A Work Project, presented as part of the requirements for the Award of a Master's degree in

Business Analytics from the Nova School of Business and Economics.

# Training Tabular Generative Adversarial Networks in a Federated Learning Framework for Generating Realistic, Non-sensitive Data for Modatta: A Case Study for State-of-the-Art Recommendation Systems

Ana Carolina Mareco Sena

Work project carried out under the supervision of:

Qiwei Han

And in collaboration with

Rodrigo Moretti and Eduardo Pinto

Basto (Modatta)

17/05/2023

**Abstract**: This work project explores the usefulness of a privacy-preserving framework that enables Modatta to make personalized recommendations to their users through their application. For that reason, the Federated Training of Generative Adversarial Networks for Tabular data was studied, and its performance was evaluated on anonymous data. Choosing the right users for campaigns has a huge impact on Modatta's user experience and satisfaction, therefore, state-of -the-art Recommendation Systems were investigated.

**Keywords**: Data-Privacy, Generative Adversarial Networks, Federated Learning Systems, Tabular Data, Machine Learning, Deep Learning, Collaborative Filtering.

**Acknowledgements**: I would like to thank Professor Qiwei Han and Eduardo Pinto Basto and Rodrigo Moretti from Modatta, for the availability, guidance and valuable insights provided throughout the project. Additionally, to my colleague, Ana Morais, for being a key element in the conclusion of this project, and to my family and friends for all the support.

This work used infrastructure and resources funded by Fundação para a Ciência e a Tecnologia (UID/ECO/00124/2013, UID/ECO/00124/2019 and Social Sciences DataLab, Project 22209), POR Lisboa (LISBOA-01-0145-FEDER-007722 and Social Sciences DataLab, Project 22209) and POR Norte (Social Sciences DataLab, Project 22209).

## 1.Introduction

The development of a digital economy has led to many changes in the way that businesses are made, firms interact, and consumers behave (Deloitte 2021). An example is how companies target their customers. With the Internet, companies were able to transition from traditional marketing and sales strategies, such as physical stores, newspapers, and radio, to digital approaches like email and banner ads (Digital Marketing Institute 2016). The further progress of the Internet and the increasing emergence of e-commerce and social media platforms allowed the expansion of these simpler approaches to improved data-driven solutions. Data-driven strategies are present in everyone's daily life. For instance, marketing strategies often rely on data to better understand their target audience and create more effective campaigns. However, to achieve this goal, it is necessary to have mechanisms in place that can leverage data into powerful insights.

Recommender Systems (RSs) have shown to be very useful mechanisms to serve that purpose. RSs are a type of Machine Learning (ML) algorithm that, based on user's preferences and behaviors, allow companies to make personalized recommendations of products and publicity, contributing to customer satisfaction and company success (Bobadilla et al. 2023).

As a data-driven solution, RSs highly rely on data quantity and quality, which requires the collection of large quantities of sensitive data. However, this process is associated with several legal and ethical challenges, especially regarding privacy, as many companies are not transparent in the way they collect and use data. Google alone, for example, tracks about 40% of web traffic (Fox 2022). Most users are aware of this problematic and feel unsafe, in fact, statistics show that 93% of Americans considered it important to be able to control who could access their personal data and 66% of consumers wanted more governments to pass laws (Fox 2022).

To combat this issue, privacy policies, such as General Data Protection Regulation (GDPR), started to emerge to protect consumers' privacy. GDPR has proven its effectiveness in enforcing privacy protection, as demonstrated by the case of Marriott International, who was fined £18.4 million by the Information Commissioner's Office for failing to secure guests data (Page 2020).

This need for quality data poses many other challenges, for example 46% of industry marketing professionals find lack of data the key challenge (DMA 2022). Data brokerages have emerged as a solution, consolidating information from diverse sources, and making it available for companies to buy for various purposes (Usercentrics 2021). Digital ad platforms, like Facebook, also offer opportunities for targeted advertising, as they monetize user traffic by displaying targeted ads to visitors (Diao 2022).

Having this increasing need for privacy protection in mind, in 2019, Rodrigo Moretti and Eduardo Pinto Basto, launched the pilot for their start-up, Modatta. Modatta's mission is to enable both people and companies to interact and share their data while promoting privacy, transparency, and a fair share of value (Modatta 2023). So, it contrasts with businesses like the ones mentioned above of data brokerages and ad platforms that monetize online users' data only for their own good.

Modatta stands out by offering an efficient marketplace for data exchange where users have complete control over what data they share with companies. To achieve this, Modatta acts only as an intermediary without ever collecting and storing sensitive data from users. This is beneficial for both users, who regain power and can benefit monetarily from their data, and for companies, who can reduce their costs and risks of managing data and still efficiently target their audience while at the same time building positive relationships with their customers.

Modatta's current user targeting approach relies solely on direct segmentation using profile characteristics specified by companies, resulting in inefficiency, lower profits, and potentially

low satisfaction. This work project proposes to enhance Modatta's platform user experience by improving and evaluating an existing privacy-preserving framework that contemplates the usage of generative models (GM) trained in a decentralized manner that enable to securely use data to accurately target users without ever compromising their right to privacy.

Based on literature review, CTGAN emerged as a top-performing GM for tabular data, Federated Learning (FL) training also appeared to be the best decentralized approach to use when data is distributed among various devices. Findings also revealed that, using FL, CTGAN's performance suffered when data quantity was relatively small. This pose as a challenge to Modatta as their users also own a small quantity of data in their devices.

The rest of the work has been structured as follows: **section 2** discusses findings of prior work projects, **section 3** presents a literature review on GM and decentralized ML approaches, **section 4** performs an exploration and treatment to data, **section 5** defines the experimental procedures undertaken to address the research objectives, and presents and discusses the results, **section 6** draws conclusions from previous sections and proposes recommendations for future research, and finally **sections 7** and **8** explore concrete RS models.

## 2. Previous Works

In this segment, the focus will be on reviewing the scope and outcomes obtained by Carolina Lucas, Zhenze Wu, Patrícia Macedo, and Emila Aguiar in their individual and group contributions for their final work projects (Aguiar 2021; Lucas 2021; Macedo 2021; Ze 2021).

Bounded by Modatta’s commitment and responsibility to secure users’ right to privacy, the main objective of the group was to investigate the possibility of Modatta to obtain valuable and usable representations of their users while fully ensuring privacy preservation. To do so, they have designed a privacy-preserving framework (Appendix 1).

Depicting the delineated framework, they have started by proposing the usage of a Generative Adversarial Network (GAN), trained in a decentralized way, more specifically with Partially Local Federated Learning (PLFL) to generate users' synthetic data, that could be used to train a RSs. For the RSs, they have discussed the possibility of using two ML algorithms, Deep Interest Networks (DIN) and HyperML. DIN is a neural network (NN) architecture RS that by using an activation unit takes into consideration the relevance of historical behaviors of a user to adaptively calculate the respective representation (Zhou 2018). In turn, HyperML is a metric learning model for one-class collaborative filtering that represents users and items' embeddings in hyperbolic space (Vinh Tran, et al. 2020).

Upon examination of the chosen methods, it became apparent that the GAN architecture chosen was originally made for image generation rather than tabular data, which hindered the synthetic generated representations of users (Xu et al. 2019). This represents an issue since Modatta users' personal data is structured in a tabular format. The synthetic generated data was also only used to evaluate the DIN model. Regarding the testing of the usefulness of the PLFL framework, the model chosen for training was not a GAN, as suggested by the framework, but a Matrix Factorization, a model typically used for making user recommendations. Furthermore, it is important to note that in the previous works, the framework was neither trained as a whole nor real data was used. To culminate the lack of real data, random samples were generated to simulate Modatta users' interests and assess the constructed models. Consequently, this prevented us from having a true perception of the framework's performance and viability.

In sum, the key insight from their works was the design of a theoretically sound framework that would allow Modatta to train a RS model using synthetic data generated by a GAN trained with a FL approach. This framework ensures that Modatta never has direct access to users' data for model training, thereby never compromising their privacy while still providing personalized offers in the app. Therefore, the focal points of this project involve making necessary

adjustments to the models used to better correspond to the reality of Modatta users' data, which in turn will enable a more accurate analysis of the real performance and potential of such framework.

## 3.Literature Review

In this section a thorough and meticulous evaluation of the existing scientific research works on GANs, and FL will be outlined. This includes their definitions, variations, implementations, and applications adaptable to the case in hand, concretely, of tabular data. Furthermore, the interconnection between GANs and a Federated Learning System (FLS), along with the primary risks associated, will be explored. The discoveries derived from this exploration will aid in determining the most suitable approach for achieving the objectives of this project.

### 3.1.GAN

#### 3.1.1. History of GANs

The first GAN was invented by Ian Goodfellow in 2014 and, since then, has sparked enthusiasm and imagination among academic and industry experts due to its versatility and possibility of usage (Langr and Bok 2019, 12). GANs belong to a group of ML models denominated by GM, where Bayesian Networks and Variational Autoencoders (VAEs) are also included and are designed to generate synthetic data from some distributions of real data (Xu 2020).

GANs appeared in the context of image generation and to illustrate their "power", they have enabled the generation of hyper-realistic fake face images with high quality (Appendix 2), something never achieved before with the existing GM.

#### 3.1.2. Vanilla GAN Architecture

A Vanilla GAN architecture consists of two simultaneously trained NNs, the Generator (G) and the Discriminator (D). The G's main purpose is to generate realistic synthetic data that resembles the characteristics of the training real data and is undistinguishable by the D. The D's

goal is to correctly classify an example as real from the training sample or fake from the synthetic data generated by the G. Training GANs can be a challenging task since both NNs are trained by backpropagation by using the D's loss. Besides, the two networks have a competing objectives, i.e., they are playing a zero-sum min-max game where, as one improves, the other deteriorates (Langr and Bok 2019, 11). Equation 1 in Appendix 3 represents the loss function to be optimized.

A GAN should be trained until convergence, where it reaches what is called in game theory, a Nash Equilibrium, which occurs when the G produces fake examples that are undistinguishable by the D, and the D can at best randomly guess if an example is real or fake. This is extremely difficult to happen, though this has not impeded the usage and usefulness of GANs even if not trained until convergence, as they still achieved remarkable empirical results, both in the quality of data generated and in processing speed results (Langr and Bok 2019, 11-12).

### 3.1.3. Variations of GANs for Tabular Data

Since the appearance of the first GAN variations have been developed with modifications to the original architecture to adapt to different types of data and improve performance. Data represented in rows and columns is designated tabular data and it represents one of the most common data formats used to store and treat data (Banerjee 2021). Besides possibly containing both discrete and continuous values in their columns, it is rather usual to find sensible data present as well (Xu et al. 2019). medGAN, TableGAN, and CTGAN are all examples of variations of GANs specifically developed for tabular data.

The ability to generate synthetic realistic data can be seen as useful when: 1) there is insufficient data available, for example, due to high acquisition costs; 2) there are quality issues in existing data, for example, missing values; 3) data is imbalanced, which is proven to generate poor

quality models if this problem is not treated before training; and finally 4) there are privacy issues, that invalidate the direct usage of data (Xu 2020).

medGAN was one of the first adaptations of GANs for discrete data to appear with the purpose of generating synthetic health patient records. To capture the different distributions of multi-label discrete variables, it proposed the addition of an autoencoder to the GAN architecture, which is trained before the GAN (Choi et al. 2017).

TableGAN surged in 2018 and was based on DCGAN, a GAN using Convolutional Neural Networks (CNN) for image generation. Therefore, for synthesizing tabular data, TableGAN proposed using CNN architectures for the G and D, as well as the training of an additional NN mentioned as the Classifier (C) trained to learn the correlation between labels and other attributes from the original data. Regarding the loss function, they adapted one from DCGAN and designed two more loss functions that enhanced the model's security and privacy (Park et al. 2018).

CTGAN was proposed one year later, in 2019, and sought to solve the flaws of the above-mentioned models that could not simultaneously tackle mixed data types in in different columns, non-Gaussian distributions of columns, multimodal distributions of columns, sparse one-hot encoded vector as input, imbalance categorical columns, and high dimensionality of tabular data (Xu 2020), as shown in Table 1.

| Problems | medGAN | TableGAN | CGTAN |
|---|---|---|---|
| Mixed data types | ✓⋆ | ✓⋆⋆ | ✓ |
| Non-Gaussian distributions | x | x | ✓ |
| Multimodal distributions | x | ✓ | ✓ |
| Sparse one-hot-encoded vectors | x | x | ✓ |
| Imbalanced categorical columns | x | ✓ | ✓ |
| High dimensionality | ✓ | ✓ | ✓ |
| Lack of training data | x | x | x |
| Missing values | x | x | x |

Table 1 – This table summarizes the capabilities of different models to solve different problems. (⋆ indicates it can model binary and count variables; ⋆⋆ indicates it can model binary and continuous variables). Adapted from (Xu 2020).

To overcome those issues, CTGAN suggested a mode-specific normalization, a conditional generator, and a specific training-by-sampling methodology. Correctly representing data for the training of NN is crucial, and while discrete columns are easier to represent as one-hot vectors, continuous columns are more complicated. Opposite to the min-max normalization adopted by medGAN and TableGAN, CTGAN defines a customized mode-specific normalization for continuous columns where each column is processed independently and where each value is represented as a one-hot vector indicating the mode, and a scalar indicating the value within the mode. The conditional generator introduced helps to solve the problem of random sampling the input vector for the G that does not account for the imbalances on categorical columns leading to poor quality of synthesized data. The objective of the conditional generator is to sample all categories evenly during the training process, hence, the G is generating data subjected to a condition. Because of this special generator, a training-by-sampling approach by the discriminator should also be adopted to adequately access the outputs of the generator. It is also worth mentioning that the D in CTGAN is defined as a critic neural network like in the Wasserstein GAN (WGAN) case. Both networks' structures are presented in Appendix 4 (Xu et al. 2019).

Upon reviewing papers and comparing results from GMs for tabular data that incorporate GANs, it appears that CTGAN stands out as one of the most complete models in this context.

### 3.2. FL

#### 3.2.1. An Introduction to FL

Many factors, like the development of smartphones, the appearance of Internet-of-Things (IoT) devices, or, as already mentioned, the development of the Internet, have led to not only an increase in the generated amount of data but also in its dispersion (Liu et al. 2022). On the other hand, and as a response to concerns in data security and privacy, multiple laws and regulations have been created in the past years, such as GDPR in the European Union, the California Consumer Privacy Act (CCPA), or the Personal Data Protection Act (PDP) in Singapore. To some extent, these legal frameworks aim to address the vulnerabilities inherent in traditional ML models, which typically employ centralized training methods. These centralized training methods involve aggregating and storing all training data in a centralized location, with the training process managed by a singular entity (Data Science Digest 2021). However, this approach presents several challenges related to computational power, training duration, and as previously noted, data security and privacy concerns (Liu et al. 2022).

