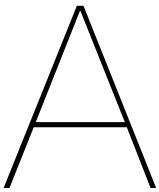
[20] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Stanford, CA, 1995.

[21] Morteza Mashayekhi and Robin Gras. Rule extraction from random forest: the rf+ hc methods. In *Canadian Conference on Artificial Intelligence*, pages 223–237. Springer, 2015.

[22] Katherine Noyes. Docker: A 'shipping container' for linux code. URL https://www.linux.com/news/docker-shipping-container-linux-code.

[23] Scheduling (computing). Scheduling (computing) — Wikipedia, the free encyclopedia. URL https://en.wikipedia.org/wiki/Scheduling_(computing).

Appendices



Infosheet

Enhancing Dynamic Market Audience Creation (Flowr)
Annalect
Presentation date: 4 July 2017
Description:

The workflow between different parties in the client company was far from optimised. Having an application that would streamline the work between those parties would save money as well as improve quality of both parties their work. An application was thus created as solution to the demand of a streamlined, well thought out workflow. The challenge here was figuring out exactly how the client wanted this workflow to work and what was supposed to be built by us. This workflow consists of the parties working with cookie data to create an online marketing campaign. There was also demand for support of machine learning algorithms. During research we found out that web frameworks offer a wide variety of different functions. From this we chose to base our decisions on what would fit the brand new workflow best. This became challenging as we did not have much experience with setting up a web application from scratch. We overcame this hurdle by communicating with our client and having weekly meetings with the client discussing what direction we were headed. During the course of the project changes were imminent, but as we met up with our client often we were able to adjust direction whenever needed. The web application we built fits perfectly with the workflow demanded by the clients. We tested this by conducting user tests and receive feedback on what could be improved. The future for this application looks bright as the client is very positive on how it functions and therefore wants to introduce it to the different parties at stake. There are features we left out however, one of those is a functioning machine learning model which could and should be made in the future. Other than that, minor improvements in UI and functionality of the back-end are recommended.

Members of the project team:
Name: Cas Bilstra
Interests: Machine learning, Management
Role & Contribution: Database, Machine Learning, Back-end development

Name: Shane Koppers
Interests:
Role & Contribution: Tableau integration, Front-end

Name: Floris List
Interests: Machine learning, Back-end structure design, UI design
Role & Contribution: UI design & implementation, Front-end and Back-end development

Name: Ramin Safarpour Erfani
Interests:
Role & Contribution: Database integration, Scheduler

All of us had the role of developer and worked on all the required documentation such as the final report and research plan.

Client, Coach:
Name and affiliation of the client: Niels Schenk, Annalect
Name and affiliation of the TU Coach: Christoph Lofi, Web Information Systems, TU Delft

Contacts:
Cas Bilstra, casbilstra@gmail.com
Shane Koppers, shanekoppers@hotmail.com
Floris List, floris.list@hotmail.com
Ramin Safarpour Erfani, erfani.ramin@gmail.com

The final report for this project can be found at: http://repository.tudelft.nl

B

Research report

Dynamic marketing audiences from big data
Research Report

C. Bilstra
R. Erfani
S. Koppers
F. List

June 8, 2017

Contents

Abstract

The clients Annalect and OmnicomMediaGroup posed us the problem of their lacking workflow when trying to implement a new system involving different groups of coworkers. As they thought this could be improved upon by making an application supporting this new workflow, our task was to create such an application. For the design a slightly changed version of the data warehouse used by Chaudhuri et al. was used. As a scheduler is needed, Apache Airflow was picked for this position. The database system is going to use PostgreSQL. The application itself will be a web application build with the Flask framework in Python. Lastly, a pluggable machine learning component will be made, with us making our own neural network as a proof of concept. For more information, please also look into the project plan.

1 Introduction

In this report the technical requirements of the system are identified and research is performed in order to be able to make well-motivated technical choices. First, the problem we will solve will be identified and assessed after which a proposed design is introduced. Research is performed on which technologies to use in order to be able to solve the stated problem.

1.1 Problem description & solution

An in-depth problem description & solution can be found in the project plan.

