MS5103 - Business Analytics Project

Final Report

Data Anonymisation using Differential Privacy

Group 21

Student Name	Student Id	Email
Eeshita Ray Chowdhury	19230471	E.Chowdhury1@nuigalway.ie
Tharunkumar Palanichamy	19230822	T.Palanichamy1@nuigalway.ie
Jayakarthi Boovendran	19230487	J.Boovendran1@nuigalway.ie
Pratyush Parida	19230623	P.Parida1@nuigalway.ie

Under the supervision of Dr Noel Carroll

Declaration of Originality

Project Details

Module Code: MS5103 **Assignment Title**: M.Sc. Business Analytics - Major Project

Group Members: Group 21

Student Name	Student Id	Email
Eeshita Ray Chowdhury	19230471	E.Chowdhury1@nuigalway.ie
Tharunkumar Palanichamy	19230822	T.Palanichamy1@nuigalway.ie
Jayakarthi Boovendran	19230487	J.Boovendran1@nuigalway.ie
Pratyush Parida	19230623	P.Parida1@nuigalway.ie

We hereby declare that this project is our original work. We have read the University *Code of Practice for Dealing with Plagiarism*[*] and are aware that the possible penalties for plagiarism include expulsion from the University.

Signature	Date
Eeshita Ray Chowdhury	04 July 2020
Tharunkumar Palanichamy	04 July 2020
Jayakarthi Boovendran	04 July 2020
Pratyush Parida	04 July 2020

Executive Summary

In this digital epoch, implementing a privacy-preserving data analysis in a customer-centric organisation is tedious. The main reason is that they have to protect the collected data from malicious attackers; also, they have to comply with the government regulations on customer privacy policies. In case of a privacy breach, organisations might lose its valuable customers and may end up paying lump-sum fine amounts. This report outlines the research carried out to perform privacy-preserving data analysis using one of the emerging privacy-enhancing technique called Differential Privacy. To show the advantages of using this privacy-preserving technique, we have performed statistical analysis on the finance and e-commerce datasets, by executing differential privacy queries along with the analytical models of Logistic Regression, Linear Regression, Customer Segmentation, Recency Frequency and Monetary (RFM) analysis in order to perform data anonymisation. Throughout the report, we have visualised the results of our analysis through useful graphical representations. The analytical tools that we used to perform differentially private analysis include Tableau, Python, TabPy and IBM differential privacy library.

Table of Contents

1. Introduction

As our interactions with the internet are growing daily, our digital footprint keeps increasing drastically. Whenever an enigma arises in our mind, we turn to our search engines for the answers, and that adds stockpile of data every time. The Forbes article states that more than 3.7 billion humans use the internet worldwide, and half of the web searches originate from mobile phones (Marr 2018). The ever-increasing use of the internet supports businesses around the globe in many ways to collect user's data. To move their businesses to the next frontier, most companies use their customers' information to analyse and uncover meaningful insights from them, and that data may contain both sensitive and insensitive information. As customer data is considered one of the most valuable treasure, few organisations built their entire business model around customer data. The data collected by organisations typically fall into four categories (Freedman, 2020).

1. Personal data- This data contains sensitive information such as name, email, gender, IP addresses of mobile phones and laptops, web browser cookies.
2. Engagement data- This data contains information such as the interaction way of the customers with mobile applications, web pages and social media sites.
3. Behavioural data- It contains the transactional data or the actions repeated by the customer such as purchase history, product usage and qualitative data.
4. Attitudinal data-consumer satisfaction data, purchase criteria and product desirability data are the attitudinal data.

Companies are collecting the above-said data by either directly asking the customers or indirectly tracking the customers mainly to enhance their business values (Freedman 2020). However, companies are aware of collecting personal data which has no immediate use, and most of them dominate by keeping their customer in the dark by controlling them (Morey et al. 2015). Hence, data sharing has raised data-privacy concerns. Government is also concerned about how organisations are making use of their customer's private information because their citizens are aware that they are monitored and worried about how enterprises will handle their data (Morey et al. 2015).

The customer-centric sectors like healthcare, finance and retail mostly rely on their customers' data mainly to enhance their business value. For example, to find a pattern to treat new patients, medical researchers require access to the medical history of previous patients diagnosed with the same disease. Researchers can share that diagnose pattern with other researchers as it helps to treat countless patients around the world. In the retail sector, from customer segmentation to market basket analysis, consumer data plays a significant role in retaining their business conditions. In some situations, without the organisation's knowledge, a customer's data gets revealed, which will result in paying a lump-sum fine amount to the governments and losing their customers' goodwill (Morey et al. 2015). Here, the breach of privacy comes into the public picture. However, data sharing in the financial and retail sectors are booming nowadays, confidentiality plays a vital role. So, in this digital epoch, customer privacy and data security have become a key concern, particularly around data collection and data analysis.

Columbia Business School (2015) states that the consumers are more aware of which data are most sensitive; and still, are willing to share it with businesses in exchange for a service or product they value. Also, the Columbia Business School claims that nearly 75% of people are more willing to share sensitive data with a brand they trust. Therefore, it is quite evident that Brand trust positively impacts consumers' willingness to share data which ultimately enables the businesses to use them for customer-centric promotional activities. The following figures represent results from the research conducted by the school relating to data sharing in six different industries such as finance, retail, telecommunications, airlines and e-commerce, and in five different countries such as the US, Canada, UK, France and India.

85%	Want to **know more information** about the data companies collect
86%	Want to **exercise greater control** over the data companies hold about them
80%	Will only **give data to a limited number of companies** they trust
75%	Are **worried sharing data makes them targets** for marketing campaigns
59%	Have taken steps to **limit companies from tracking** and advertising to them
34%	Have provided **made-up personal details** to avoid giving away personal data

Figure 1: People's concerns on personal data handling (Columbia Business School 2015)

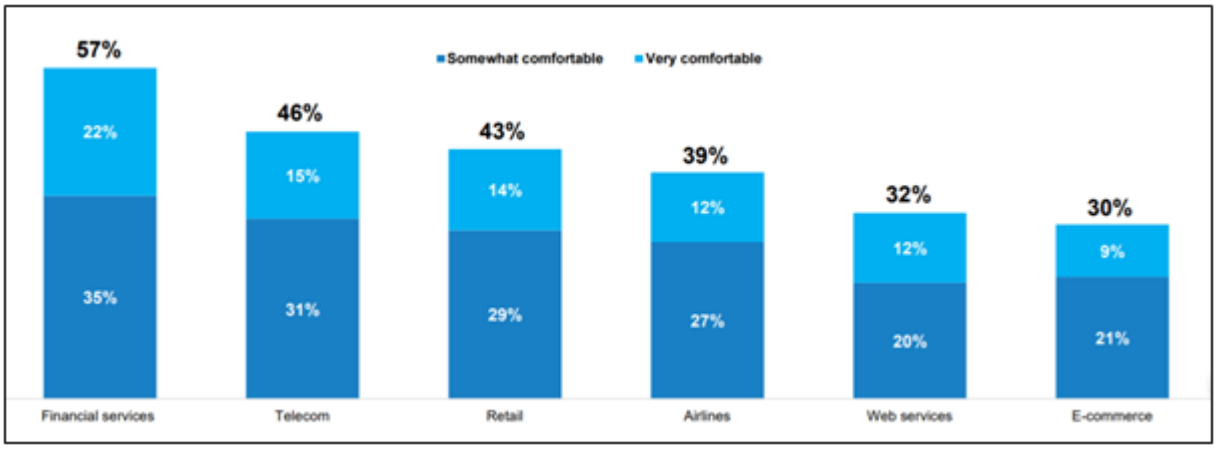


Figure 2: How comfortable are people when sharing data in different sectors (Columbia Business School 2015)

From the above figure, it is evident that in the retail sector, only a negligible percentage of consumers are very comfortable in sharing their data for processing. Therefore, it is critical to building trust with the customers by guaranteeing the preservation of valuable information from a breach, also, for appropriate use in marketing. The following figures give customers' perspective on what they consider as sensitive information and how willing are they to share it with businesses.

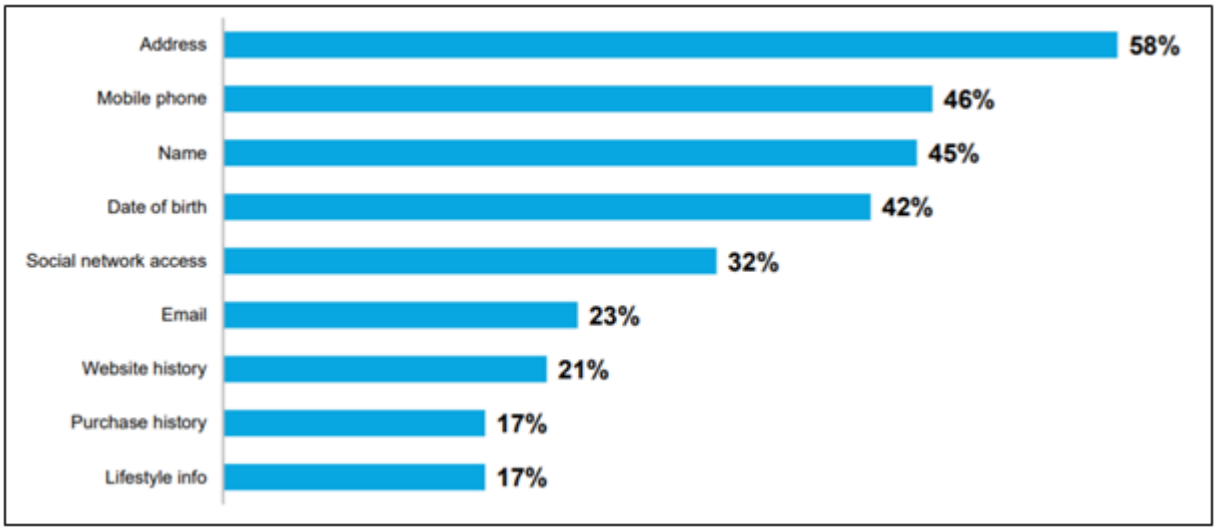


Figure 3: Personal Identifying Information (PII) and their sensitivity rate

(Columbia Business School 2015)

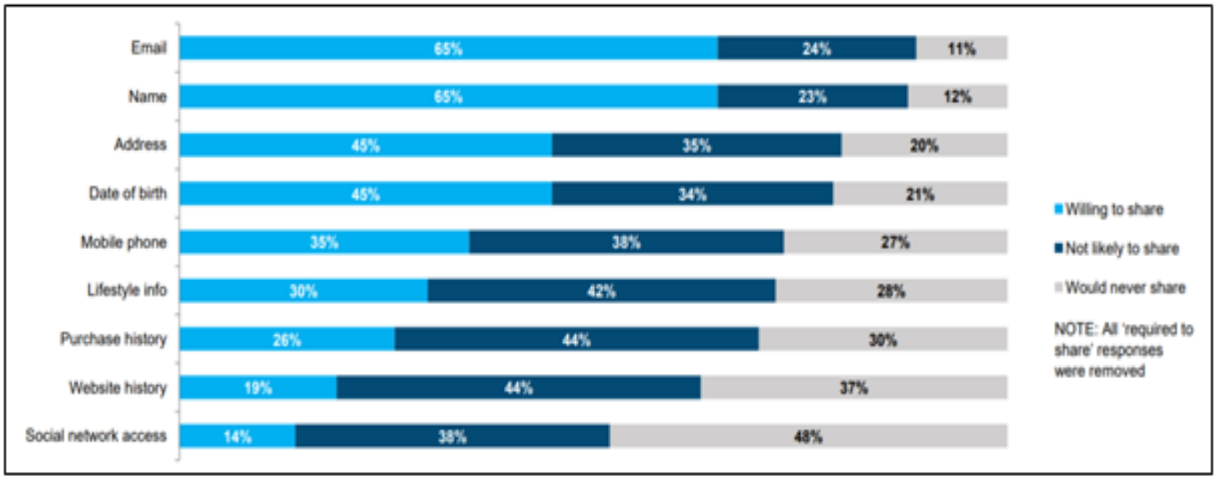


Figure 4: Willingness of consumers in sharing sensitive data

(Columbia Business School 2015)

In addition to Personal identifying information (PII), a few sectors also expects non-PII information such as hobbies, lifestyle, and social data to provide more exciting offers, recommendations, coupons or networked connections. The primary issue that prevents consumers from sharing the information mentioned earlier is the fear of breach or misuse. Therefore, preserving customer information should be a significant step for businesses in building trust. This study covers one such technique that not only safeguards customer information but also allows for data analytic operations to become more customer-centric.