Growing awareness of data protection legislation that makes data aggregation almost impossible, especially if sensible data is in question, highlights the need for alternative, more secure methods of managing data when training ML models (Yang et al. 2019 cited in Lyu, Yu, and Yang 2020).

The idea of FL appeared in 2016 (McMahan and Ramage 2017) as a possible solution for this emerging issue and proposed collaborative training ML models while taking advantage of distributed data and distributed computing resources. In FL, the main idea is to bring the code to the data instead of transporting the data to the code. Opposed to a centralized machine learning approach, FL is considered a distributed machine learning approach. In practice, the FL model is trained across multiple local devices where the data is stored, and only intermediate data, such as gradients or models, is exchanged between devices. This way, by keeping training

data decentralized, FL effectively enhances privacy and security, as it reduces the risk of exposing sensitive information (Liu et al. 2022).

In essence, FL differentiates itself from centralized learning through three main aspects: 1) it does not ever allow direct training data communication across devices, 2) it takes advantage of the computing resources of different devices, and finally 3) when communicating it takes advantage of encryption or other defense mechanisms to avoid security issues.

It is crucial to acknowledge that the emergence of the FL concept and consequently the first FLS was centered around the paradigm of unbalanced and non-Independent and Identical Distributed (non-IDD) data possessing similar features, distributed across multiple mobile devices (McMahan et al. 2017). Although, over time, the concept has evolved, resulting in the development and increase of various FL frameworks and systems.

According to Li et al. (2023) FLSs can be classified considering six aspects: 1) data partitioning, 2) ML model, 3) privacy mechanisms, 4) communication architecture, 5) scale of federation, and 6) motivation of federation. These are important because, depending on the specific problem we encounter, it is necessary to build an FLS accordingly. We will delve deeper into the differences these aspects represent, and, in addition, a visual scheme of this taxonomy is presented in Appendix 5.

The type of parallelism techniques that the FLS uses in the training process depends on how data is partitioned, i.e., how the training data is distributed on the feature space across the devices. In horizontal FLS, the distributed data has the same feature space, but the sample intersection is small, thus data parallelism is exploited. Note that in the case of horizontal FL, an aggregation algorithm must also be defined to aggregate the models or the gradients locally trained, the decision of this algorithm is also related to the communication structure of the system, but some of the possibilities for centralized algorithms are FedAvg, FedBCD or

FedProx. Conversely, if distributed data has a similar sample space but a different feature space, which is the case of a vertical FLS, model parallelism is used. Nonetheless, in the case of a hybrid FLS, where data may be both horizontal and vertically partitioned, no parallelism technique is used but rather transfer learning is used as a possible solution.

The type of ML model chosen to be trained using FL also influences the FLS structure and needs to be chosen consonant to the problem and type of data in question.

In FL, local training data is never shared, though there are still some concerns related to the sharing of the model's parameters and gradients since there is the possibility of a variety of attacks, such as membership inversion attacks, that can leak sensitive information from this data. These attacks are varied and can happen in several steps of the FL training. To overcome this issue, additional privacy mechanisms have been studied to be added to FLSs, the most used are Differential Privacy (DP) and Cryptographic Methods (CM). Each of these methods offers different privacy guarantees, DP for instance, guarantees individual privacy, while Secure Multi-party Computation (SMC), an example of a CM, only secures the aggregation of transferred gradients and not the final model privacy. However, when using these types of privacy mechanisms, other problems may arise, for instance regarding the quality of the model in the case of DP, and communication and computation overhead for the case of CM.

An architecture of communication must also be defined, it can either be centralized, the most common, or decentralized. In centralized design, local devices share their intermediate data with a central party, like a server, that is responsible for its aggregation and returning those results to all participating devices. As opposed, in a decentralized design, communication of parameters or gradients is shared among the devices participating, meaning that every device can make updates directly according to a certain predefined protocol (Li et al. 2023) (Appendix 6).

If the number of parties, that is devices participating in the training, are relatively small and possess a relatively large amount of data and computing capacity, we are talking about a cross-silo FLS, alternatively, if the number of devices participating in the training is relatively large and have a relatively small amount of data and computing capacity, usually being mobile devices, we are in the presence of a cross-device FLS.

To finish, when designing a FLS, the motivation of the federation should also be taken into consideration since it can influence the success of the model trained. Motivation can either be created from regulations, incentives, or even both. For example, inside companies, different parts of the company may have the motivation, because regulations may not allow the direct sharing of different data, to participate in the FL training of the certain model that is believed to be useful. Nonetheless, when examining Google Keyboard (Gboard), it becomes clear that incentives play a crucial role, meaning that users cannot be forced to participate in the federated learning process, nevertheless, the ones that effectively do participate will benefit from it as they will have access to a better version of Gboard (Yang et al. 2018). Note that the definition of incentives is challenging and should always be fair.

Up to this point, a clear and concise definition of FL and a comprehensive view of the different aspects to consider when developing FLS have been provided.

### 3.2.2. Using FL to train a GAN

Before going over previous work in the field of training GANs with FL, it is necessary to introduce two concepts: standard FL, and general distributed GAN. The first one refers to the typical FL structure with a centralized communication system. Furthermore, as outlined above, a vanilla GAN architecture consists of at least two deep NNs, the G, and the D. When trained with FL, there are several approaches that can be taken to train the GAN's NNs regarding their location, i.e., the D is always trained locally, as this is the only component that requires contact

with personal data, but the G can be either trained locally with the D, or trained in the server, which leads to the introduction of the general distributed GANs notion (Rasouli, Sun, and Rajagopal 2020). In this concept, there are various Ds distributed across clients, which are trained locally and aggregated at the server, and a single central G trained in the server.

With these initial concepts clarified, the focus can be redirected to the main topic at hand. Scientific research that explores the combination of FL and GANs started appearing in 2019 with the introduction of FedGP (Triastcyn and Faltings 2020). This framework uses a standard FL setting with Generative Differential Average-Case Privacy as a privacy guarantee. In this case, the local model parameters are aggregated using the FedAvg algorithm in a generally distributed GAN architecture. Results showed that high-quality artificial image data was generated with success and that information leakage can be reduced when training with a federated GAN.

Subsequently, further improvements were explored. For example, using a similar structure to FedGP coupled with DP as a privacy guarantee, Augenstein et al. (2020) focused on improving the model's quality by investigating mechanisms to effectively debug common data issues in normal ML projects. Moreover, to mitigate the challenge of distributed data training associated with the heterogeneity of local data distributions, Zhang et al. (2021) proposed a new approach for parameter aggregation, Universal Aggregation (UA). UA-GAN uses a standard FL setting and general distributed GANs with UA as the aggregation algorithm, which consists of aggregating the feedback of all private discriminators through their odds value. Experiments show that not only does UA-GAN meaningfully outperform AVG-GAN and MD-GAN (Zhang et al. 2021), other known federated GAN models, but also achieved a performance close to the centralized GAN (Appendix 7).

Furthermore, Rasouli, Sun, and Rajagopal (2020) introduced a novel GAN structure FedGAN, this uses local generators and discriminators which are periodically synced via an intermediary that averages and broadcasts both model's parameters. When compared to a generally distributed GAN, FedGAN results show similar performance but lower communication complexity. Additionally, the experiments were done either on image data or time series data.

While GANs to synthesize images using FL systems have been widely explored, the capacity of tabular GANs to learn from decentralized data sources is still under study. By searching the keywords 'federated learning' and 'tabular GAN', only two relevant papers to the current context were found. The first one was published in 2019, and it developed Fed-TGAN, a framework with two novel features, (i) a privacy-preserving multisource feature encoding for model initialization; and (ii) table similarity aware weighting strategies to aggregate local models to counter data skewness (Zhao et al. 2021). This framework was evaluated using the state-of-the-art tabular GAN with distinct federated frameworks besides the proposed one. The results showed that Fed-TGAN had a high capacity to generate quality synthetic tabular data in a more efficient and privacy-preserving manner.

Later, at the end of 2022, Duan et al. (2022) published HT-FED-GAN, a federated generative model for decentralized tabular data synthesis as the previous one, but incorporating DP as privacy guarantee, and more focused on tackling the problem of multimodal distributions, characteristic of continuous columns, and the high imbalance in categorical attributes, which is achieved by using a variational Bayesian Gaussian mixture model (fed-VB-GMM) and a federated conditional one-hot encoding with conditional sampling. On top of that, Duan et al. (2022) also investigated the model's privacy level against a membership inference attack evaluated using the F1 score metric. When compared with DP-FedAvg-GAN (Augenstein et al. 2020), results show that HT-FED-GAN has the best trade-off between quality and level of privacy.

Note that, particularly for tabular data, most experiments are conducted with large amounts of data and low number of clients. In the analyzed papers, the average number of clients participating was 4, and most of the scenarios had clients with over 500 hundred records each. One of the experiments in Zhao et al. (2021) simulated a client with 40k records that were all duplicates of a single row from the original dataset. Although, results showed that Fed-TGAN was able to mitigate the low-quality data of that user by using weighting strategies for the aggregation. In contrast, Duan et al. (2022) considered a scenario where each client had very few records, being the minimum 31 and the maximum 126, and still achieved good results.

Additionally, many of the encodings in this type of framework require the sending of the encoding labels to the server which presents a big threat to privacy.

### 3.2.3. Challenges of FL training

FL is a promising solution to preserving privacy of users when training ML models, yet its design still presents inherent vulnerabilities that need to be addressed. FL by nature offers a more privacy-oriented paradigm compared to centralized learning, though it still proves to be insufficient sometimes.

One notable limitation of federated learning is the potential vulnerability during the communication of model updates, which can lead to the leakage of sensitive participant data to malicious third parties or even to a compromised server, enabling various forms of attacks on the model's integrity and security (Lyu, Yu, and Yang 2020).

On the other hand, unstable and slow communication deriving from the heterogeneity of devices can also pose as a big challenge when training the models which conversely may lead to suboptimal training outcomes of models (Lim et al. 2022, 1-12). Moreover, the heterogeneity of devices can bring additional constraints to the training process, including variations in

computing capabilities and willingness to participate, which further complicate the training and impact the overall performance (Lim et al. 2022, 1-12).

In conclusion, these challenges represent potential issues to convergence and scalability of FL models. Nonetheless, they do not undermine the potential benefits of FL but, instead, highlight areas that require future research for future improvements in the field.

## 4. Data

This section presents an overview of the data used in the project and outlines the cleaning steps undertaken on data provided by Modatta. The cleaning process involved removing unmatched categories, addressing uncertainties with request identifiers, ensuring statistical significance of users and categories, and finally reformatting data structure. The resulting dataset represents unique users in rows, and their interests in Facebook categories in columns.

### 4.1. Data Description

As of now, to make campaign recommendations, Modatta takes advantage of segmentation using the user's profile information (Pinto Basto, video interview, February 25, 2023). It is important to highlight that all profile information shared in the Modatta app is optional and it never leaves users' mobile phone app. Hence, no profile information is ever collected or saved by Modatta on any central server, it is only used whenever the user decides, and if a user decides to delete Modatta's app, all information will be forever deleted (Modatta 2023).

Profile information provided by users in Modatta's app can be distinguished in two ways concerning the manner it is provided. It can be manually inserted, i.e., the user answers questions that help build their profile, or alternatively, users can choose to allow the app to automatically connect to their device or other apps, enabling, this way, the app to store information of users from these so-called connectors, which is also used to define their profiles in the app. As this is being written, Modatta has only enabled 3 connectors: the device, the

device location, and Facebook. Figure present in Appendix 8 illustrates the in-app experience of fulfilling profile information in both ways.

One of the purposes of this work project was to evaluate the privacy-preserving framework developed for Modatta with real data. Nevertheless, as mentioned, Modatta does not possess any data from users and to provide us with this kind of data Modatta needed to request authorization individually. Unfortunately, this option turned out to be unviable.

As a solution, Modatta provided some alternative data consisting of two documents:

- dump_fb_translations: A Comma-Separated Values document that has anonymous information on requests made by users to clarify which information from Facebook was taken and stored in the Modatta app.
- facebook_categories: An Excel document, that allowed to make the connection of the information of the requests and how they are identified by Modatta.

The first document is valuable since it permits to extract some profile information, more concretely Facebook categories of interest of real Modatta app users, while the second document is essential to understand the encoded information in the dump_fb_translations file, as it provides information about the meaning of each field.

Additional detailed information regarding each file is present in Appendix 9.

### 4.2. Data Treatment

The cleaning of dump_fb_translations file was necessary to define unique users and associate them with profile information, therefore a step-by-step explanation of this process is presented.

Firstly, in an exploratory data analysis on the data of dump_fb_translations file, it was discovered that there were in total 1297 categories, i.e., unique values in the *category* column,

and 859 requests identifiers, i.e., unique values in the *invocationId* column. It is worth mentioning that these categories can be organized in a hierarchical structure.

Regarding the categories present in the dump_fb_translations, 250 out of the 1297 were not present in the facebook_categories file that allowed to decode these categories, therefore, a decision was made to eliminate every row of the dump_fb_translations that referred to those 250 categories. As a result, the dump_fb_translations file ended up with 1047 categories.

On the other hand, although each request identifier information is allusive to a single user, there was uncertainty regarding whether multiple request identifiers could represent the same user. This uncertainty arises because requests are timestamped, and a user may make requests multiple times at different points in time. As a result, each request is assigned a unique request identifier, potentially leading to the same user being represented by several request identifiers.

To address this and to obtain a sample that with some level of confidence represents unique users, some assumptions were made as to what were considered requests from the same user. Therefore, subsequently to the first step, information from each unique request identifier was aggregated into sets and, the Jaccard Similarity Coefficient (JSC), a common metric used to measure the similarity between two sets, was calculated between every pair of sets. An explanation of the formula of the JSC and a graphical visualization of the results are shown in Appendices 10 and 11. Considering the results, it was determined to eliminate one unique request identifier of the pairs that had a JSC above 0.85, which lead to the deletion of 230 requests identifiers, remaining 629 unique requests identifiers that, from this point on, will be considered unique users.

Proceeding with the data treatment, a more careful investigation of the relationship between users and categories was conducted as a third phase, which can be seen in graphics of Appendix 12. After establishing that 30 occurrences represented the minimum threshold for achieving

statistical significance in this study, for both the number of users who liked a particular category and the number of categories liked by a specific user, it was proceeded with the withdrawal of categories that were not liked by at least 30 different users and the elimination of users who were not interested in at least 30 categories. By doing so, and certifying both conditions were satisfied, the dump_fb_translations data contained the representation of 540 unique users and 558 unique Facebook categories (Appendix 13).