1.2 Requirements

When looking at the problem this project will solve, some technological requirements can be identified. The system will have to do calculations and processing on an ever expanding pool of data. Therefore, scalability in performance is very important when machine learning models are applied, but also scalability in storage is important as the data can become larger than one machine is able to handle smoothly. Processing must be performed daily on new data. The system will be used by display consultants on their own machines, which means the system should have an user interface which is easy to use. Similar systems will be investigated in section 2 after which a design in which these requirements can be met will be proposed.

2 Design

Not only the use of possible technologies needs to be researched, also the overall design of the system should be evaluated. A rather old but popular system for

data warehouses is the one used by Chaudhuri et al. [2]. Our system is based on the one they proposed in figure 1 of their paper, but more specified to our specific use case. An overview of our system can be found in Figure 1. As this system is used so widely, it has proven itself and should be a good choice for this project.

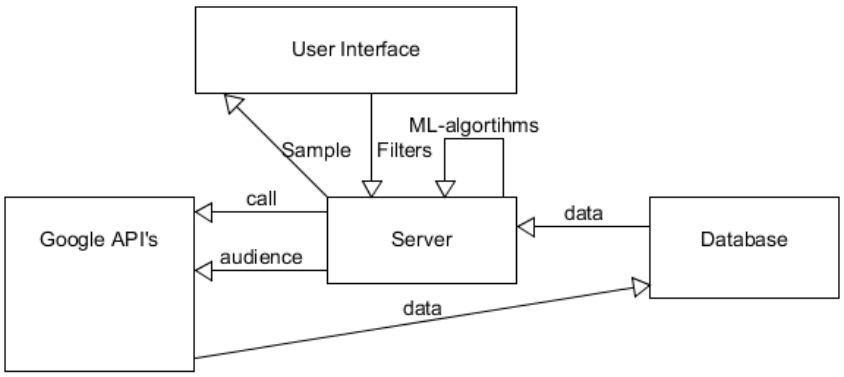


Figure 1: Overview of the proposed design

The external source of the data is Google, and with API's this data can be fetched regularly. This is the task of the server, which will have scheduled tasks. This data will be put in the database. Every time data gets fetched from google, the data changes and algorithms, like machine learning algorithms as described in section 4 must be run on the data. But also the query as defined by data analysts together with the filters as specified by the display consultants should be run on the new data daily, as different audiences will result from new data, and these resulting audiences should be sent to Google again. Users (display consultants) will be presented a view on the data provided by the data analysts. They will visualise this view using Tableau [1], data visualization software, and select some filters on which the final audience will be built. In section 3, there will be elaborated on the technical choices for individual parts of the design.

3 Technical choices

With an application of the scale of this project there are a lot of options when it comes to the usage of different techniques and technologies. This project is going to use state-of-the-art technologies, as this offers a fair challenge to the programmers and this way it is possible to push innovation. This section will discuss the technical choices for each part of the application. A short summary

[1] `https://www.tableau.com/`

for each consideration is discussed and a concluding paragraph that justifies the choice to go for a specific technique follows.

3.1 Scheduler

Once the resulting audience has been created, the list of user id's is sent to Google. This process is done using a scheduler, an ETL (Extraction, Transformation, Load) workflow framework. These ETL workflows are commonly used to union data from different databases into another database, mostly a datawarehouse[3]. A scheduler is a piece of software that can programmatically schedule workflows. In our proposed design, a scheduler is a mandatory component as we need to continuously send data to the Google API (containing the user id's of the resulting audience). It is important for the data to be correctly sent, we need to be highly confident that the request has succeeded and that we are notified on error. By being able to programmatically author schedules and workflows, it is possible to create a robust component that is fail-safe. A scheduler will work directly with the database.

3.1.1 Apache Airflow

Airflow uses Directed Acyclic Graphs (DAGs) to author workflows. The Airflow scheduler executes your tasks on an array of workers while following the specified dependencies. Airflow pipelines are configuration as code (Python), allowing for dynamic pipeline generation. Airflow expects workflows to be mostly static or slowly changing. Airflow lacks testing tools as well as extensive documentation.