1.1. Problem Statement

Businesses collect, store and analyse customer data to retrieve valuable information that could support data-driven decision making. In general, companies make use of internal or third-party analysts to help mine insights from the data. Companies then convert these insights into executable action plans or business strategies. However, it is quintessential for the companies to strictly maintain the confidentiality of the customer data while mining for insights. To facilitate this, companies must adhere to the data regulation rules enforced by the government. According to article 5 of the General Data Protection Regulation (GDPR), standard principles of data protection are:

1. Lawfulness, fairness, and transparency
2. Purpose Limitation
3. Data Minimization
4. Integrity and Confidentiality
5. Accountability

Data leakage, in a few cases, can occur without the knowledge of the company or the customers. For instance, in early 2018, Cambridge Analytica harvested personal data of millions of individuals from Facebook profiles without their consent and used it for political advertisements (Wong 2019).

For the past few years, retailers have utilised customer data to establish better customer relationships, provide customised retailing experience to the customers and accomplish competitive advantage in the market. From facial recognition features to location trackers, the retailers gather data using the latest available technologies in the market. Though these data can benefit the business, there is a risk of a data breach. Therefore, it is critical to attend to the data protection standards to prevent the data from unauthorised on unintentional access.

The complexity of data analysis increases with an increase in the volume of data. Therefore, almost two-thirds of the organisations have invested in more than fifty information systems such as the spreadsheets, CRM tools, data warehouses servers and ETL systems to aid the analysis process. Further, in real-time, data is spread across different BI systems, as mentioned earlier, and implementing privacy standards on the data often impose a challenge of bringing the data on to the same platform. Also, many retailers find it challenging to pay for the privacy programs which are required. These issues act as a blocker for providing adequate privacy protection to customer data.

Such privacy related issues have raised concerns and increased the need for better privacy technologies which can provide insulation to any potential attack or leakage. As discussed earlier, the data may reside at multiple systems. This means that taking the data back is not an effective solution after a data breach. In response to these flaws of privacy technologies, a new methodology has emerged which is called **Differential Privacy (DP).**

The primary objective of this project is to provide better insights for retail sector by effectively analysing customer data without breaching customer privacy policies which in turn helps to enhance the performance of their business and helps them to stay ahead in the competitive market. The above problem requires a solution which needs to be unique from the existing privacy-enhancing methods that offer more good insights and outcomes.

2. Differential Privacy

This project aims to employ Differential Privacy (DP), an emerging **privacy-enhancing technique** to protect the privacy of customer's data while analysing their information to gain business insights from it.

2.1 What is Differential Privacy?

Differential Privacy (DP) is a privacy-enhancing mechanism which provides a robust mathematical definition of privacy in terms of machine learning and statistical analysis (Nguyen 2019). DP's main aim is to provide a guarantee for a privacy-preserving data analysis process. Informally, the guarantee says that the behaviour of the mechanism is essentially unchanged independent of whether any individual opts into or opts out of the data set. Further, differential privacy intends to provide data anonymity while preserving admittance to a high volume of valuable data. For the chosen retail dataset differential private queries will be used to differentiate how differential privacy varies from the standard visualisation methods which can be visualised easily, keeping in mind the privacy constraint.

Differential privacy, a new form of data science, is used by tech giants like Google, Uber and Apple to mine large of amount of data without violating their user's privacy. Also, it was implemented by federal agencies such as the U.S. Census Bureau (Wood et al., 2018). Google uses differential privacy to weigh how popular a specific restaurant's dish is on Google Maps and for improving Google fi (Choi 2018). Apple uses this method to find the popular emoji used by its users without reading the text by injecting some noise into it (Anon 2017). World's leading tech giants have already adopted this mechanism to mine the user's information without violating their privacy and mainly because of its promising outcomes (Khalid 2018). So, differential privacy can be an alternate approach for companies who wish to protect their customer's privacy and to protect them from a wide range of privacy attacks.

2.2 How it works?

Differential privacy is based on the injection of randomised noise into the data analysis process that limits the ability of an outsider attempting to deduce the inputs to an analysis from the results of the analysis. Further, it provides a formal guarantee that the output of an analytics operation will not reveal sensitive information about any individual present in the database.

In this case, for removing private information in the data, a Differential Privacy algorithm was designed to perform the analysis of data. For each entry,

- Flip a coin.
- If heads, return the answer in the entry.
- If tails, then flip a second coin and return "yes" if heads and "no" if tails.

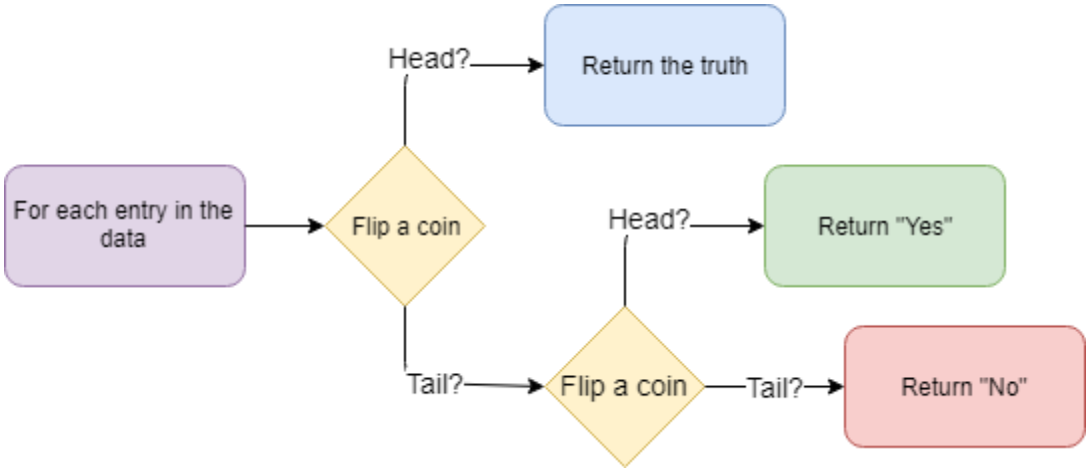


Figure 5: Flow diagram of the Differential privacy algorithm (Nguyen 2019)

2.3 The Promise of Differential Privacy

- DP describes a promise made by a data holder that users will not be affected by allowing their data to be used in any study or analysis, no matter what other studies, data sets, or information sources, are available.
- DP database mechanisms can make confidential data widely available for accurate data analysis, without resorting to clean data rooms, data usage agreements, data protection plans, or restricted views.
- In general speaking of data mining, the richer the data, the more interesting and useful it is. So, when doing such analysis of interesting data, the privacy needs to be taken as an important aspect and exploration are made. This has led to notions of "anonymization" and "removal of personally identifiable information," where the hope is that portions of the data records can be suppressed, and the remainder published and used for analysis (Dwork and Roth, 2013, p. 05-06).

2.4 Differential Privacy Libraries

There are several general-purpose libraries for developing differential privacy applications. However, the most renowned are Google Differential Privacy Library and IBM Differential Privacy Library.

2.4.1 Google Differential Privacy Library

Programmed in C++, the Google differential privacy library focuses primarily on providing differential privacy to PostgreSQL based applications. The core concept of differential privacy centers around aggregating data to prevent revealing sensitive personal information. Therefore, the library offers a variety of standard **statistical functions** like (count, sum, mean, variance, and standard deviation) for aggregating data and a **PostgreSQL extension** to execute differentially private SQL queries in PostgreSQL (Google et al. 2018).

The library encapsulates its functionalities in a SQL extension that offers data anonymization at the database level and reports the anonymized results to the user. However, one of the critical bottlenecks of implementing the library is that it intends to support differential privacy only in PostgreSQL. Since one of the primary objectives of the project is to facilitate data sharing regardless of the underlying storage technology, also, the majority of the financial firms and enterprises make use of standard MySQL databases for their operations, it is challenging to use Google differential privacy for enforcing data anonymization. Further, the installation process is also complicated and time-consuming.

2.4.2 IBM Differential Privacy Library

The IBM differential privacy library, also known as 'Diffprivlib', is an open-source library for developing Python-based differential privacy applications. The library includes several mechanisms for employing differential privacy alongside numerous tools and models for machine learning and data analytics tasks (Holohan et al. 2019).

The library leverages the functionality of the NumPy and Scikit-learn packages, making the models instantly recognizable. Further, the parameters that define differential privacy are set as default parameters ensuring convenience for all. Just like the Scikit-learn, diffprivlib machine learning models can be trained in just two lines of code. The library is famous among the data science practitioners because of the machine learning capabilities, accessibility and active user community of Python programming language.

3. Potential audience

Any sector that handles sensitive data and is prone to a data breach can be the potential audience of differential privacy applications. For instance, industries like finance, retail and healthcare can benefit from employing differential privacy techniques. In our initial work with Query Layer, we incorporated differential privacy solutions to a financial dataset using the Google differential privacy library. However, the study focuses primarily on the retail sector, where we implemented differential privacy solutions to an e-commerce dataset using the IBM differential privacy library. Therefore, the target audience of this study are the **Retail Industries**.

4. Initial Research with QueryLayer

Query Layer is a Business Intelligence (BI) platform that facilitates data-sharing that is compliant with regulatory principles, protects the privacy of customers and safeguards the confidentiality of institutions' business processes. In real-time, the Query Layer serves as a middleware that bridges the gap between the data source and BI/Visualization. In simple terms, the Query Layer takes in users' constraints in the form of standard SQL queries, executes them, and maps the results on to visualizations. In Query Layer, the privacy of the data is preserved at the query level where the differential privacy is employed.