To finish, a new dataframe was created that resulted from the merge of the cleaned dump_fb_translations and facebook_categories files. In this dataframe, named "cleaned_df", with shape (540,558), each row represented a unique user and each column corresponded to a single Facebook category. The values of all columns are binary, with 1 indicating that the user has interest in that specific category, and 0 suggesting that it is unknown if the user has interest in that category.

More information regarding this new dataframe can be found in Appendix 14. This dataframe serves as a starting point for the experiments conducted.

## 5. Methods and Results

This section first aims to conduct a study to identify the optimal GM for the current use case. Once the best model is determined, it will be integrated into a FL framework and assessed across multiple scenarios.

### 5.1. Generative Models

#### 5.1.1. Generative Models Selection

To confirm that the state-of-the-art tabular GAN, specifically CTGAN, is the most suitable for the case in hand, it was compared with three other known models. Among the chosen ones for the comparison, one is a statistical model, the Gaussian Copula (GC), while the other two are

deep learning models, namely, the Triple-Base Variational Autoencoder (TVAE) and a simple image GAN.

VAE is an autoencoder, a model that learns to compress and reconstruct data, that is enhanced by controlling the compressed representations using regularization techniques. TVAE is an improved variation of VAE that uses a triplet loss function on the average representations to capture important features (Rocca 2019).

Classical Copula correlation models are characterized by preserving the individual characteristics of each variable while establishing a joint probability distribution for a selected group of variables. The GC, for instance, maps the marginal distribution of each variable to the standard normal distribution (Analystprep 2019).

The CTGAN and GAN are generative adversarial networks aforementioned in section 3. 1..

The CTGAN, the GC, and the TVAE were trained using the built-in functions from the Synthetic Data Vault (SDV) library, CTGANSynthesizer, TVAESynthesizer, and CopulaGANSynthesizer, respectively. The simple image GAN was constructed based on a simple GAN for images with the necessary adaptations made for handling tabular data (Langr and Bok 2019, 36-50).

### 5.1.2. Generative Models Evaluation Procedures

To evaluate the performance of the different models, the cleaned_df resultant from the data treatment step, was chosen to be used. Upon examination, cleaned_df proved to be highly squared, meaning that the proportion of rows, users, to columns, categories, is very close in magnitude (30:31). This can present a challenge to the training process of any of the chosen models, therefore, a column reduction was made to achieve a reasonable number of columns that allowed training accurate models with the available number of rows.

A few techniques were explored to make this reduction, namely, 1) random selection of columns, 2) election of the columns with the highest frequency of the value one, 3) categories hierarchy flattening and finally 4) columns selection based on proximity to average. As the first three failed to do an accurate representation of the real data, the fourth was the preferred technique for the selection of columns. This selection was made by keeping the columns whose frequency of the value one was close to the mean count of users per column. The number of columns kept was selected with the goal of maintaining a similar proportion of one value as the original data. Therefore, the dataframe obtained from the chosen reduction, named reduced_df, had 522 users and 34 categories.

In addition, the datatype was changed to Boolean, as this data type better aligns with the specific requirements and conventions of the SDV library, ensuring proper compatibility and evaluation of the GMs. Further information on the column reduction approaches and the final dataframe obtained can be found in Appendix 15.

To properly evaluate the models' results, it was decided to evaluate the synthetic generated samples generated from each trained model based on four metrics: 1) TVComplement, 2) ContigencySimilarity, 3) CategoryCoverage, and 4) NewRowSynthesis. All these metrics come from the SDMetrics library, a library built with the concrete mission of evaluating synthetically generated tabular data.

TVComplement ignores missing values and calculates the similarity of a real column and the corresponding synthetic one in terms of shape. A value equal to 1 indicates that real and synthetic data shape are equivalent and a value equal to 0 point out that real and synthetic data shape are entirely distinct.

CategoryCoverage evaluates if synthetic data columns cover all the possible original categories that exist in the corresponding real columns. For the current case there are only two categories

in each of the columns of the dataframe in use, True and False. A value of 1 means that all the synthetic columns cover all the possible categories, and a value of 0 means the opposite.

NewRowSynthesis assesses how many rows of the synthetic generated data are exactly equal to rows in the real data. The closer this metric is to 1 the fewer rows sampled data and real data have in common.

ContingencySimilarity determines the similarity of a pair of columns both in real and in synthetic data, however, columns must be compatible and missing values are treated as an additional category. A score of 1 means that all contingency tables between every pair of columns are the same both in real and synthetic data and 0 means that they are opposites.

Appendix 16 contains mathematical formulas and additional information related to these metrics.

With that being stated, each model was trained with a batch size of 50, when possible. For each trained model, five samples of 522 records were sampled and individually evaluated using enumerated metrics. The results correspond to the simple average of the metrics of each sample for each model.

### 5.1.3. Generative Models Results and Discussion

Table 2 presents the results of the metrics for the trained models. By examining the results, it becomes clear that the image GAN model exhibits a notably poor performance. This outcome aligns with the earlier theoretical discussion in section 3.1, highlighting the challenges faced by normal image GANs when generating synthetic data for tabular datasets. Note that, the score in NewRowSynthesis can be misleading, as the low scores obtained in the other three metrics indicates that the model could not capture the patterns of each column, 0,4265 and 0,5382 scores obtained for the TVComplement and CategoryCoverage respectively, and relations between them, indicated by ContingencySimilarity score of 0.

Despite demonstrating good results on ContingencySimilarity, TVComplement and CategoryCoverage metrics, the TVAE model presents a very poor result in the NewRowSynthesis indicator of 0,1364, which means that 86.36% of the generated rows are equal to original data rows, which consequently influences the positive results of the above-mentioned metrics.

When considering, the GC and CTGAN models both achieved superior overall performance.

| Models | Contingency Similarity | TV Complement | New Row Synthesis | Category Coverage |
|---|---|---|---|---|
| GC | 0,8514 | 0,9229 | 0,9835 | 1,0000 |
| TVAE | 0,9186 | 0,9838 | 0,1364 | 1,0000 |
| CTGAN | 0,9186 | 0,9838 | 1,0000 | 1,0000 |
| GAN | 0,0000 | 0,4265 | 1,0000 | 0,5382 |

Table 2 – Evaluation Metric Results of the GM Models trained

As the goal is to input the data generated from the chosen model into the RS, it is key that the synthetic data successfully captures the patterns and dependencies present in the original data, enabling the RS to target users accurately. In addition, in line with Modatta's commitment to data privacy, it is crucial that the generated records do not match the original ones. Therefore, striking a balance between similarity and novelty becomes essential. This trade-off ensures that the established data-privacy policies are upheld while maintaining the efficiency and effectiveness of the RS.

Summarizing, although GC model achieved respectable scores, 0,8514, 0,9229, and 0,9835 on ContingencySimilarity, TVComplement and NewRowSynthesis, respectively, it was outperformed by CTGAN model, that obtained scores of 0,9186, 0,9838 and, 1 in the same metrics. Hence, the CTGAN model emerged as the preferred choice based on its superior performance.

## 5.2. Federated Training of CTGAN

### 5.2.1. Federated Training Methodologies

To build a federated framework for a CTGAN for this project, it was decided to use the Flower Library. Flower is an interoperable friendly federated learning framework that oversees several challenges inherent to FL, such as data and device heterogeneity and limitations of computational resources, hence, allowing the execution of large-scale FL experiments in heterogeneous device scenarios.

Locally, each client trains the model with real data, the local parameters resulting from this computation are then sent to a central server which is responsible for parameters aggregation. The resulting aggregated parameters are then sent to each local device that updates its local ones and retrains the model. This cycle is repeated for a determined number of rounds. For parameter aggregation, algorithms like FedAvg are provided by Flower.

The process described above is orchestrated on the server side by three main parties, the ClientManager, the FL loop, and a user-customizable Strategy (Beutel et al. 2022). Each connected client is represented by a ClientProxy object that handles the communication of Flower Protocol messages for the client. Each set of these objects is administrated by the ClientManager from where the server components sample clients.

The FL loop is the core of the FL process. It manages the entire learning process but does not make decisions about how to proceed. Rather, it delegates these decisions to the implementation of the configured strategy. The FL loop asks the Strategy to configure the next round of FL, sends those configurations to the affected clients, receives the resulting client updates (or failures), and delegates result aggregation to the Strategy.

The communication between the client and server is executed under a serialization process, i.e., the clients receive messages as raw bytes, deserialize and execute them, the results are then

reserialized and communicated back to the server. Additionally, Flower ensures client privacy by incorporating secure aggregation protocols, namely SecAgg and SecAgg+, during the training process.

Another important component is Virtual Client Engine (VCE) that allows for the maximization of the use of available hardware through the virtualization of clients, thus simplifying the parallelization of jobs and enabling large-scale FL workloads to be run with minimal overheads in a scalable way. An illustration of Flower architecture can be found in Appendix 17 (Beutel et al. 2022).

In this work, the integrated model in the Flower Framework corresponds to the CTGAN from the SDV library but due to the necessity of accessing some of the model parameters, the development files of the model were adapted (SDV-DEV 2023).

### 5.2.2. Federated Training Evaluation Procedures

To evaluate the quality of CTGAN under a FL setting, a dataset, named as augmented_df, of 3000 users generated using CTGAN previously trained and evaluated in section 5.1. was used. The augmented dataset was divided among different numbers of clients to generate four distinct scenarios, described in Table 3.

| Scenarios | Training | # Clients | # Rows per Client | # Rounds | # Epochs | Batch Size |
|---|---|---|---|---|---|---|
| 0 | Centralized | - | - | - | 300 | 500 |
| 1 | FL | 3000 | 1 | 30 | 10 | 2 |
| 2 | FL | 75 | 40 | 30 | 10 | 40 |
| 3 | FL | 15 | 200 | 30 | 10 | 200 |

Table 3 – Experiment Scenario Definition

Each of the three scenarios under FL training used the Flower simulation module, set with 30 rounds of training each with a sample of 30% of the total number of clients, and a FedAvg aggregation strategy. In each round, the CTGAN model was trained for 10 epochs in each

selected client. In addition, the batch size for each epoch varied across the scenarios, with values of 2, 40, and 200. Furthermore, the previously mentioned scenarios were compared to a baseline simulating a centralized approach, scenario 0. This scenario involved training the CTGAN model with the entire dataset using the default batch size and number of epochs, 500 and 300, respectively.

To evaluate the performance of the trained models in each of the defined scenarios, a sample with 3000 rows was generated from each trained model and evaluated using the same metrics described in section 5.1.2., together with CategoricalCAP, a metric also from the SDMetrics library designed to measure the risk of disclosing sensitive information through an inference attack. This last metric assumes the attacker possesses a certain number of columns of real data along with the entire synthetic data. This metric is suitable for categorical and boolean data and returns a score from 0 to 1, where 0 represents high unsafety and 1 full safety from the attack.

### 5.2.3. Federated Training Model Results and Discussion

The results of the models trained under the different scenarios can be found in Table 4.

| Scenarios | Contingency Similarity | TV Complement | New Row Synthesis | Category Coverage | Categorical Cap | Training Time (s) |
|---|---|---|---|---|---|---|
| 0 | 0,8559 | 0,8995 | 0,9987 | 1,0000 | 0,9986 | 108,00 |
| 1 | 0,2248 | 0,4762 | 0,0000 | 0,5000 | 0,0000 | 6688,37 |
| 2 | 0,7138 | 0,7715 | 1,0000 | 1,0000 | 1,0000 | 297,39 |
| 3 | 0,7140 | 0,7743 | 1,0000 | 1,0000 | 1,0000 | 281,40 |

Table 4 - Experiment Scenarios Metrics Results

It comes as no surprise that the model trained using centralized learning, scenario 0, demonstrates a high capacity to generate synthetic data that closely resembles the original dataset. This is evident as it presents a CategoryCoverage score of 1, meaning that the model captures all possible categorical variables of original data, a ContingencySimilarity of 0,8559 and a TVComplement of 0,8995, demonstrating that it can learn columns patterns and preserve

their relations, and a score NewRowSynthesis score of 0,9987 which indicates that the model mostly generates new rows different from the originals.

For model training under FL, scenarios 2 and 3 emerged as the top performers. Both achieved a score of 1 for CategoryCoverage and NewRowSynthesis metrics. Scores of 0,7138 and 0,7140 for ContigencySimilarity, respectively, and TVComplement scores of 0,7715 and 0,7743, respectively, were also achieved. These scores exhibit reasonably good performance regarding models' ability in learning and representing column patterns and relationships.

The model trained in scenario 1, the scenario which closely resembles Modatta case, exhibits the poorest performance across all evaluation metrics compared to the other trained models. The ContingencySimilarity score of only 0.2248 indicates a significant inability to capture the relationships between columns. Additionally, the TVComplement and CategoryCoverage scores of 0.4762 and 0.5, respectively, highlight the model's inability to learn the distributions and categories values of the columns effectively. Furthermore, the model completely fails to generate novel data, as indicated by a NewRowSynthesis score of 0.

Considering the satisfactory performance of the models in the other scenarios, it is reasonable to state that these low results are highly influenced by the limited quality and quantity of data each client possesses, one row, this is backed by the findings discussed in section 3.2.2., where the mentioned experiment of Fed-TGAN showed that local models without enough quality and quantity of data may have a negative impact on the performance of the final model.

Regarding CatgeoricalCAP metric, models trained in scenarios 2 and 3 are the safest against these types of attacks, attaining a score of 1. Nonetheless, the model trained in scenario 0, i.e., in a centralized manner, also attains a very high score of 0,9986, meaning that the FL models did not stand out as safer. Nonetheless, this metric should not be taken as a guarantee, firstly

because it may not be adequate to the case under analysis, and secondly because a single metric is not enough to demonstrate the privacy guarantees of a model.

The time necessary to train the model was also taken into consideration. Federated training requires significantly more time compared to centralized training, specifically an increase of 175% and 161% for scenarios 2 and 3, respectively. This emphasizes challenges for FL training, as mentioned in section 3.2.3.

In summary, the results indicate that the performance of FL models is directly related to the amount of data available on each client device. Under favorable conditions, FL models can achieve comparable results to those obtained through centralized training approaches. Nevertheless, concerning training time, models in FL scheme take significantly more time. The presented findings lead to conclude that for Modatta to effectively implement this framework for the generation of synthetic data of users, it is necessary to either restructure the data available on each user device or explore alternative FLS and implementation frameworks more capable of handling such small data quantities.