3.1.2 Luigi

Luigi uses the same methods as Airflow except that it is more mature and is more thoroughly tested. However, Luigi lacks a lot of interesting features compared to Airflow: state persisting (in case of failures this comes in handy), execute DAGs in parallel, can't execute DAG directly (have to create a special sink task) and ironically, does not have a built-in scheduler and relies on cron to do scheduling which is very inconvenient in terms of monitoring. In addition to aforementioned, Luigi lacks an intuitive, extensive user interface.

3.1.3 Pinball

Pinball subsequently uses DAGs to author workflows. Does have the ability to execute DAGs and the ability to process DAGs in parallel. In addition, Pinball is thoroughly tested while maturity is low. However, Pinball does not come with a lot of contributed software meaning that it is our task to write various adapters to different systems (PostgreSQL, MySQL, Hadoop, Hive, etc.). There also is very little documentation.

3.1.4 Conclusion

First of all, It is mandatory for our ETL workflows to continue where it left off in the case of an error. Due to Luigi's inability to persist states, Luigi is the first to fall off. Second, Pinball's incomplete documentation in addition to very little contributed software makes Pinball the second to fall off. With those options out of the way only Airflow is left, therefore we chose Apache Airflow as our ETL workflow framework. Airflow contains an extensive UI and advanced features like state database persisting and parallel DAG execution.

3.2 Database

There are two different types of databases: relational and non-relational as well as several different database managers: PostgreSQL, MySQL, MongoDB, SQLite. Each having their advantages and disadvantages which are mostly based on the purpose of the database.

3.2.1 Apache MySQL

MySQL is a relational database and thus its organisation is based on the relational model of data. It is probably the most widely used database manager today and is backed by a massive array of documentation. MySQL offers high security, scalability, speed and a smooth learning curve. MySQL does, however, come with some reliability issues during transactions, concurrent read-write operations can be problematic for example.

3.2.2 PostgreSQL

PostgreSQL is just like MySQL, a relational database. PostgreSQL is the most advanced relational database management system today and differs from MySQL in that it is highly programmable and therefore extendable. Subsequently, PostgreSQL is very good at concurrency and handling multiple tasks efficiently, making PostgreSQL a good candidate for handling big data where data integrity and reliability are an absolute necessity. There is, however, one drawback that should be considered: speed. PostgreSQL is not the best at heavy read-operations, if all one needs are heavy read-operations, PostgreSQL is not the tool to go for.

3.2.3 MongoDB

MongoDB is quite different to the two preceding database managers. MongoDB uses NoSQL and is a non-relational database. The thing to remember about most NoSQL databases is they rely on the concept of indexed key-value pairs (and sharing nodes based on an unique key). They are designed to read and write single key-value pairs at a time and are most commonly used as a tertiary storage platform for transactional systems. Doing the equivalent of a "full table

scan" in MongoDB is an extremely slow operation which is very common in analytics.

3.2.4 Conclusion

MySQL is a good option in terms of speed, scalability and security. It comes with a lot of documentation and is the most commonly used database manager. It lacks however, reliability and concurrency, which are both absolute necessities. It seems like MongoDB is not really suitable as during this project we will mostly work with relational data making NoSQL not the preferred option. PostgreSQL has data integrity and reliability as top priority while being very extensive and programmable. For this project, PostgreSQL is the preferred database manager.

3.3 User management

In our system, Display Consultants and Data Analysts must be able to identify themselves in order to communicate with each other. This means a user management system, together with a database to save the user information, should be present. Choices for a suitable database are elaborated upon in section 3.2. We could write our own user management system, but this would take a lot of time and is a project on itself. Furthermore, this would introduce a lot of testing as user management has to be very safe. Therefore, we will use Flask-user [2] for our user system. This addition to flask contains everything we need, and can be adopted to our wishes fairly easy. It uses Flask-login to provide a login system at the same time. It is beyond the scope of this project to provide a fully fail-proof extremely well-tested user management system, but by using these libraries we can provide the user with a good system. Flask-user is the most popular user management system for flask, other user management systems in python are for other frameworks, for example Django, which we will not use as can be read in section 3.4.1.