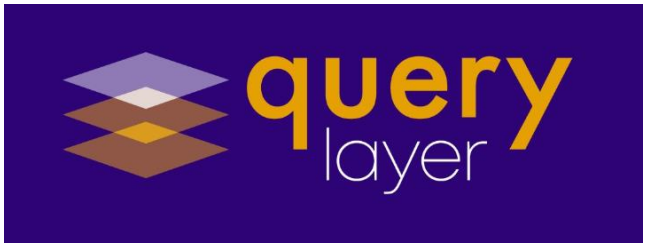


Figure 6: Query Layer logo (https://www.querylayer.com/)

One of the key characteristics of the Query Layer is that it enables the data owner to define the level of access that the data users can have on the data. Moreover, the Query Layer allows integration with popular BI tools. As the Query Layer is compatible with major BI tools, it does not demand any remodeling to the existing system architecture. It is impressive that the Query Layer can work on any data regardless of their type and where they are stored. Furthermore, the Query Layer platform offers live-data sharing from a single data source through incorporating the access rules set by the data owner, to be precise, is no data is copied or replicated. The Query Layers also offers major data visualizations like the bar chart, pie chart, line graphs, area charts, scatter plots, box plots, cohorts, sunburst sequence maps, and features like filters and dashboard level parameters to interact with the users.

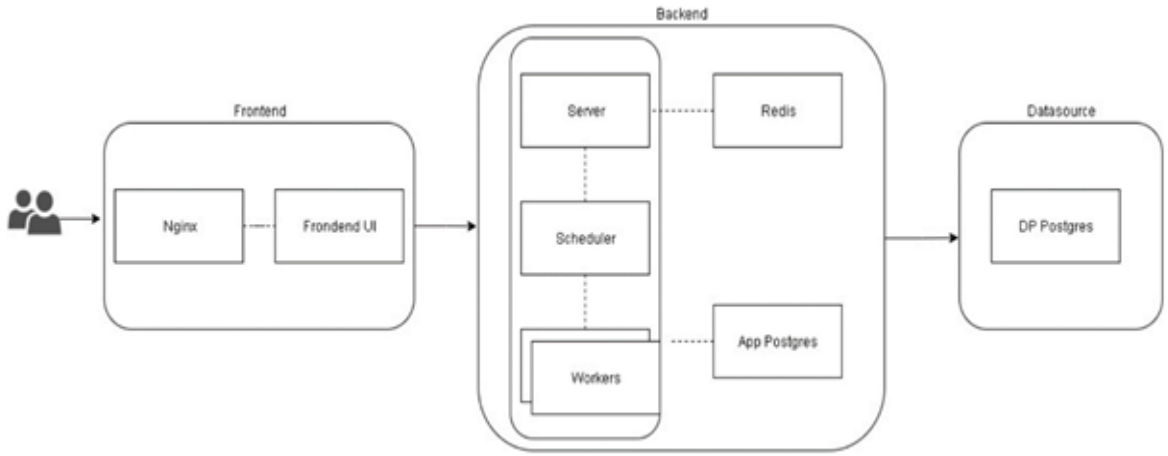


Figure 7: Data flow diagram of Query Layer

4.1. Privacy Solutions to Fintech

In our initial work with Query Layer, we focused primarily on providing differential privacy solutions relating to financial data. In general Fintech companies, comprise of operations such as online payments, money transfers, loans approvals, crowdfunding, asset management and customer profiling, which have high risks of a data breach if not administered properly. Therefore, the primary objective of the research is to accomplish data anonymisation by applying differential privacy, with the tools and techniques offered by the Query Layer, eventually, offer more beneficial insights and outcomes to improve performance of Fintech companies.

4.2. Underlying Concept of Differential Privacy

Many of the world's top MNC's such as Apple, IBM and Google employ Differential privacy as a reliable technique to analyse customer data anonymously. The underlying concept of differential privacy is the 'Addition of random noise', that is, a certain amount of noise is injected to the data, modifying it, and eventually making it impossible to trace back the original data. The Google differential privacy library handles this addition of noise using a SQL extension. To be precise, the SQL extension adds noise to the data at the database level through the use of differentially private SQL queries in order to obtain anonymised resultsets.

4.3. Dataset Description

We made use of a publicly available credit card dataset to implement differentially private solutions using the tools offered by Query Layer. The dataset contains historical credit card data from January 2009 to April 2019 containing infomation on the volume of general-purpose credit cards originated each month, year over year changes in the number and volume of credit cards, inquiry index, credit tightness index, and non-identifiable customer information such as age, gender and geolocations.

- **Dataset**: Consumer Financial Protection Bureau (CFPB) - Credit cards
- Available at https://www.consumerfinance.gov/data-research/consumer-credit-trends/

4.4. Exploratory Data Analysis using Query Layer

As an initial step, we performed exploratory data analysis on credit card data to showcase the visualisation features offered by Query Layer. We analysed critical data points such as the number of credit card originations, aggregate credit limits over the years, credit inquiry index, credit tightness index, customer credit scores, credit limit and purchase rate using differentially private SQL queries. In addition to the before-mentioned factors, we also included factors like the gender, age group and location of customers in the analysis.

We then considered the credit card users with positive and negative credit card balance along with Geo Spatial and Age-wise Analysis of Credit Card Usage. For the same, we considered metrics such as Customer wise Credit Card Balance Amount, Year-wise Balance, Spending and Deposit Pattern of a customer, Country-wise Average Balance Amount, Age-wise Average Balance Amount, Age-wise Average Withdrawal/Spending Amount and Age wise Average Deposit Amount.

We mapped the query results from both standard SQL and differentially private SQL queries on to appropriate visualizations using the visualization tools offered by the Query Layer. Finally, we created interactive dashboards by grouping the visualizations relating to the parameters mentioned earlier. A link to the video demonstrating the initial analysis is included in the Appendix.

Sample Queries and Results

- ***Customer with Negative Balance Query*:**

  select cust_id, sum(balance_amt) as SUMBA, ANON_SUM(balance_amt) as SUMBA_0, ANON_SUM(balance_amt,.1) as SUMBA_1, ANON_SUM(balance_amt,.2) as SUMBA_2 from dp_transactions group by cust_id having sum(balance_amt)<0;

cust_id	sumba	sumba_0	sumba_1	sumba_2
C12815	-3,275,840,000,000.00	-549,755,813,888.00	-549,755,813,888.00	-532,575,944,704.00
C10506	-3,772,120,000.00	838,860,800.00		805,306,368.00
C16496	-81,413,100,000,000.00	-545,460,846,592.00	-481,036,337,152.00	-549,755,813,888.00
C17580	-2,494,860,000,000.00	-549,755,813,888.00	-515,396,075,520.00	-532,575,944,704.00
C16329	-16,047,600,000,000.00	-549,755,813,888.00	-549,755,813,888.00	-549,755,813,888.00
C10994	-7,122,920,000,000.00	-547,608,330,240.00	-515,396,075,520.00	-498,216,206,336.00
C13483	-24,310,800,000.00			
C10667	-52,860,000,000,000.00	-545,460,846,592.00	-549,755,813,888.00	-515,396,075,520.00

Figure 8: Query Result from Query Layer

Explanation: *Just for an explanation, we have kept the real average of positive balance i.e.* ***sumba*** *of customers as shown above. But when the noise is added by using anon_avg(amount,.1) i.e.* ***sumba_0*** *or anon_avg(amount,5) i.e.* ***sumba_1*** *or anon_avg(amount,10) i.e.* ***sumba_2*** *where .1, 5, 10 are the privacy budget to the aggregation performed. In the Query Layer, when we create the dashboards with parameter which is just like filters where the user selects the aggregation of their choice -> sumba_0/ sumba_1/ sumba_2 which is with privacy budget and perform exploratory analysis. We will ignore the real average and give option to choose one of the noised averages so that the real data is hidden. In this way, we promote data analysis using Differential Privacy.*

- ***Least Balance Customer C10667 Query***

  select extract(year from trans_date) as YEAR , avg(balance_amt) as BA, anon_avg(balance_amt) as BA_0, anon_avg(balance_amt,.1) as BA_1, anon_avg(balance_amt,.2) as BA_2 from dp_transactions where cust_id='C10667' group by YEAR;

Table | Balance Trend Of Customer C10667 × | + New Visualization

year	ba	ba_0	ba_1	ba_2
2,015.00	-1,626,999,275.90	-1,626,990,113.68	-1,627,131,840.98	-1,622,907,489.71
2,016.00	-1,649,202,418.34	-1,649,482,190.03	-1,645,249,569.03	-1,650,063,314.20
2,017.00	-1,883,367,847.58	-1,668,118,574.27	-1,668,263,303.73	-1,668,263,303.73
2,018.00	-1,895,838,730.67	-1,712,606,162.81	-1,711,670,790.02	-1,714,733,155.30
2,019.00	-1,900,035,309.71	-1,899,697,073.23	-1,879,048,192.00	-1,903,451,415.27

Figure 9: Query Result from Query Layer

We have created the query with parameters(filters) and have opened the option to the user to choose which customer they want and what type of aggregation with an anonymization average to implement differential privacy.

Figure 10: Sample Query1 with Parameter (filter option)

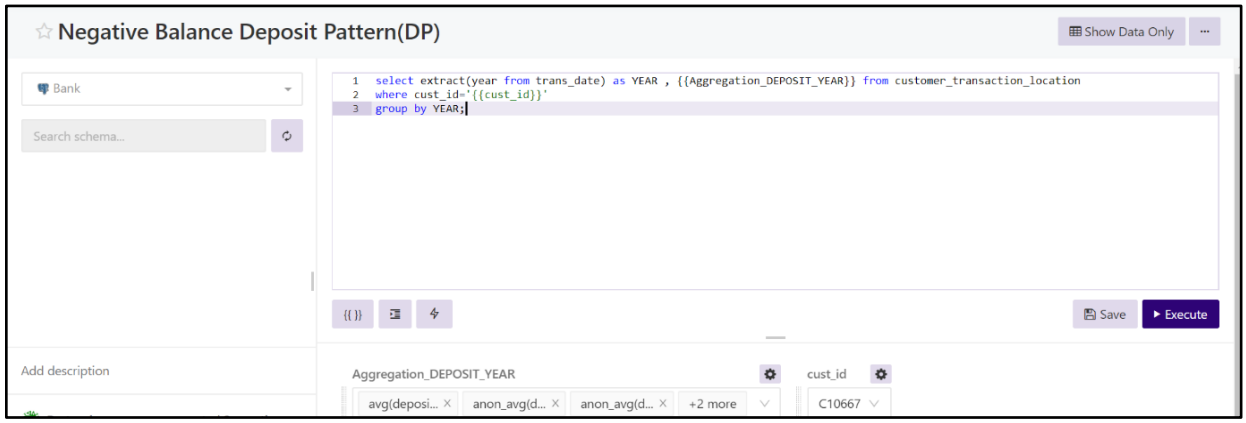


Figure 11: Sample Query 2 with Parameter (filter option)

Figure 12: Sample Query3 with Parameter (filter option) exhibiting dropdown

Visualization Report:

In this part, we have created the visualization report with the metrics as discussed above. Privacy Budget figures are placed in form of filters, where the user picks the value from filter and proceed with data analysis to look at various customer patterns to take some business decisions. The interesting part is we have explored the customer but no where used their real value. Some of the sample visualization reports are as follows.