## 6. Conclusion and Future Works

In conclusion, this paper presents a case study on a privacy-preserving framework suggested for Modatta. This framework proposed to generate realistic, non-sensitive data using a tabular GAN trained with FL, ensuring that Modatta never has direct access to users' personal data for training a RS model, thereby preserving their privacy while still providing personalized offers in the app.

The results of the case study firstly show that regarding GM, CTGAN outperformed the other studied models, establishing itself as the best possible choice for the synthetization of tabular data. Secondly, training this model with FL proved to be a feasible option for the preservation

of the privacy of user's data when training. Nonetheless, the results also showed that this is only true when each participating client in the training has sufficient and good-quality data.

It is to mention that the explored topics are emerging areas of research, especially the interconnection of tabular GANs and FL, meaning that such areas are subjected to future improvements, and it is important to follow them when considering a real-life implementation.

Additionally, it is important to acknowledge that the data used in the study was not exhaustive in capturing the complete range of information that users can potentially provide in the app, which may interfere with a comprehensive assessment of the GMs. The experiments made, highlighted another possible concern regarding how Modatta users' data is structured, and how it can influence the training and performance of decentralized GMs. Therefore, when implementing this framework, it is key to research mechanisms able to mitigate such problems.

Moreover, the Flower framework used to implement the FL strategy is still under development, which presents several limitations, such as regarding privacy guarantees, as it does not provide any additional mechanism to further ensure clients' data security which is crucial in Modatta's case. Hence, other FL implementation systems should be considered for evaluation.

To summarize, this project has demonstrated the potential applicability of federated GM in the construction of a privacy-preserving RS solution. Nevertheless, further investigations are needed to tackle the identified challenges.

## 7. A Case Study for State-of-the-Art Recommendation Systems

### 7.1. Motivation

Since its appearance alongside the Internet, Recommender Systems (RSs) have played a crucial role in assisting users to find relevant information. With the constant increase of data availability, they become more essential every day. In 2017, the amount of data generated globally was estimated to be 26 zettabytes and is expected to reach 181 zettabytes by 2025 (Taylor 2022). RSs are key to tackle this data overload problematic.

RSs have been widely adopted by worldwide companies and are mainly used in e-commerce and online advertisement to help increase revenues, improve user experience, drive web traffic, and decrease workload (Section 2021).

Following the goal of this project, this section will study traditional RS approaches to complement the framework explained in the previous sections to build an efficient privacy-preserving RS that surpasses the privacy challenges associated with RSs and allows Modatta to have a more efficient marketplace for consented personal data empowered solutions (Modatta 2023).

Hereupon, this segment will firstly address a trivial (7.2) Introduction to Recommender Systems, followed by (7.3) Data Treatment and (7.4) Models used for the mentioned purpose, subsequently the (7.5) Models Evaluation is discussed, and the section ends with the (7.6) Conclusion.

### 7.2. Introduction to Recommender Systems

Recommendation systems can be defined as a subclass of machine learning models that aim to give personalized recommendations of products and services that are relevant to users through information collection. Figure 1 illustrates the taxonomy of RSs techniques. The

recommendation process has two main phases, the information collection, and the learning phases.

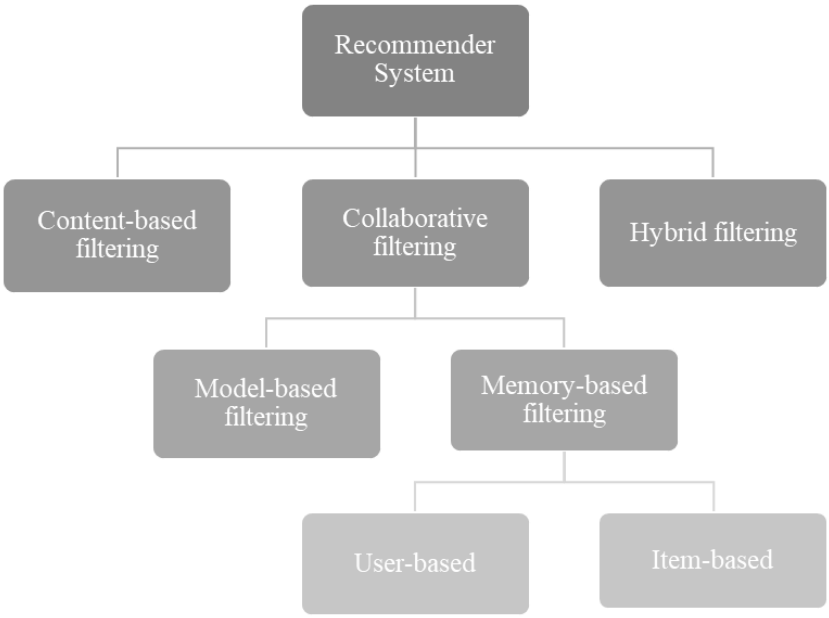



Figure 1 – Recommender Systems Techniques. Adapted from (Isinkaye et al. 2015).

The quality of the RS largely relies on its ability to represent the user's current interests (Isinkaye et al. 2015). In the first phase, the system must gather as much relevant information as possible about both users and items, to aid the construction of an accurate model. Items data refers to its attributes, features, and other characteristics, and can be collected from external sources such as reviews, descriptions, and tags, while users' information includes several attributes such as demographics, preferences, and past behavior, and can be provided in three ways, through implicit, explicit and hybrid feedback.

In implicit feedback, the system automatically infers users' preferences by observing their behavior through actions such as views, purchases, and clicks. On the other hand, in explicit feedback, information is provided directly by the user who is prompted by the system to express their opinion over ratings, reviews, or likes/dislikes. Lastly, hybrid feedback is a combination of both types of mentioned feedback. Although implicit feedback is less detailed, it is easier to

collect compared to explicit feedback, as it does not require any effort from the user, therefore being more abundant and more commonly used in RSs. Alternatively, hybrid feedback can be used to minimize their weaknesses since it incorporates elements from each.

Subsequently, in the learning phase, a learning algorithm is applied to the collected information to exploit users' relevant features for recommendation (Jariha 2018). Depending on the information available, the recommender system can be classified mainly in three ways, content-based, collaborative filtering, or hybrid.

Content-based (CB) systems utilize additional information about items to generate personalized recommendations based on the user's previously liked items (Shetty 2023). By analyzing the attributes or characteristics of items, CB systems identify similar items that align with the user's preferences, resulting in meaningful and relevant recommendations. These systems typically use models that measure the similarity between items, based on their attributes or features. Some commonly used models are Term Frequency-Inverse Document Frequency (TF-IDF), Naïve Bayes Classifier, and Neural Networks (Isinkaye et al. 2015).

Contrarily, in collaborative filtering (CF), recommendations are generated based on the underlying pattern resulting from collaborating past interactions of multiple users with different items, instead of considering item characteristics. CF has two types of approaches, item-item, and user-user. In the first one, recommendations are made by identifying items like the ones previously liked by the user. The system measures the similarity between items based on their attributes or features and suggests items that align with the user's preferences. In the second one, recommendations to users are generated using the distance between users based on their information. Furthermore, CF techniques can be divided into memory-based and model-based. In the first one, no latent model is assumed, therefore recommendations are achieved by using one of the two previously mentioned approaches to build a similarity matrix between users and

items and suggest the most similar. Similarity is calculated using distance metrics such as Euclidean distance, Manhattan distance and cosine similarity (Shetty 2023). In the second one, using machine learning or data mining techniques, predictive models are used to generate recommendations. The models take as input parameterized information features to solve an optimization-related problem. Common model-based CF are Singular Value Decomposition (SDV), Matrix Completion Technique, and Clustering (Isinkaye et al. 2015).

Finally, hybrid systems are optimized by combining both CB and CF. Combining algorithms can be done separately with joint results/recommendations, or directly constructing a unique model that consolidates both techniques. A commonly known used hybrid system is LightFM (Casalegno 2022).

CF is very practical as it does not require explicit information regarding users or items, however, since it relies on past feedback it is susceptible to cold start problems, i.e., if a new user/item is introduced the model cannot do accurate recommendation because it does not have any knowledge on them. CB systems tackle this challenge by using user/item characteristics to generate recommendations, though this comes with a caveat of high dependency on descriptive data. Individually, CB systems and CF present numerous challenges like the ones described. Hybrid systems surge to overcome these problems by combining the strengths of each technique, nonetheless, they generally require high computational complexity which might present a challenge in real-life applications.

### 7.3. Data Treatment

As mentioned, recommender systems quality is highly dependent on the quantity and quality of information, hence, in this section, two data frames will be taken into consideration for model evaluation. These are comprised of the cleaned_df described in section 4.2 (Appendix 14), and the augmented_df, used as input to the federated CTGAN mentioned in section 5.2.2.

One of the objectives of this section is to evaluate the influence of the number of categories on the same sample of users. To achieve this, the goal was to augment the number of rows in the dataset without reducing the number of columns. Due to the constraints imposed by the size and structure of the cleaned_df dataset, more complex models were not feasible to train. As a result, two simpler augmentation techniques were chosen to effectively work within these limitations and extract valuable insights from the data.

The first technique, random binary addiction, generated random rows based on a Bernoulli distribution with probability 0.2666, corresponding to the percentage of ones in the original data. The second technique, random variation augmentation, generated new rows based on segments of randomly selected original rows.

To evaluate the performance of each technique, three aspects were considered, the similarity between users using JSC (Appendix 10), the distribution of user and category count per category and user, respectively, and overall resemblance to the original data using standard statistical metrics and Jensen-Shannon divergence (JSD) (Appendix 18) for measuring the dissimilarity between distributions.

Analyzing the obtained results in Appendix 19, it is possible to see both approaches have very satisfactory results metrics-wise. Both showed good results in generating non-overlapped novel data (low JSC) (Appendix 19, Graphic 4). Respecting statistical representation, binary addiction slightly outperformed variation augmentation, showing higher proximity to the original data in terms of percentage of class 1, mean, standard deviation, and skewness. Nevertheless, the second method manages to somewhat capture the overall distribution pattern more closely (lower JSD) (Appendix 19, Table 7). Regarding the distribution of category count per user, the center in the binary approach is closer to the original average, however, the distribution pattern presents high central density which does not occur in the original data. For variation

augmentation, the distribution pattern illustrates several spread-dense points, better resembling the original distribution. Concerning the distribution of user count per category, both methods perform well (Appendix, Graph 5).

Hereupon, while random binary addiction better preserves the statistical properties of the original dataset, random variation augmentation captures the underlying patterns of user-item interactions more accurately. As the goal is to generate more truthful data, the second technique proves to be more effective, thus the results obtained from it will be used for model assessment in the following sections.

Summarizing, the datasets that are going to be used hereafter have 3000 unique users with a unique set of preferences for each user, and 34 and 558 unique categories for augmented_df and the augmented cleaned_df, respectively.

It is to mention that the data in use only comprehends users known preferences, which were collected in an undisclosed manner making it impossible to know if they were explicitly given by the user or inferred by its behavior, hence falling within the scope of implicit data. It is worth mentioning that Modatta's platform allows users to provide additional information, such as profile attributes (explicit feedback). However, it is important to note that the current data being used does not include this specific type of information.

### 7.4. Models

For this section, three traditional recommendation system models will be studied for the purpose of generating accurate recommendations for users. Taking into consideration the type of data provided, the models were chosen based on their ability to handle implicit feedback.

#### 7.4.1 Item-Item Based

An Item-Item model is a memory-based CF type of recommender system. This method finds the closest items by calculating the weighted average of the classification for each item, where

the weights are determined by an item-item similarity matrix calculated using the pairwise distance function for the distance metric of choosing. This metric supports three distance metrics, Euclidean, Manhattan and cosine similarity.

### 7.4.2. Alternating Least Squares (ALS)

Alternating Least Squares is a model-based CF type of recommender system. For this use case, the model was trained using the implicit library that follows the model implementation in (Hu, Koren and Volinsky 2008). In the mentioned paper, ALS considers the varying confidence levels of positive and negative preferences in the data and uses matrix factorization to decompose the user-item interaction matrix into two lower-rank matrices, one representing user embeddings and the other representing item embeddings.

The reconstructed interaction matrix is obtained by multiplying the user and item embeddings matrices together. The reconstruction error measures how well the model can reconstruct the observed interactions using the learned embeddings.

The ALS algorithm alternates between fixing one set of embeddings and optimizing the other, iteratively updating them to solve a least-squares problem, i.e., to minimize the reconstruction error. After training, the predictions are calculated by taking the dot product between the user embedding and the item embeddings. More detailed information regarding the structure and computations used can be found in (Hu, Koren and Volinsky 2008).

### 7.4.3. LightFM

LightFM is a hybrid type of recommender system that incorporates both item and user metadata into the traditional matrix factorization algorithms, meaning it is adapted to implicit and explicit feedback, tackling the cold start problem. For this use case, the model was trained using the LightFM library that follows the model implementation in (Kula 2015). The model learns to optimize a hybrid training objective that combines both collaborative filtering and content-

based objectives. It aims to predict user-item interactions while also considering item features and their relevance to users.

The model uses asynchronous stochastic gradient descent for optimization to update the model parameters iteratively. The training process involves maximizing the likelihood of the data conditional on the parameters, which is a combination of collaborative and content-based objectives.

Predictions are obtained by calculating a predicted preference score based on a combination of collaborative filtering and content-based signals. The model can generate personalized recommendations by ranking items based on their predicted preference scores and recommends the top-N items with the highest scores to the user.

The model also supports different loss functions, such as logistic loss, Bayesian personalized ranking (BPR), or WARP (Weighted Approximate-Rank Pairwise).

### 7.5. Models Evaluation

To assess model quality each of the mentioned methods was trained and evaluated using a train and test set, respectively, of the augmented cleaned_df and augmented_df mentioned in part 8.3. Additionally, each of the models was fine-tuned for each train set separately.

Modatta's goal is to recommend marketing campaigns to users with higher likelihood of response, thereat the best model was chosen based on the precision score that, by allowing to minimize the number of users targeted as likely responsive that are in fact not likely to respond (false positives), helps to optimize the resource allocation and increase profits. Moreover, other metrics were also calculated to help evaluate each model (Appendix 20). The results obtained from the best models for each dataset can be found in tables 5 and 6.

| | ROC AUC | Accuracy | Precision | Recall | F1 |
|---|---|---|---|---|---|
| Item-Item | 0,522 | 0,735 | 0,581 | 0,061 | 0,110 |
| ALS | **0,803** | **0,836** | **0,709** | **0,727** | **0,718** |
| LightFM | 0,511 | 0,661 | 0,500 | 0,043 | 0,080 |

Table 5 – Evaluation metrics results of Item-Item based, ALS and LightFM models for augmented_df.

| | ROC AUC | Accuracy | Precision | Recall | F1 |
|---|---|---|---|---|---|
| Item-Item | **0,536** | 0,810 | 0,585 | **0,087** | **0,151** |
| ALS | 0,523 | **0,811** | **0,598** | 0,056 | 0,102 |
| LightFM | 0,502 | 0,737 | 0,529 | 0,005 | 0,011 |

Table 6 – Evaluation metrics results of Item-Item based, ALS and LightFM models for augmented cleaned_df.