3.4 Application

There are plenty of options to choose from when it comes to creating software to solve the problem as stated in the introduction. The main distinction between application types this research report will focus on are Desktop Applications and Web Applications. The following two paragraphs will discuss both options in more detail.

Desktop Applications Applications that run only locally by using an executable for example, have the advantage of not having to use a server. This advantage manifests itself in several ways. Firstly, writing and testing code becomes easier as it can be executed immediately locally, while a server would need live deployment or some kind of server simulation. Secondly, using the

[2] `https://pythonhosted.org/Flask-User/`

application would not need any kind of Internet connection as long as all the data is also stored locally. Lastly, there is a trade-off when running calculations at runtime locally. In a desktop application there is no longer a possibility for a server to overload, but this does mean the local machine needs sufficient processing power itself. A big disadvantage of a desktop application is that executables are usually not too multi-platform friendly. Web applications work on every device with an internet connection and a browser, which most devices already have. When stakeholders, using a wide range of devices need to use the application, being able to use it on all of those devices is of importance. This is hard to achieve when using a desktop application.

Web Applications Web applications are used and operated from within a web browser (Google Chrome, Firefox, etc) with access to the Internet. The client (web browser) has direct contact with a server, this server supplies the client with a User Interface in which the user can interact with the application. This interaction consists of server side processing.

A huge advantage of web applications over desktop applications is that intense processing is (or can be) done on the server side of the application. When it comes to processing a lot of data, having a server that does all the number crunching is a much more pleasant option than having a local machine do it. Besides that, servers are often more powerful than local machines. The more data that needs processing, the more processing power is needed which results into mandatory server upgrades or even considering a distributed server that can reduce load by distributing them over several servers effectively solving the problem in parallel. These upgrades, are very easy to realise on a rather cost effective scale. When a Desktop App is expected to process big data and this needs to be possible on machines of multiple users, it is simply not feasible to upgrade each individual machine from a cost effective standpoint.

Web applications are also easily accessible, a user can access it whenever, wherever one deems fit given that an Internet connection is available. The only thing a web app requires is a working network connection and a device (client) that is capable of running a web browser. Since many modern devices are equipped with hardware that fulfil said dependencies it is save to say that claimed accessibility is true.

A drawback of using a web app is that concurrent users cause a decrease in user experience. Server problems might occur when there is a big load on the server. Problems like speed and erroneous output or failing processes are most common to happen when a server experiences a heavy load. This problem, however, can be solved by expanding the server resulting in more processing power. Yet again, this is in principle more cost effective than upgrading local machines when using a desktop app.

Another difficulty to overcome when using a Web App is security flaws. The traffic that is going to move over the Internet will often be sensitive. When handling sensitive data security is very important. If there is a lack of security measures this sensitive data can be intercepted by third parties that should not

have access to this data in the first place. Therefore a Web App must have proper security measures to avoid any breaches.

Conclusion All things considered, a Web Application would be the better of the two posed options. The platform that is going to be built needs to support multiple (concurrent) users that get the exact same user experience. The strongest reason to go for a web application is that it is easily scalable. When queries are run over vast amounts of data it is good to be able to scale to a sufficient degree where all these queries can be run smoothly. Since only one device, the server, needs to upgrade when needed it is rather cost effective when compared to having to upgrade every individual machine. A Web Application will also provide a solid and easy way of implementing a working User Interface. This UI will be built just like any regular website. This process will not take much time and the team is already experienced with development of websites. The deployment of this application only takes the server into consideration. The server needs only to host this application and can then serve a multitude of clients. If a device supports the client, the user will be able to use the application on that device.

3.4.1 Framework

Now that we have decided to use a web application, there are still multiple ways to create one. A web application traditionally has a front-end and a back-end. There are multiple frameworks who can serve the job of either one, however frameworks also exist which do both. The advantage of using a framework that serves both front- and back-end, is that there is no need for enhanced client-server communication. Also only one framework needs to be learned by the team, which can considerably speed up the process of actually creating the web application.