Figure 13: Visualization report of Year wise Analysis of credit card holding customers where the user chooses the month and year from the drop down to drill down their analysis.

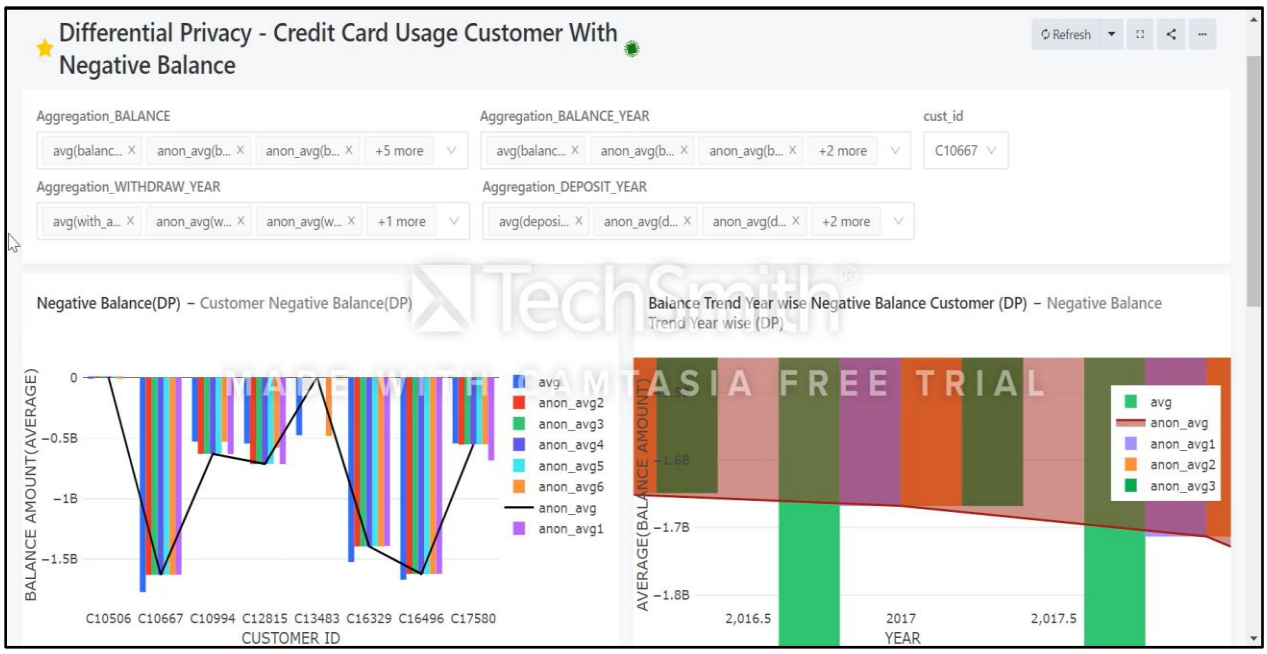


Figure 14: Visualization where all the filter options of differential privacy selected for Credit Card Usage Customer with Negative Balance

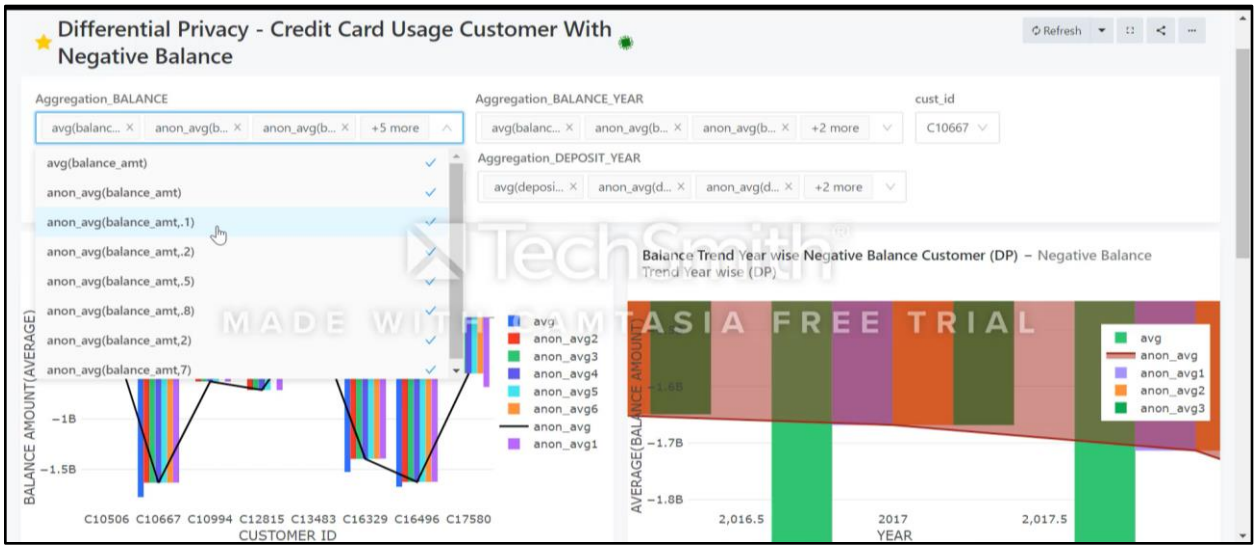


Figure 15: User Option to select any of the Privacy Budget to implement Differential Privacy for a credit card user with negative credit balance

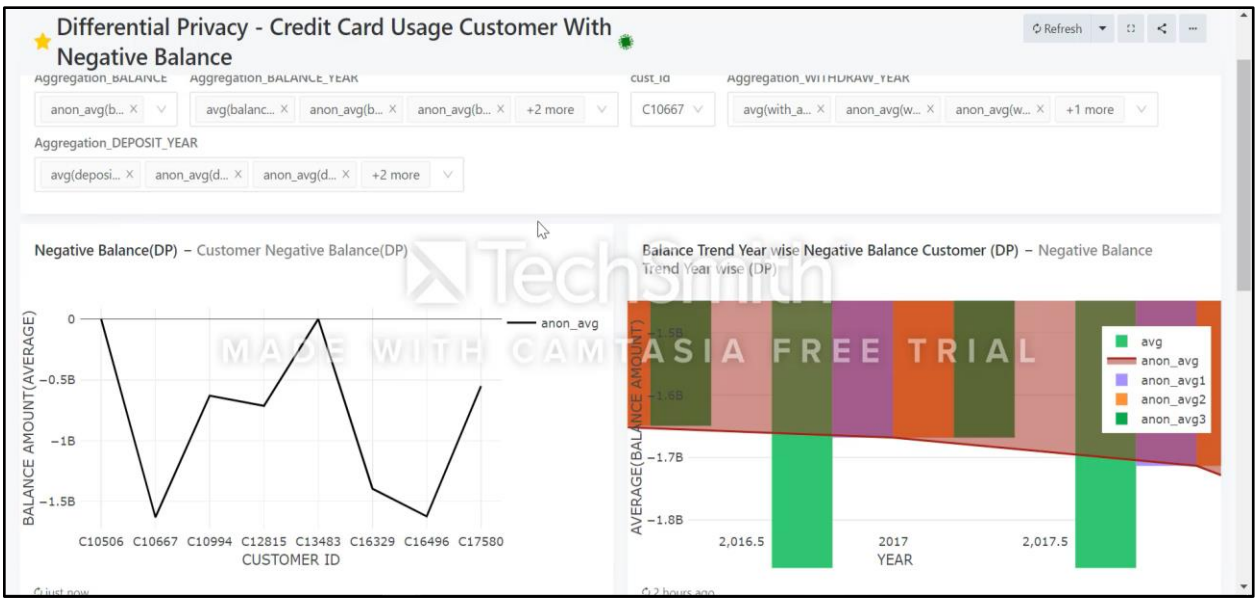


Figure 16: User selects the Privacy Budget 0 (anon_avg) to implement Differential Privacy on the Negative Balance of credit card usage customer.

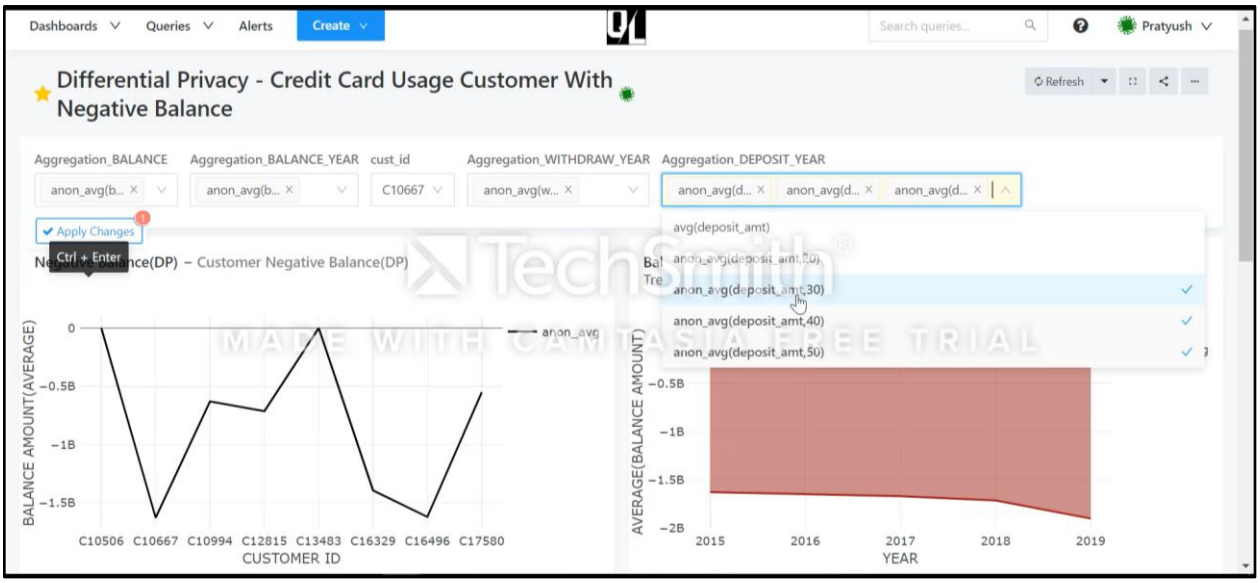


Figure 17: User highlighting the Privacy Budget 30 (anon_avg,deposit_amt,30) to implement Differential Privacy on the deposit amount by the credit card customer users with negative credit balance