For augmented_df, ALS is the model with best performance regarding every score where it presents good results (all above 0.7). For the augmented cleaned_df, ALS approach still performs best regarding accuracy and precision but performs badly in the rest.

Comparing the overall results obtained in each dataset, first, it is to note that, Item-Item approach and LightFM got better results with the augmented cleaned_df. However, ALS performance highly deprecated concerning ROC AUC, Recall and F1, indicating a high prediction of false negatives, i.e., prediction of likely responsive users as non-likely responsive. Moreover, the ROC AUC scores close to 0.5 indicate that the model has low capacity distinguishing positive and negative instances.

Item-Item and LightFM models were able to leverage the increase of information to produce better recommendations (higher accuracy and precision), still, the lower recall and F1 scores obtained in both datasets might be due to the sparsity and higher complexity of relations between categories that the models are unable to capture. The dimensionality increase

probability accentuated this characteristic of the data causing ALS to also underperform in that task.

Summarizing, results suggest that ALS might be the most suitable model for both scenarios, as it achieves better overall performance across various evaluation metrics and can provide more trustworthy recommendations.

### 7.6. Conclusion

In this section, three RSs models were evaluated on a low and high dimensionality scenario, Item-Item based, ALS and LightFM models. All were able to generate accurate and precise recommendations in both setups but in most cases failed to capture the complex relations between items in the high dimensionality dataset. Globally, ALS showed better performance in both datasets, however, not only is this model suited for implicit data exclusively, but also underperforms in higher dimensional implicit data.

Typically, Modatta users locally store implicit and/or explicit data, despite the fact ALS outperforms the others with the provided data, for real-life applications it might not be the case and it needs to be reassessed. Furthermore, the sparsity of the data also needs to be addressed to improve the quality of recommendations as, within our goal, having a low recall score has adverse implications.

**References List**


Aguiar, Emilia Cristina Ribeiro. 2021. “HyperML and Deep Interest Network to Build a Recommender System for Modatta: Data Privacy in Federated Learning.” Master’s thesis, School of Business and Economics of UNL.http://hdl.handle.net/10362/140157.

Alibaba. 2023. "Curvature-Learning-Framework" GitHub. https://github.com/alibaba/Curvature-Learning-Framework

Analystprep. 2019. “Financial Correlation Modeling – Bottom-Up Approaches.” Analystprep. April 23, 2019. https://analystprep.com/study-notes/frm/part-2/market-risk-measurement-and-management/financial-correlation-modeling-bottom-up-approaches/.

Augenstein, Sean, H. Brendan McMahan, Daniel Ramage, Swaroop Ramaswamy, Peter Kairouz, Mingqing Chen, Rajiv Mathews, and Blaise Aguera y Arcas. 2020. “Generative Models for Effective ML on Private, Decentralized Datasets.” In *International Conference on Learning Representations (ICLR)*. https://iclr.cc/virtual_2020/poster_SJgaRA4FPH.html.

Banerjee, Sayar. 2021.“Fast Feature Engineering in Python: Tabular Data” Medium. April 11, 2021. https://towardsdatascience.com/fast-feature-engineering-in-python-tabular-data-d050b68bb178

Beutel, Daniel J., Taner Topal, Akhil Mathur, Xinchi Qiu, Javier Fernandez-Marques, Yan Gao, Lorenzo Sani, et al. 2022. “Flower: A Friendly Federated Learning Research Framework.” Computing Research Repository (CoRR). https://doi.org/https://doi.org/10.48550/arXiv.2007.14390.

Bobadilla, Jesús, Abraham Gutiérrez, Raciel Yera, and Luis Martínez. 2023. “Creating Synthetic Datasets for Collaborative Filtering Recommender Systems Using Generative

Adversarial Networks." *Computing Research Repository (CoRR)*, March. https://doi.org/https://doi.org/10.48550/arXiv.2303.01297.

Casalegno, Francesco. 2022. "Recommender Systems – A Complete Guide to Machine Learning Models" Towards Data Science. November 25, 2022. https://towardsdatascience.com/recommender-systems-a-complete-guide-to-machine-learning-models-96d3f94ea748

Choi, Edward, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F. Stewart, and Jimeng Sun. 2017. "Generating Multi-Label Discrete Patient Records Using Generative Adversarial Networks." In *Proceedings of the 2nd Machine Learning for Healthcare Conference:* 286–305. PMLR. http://proceedings.mlr.press/v68/choi17a

Data Science Digest. 2021. "Centralized Learning vs. Distributed Learning." Medium. May 15, 2021. https://digestize.medium.com/centralized-learning-vs-distributed-learning-c75ee9e94423.

Deloitte. 2021. "What Is Digital Economy?" Deloitte. 2021. https://www2.deloitte.com/mt/en/pages/technology/articles/mt-what-is-digital-economy.html.

Diao, Sandy. 2022. "Top Digital Advertising Platforms For Effective Campaigns." Descript. January 12, 2022. https://www.descript.com/blog/article/top-digital-advertising-platforms-for-effective-campaigns.

Digital Marketing Institute. 2016. "The Evolution of Digital Marketing: 30 Years in the Past & Future." Digital Marketing Institute. October 4, 2016. https://digitalmarketinginstitute.com/blog/the-evolution-of-digital-marketing-30-years-in-the-past-and-future.

DMA. 2022. “DMA Insight: Data and Training Are Key Challenges Facing the Industry.” DMA. December 13, 2022. https://dma.org.uk/research/dma-insight-data-and-training-are-key-challenges-facing-the-industry.

Duan, Shaoming, Chuanyi Liu, Peiyi Han, Xiaopeng Jin, Xinyi Zhang, Tianyu He, Hezhong Pan, and Xiayu Xiang. 2022. “HT-Fed-GAN: Federated Generative Model for Decentralized Tabular Data Synthesis.” *Entropy* 25 (1): 88. https://doi.org/10.3390/e25010088.

Fox, Sapphire. 2022. “Data Privacy Statistics, Facts & Trends of 2023: Your Data Is the New Oil.” Cloudwards. September 29, 2022. https://www.cloudwards.net/data-privacy-statistics/.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial networks." Communications of the ACM 63, no. 11 (2020): 139-144. https://doi.org/10.48550/arXiv.1406.2661

Hu, Yifan, Yehuda Koren, and Chris Volinsky. 2008. "Collaborative filtering for implicit feedback datasets." In 2008 Eighth IEEE international conference on data mining, pp. 263-272. Ieee. https://doi.org/10.1109/ICDM.2008.22

Isinkaye, Folasade Olubusola, Yetunde O. Folajimi, and Bolande Adefowoke Ojokoh. 2015. "Recommendation systems: Principles, methods and evaluation." Egyptian informatics journal 16, no. 3 (2015): 261-273. https://doi.org/10.1016/j.eij.2015.06.005

Jariha, Priyanka, and Sanjay Kumar Jain. 2018. "A state-of-the-art Recommender Systems: An overview on Concepts, Methodology and Challenges." In 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT): 1769-1774. https://doi.org/10.1109/ICICCT.2018.8473275

Kula, Maciej. 2015. "Metadata embeddings for user and item cold-start recommendations." arXiv preprint arXiv:1507.08439.

Langr, Jakub, and Vladimir Bok.2019. GANs in Action: Deep Learning With Generative Adversarial Networks. 1st ed. New York: Manning Publications Co.

Li, Qinbin, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. 2023. "A Survey on Federated Learning Systems: Vision, Hype and Reality for Data Privacy and Protection." *IEEE Transactions on Knowledge and Data Engineering* 35 (4): 3347–66. https://doi.org/10.1109/TKDE.2021.3124599.

Lim, Wei Yang Bryan, Jer Shyuan Ng, Zehui Xiong, Dusit Niyato, Chunyan Miao. (2022). Federated Learning Over Wireless Edge Networks. 1st ed.Switzerland: Springer.

Liu, Ji, Jizhou Huang, Yang Zhou, Xuhong Li, Shilei Ji, Haoyi Xiong, and Dejing Dou. 2022. "From Distributed Machine Learning to Federated Learning: A Survey." *Knowledge and Information Systems* 64 (4): 885–917. https://doi.org/10.1007/s10115-022-01664-x.

Lucas, Carolina Carvalho. 2021. "HyperML and Deep Interest Network to Build a Recommender System for Modatta: Measuring the Effectiveness of Targeting Users with New Offers." Master's thesis, School of Business and Economics of UNL. http://hdl.handle.net/10362/140156.

Lyu, Lingjuan, Han Yu, and Qiang Yang. 2020. "Threats to Federated Learning: A Survey." *Computing Research Repository (CoRR)*, March. https://doi.org/https://doi.org/10.48550/arXiv.2003.02133.

Macedo, Patrícia Alexandra Cravo. 2021. "HyperML and Deep Interest Network to Build a Recommender System for Modatta: Targeting Customers for Campaign Offers in a Two-

Sided Market." Master's thesis, School of Business and Economics of UNL. http://hdl.handle.net/10362/140149.

McMahan, Brendan, Eider Moore, Daniel Ramage, Daniel Hampson, and Blaise Aguera y Arcas. 2017. "Communication-Efficient Learning of Deep Networks from Decentralized Data." *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)* 54 (JMLR: W & CP): 1273–82. http://proceedings.mlr.press/v54/mcmahan17a?ref=https://githubhelp.com

McMahan, Brendan, and Daniel Ramage. 2017. "Federated Learning: Collaborative Machine Learning without Centralized Training Data." Google. April 6, 2017. https://ai.googleblog.com/2017/04/federated-learning-collaborative.html.

Modatta. 2023. "FAQ." Modatta. 2023. https://www.modatta.com/faq.

Mokhtarani, Shabnam. 2021. "Embeddings in Machine Learning: Everything You Need to know" Featureform. August 26, 2021. https://www.featureform.com/post/the-definitive-guide-to-embeddings

Nickel, Maximillian, and Douwe Kiela. 2017. "Poincaré embeddings for learning hierarchical representations." Advances in neural information processing systems 30. https://proceedings.neurips.cc/paper/2017/hash/59dfa2df42d9e3d41f5b02bfc32229dd-Abstract.html

Page, Carly. 2020. "Marriott Hit With £18.4 Million GDPR Fine Over Massive 2018 Data Breach." Forbes. October 30, 2020. https://www.forbes.com/sites/carlypage/2020/10/30/marriott-hit-with-184-million-gdpr-fine-over-massive-2018-data-breach/.

Park, Noseong, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. 2018. “Data Synthesis Based on Generative Adversarial Networks.” *Proceedings of the VLDB Endowment* 11 (10): 1071–83. https://doi.org/10.14778/3231751.3231757.

Rasouli, Mohammad, Tao Sun, and Ram Rajagopal. 2020. “FedGAN: Federated Generative Adversarial Networks for Distributed Data.” *Computing Research Repository (CoRR)*, June. https://doi.org/https://doi.org/10.48550/arXiv.2006.07228.

Rocca, Baptiste. 2019. "Introduction to recommender systems" Towards Data Science. June 3, 2019. https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada

Rocca, Joseph. 2019. “Understanding Variational Autoencoders (VAEs)”. Towards Data Science. September 24, 2019. https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

SDV-DEV. 2023. “CTGAN.” GitHub. 2023. https://github.com/sdv-dev/CTGAN.

Section. 2021. “Recommender System Explained” Section. May 3, 2021. https://www.section.io/engineering-education/recommender-system-explained/

Shetty, Badreesh. 2023. “An In-Depth Guide to How Recommender Systems Work” builtin. March 2, 2023. https://builtin.com/data-science/recommender-systems

Taylor, Ptroc. 2022. “Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2020, with forecasts from 2021 to 2025” Statista. September 8, 2022. https://www.statista.com/statistics/871513/worldwide-data-created/

Tibco. “What is Hierarchical Data?” Tibco. Accessed April 15, 2023. https://www.tibco.com/reference-center/what-is-hierarchical-

data#:~:text=Hierarchical%20data%20is%20a%20data,a%20hierarchy%20of%20connected%20data

Triastcyn, Aleksei, and Boi Faltings. 2020. “Federated Generative Privacy.” *IEEE Intelligent Systems* 35 (4): 50–57. https://doi.org/10.1109/MIS.2020.2993966.

Usercentrics. 2021. “Data Is the New Gold – How and Why It Is Collected and Sold.” Usercentrics. October 21, 2021. https://usercentrics.com/knowledge-hub/data-is-the-new-gold-how-and-why-it-is-collected-and-sold/.

Vatsal. 2021. "Recommendation Systems Explained" Towards Data Science. July 12, 2021. https://towardsdatascience.com/recommendation-systems-explained-a42fc60591ed

Vinh Tran, Lucas, Yi Tay, Shuai Zhang, Gao Cong, and Xiaoli Li. 2020. “HyperML: A Boosting Metric Learning Approach in Hyperbolic Space for Recommendation Systems.” The Thirteenth ACM International Conference on Web Search and Data Mining (WSDM’20) 9. https://arxiv.org/pdf/1809.01703.pdf.

Xu, Lei. 2020. “Synthesizing Tabular Data using Conditional GAN” Master’s thesis, Massachusetts Institute of Technology. https://dai.lids.mit.edu/wp-content/uploads/2020/02/Lei_SMThesis_neo.pdf

Xu, Lei, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. 2019. “Modeling Tabular Data Using Conditional GAN.” In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)* http://arxiv.org/abs/1907.00503.

Yang, Timothy, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kongs, Daniel Ramage, and Françoise Beaufays. 2018. “Applied federated learning: Improving google keyboard query suggestions.” https://doi.org/10.48550/arXiv.1812.02903

Ze, Wu Zhen. 2021. “HyperML and Deep Interest Network to Build a Recommender System for Modatta: Data Privacy with GAN.” Master’s thesis, School of Business and Economics of UNL. http://hdl.handle.net/10362/140159.

Zhang, Yikai, Hui Qu, Qi Chang, Huidong Liu, Dimitris Metaxas, and Chao Chen. 2021. “Training Federated GANs with Theoretical Guarantees: A Universal Aggregation Approach.” In *International Conference on Learning Representations (ICLR 2021)*. https://doi.org/https://doi.org/10.48550/arXiv.2102.04655.