There are too many web frameworks to cover each and every one individually, so we will only discuss some of them. There is an important distinction between full-stack and non-full-stack. Full-stack can offer a more complete package which usually has all the tools you would need to create a web-application. The learning curve might be steeper, as there is just more to begin with. Non-full-stack frameworks usually have less services built-in, which will need integration of other libraries or frameworks. The upside of this, is that when creating a small application and a full package is not necessary, start-up time can be reduced. Of course each web framework is using a certain language, which each has its own pros and cons.

PHP PHP is a server-side scripting language, with popular frameworks including Laravel and Symfony. Although PHP has proven itself useful in web-development, as a machine learning tool, it is quite lacking. The language is

not built to handle the amount of computational power machine learning requires and partly because of this, there are not many tools supporting machine learning for it.

Python There are a couple advantages using Python. First of all, it is quite easy to learn and it has very strong and a large set of libraries which can prove useful especially when doing machine learning. The two main python web-frameworks we will be looking at are Django and Flask. Of course there are many more, but these two offer the advantage that they can both use Bootstrap or Materialize CSS to create layouts, and both are html-frameworks the team is already familiar with. Django is a full-stack, batteries included framework, which means all the basic things you would need for creating a web-application are already there. There are services ready like a database, easy to create admin pages and build-in security. Flask is a micro-framework, it is more bare bones in the sense that it has less features, but it is easy to add features you would need on the run. Web development one drop at a time, as their slogan goes. Although Django has a steeper learning curve, when working in Flask we would need to spend time on integrating certain functionality that might already be present in Django natively. Flask, however, can be easier to set up and get going quicker. The application being built is not going to have a complex front-end, while at the back-end it will need to be able to do complex computations when applying machine learning.

Conclusion When firstly looking at languages, Python will be most useful to us when implementing a machine learning model. Secondly, as the front-end is going to be simple and we will mainly focus on some computation features such as filtering tables and implementing the machine learning model, we do not have a need for a full-stack framework, but could really benefit from a quick start up. For these reasons, Flask seems like the framework to go for.

4 Machine learning

Expert knowledge has been a main driving force behind practice of processes of many varieties. Nowadays the focus is shifting to machine learning combined with expert knowledge instead of solely relying on expert knowledge. Machine learning has a lot to offer when it comes to automation of processes previously done by human experts. Machine learning algorithms that process a lot of data are able to see connections and rules within a given dataset that are invisible to the human interpreter. Experts do need to control the output of such machine learning algorithms in order to see if the algorithm shows expected behaviour, which means that their role is still crucial.

This also applies to the problem that is going to be solved with this project. A machine learning algorithm that is going to be constructed will output suggestions for audiences based on similar audiences the experts wanted. These

similar audiences also provide information on whether an individual in that audience was well placed. Well placed means that an individual interacted with the advertisement either by clicking on it or watching it. The algorithm will therefore use previous audiences as training and test data in order to classify new input data to a new audience if the individual has a high chance of conversion. The input will exist of an unique user with a set of features that are deemed important for the classification.

Classifying algorithms (classifiers) come in different shapes and sizes. Each of these different implementations have their advantages and disadvantages. There are different types of classifiers, namely interpretable and non-interpretable classifiers. We will be going for a non-interpretable classifier, as that type of classifiers offer the most accurate solutions. A drawback of this type of classifier is that it is, as the name states, not interpretable. This, however, will not be a problem since little interpretation of the model is needed. The only thing that matters is that the classifier successfully classifies the input to the correct audiences. Simple tests in the learning process provide insight into whether this classification is done successfully or not.

Having settled with a non-interpretable kind of classifier comes a next step, what exact kind of non-interpretable classifier is going to be implemented. For this there is a wide range of options, but in order to limit the search space only Neural Networks and Random Forests will be considered, as these are viable options that produce significant results which are feasible within the timeframe of this project. Each of the two options will be discussed in the coming two paragraphs.