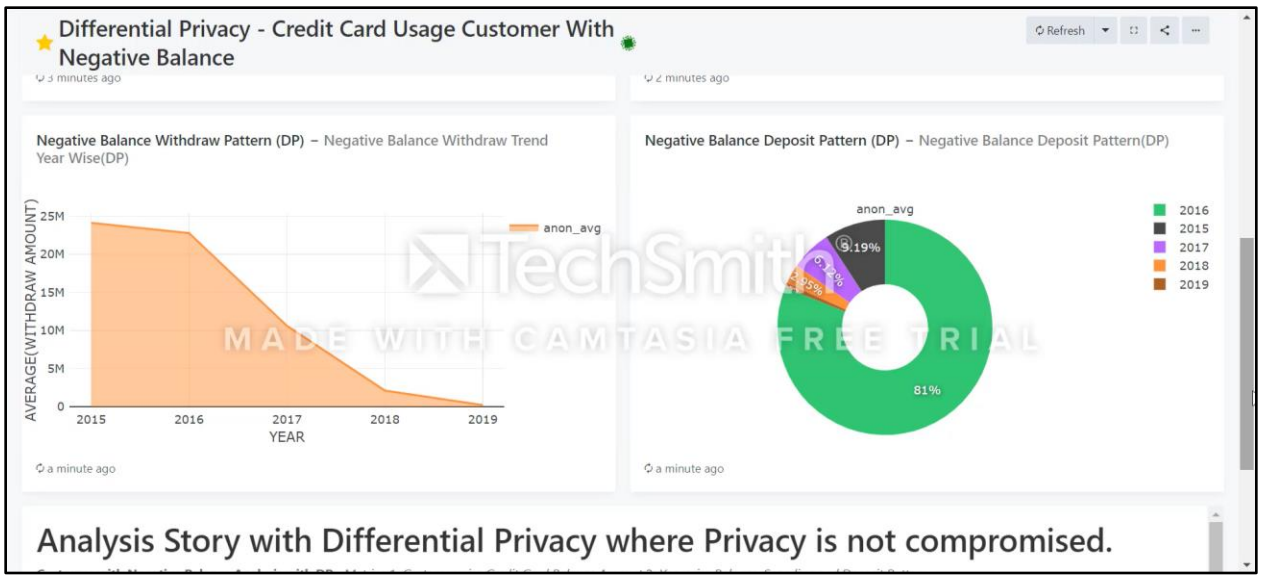


Figure 18: User selects the Privacy Budget 0 (anon_avg) to implement Differential Privacy in understanding the Withdraw Pattern and Deposit Pattern of a Credit Card Usage Customers with negative credit balance

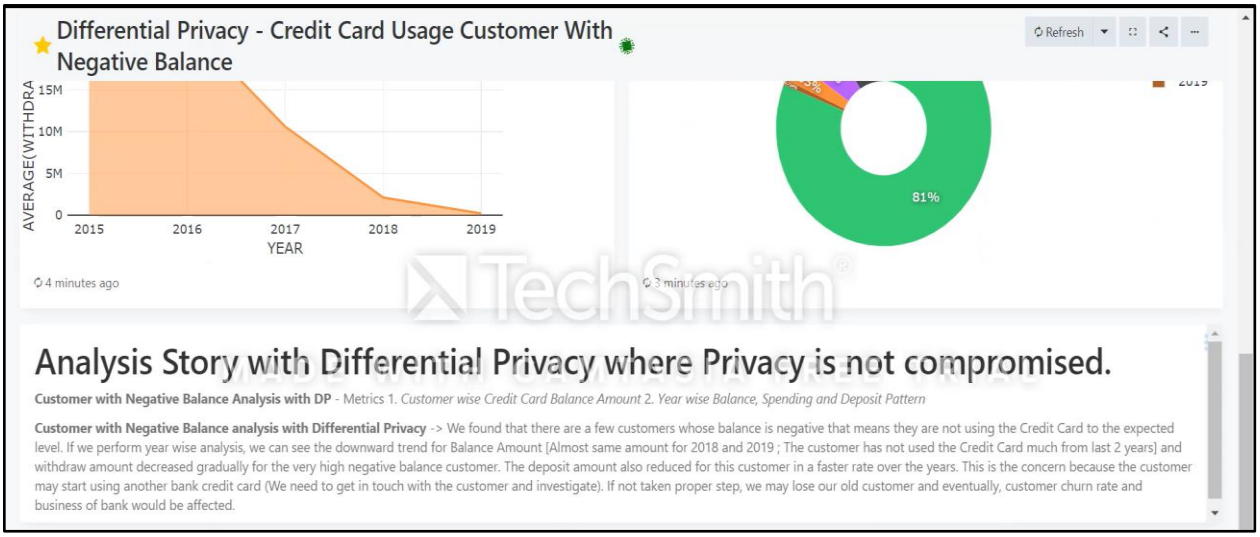


Figure 19: Credit Card Customer analysis report with negative credit balance

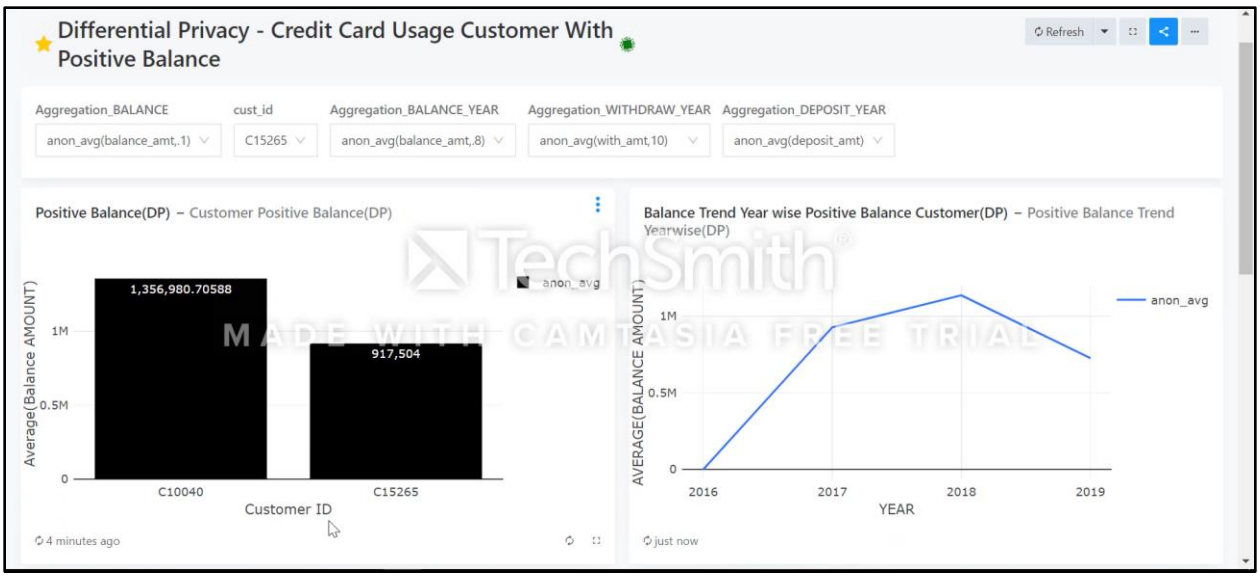


Figure 20: User selects the Privacy Budget .1(anon_avg,balance_amt,.1) on Balance, Privacy Budget .8(anon_avg,balance_amt,.8) on Balance Amount, Privacy Budget 10(anon_avg,with_amt,10) on Withdraw Amount and Privacy Budget 0(anon_avg,deposit_amt,30) on Deposit Amount to implement Differential Privacy for positive credit balance customer

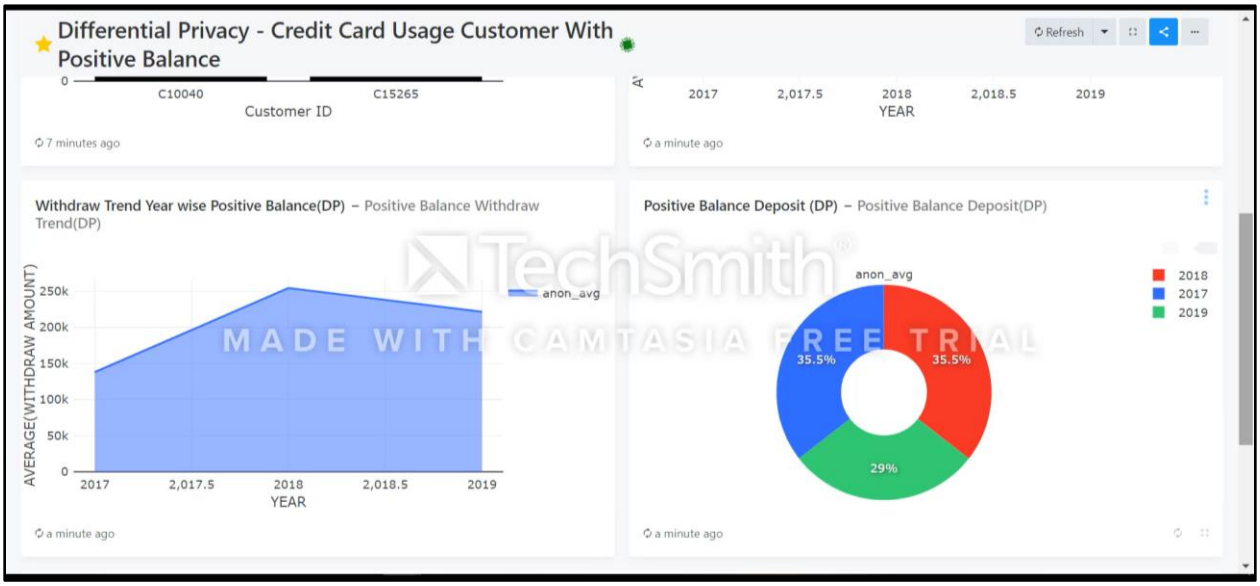


Figure 21: Pattern of Withdraw and Deposit shown after implementing differential privacy for positive credit balance customer