Zhao, Zilong, Robert Birke, Aditya Kunar, and Lydia Y Chen. 2021. “Fed-TGAN: Federated Learning Framework for Synthesizing Tabular Data.” *Computing Research Repository*. https://doi.org/https://doi.org/10.48550/arXiv.2108.07927.

Zhou, Guorui, and, Xiaoqiang Zhu, Chengru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, Kun Gai. 2018. “Deep interest network for click-through rate prediction”. *In Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining: 1059-1068.* https://doi.org/10.1145/3219819.3219823

# Appendices

## Appendix 0

## Table of Contents

## Appendix 1

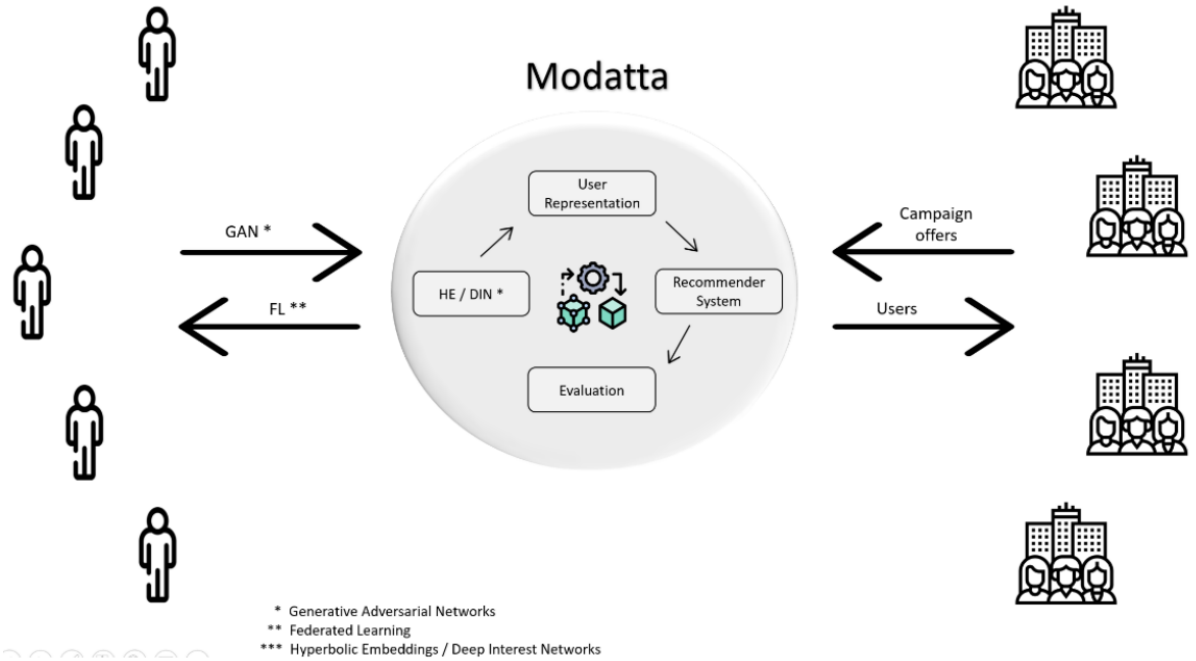



Figure 1 – Proposed Privacy-preserving framework for Modatta. Retrieved from (Lucas, 2021).

## Appendix 2


Figure 2 – Progress in Human face generation by ML models. (Source: "The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation," by Miles Brundage et al., 2018, https://arxiv.org/abs/1802.07228)

Figure 3 – Generated Images by GANs. (Source: “Progressive Growing of GANs for Improved Quality, Stability, and Variation,” by Tero Karras et al., 2017, https://arxiv.org/abs/1710.10196.)

**Appendix 3**

$$\min_G \max_D \quad V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D\,(x)] + \mathbb{E}_{z\sim p_z(z)}\left[\log\left(1 - D\big(G(z)\big)\right)\right]$$

Equation 1 – Objective function used in Generative Adversarial Networks (GANs). Retrieved from (Goodfellow et al., 2014).

Breaking down the equation we have:

- "E_x[log(D(x))]" calculates the average of the logarithm of the discriminator's output (D(x)) when presented with real data samples x, i.e., measures how well D can classify real data.
- "E_z[log(1 - D(G(z)))]" calculates the average of the logarithm of 1 minus the discriminator's output (D(G(z))) when presented with generated data samples G(z), i.e., measures how well D can distinguish between generated data and real data.

By optimizing this objective function, G and D engage in a competitive process, with G improving its ability to generate realistic data and D improving its ability to distinguish between

real and generated data. The overall goal is to reach an equilibrium where G generates data that is indistinguishable from real data, and D cannot reliably differentiate between the two.

**Appendix 4**

$$\begin{cases} h_0 = z \oplus cond \\ h_1 = h_0 \oplus \text{ReLU}\left(\text{BN}\left(\text{FC}_{|cond|+|z|\to 256}(h_0)\right)\right) \\ h_2 = h_1 \oplus \text{ReLU}\left(\text{BN}\left(\text{FC}_{|cond|+|z|+256\to 256}(h_1)\right)\right) \\ \hat{\alpha}_i = \tanh\left(\text{FC}_{|cond|+|z|+512\to 1}(h_2)\right) & ,1 \le i \le N_c \\ \hat{\beta}_i = \text{gumbel}_{0,2}\left(\text{FC}_{|cond|+|z|+512\to m_i}(h_2)\right) & ,1 \le i \le N_c \\ \hat{\mathbf{d}}_i = \text{gumbel}_{0,2}\left(\text{FC}_{|cond|+|z|+512\to |D_i|}(h_2)\right) & ,1 \le i \le N_d \end{cases}$$

Equation 2 – Description of Conditional Generator $\mathcal{G}(z, cond)$- Adapted from (Xu et al. 2019).

$$\begin{cases} h_0 = \mathbf{r}_1 \oplus \ldots \oplus \mathbf{r}_{10} cond_1 \oplus \ldots \oplus cond_{10} \\ h_1 = \text{drop}(\text{leaky}_{0,2}(\text{FC}_{10|\mathbf{r}|+10|cond|\to 256}(h_0))) \\ h_2 = \text{drop}\left(\text{leaky}_{0,2}\left(\text{FC}_{256\to 256}(h_1)\right)\right) \\ \mathcal{C}(\cdot) = \text{FC}_{256\to 1}(h_2) \end{cases}$$

Equation 3 – Description of Critic architecture with pac size 10 $\mathcal{C}(r_1, \ldots, r_{10}, cond_1, \ldots, cond_{10})$. Adapted from (Xu et al. 2019).

These two equations describe the network structure of CTGAN. To capture correlations between columns in a row, fully connected networks are employed in both the generator and critic. These networks consist of two hidden layers each. In the generator, batch-normalization and ReLU activation functions are used. The row representation is generated by combining multiple activation functions. Specifically, scalar values ($\alpha_i$) are generated using the tanh activation function, while the mode indicator ($\beta_i$) and discrete values ($d_i$) are generated using the Gumbel softmax activation function. In the critic, the leaky ReLU function and dropout are utilized on each hidden layer. This architecture and selection of activation functions allow for

the capture of all possible correlations between columns, considering the absence of local structure within rows.

## Appendix 5

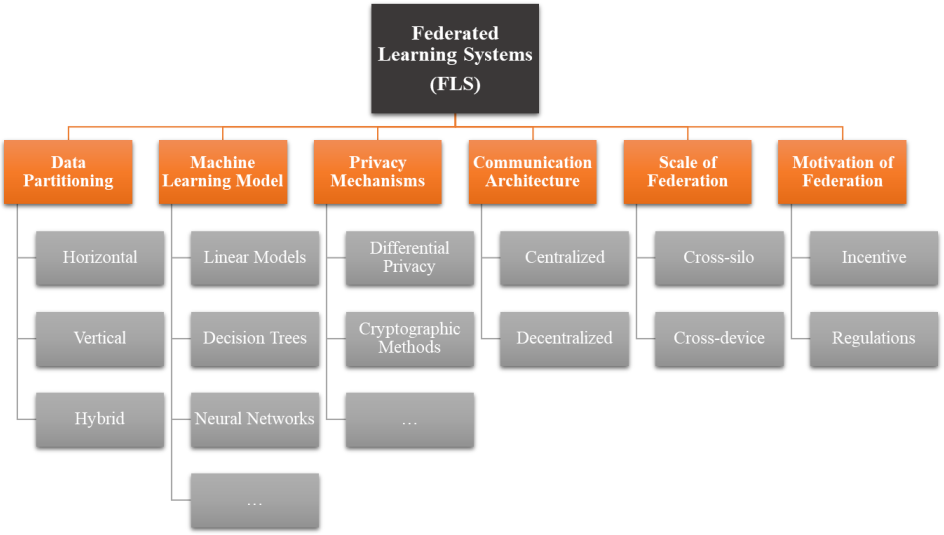



Figure 4 – Picture demonstrating a Federated Learning Systems Taxonomy.

## Appendix 6

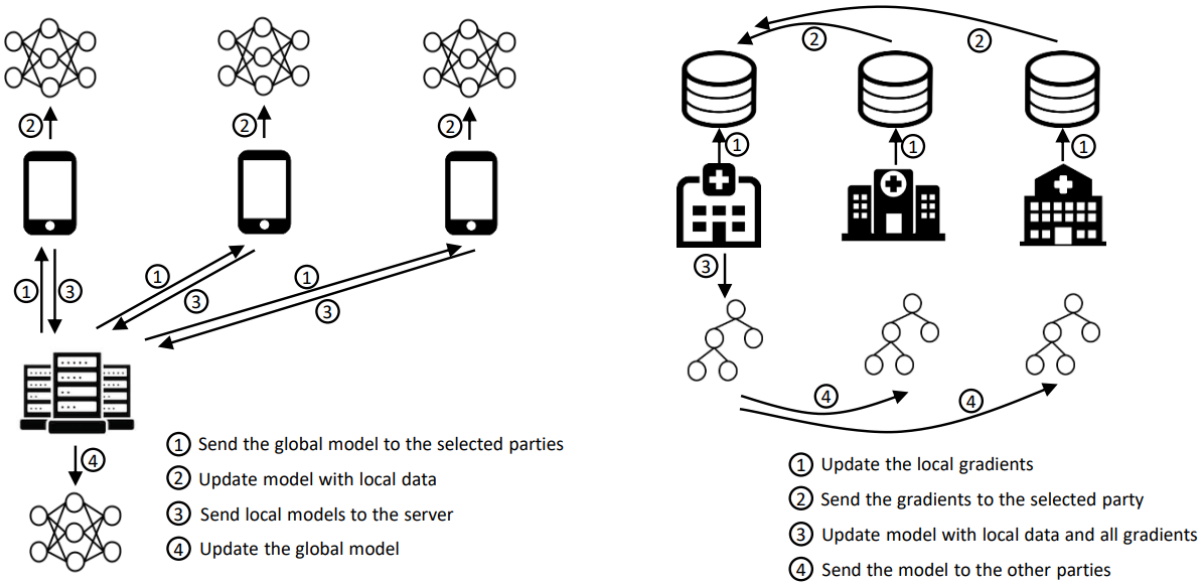



Figure 5 – Visual representation of the centralized (left) and decentralized FL architecture (right). Retrieved from (Li et al. 2023)

## Appendix 7

Figure 6 – Generated images by the UA-GAN. Retrieved from (Zhang et al. 2021).

## Appendix 8

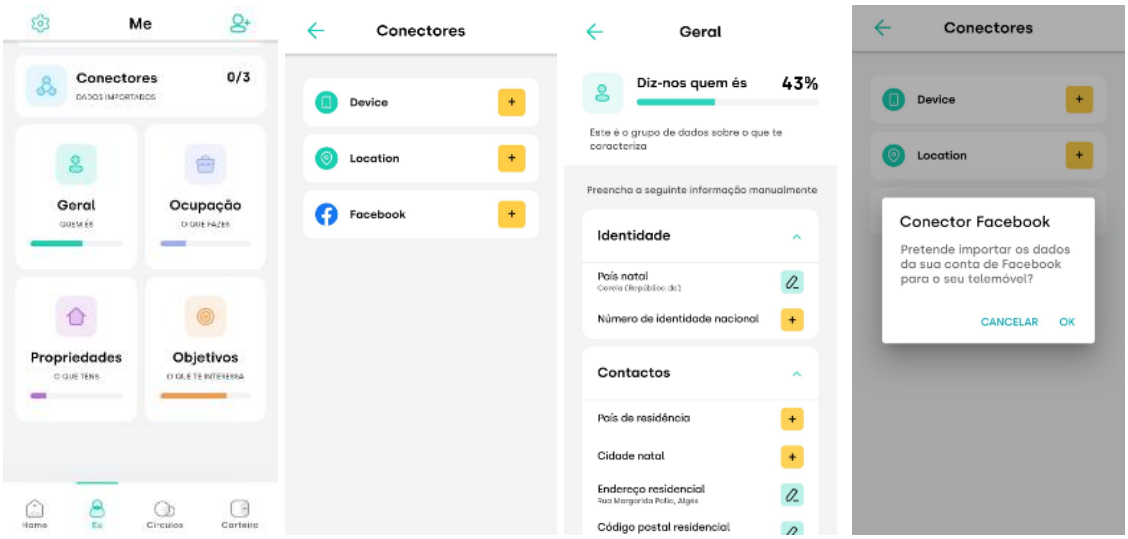



Figure 7 – Modatta's app user interface examples, from left to right, profile information, connectors, specified information on the section “general” of profile information, and pop-up warning asking to connect app to Facebook.