Neural Network A Neural Network (NN) is based on the workings of a human brain[4]. A NN is a set of subsequent layers $l \in L$ where $L = \{l_0, l_1, ..., l_n\}$ and $n \in \mathbb{N}$. Each layer consists out of a set of nodes called neurons, $l_i = \{c_0, ..., c_n\}$ where n can be different for each layer l_i. Every neuron from layer l_i is fully connected by an edge to every neuron in layer l_{i+1}. Each of the neurons present in the model have their own weights w. Whenever the algorithm is run the input must surpass a certain weight of neuron c_i in order to be passed on to the next layer through neuron c_i.

The training of a NN is (almost) always done by means of backpropagation. There are multiple types of training, namely supervised, unsupervised and reinforcement learning. This paragraph will only look into supervised learning, since it is the only relevant type for this project. Supervised learning means that the training data also has an expected output value bound to the input value. This means that a NN will able to correct itself when a classification is made. If the output turns out to be wrong compared to the expected output this error is backpropagated into the NN and that will lead to a weight change in the neurons that contributed to the classification.

Random Forest A Random Forest (RF) is a type of Decision Tree Ensemble. A RF is composed of a selection of Decision Trees (DTs) that have been generated by bagging[1]. When this process is randomised the resulting Forest is called a Random Forest. When this RF is fed with data every DT in the RF will make a classification on its own. If all the DTs have made a classification a majority vote over those classifications will decide the final classification of the RF as a whole. The RF model is one of the strongest and most accurate classifiers out there at the moment. It has also been shown that a RF is not prone to overfitting the dataset and is able to accurately classify on real world data.

Both models are very strong when it comes to sophisticated classification. It will, however, be more difficult to implement a sufficient functioning RF than a NN, since the majority of the team has already implemented a NN before. This also is the main reason for choosing a Neural Network as model that is going to be used for this project. The implementation of a NN is feasible in the given time and it is also feasible to have a proper functioning one. Which leads to the following section, the outline of the implementation of the NN.

4.1 Implementation outline

As stated previously, a Neural Network is going to be build. This NN is going to function as an optimising classifier. This means that given a certain test set, it will be trained to classify new data as optimally as possible. The NN will be specific to a certain advertisement campaign and will only function properly when used for similar campaigns. The following paragraphs will discuss the different components of the NN.

Feature vector It is difficult to specify what the features are that will compose the feature vector. This will be defined during the course of the project.

Amount of neurons The input neurons will be equal to the amount of features that are provided by the feature vector. The output will only consist out of one neuron, because only a yes/no answer is expected. We decided to do it that way, because then we overcome the difficulty of making the model interpretable. Its only important to know whether given datapoint belongs to an audience for a new or existing campaign. The neurons in the hidden layer amount to a number $\leq$ input neurons and $\geq$ output neurons

Amount of layers The two base layers (input and output) will be present along with a n amount of hidden layers. This number has still to be decided upon.

Test set The test set will be previously (hand tailored) audiences, with knowledge over conversions of individuals in those audiences. From previous results

of a campaign the effectiveness of the advertisement on each individual will be determined. This effectiveness means that a user interacted with a displayed advertisement, either by clicks or conversion. The NN will then use this test data to see if given user was an effective part of that audience. If false the NN will learn to classify similar people as unfit for a similar effective audience and vice versa if true. Optimisation of new audiences will be implied by this way of learning. Only effective similar users will be considered for a new audience by the NN.

Input The input of the trained NN will be a feature vector x that contains all features deemed to be of importance for the classification.

Output The output is a simple yes/no answer. This yes/no answer will be decided by a Sigmoid Function over the value propagated to the output neuron. If the output is yes (true) then the datapoint will be added to the resulting audience.

Performance optimisation The NN will be optimised performance wise by conducting a Sensitivity Analysis. A Sensitivity Analysis changes one or multiple feature values to see fluctuations in the resulting classification. This information is then used to decide whether such features are of great importance and should therefore bare stronger weights in the NN. Besides optimisation it will also give insight into the working of the NN and it will provide insight in to what features are important in the data set.