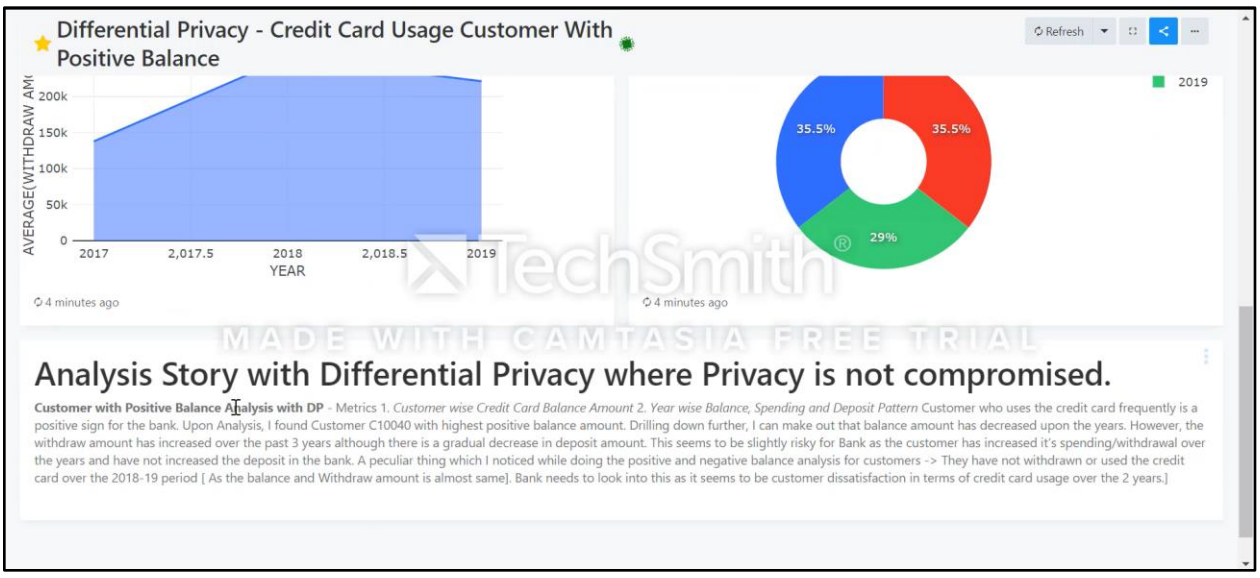


Figure 22: Credit Card Customer analysis report with positive credit balance

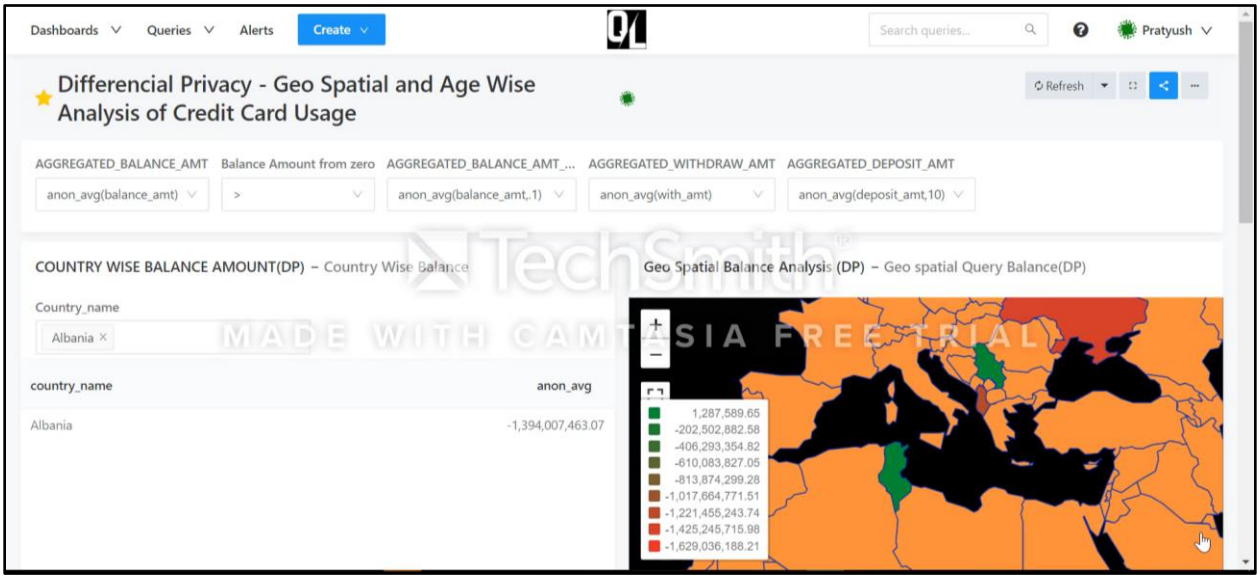


Figure 23: Differential Privacy(anon_avg) implementation on geo spatial analysis where red – Bad Performance in terms of credit card usage by customers, green – Excellent usage and Orange – Non-operational areas

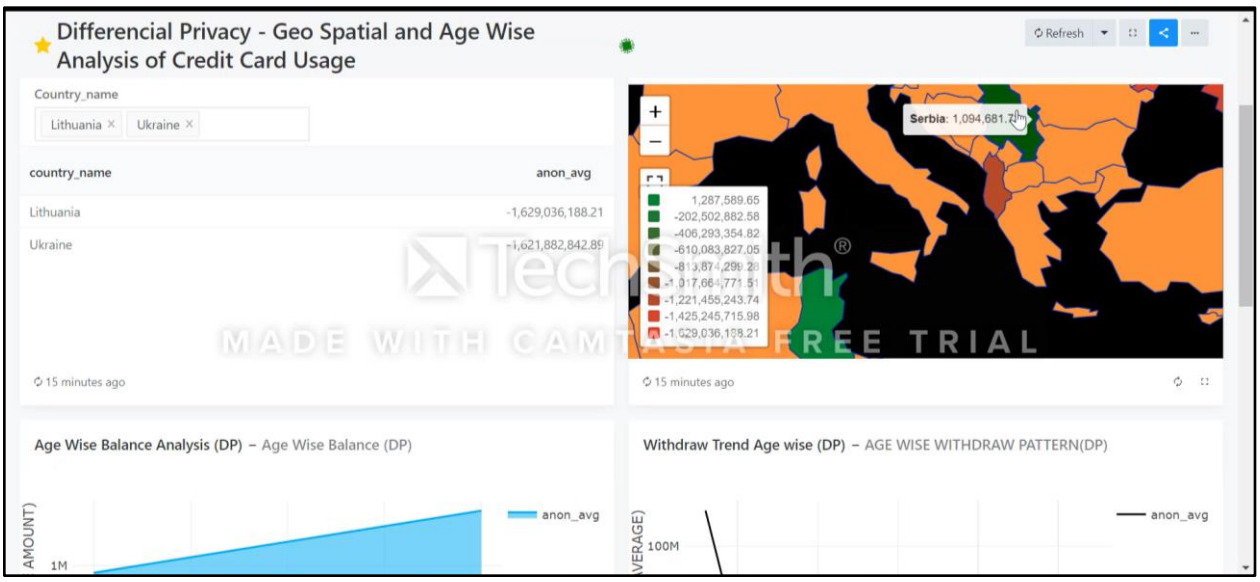


Figure 24: Highlighting the Country along with anonymization value

Figure 25: Age wise Balance Analysis and Withdraw Pattern for positive balance credit customer by implementing Differential Privacy

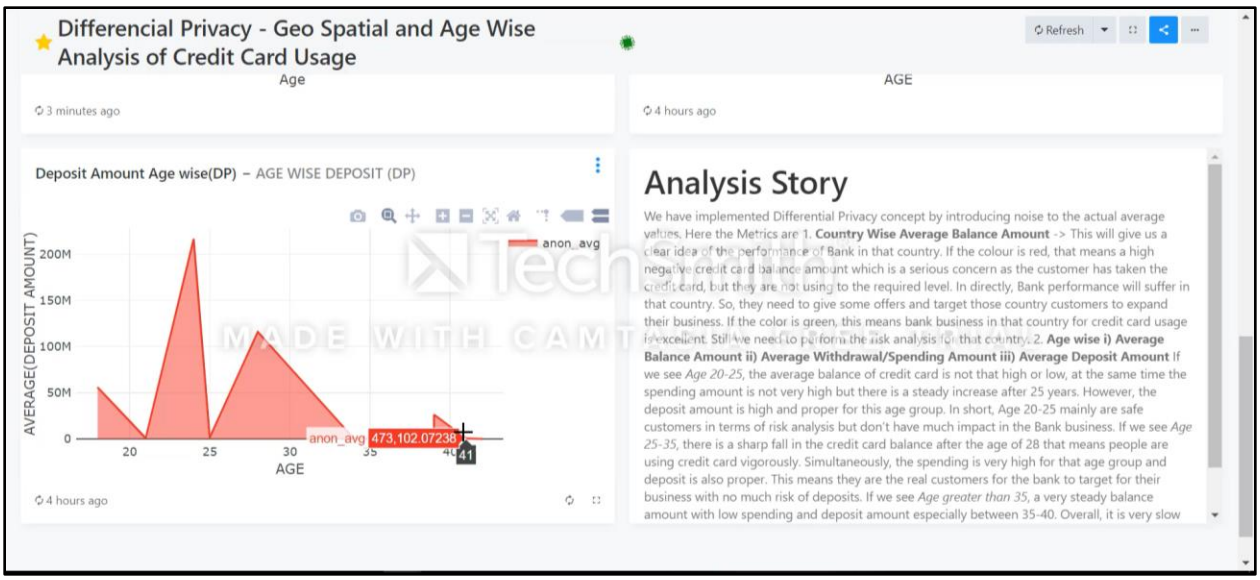


Figure 26: Age wise Deposit Pattern of a positive balance credit customer by implementing Differential Privacy along with the report

4.5. Challenges

As mentioned earlier, the real power of the application lies at its query layer where Differential Privacy is deployed. Currently, the query layer is designed based on Google's Differential Privacy Library that offers anonymisation at the database level. However, one of the significant challenges is that the library intends to support employing differential privacy in PostgreSQL databases only.

One of the primary objectives of the project is to facilitate data sharing regardless of the underlying storage technology. Since the currently employed differential privacy library supports for only PostgreSQL and most of the financial firms make use of standard MySQL databases for their operations, it is hard to satisfy this objective. Therefore, as an alternative solution, we considered a library that supports implementing differential privacy at the code level, making it compatible with any underlying database systems.

5. Implementation of Differential Privacy in Retail Sector

In the alternative approach, we propose to make use of IBM's Python-based differential privacy library for the core functionality. As mentioned earlier, the library aids building differential privacy solutions that not only support data anonymisation despite the underlying storage technology but also offer machine learnings capabilities to perform advanced analytics to data.

5.1 Technical Architecture

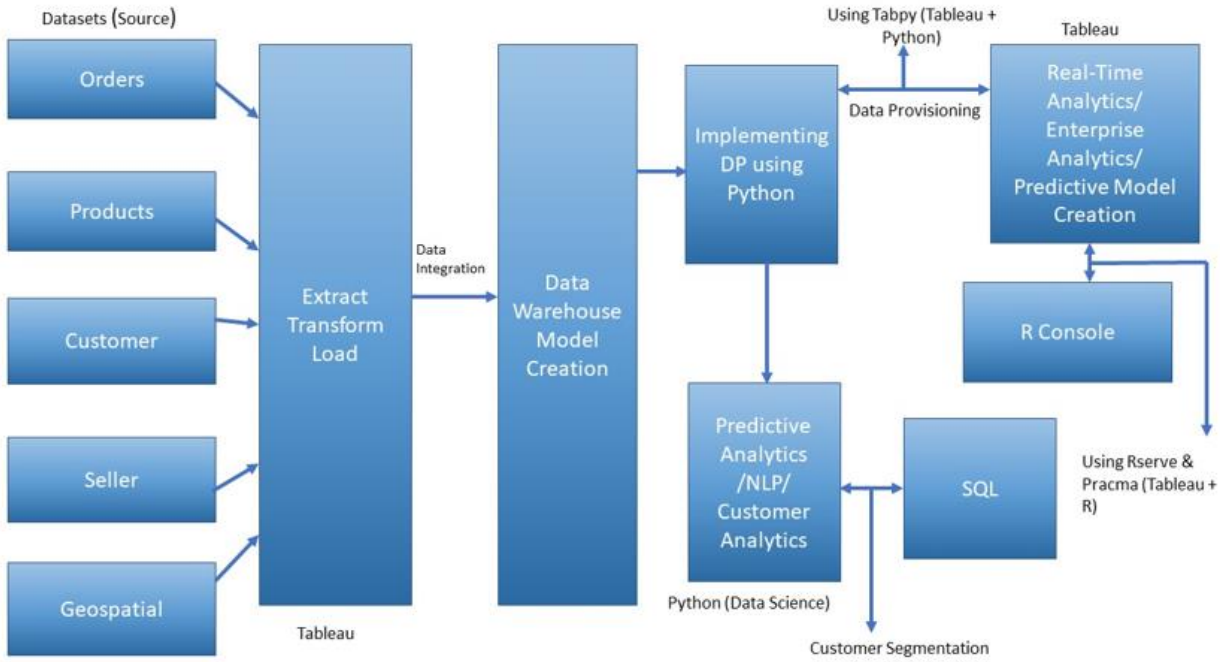


Figure 27: Architecture of the proposed methodology

The chosen tools & technologies are compatible with the existing data mining framework based on SQL, Python and Tableau. The data from various sources are integrated using Extract, Transform, and Load which later creates a Data Warehouse Model. The enterprise warehouse data is being anonymized using differential privacy with the help of Python. The data can be directly used for data provisioning by implementing Real-Time Analytics, or Enterprise Analytics using Tableau Desktop. The other way around could be taking the same data for Predictive Analytics or creation of Machine Learning Models and draw valuable insight from the data.