## Appendix 9

| File Name | Size | Description | # Columns | # Rows |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| dump_fb_translations .csv | 56,8 MB | Anonymous information of requests made by users of Modatta app | 5 | 580 000 |
| facebook_categories. xlsx | 149 KB | Codes and description of Facebook Categories | 9 | 1655 |

Table 1 – Summary of Documents Received

| Column Name | Type | Description |
|---|---|---|
| *id* | int64 | Line counter of entries |
| *category* | int64 | Code from Facebook that identifies a category user has shown interest on Facebook |
| *invocationId* | object | Request identifier |
| *created_at* | object | Timestamp of the request creation |
| *updated_at* | object | Timestamp of the request update |

Table 2 – dump_fb_translations.csv

| Column Name | Type | Description |
|---|---|---|
| *MASTER_CATEGORY_PROFILE_ID* | object | Master category code |
| *MASTER CATEGORY* | object | Master category description |
| *Unnamed: 2* | None | None |
| *Unnamed:3* | None | None |
| *PROFILE_ID* | object | Profile code of Modatta app for each topic of Facebook |
| Facebook ID | object | Code from Facebook that identifies a certain topic user can show interest on Facebook |
| Tag EN | object | Corresponding topic name in English |
| Tag PT | object | Corresponding topic name in Portuguese |
| Tag ES | object | Corresponding topic name in Spanish |

Table 3 – facebook_categories.xlsx

## Appendix 10

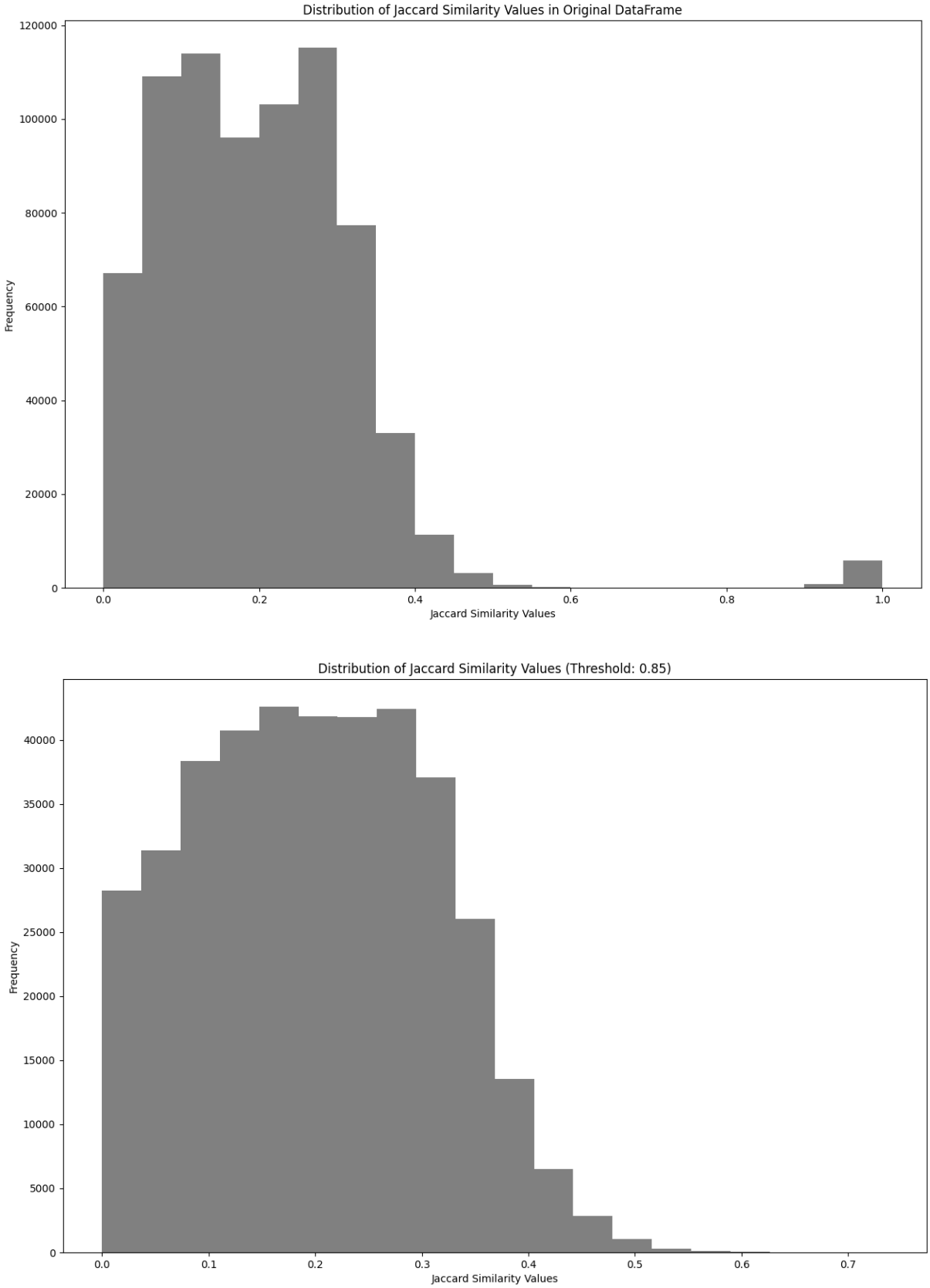
Jaccard similarity is a measure of similarity between two sets, A and B. It is defined as the size of the intersection of the sets divided by the size of the union of the sets, as follows:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Equation 4 – Jaccard Similarity Coefficient Formula.

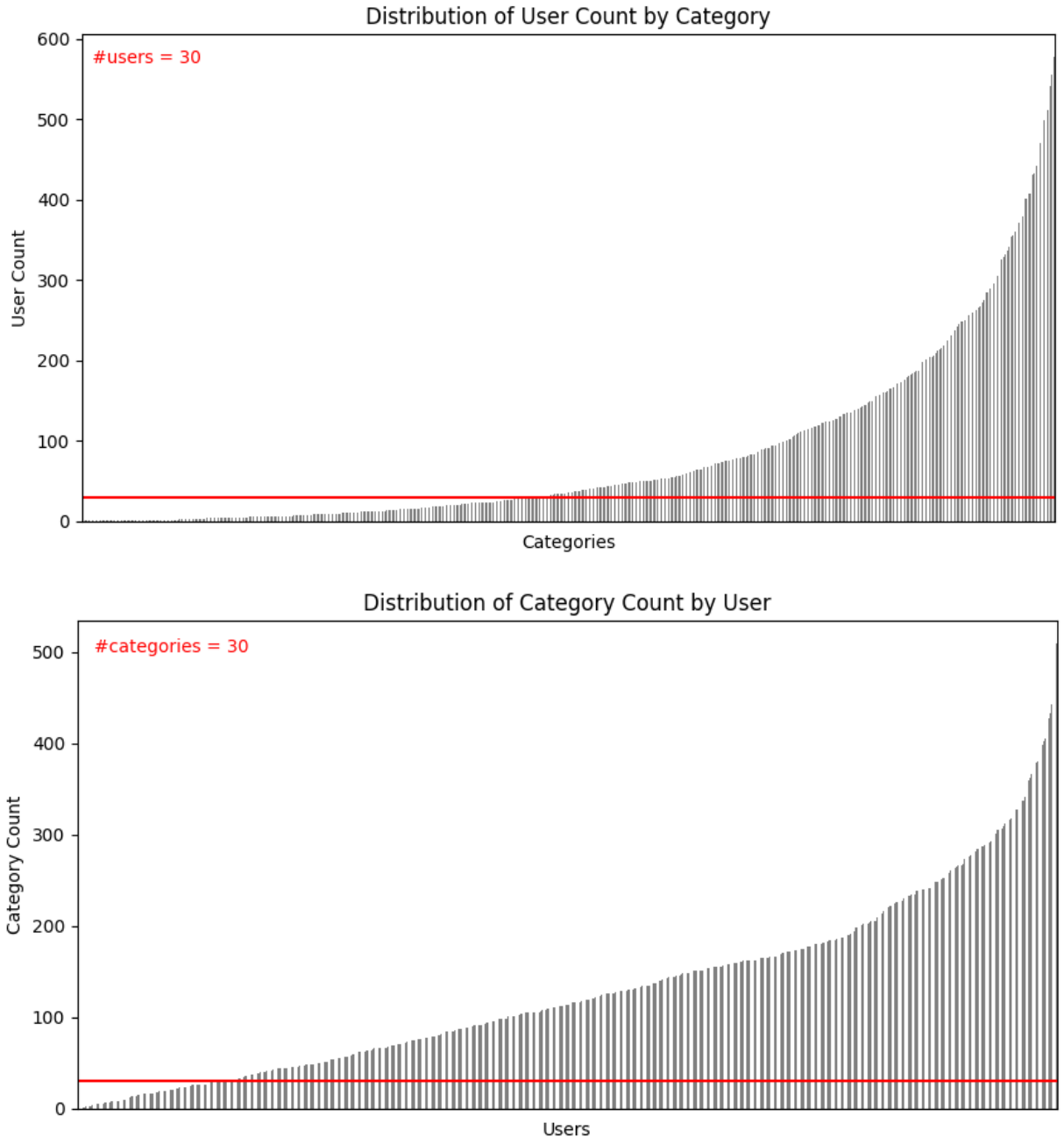
The Jaccard similarity can range between 0 and 1, where 1 indicates the sets are identical and 0 indicates no similarity between the sets.

## Appendix 11



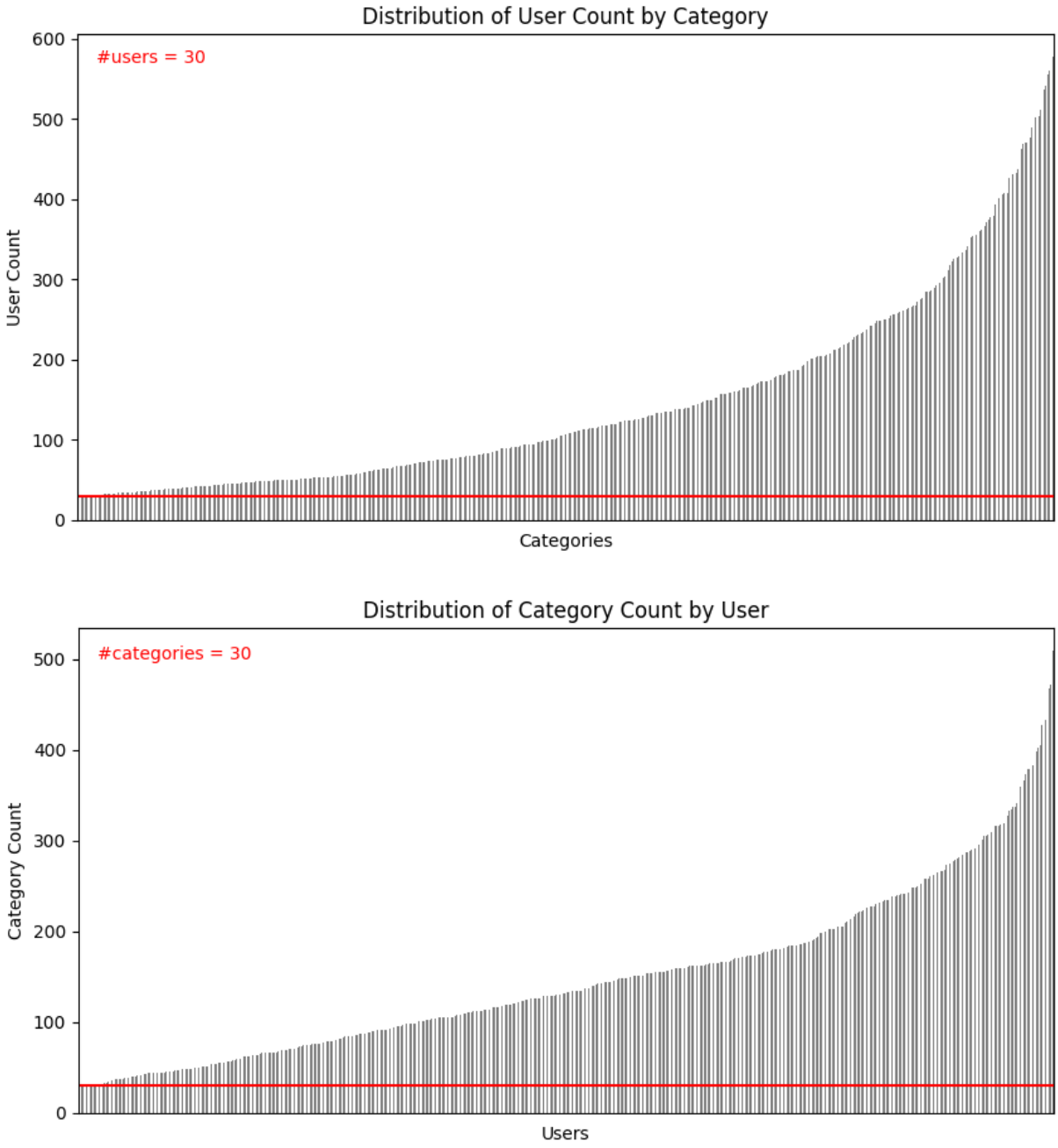

Graphic 1 – Jaccard Similarity Scores values distribution of the pairs on the original data (top) and of the pairs after removing one element of them with a score above 0,85 (bottom).

## Appendix 12




Graphic 2 – Distribution of user count by category (top) and distribution of category count by user (bottom) originally.

## Appendix 13




Graphic 3 – Distribution of user count by category (top) and distribution of category count by user (bottom) after removing categories and users with a frequency inferior to 30.