References

[1] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[2] Surajit Chaudhuri and Umeshwar Dayal. An overview of data warehousing and olap technology. *ACM Sigmod record*, 26(1):65–74, 1997.

[3] SAS. What is etl?, 2017. [Online; accessed 26-April-2017].

[4] WenJun Zhang. *Computational ecology: artificial neural networks and their applications*. World Scientific, 2010.

C

SIG Evaluation

De code van het systeem scoort 3 sterren op ons onderhoudbaarheidsmodel, wat betekent dat de code gemiddeld onderhoudbaar is. De hoogste score is niet behaald door lagere scores voor Unit Size en Unit Inerfacing.

Voor Unit Size wordt er gekeken naar het percentage code dat bovengemiddeld lang is. Het opsplitsen van dit soort methodes in kleinere stukken zorgt ervoor dat elk onderdeel makkelijker te begrijpen, te testen en daardoor eenvoudiger te onderhouden wordt.

Het is hierbij belangrijk om te bedenken wanneer je een nieuwe methode moet introduceren. Jullie houden bij views.py bijvoorbeeld sterk vast aan de standaardindeling van het framework: alle code voor het afhandelen van de route naar index staat in de methode index(). Deze aanpak werkt niet meer op het moment dat je te veel logica hebt. In het voorbeeld van index() heb je duidelijk twee types functionaliteit: het blok voor de consultants en het blok voor de data-analisten. Het is beter om deze blokken functionaliteit naar aparte methodes op te splitsen. Als je dit niet doet wordt je systeem op termijn steeds slechter onderhoudbaar op het moment dat de functionaliteit blijft groeien.

Voor Unit Interfacing wordt er gekeken naar het percentage code in units met een bovengemiddeld aantal parameters. Doorgaans duidt een bovengemiddeld aantal parameters op een gebrek aan abstractie. Daarnaast leidt een groot aantal parameters nogal eens tot verwarring in het aanroepen van de methode en in de meeste gevallen ook tot langere en complexere methoden.

In jullie geval is de constructor van GoogleDCMPlugin_SIGPROOF.py hier een voorbeeld van. Overigens hebben jullie dit bestand blijkbaar gecensureerd omdat wij een deel niet mochten zien, wat het wat moeilijker maakt om goede feedback te geven. Desondanks is de hoeveelheid parameters in de constructor eigenlijk te veel. Je kunt de leesbaarheid vergroten door van de niet-kritieke parameters properties te maken.

Als laatste nog de opmerking dat er nog erg weinig testcode is. Het is sterk aan te raden om in ieder geval voor de belangrijkste delen van de functionaliteit automatische tests gedefinieerd te hebben om ervoor te zorgen dat eventuele aanpassingen niet voor ongewenst gedrag zorgen.

Over het algemeen scoort de code dus gemiddeld, hopelijk lukt het nog om dit niveau tijdens de rest van de ontwikkelfase nog iets te laten stijgen.

D

Original Project Description

D.1. Project description

In the current project you will work on building a system that allows our campaign consultants to build custom ‘audiences’ that they can use to target specific advertisements to. You will build a basic UI where people can define the criteria that demarcate which cookies should belong to a specific audience. A backend system will then integrate with our big data infrastructure to continuously process incoming datasets in order to keep the custom audience up to date, and integrate with our media buying systems to update audiences in these external systems. Ultimately our data scientists should be able to add machine learning algorithms that calculate who should belong to a specific audience. If you are interested you will have the possibility to build prototypes of models in Spark yourself.

D.2. Company description

Annalect is part of OmincomMediaGroup, one of the largest media agencies in the world. At Annalect we provide insights from data and analytics that our internal and external customers use to make their advertising smarter. We consider marketing to be a really big playground that allows us to come up with smart ways to handle and make use of the huge amounts of data that are being generated.

Within the Netherlands we are building out our data and analytics infrastructure in order to be able to automate and scale out some of the use cases we have built for individual clients. There are many projects we are working on at the moment. One in particular seems interesting for Computer Science students because it encompasses a wide set of skills needed to successfully complete an IT-project.