## Appendix 14

| cleaned_df | |
|---|---|
| # Rows | 540 |
| # Columns | 558 |

| | |
|---|---|
| # Null Values | 0 |
| # Duplicates | 0 |
| Data types | object |
| Data Values | {0,1} |

Table 4 – Description of cleaned_df

| | Businesses \| Shopping & retail \| Bridal shop | Businesses \| Vehicle, aircraft and boat \| Motorcycle repair centre | Non-business places \| Religious place of worship \| Evangelical Church | Businesses \| Local service \| Computer repair service | Businesses \| Shopping & retail \| Shoe shop | ... | Businesses \| Advertising/marketing \| Media agency | Businesses \| Property \| Property developer | Businesses \| Travel & transport \| Boat hire | Other \| Ticket sales | Businesses \| Food & drink \| Hot dog restaurant |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 00-00-00 | 1 | 1 | 1 | 1 | 1 | ... | 0 | 0 | 0 | 0 | 0 |
| 00159fc2-dac5-4a9e-af79-9acc395fda30 | 0 | 0 | 1 | 1 | 1 | ... | 0 | 0 | 0 | 0 | 0 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| ff48a58d-0eac-4582-b976-c8875dfd915d | 0 | 0 | 0 | 1 | 0 | ... | 0 | 0 | 0 | 0 | 0 |
| ffeba485-95a4-441d-b44e-dce13d361c41 | 0 | 0 | 0 | 0 | 1 | ... | 0 | 0 | 0 | 0 | 0 |

540 rows × 558 columns

Figure 8 – Visual representation of cleaned_df dataframe

## Appendix 15

The table displays the percentage of the value 1 and the categories count by main category in the original dataframe and those obtained using the following column reduction techniques: 1) random selection of columns, 2) election of the columns with the highest frequency of the value one, 3) categories hierarchy flattening and 4) columns selection based on proximity to average. There are 7 main categories in the dataframe, Businesses (B), Community organization (CO), Interest (I), Media (M), Non-business places (NBP), Other (O), Public Figure (PF)

| | | Original Data | 1) | 2) | 3) | 4) |
|---|---|---|---|---|---|---|
| | # Columns | 558 | 28 | 30 | 142 | 34 |
| | Overall % 1 | 27 | 47 | 85 | 57 | 26 |
| Distribution of categories across main categories | B | 400 | 4 | 8 | 87 | 22 |
| | CO | 14 | 4 | 0 | 1 | 2 |
| | I | 4 | 4 | 1 | 1 | 0 |
| | M | 25 | 4 | 3 | 11 | 3 |
| | NBP | 30 | 4 | 0 | 4 | 0 |
| | O | 59 | 4 | 12 | 30 | 5 |
| | PF | 26 | 4 | 6 | 8 | 2 |

Table 5 - Comparison of original dataframe with the obtained dataframes from the column reduction approaches. Include number of columns, overall percentages of 1, and distribution of categories by main categories.

| reduced_df | |
|---|---|
| # Rows | 522 |
| # Columns | 34 |
| # Null Values | 0 |
| # Duplicates | 22 |
| Data types | Boolean |
| Data Values | $\{True, False\}$ |

Table 6 - Description of reduced_df.

| | Businesses \| Shopping & retail \| Toy shop | Other \| TypeAhead \| Sport | Businesses \| Education \| Driving school | Businesses \| Finance \| Bank | Businesses \| Advertising/marketing \| Marketing agency | ... | Public figure \| Designer | Businesses \| Medical & health \| Doctor | Businesses \| Shopping & retail \| Outdoor & sporting goods company | Businesses \| Medical & health \| Nutritionist | Businesses \| Shopping & retail \| Discount shop |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | False | True | True | False | True | ... | True | True | True | True | False |
| **1** | False | True | False | True | True | ... | True | True | True | True | True |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **520** | False | True | False | False | False | ... | True | False | False | False | False |
| **521** | False | False | False | False | True | ... | False | False | False | False | False |

522 rows × 34 columns

Figure 9 – Visual representation of reduced_df dataframe.

## Appendix 16

### TVComplement

The calculation utilizes the Total Variation Distance (TVD), which measures the disparities in probabilities between the categories of the real (R) and synthetic (S) columns. The frequency of each category value is transformed into a probability before comparing the differences using the TVD formula:

$\delta(R,S) = \frac{1}{2}\sum_{\omega\in\Omega}|R_\omega - S_\omega|$, where ω describes all possible categories in a column, Ω

Equation 5 – Total Variation Distance (TVD) Calculation

score=1- $\delta(R,S)$

Equation 6 - TVComplement Score Calculation

**CategoryCoverage**

To calculate the score, the metric counts the number of distinct categories, c, present in the real column r. It then determines the number of those categories that appear in the synthetic column, s. Finally, it computes the proportion of real categories covered by the synthetic data described in equation 7.

$$score = \frac{c_s}{c_r}$$

Equation 7 – CategoryCoverage Score Calculation

**NewRowSynthesis**

This metric computes the proportion of rows in synthetic data that match with rows in real data.

$$score = 1 - \frac{matching\ synthetic\ rows}{total\ synthetic\ rows}$$

Equation 8– NewRowsSynthesis Score Calculation

**ContingencySimilarity**

For a pair of columns, A and B, the metric computes a normalized contingency table that illustrates the proportion of rows with each combination of categories in A and B. It then measures the difference between the contingency tables using the Total Variation Distance. Finally, the distance is subtracted from 1 to ensure that a higher score signifies greater similarity. R and S denote the real and synthetic frequencies of those categories.

$score = 1 - \frac{1}{2}\sum_{\alpha \in A}\sum_{\beta \in B}|S_{\alpha,\beta} - R_{\alpha,\beta}|$, where α and β, represent all columns possible categories in columns A and B, respectively

Equation 9 - ContingencySimilarity Score Calculation

**Appendix 17**

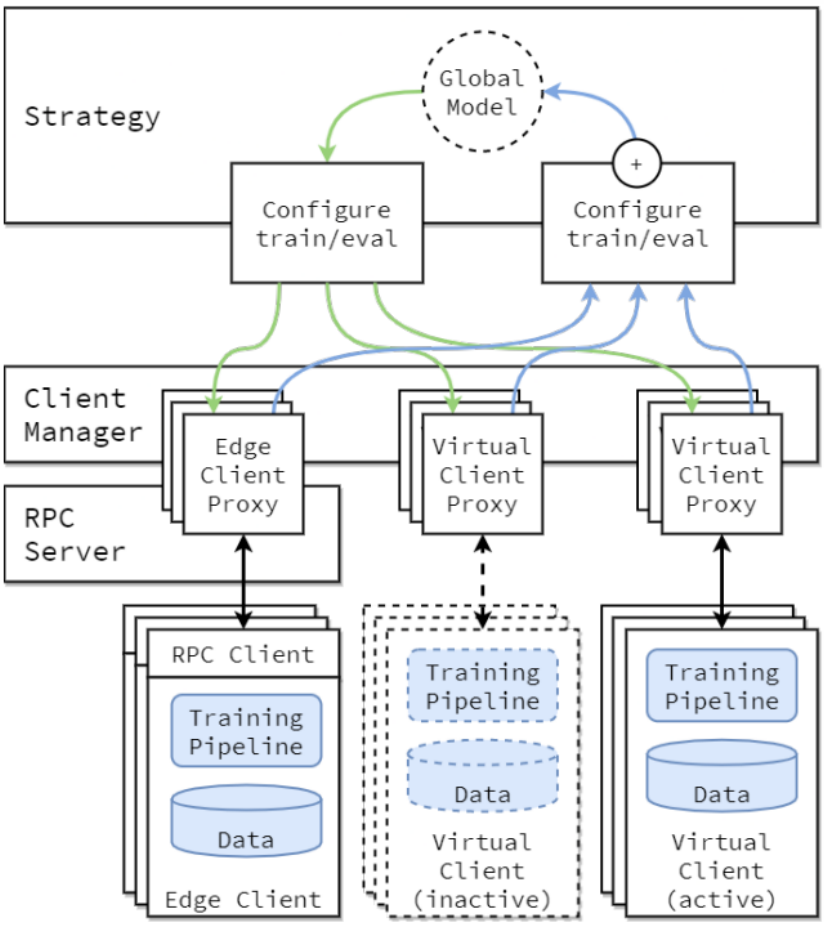



Figure 10 – Illustration of Flower architecture. Retrieved from (Beutel et al. 2022).

**Appendix 18**

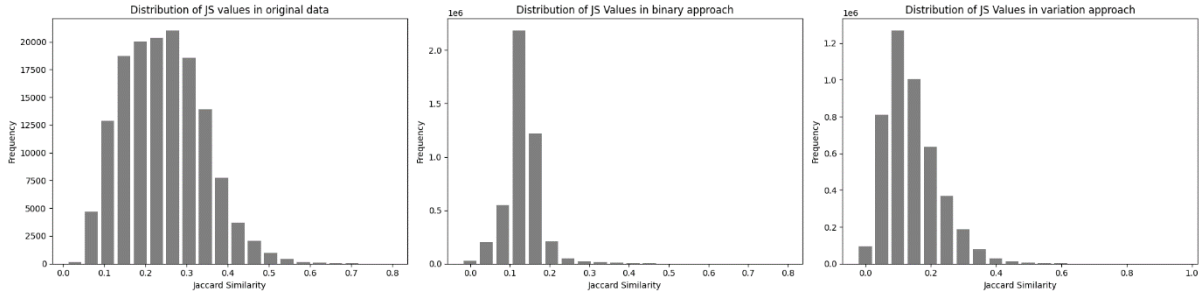
Jensen-Shannon Divergence metric measures the similarity between two distributions and can be defined as follows:

$$JSD\big(p(x), q(x)\big) = \frac{1}{2} D\left(p(x),\ \frac{1}{2}p(x) + q(x)\right) + \frac{1}{2} D\left(q(x),\ \frac{1}{2}p(x) + q(x)\right) \text{ where } D(\alpha, \beta)$$

denotes the divergence between probability distributions $\alpha$ and $\beta$

Equation 10 – Jensen-Shannon Divergence Calculation

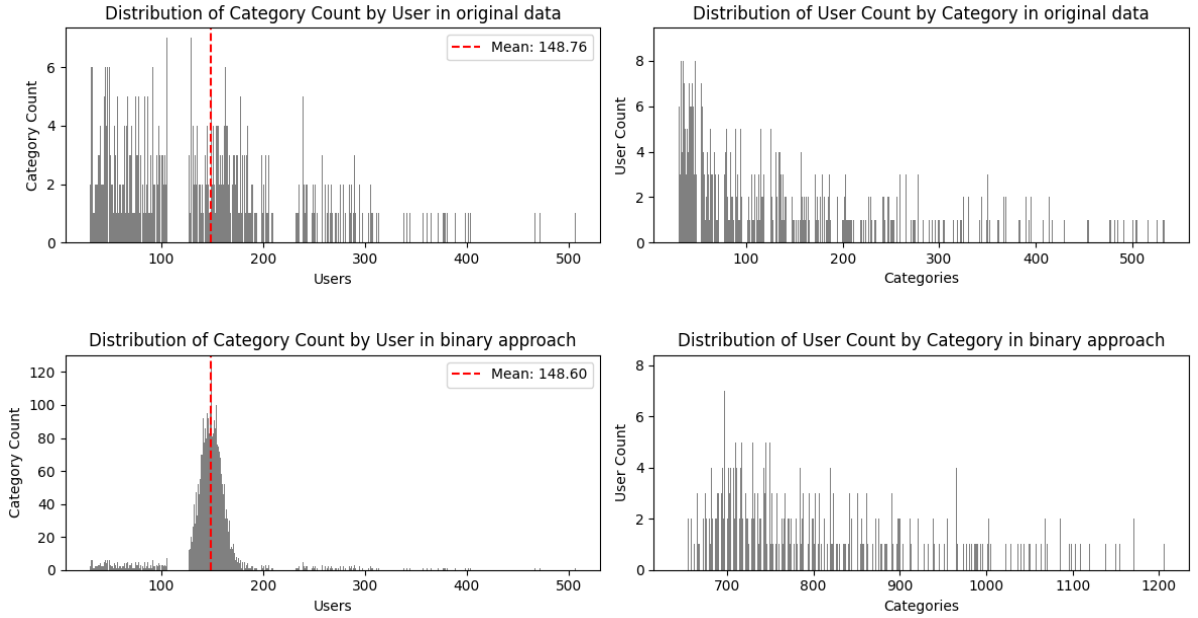
**Appendix 19**

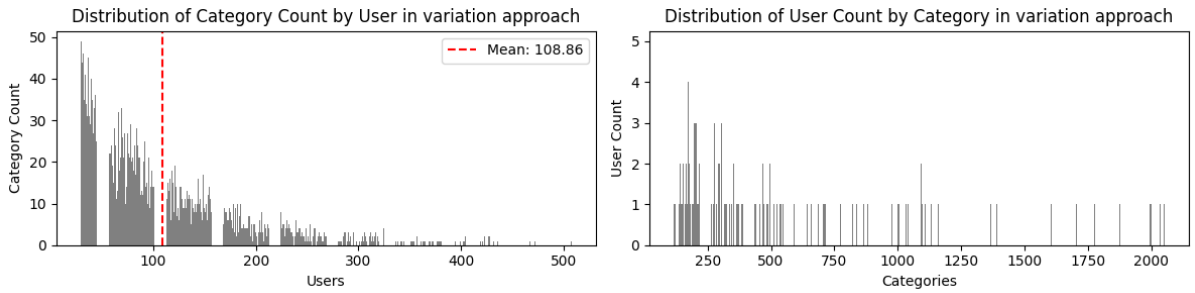



Graphic 4 – Jaccard Similarity Scores values distribution of the original data (center), of the data generated from the binary approach (center) and of the data generated from the variation approach (right).

| | # Duplicate Rows | Overall % of 1 | Mean | Standard Deviation | Skewness | Average JSD | Average JSC |
|---|---|---|---|---|---|---|---|
| Original Data | 0 | 26,660 | 0,267 | 0,442 | 1,056 | | 0,263 |
| Binary Approach | 0 | 26,630 | 0,266 | 0,442 | 1,057 | 0,095 | 0,153 |
| Variation Approach | 0 | 19,510 | 0,195 | 0,396 | 1,539 | 0,057 | 0,167 |

Table 7 – Statistical metrics results in the original data, the data generated from the binary approach and data generated from the variation approach.






Graphic 5 – Distribution of category by user (left) and distribution of user count by category (right) of the original data (top), of the data generated from the binary approach (center) and of the data generated from the variation approach (bottom).

**Appendix 20**

TP = True positives; TN = True negatives; FP = False positives; FN = False negatives

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Equation 11 – Accuracy Score Calculation.

$$\Pr e\,cision = \frac{TP}{TP + FP}$$

Equation 12 – Precision Score Calculation.

$$Recall = \frac{TP}{TP + FN}$$

Equation 13 – Recall Score Calculation.

$$F1 = 2 \cdot \frac{\Pr e\,cision \cdot Recall}{\Pr e\,cision + Recall}$$

Equation 14 – F1 Score Calculation.

ROC AUC is obtained by plotting the true positive rate against the false positive rate curve at different classification thresholds and measuring the area under the curve.

**Appendix 24**

**Acronym Glossary**

**ALS -** Alternating Least Squares

**CB -** Content-based

**CCPA** – California Consumer Privacy Act

**C** – Classifier

**CF -** Collaborative filtering

**CM** – Cryptographic Methods

**CNN** – Convolutional Neural Network

**D** – Discriminator

**DIN** – Deep Interest Network

**DP** – Differential Privacy

**FL** – Federated Learning

**FLS** – Federated Learning System

**GDPR** – General Data Protection Regulation

**G** – Generator

**GAN** – Generative Adversarial Networks

**GC** – Gaussian Copula

**IoT** – Internet of Things

**JSC** – Jaccard Similarity Coefficient

**JSD** – Jensen-Shannon Divergence

**ML** – Machine Learning

**NN** – Neural Network

**Non-IDD** – Non-Independent and Identical

**PDP** – Personal Protection Act

**PLFL** – Partially Local Federated Learning

**RS** – Recommendation System

**SDV** – Synthetic Data Vault

**SMC** – Secure Multiparty Computation

**TVAE** – Triple-base Variational Autoencoder

**TF-IDF** - Term Frequency-Inverse Document Frequency

**UA** – Universal Aggregation

**VAE** – Variational Autoencoder