6. Analysis Methodology

A robust and well-proven methodology which provides a structured approach to plan a data mining project is CRISP-DM or cross-industry process for data mining. CRISP-DM is a flexible method, mainly when using analytics to solve bothersome business issues. The main reason to adopt this CRISP-DM method is that it supports projects with project management directions so one can easily follow the stages defined in CRISP-DM to ensure to deliver a quality outcome (Kumar 2019). Primarily based on an idealized sequence of events, the tasks of the project are carried out through the entire project lifecycle.

Figure 28: CRISP Datascience Methodology (Wikimedia Commons, 2012)

6.1. Business understanding

The first stage of CRISP-DM, which mainly focuses on understanding the objectives and requirements of the project in terms of business perspective and then helps in converting this knowledge into a data mining problem. The desired outputs of the project are determined in this stage using. The primary aim of this project is to display how differential privacy technique will help the organization trying to find the business values from the data they collect from their customers without violating their privacy policies.

6.2. Data understanding

The study focuses primarily on employing data anonymization to a publicly available e-commerce dataset using the IBM differential privacy library.

a) Dataset Description

- **Dataset :** Brazilian e-commerce retail dataset by Olist
- Available at https://www.kaggle.com/olistbr/brazilian-ecommerce
- **Number of transactions** : 100k orders relating to more than seventy product categories like health and beauty, electronics, food and drinks, and home appliances.

The selected dataset has information on orders made at multiple marketplaces in Brazil from 2016 to 2018. Further, the dataset comprises of data captured from multiple dimensions like order status, customer details, price, payment and freight performance, geospatial data like latitude and longitude coordinates, product attributes and finally customer reviews to products purchased. Since this is a real commercial data, sensitive user and product information has been anonymised. Figure 29 shows the relationship between the different entities of the dataset.

b) Entity Relationship Diagram

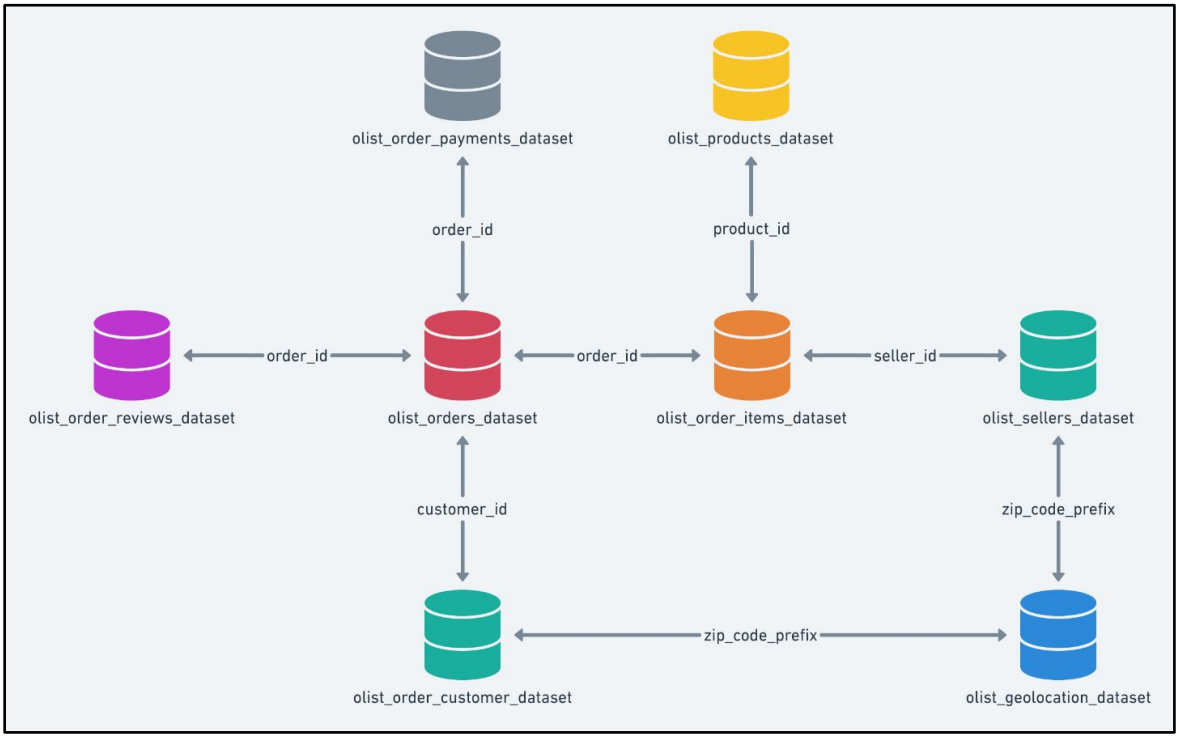


Figure 29: Entity Relationship diagram of the e-commerce dataset (Kaggle 2018)

c) Key Datapoints

1. Olist customer dataset.csv

Column name	Description
customer_id	Unique customer ID for each order placed in olist
customer_unique_identifier	Unique identification value for each customer
customer_zip_code	First 5 digits of customer zip codes
customer_city	Name of the city where customer lives
customer_state	Name of the state where customer lives

2. Olist_geolocation_dataset.csv

Column name	Description
geolocation_zip_code	First 5 digits of zip codes
geolocation_lat	Latitude
geolocation_lng	Longitude
geolocation_city	City name
geolocation_state	Name of the state

3. Olist_order_items_dataset.csv

Column name	Description
order_id	Unique order ID
order_item_id	Sequential number identifying number of items included in the same order.
product_id	Product unique identifier
seller_id	Seller unique identifier
shipping_limit_date	Shows the seller shipping limit date for handling the order over to the logistic partner.
price	Price of the product
freight_value	Item freight value item (if an order has more than one item the freight value is splitted between items)

4. Olist_order_payments_dataset.csv

Column name	Description
order_id	Unique order ID
payment_type	Payment method chosen by the customer.
payment_installments	Number of installments
payment_value	Transaction value.

5. Olist_order_reviews_dataset.csv

Column name	Description
review_id	Unique review identifier
order_id	Unique order ID
review_score	Satisfaction survey from the scale of 1 to 5
review_comment_title	Comment title from the review left by the customer, in Portuguese.
review_comment_title	Comment message from the review left by the customer, in Portuguese.
review_creation_date	Shows the date in which the satisfaction survey was sent to the customer.
review_answer_time	Shows satisfaction survey answer timestamp.

6. Olist_orders_dataset.csv

Column name	Description
order_id	Unique order ID
customer_id	Unique customer ID for each order placed
order_status	Reference to the order status (delivered, shipped, etc).
order_purchase_time	Shows the purchase timestamp.
order_approved_at	Shows the payment approval timestamp.
order_delivered_carrier_date	Shows the order posting timestamp. When it was handled to the logistic partner.
order_delivered_customer_date	Shows the actual order delivery date to the customer.
order_estimated_delivery_date	Shows the estimated delivery date that was informed to customer at the purchase moment.

7. Olist_sellers_dataset.csv

Column name	Description
seller_id	Seller unique identifier
seller_zip_code_p	First 5 digits of seller zip code
seller_city	Seller city name
seller_state	Seller state

8. Olist_products_dataset.csv

Column name	Description
product_id	Product unique identifier
product_category_name	Root category of product name in Portuguese.
product_name_length	Number of characters extracted from the product name.
product_description	Number of characters extracted from the product description.
product_photos_qty	Number of product published photos
product_weight_g	Product weight measured in grams.
product_length_cm	Product length measured in centimeters.
product_height_cm	Product height measured in centimeters.
product_width_cm	Product width measured in centimeters.

9. product_category_name_translation.csv

Column name	Description
product_category_name	Category name in Portuguese
product_category_name_english	Category name in English

c) Implementing Data Schema in Tableau Desktop

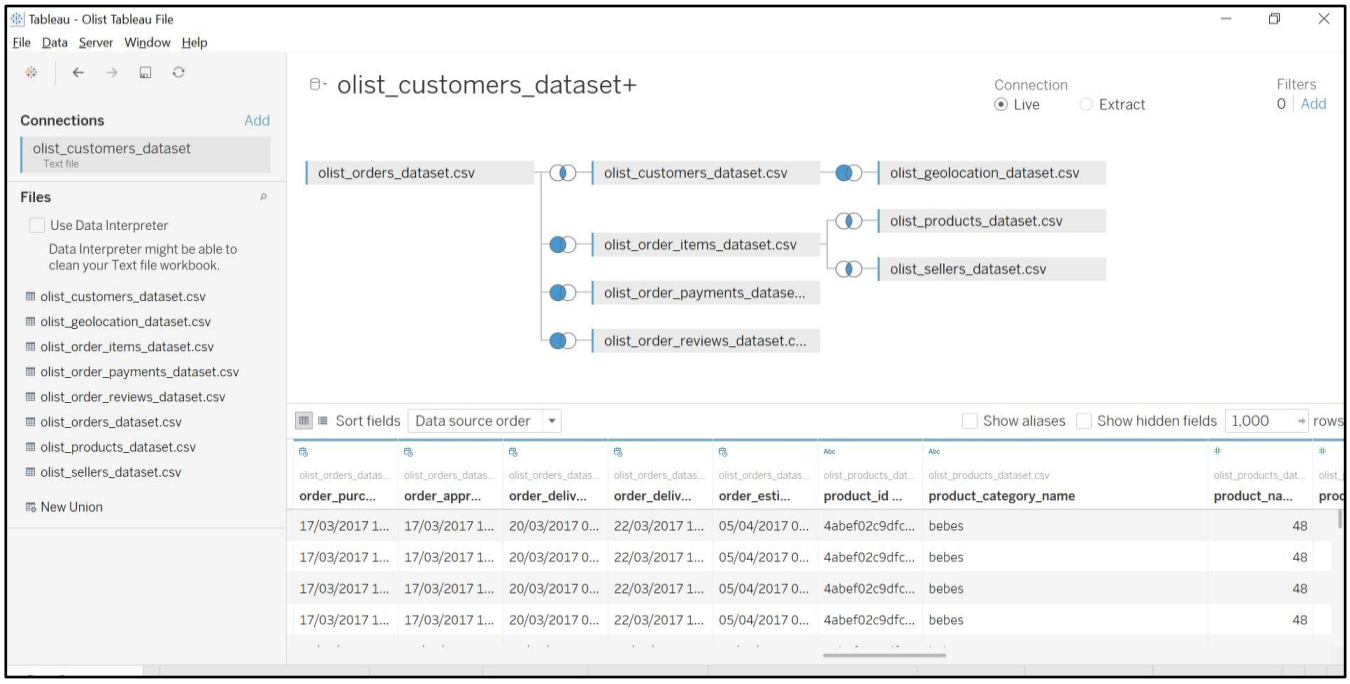


Figure 30: Snowflakes Schema

Explanation:

In order to create a logical model to address the unique needs of huge datasets for analytics purpose (OLAP), we prefer to finalize a data model. Looking at the dataset and the referential integrity forming from it, we have decided to go with Snowflakes Schema. In this case, Order Dataset is considered as the fact table which has the primary information of all other dimension tables i.e. Customer, Items, Payment, and Reviews. The dimension tables primary keys are present as the foreign key in the fact table and joined with it after decoding the relationship whether one-one or one-many or many-many. The type of the relationship decides the Join Method. The dimension tables are further normalized to Geolocation, Product, and Seller tables, which connects them further deciding upon the relationship. Finally, the model formed is a “Snowflakes Schema”.

6.3. Data preparation

As we have finalized the model as Snowflakes Schema, we have decided to transform the data and create the enterprise data warehouse accordingly. In the process of data preparation, we have considered all the possible sources that can be used for analysis. Next, we have extracted those sources, and transformed them with proper look up among Fact Table and Dimension Tables. Besides this, as it is a Snowflakes Schema, we therefore must decide the relation between the dimension tables to normalize it. Finally, the transformed model is loaded and used as single source of truth for analysis.

6.4. Modelling

In this phase, we apply differentially private machine learnings models and statistical functions to the data to explore the capabilities of predictive analytics, Customer analytics and Natural language processing. A detailed explanation is included under the section Employing Differential Privacy with Python

6.5. Evaluation

Each implemented machine learning model is evaluated for its accuracy. Further, other performance metrics like confusion matrix, F1 Score, RMSE value, Log-loss score and Jaccard Similarity score is also computed. In addition to that, a comparison of non-private and differentially private model is given under each machine learning model, also, we illustrated how selecting proper values of privacy budget impacts the overall accuracy of the model. A detailed explanation is included under the section Employing Differential Privacy with Python**.**

7. Differential Privacy with IBM Differential Privacy Library

a) Setup

- **Installation with pip**

  The library runs with Python 3, and the installation is simple and straight-forward. One of the ways to install Diffprivlib is through a pip command.

```
pip install diffprivlib
```

- **Manual Installation**

  Alternatively, the library can be installed manually by cloning the git repository.

```
git clone https://github.com/IBM/differential-privacy-library
```

  Then, from the project folder, execute the following.

```
pip install .
```

b) Components

The library comprises of three main components.

- **Mechanisms**: Diffprivlib provides an extensive collection of mechanisms which acts as the fundamental building blocks of differential privacy as they are responsible for handling the addition of noise. Further, Mechanisms like Laplace and Gaussian can be used to perform computations that are not covered by the pre-built algorithms.
- **Tools**: Diffprivlib has a collection of tools and utilities for simple data analytics and statistical operations. Also, has definitions for aggregations with differential privacy like mean, variance and standard deviations.
- **Models**: Diffprivlib offers a variety of differentially private machine learning models (Holohan et al. 2019)

7.1. Mechanisms

Given below is the list of Mechanisms offered by the Diffprivlib library. Out of which, in this study, we will be focusing on the **Laplace mechanism** for employing differential privacy.

Mechanism name
DPMechanism
TruncationAndFoldingMixin
Binary
Exponential
ExponentialHierarchical
Gaussian
GaussianAnalytic
Geometric
GeometricTruncated
GeometricFolded
Laplace
LaplaceTruncated
LaplaceFolded
LaplaceBoundedDomain
LaplaceBoundedNoise
Staircase
Uniform
Vector

Figure 31: List of Diffprivlib Mechanisms (Holohan et al. 2019)

Holohan et al. (2019) claims that the mechanisms, as mentioned above, have been built primarily for their inclusion in more complicated applications and machine learning models; however, they can also be executed separately. Further, these mechanisms have specific functions that reduce the need for code. For instance, the **set_epsilon()** and **set_epsilon_delta()** methods enable assigning the ϵ and δ parameters to the model. In addition to that, every mechanism has **set_sensitivity()** and **randomize()** methods to take in non-private input values and return a differentially private output. Given below is a sample implementation of the Laplace mechanism.

```
>>> from diffprivlib.mechanisms import Laplace

>>> laplace = Laplace()
>>> laplace.set_epsilon(0.5)
>>> laplace.set_sensitivity(1)
>>> laplace.randomise(3)
5.835104866820303
```

Figure 32: Laplace Mechanism for Adding Noise (Holohan et al. 2019)

7.2. Diffprivlib Tools and Utilities

Diffprivlib contains several essential tools and utilities to employ differential privacy. Currently, the module offers histogram functions to compute differentially private histograms, and **statistical functions** to calculate the differentially private variance, mean, and standard deviation.

Tools	Notes
`histogram`, `histogram2d`, `histogramdd`	Histogram functions mirroring and leveraging the functionality of their NumPy counterparts, with differential privacy (using the `GeometricTruncated` mechanism)
`mean`, `var`, `std`	Simple statistical functions mirroring and leveraging their Numpy counterparts, with differential privacy (using the `Laplace` and `LaplaceBoundedDomain` mechanisms)

Figure 33: List of Diffprivlib tools (Holohan et al. 2019)

All these tools take in an array of values as input. Therefore, they leverage the functionalities of the corresponding functions in NumPy, to ensure ease of use. In addition to the statistical functions mentioned before, we added definitions for sum and count using the Laplace mechanism. A Jupyter notebook containing the implementations of various Diffprivlib tools is included here with Diffprivlib Tools & Utilities.

7.3. Machine Learning Models

The machine learning models of diffprivlib is designed in a way to mirror the syntax and behaviour of the Scikit-learn models, which, facilitates an effortless transition from Scikit-learn machine learning model to diffprivlib. Following are a few machine learning models offered by the library.

Model	Notes
Supervised learning	
`LogisticRegression`	Implements the logistic regression classifier in [3] (using the `Vector` mechanism), with minor changes to allow for non-unity data norm and to allow integration with the corresponding classifier in SKLearn.
`GaussianNB`	Implementation of the Gaussian naïve Bayes classifier presented in [16] (using the `Laplace` and `LaplaceBoundedDomain` mechanisms) with minor amendments[4]
Unsupervised learning	
`KMeans`	Implementation of the *k*-means algorithm presented in [15] (using the `GeometricFolded` and `LaplaceBoundedDomain` mechanisms)

Figure 34: List of Diffprivlib Machine learning models (Holohan et al. 2019)

For instance, implementing a Gaussian naive Bayes classifier with differential privacy is as simple as replacing the Scikit-learn import statement with the corresponding import from the diffprivlib and run the analysis in the same way.

Default Parameters:

Diffprivlib encourages optimizing the model's sensitivity and noise-addition through appropriately defining the model parameters. The ϵ value is specified as a parameter to the model during initialization (e.g., GaussianNB(epsilon=1.0)); otherwise, a default of $\epsilon = 1$ is applied. In addition to that, parameters like range and maximum data norm are computed during model fitting if not explicitly defined. However, it is a best practise to define the model's parameters to ensure no privacy leakage and optimal noise-addition

Privacy Warnings:

If the parameters are not specified explicitly before training the model or if misconfigured, a warned relating to a possible privacy leak is issued. However, the model proceeds to execute, with the parameters computed during model training. Some of the commonly occurring privacy warnings are 'PrivacyLeakWarning' and 'DiffprivlibCompatibilityWarning'. Below is an instance that produces a privacy warning.

```
from diffprivlib.models import GaussianNB
nb=GaussianNB()
X=np.array([[-1,-1],[-2,-1],[-3,-2],[1,1],[2,1],[3,2]])
y=np.array([1,1,1,2,2,2])
nb.fit(X,y)
nb.predict([[-0.5,-2],[3,2]])
```

```
C:\Anaconda\lib\site-packages\diffprivlib\models\naive_bayes.py:93: PrivacyLeakWarning: Bounds hav
e not been specified and will be calculated on the data provided. This will result in additional p
rivacy leakage. To ensure differential privacy and no additional privacy leakage, specify bounds f
or each dimension.
  "privacy leakage, specify bounds for each dimension.", PrivacyLeakWarning)
```

Figure 35: PrivacyLeakWarning in a GaussianNB model

The warning suppresses by initializing the GaussianNB model with bounds([(-3, 3), (-2, 2)]), as the values in each column of X fall within those bounds.

```
from diffprivlib.models import GaussianNB
nb=GaussianNB(bounds=[(-3,3),(-2,2)])
X=np.array([[-1,-1],[-2,-1],[-3,-2],[1,1],[2,1],[3,2]])
y=np.array([1,1,1,2,2,2])
nb.fit(X,y)
nb.predict([[-0.5,-2],[3,2]])
```

```
array([2, 2])
```

Figure 36: GaussianNB model with bounds

8. Employing Differential Privacy with Python

The differential privacy library has many potential applications in real-time. From a simple linear regression to complicate NLP and customer segmentation, the library can be employed for various machine learning applications. In the following sections covers the use of differential privacy in Predictive analytics, Natural Language Processing (NLP), and Customer Analytics. Before we cover on to these machine learning models, let's first understand what the actual need for data anonymization in the retail industries is.

Why do we require Differential Privacy in Retail?

The modern retail environment is dynamic and unpredictable. With digitally empowered customers, there is a compelling need for companies to continually adapt to changing customer requirements and deliver customer-centric services without exposing valuable customer information. On the other hand, the customers are also keen on knowing how the companies are processing their data, and more are concerned with sharing their data in the first place. In order to cater to the privacy needs of the digitally empowered community, it is quintessential for the companies to adhere to data privacy laws and regulations such as the General Data Protection Regulation 2016/679. Savage (2018) identifies the top issues of GDPR on retailers as the follows.

- **Profiling**

Retailers profile customers based on the usage of loyalty cards, online behaviour, purchase or search history, even by tracking the instore movements of customers using CCTV footages and sensors. Generally, these data are collected in an automated method to analyse and predict the personal preferences of customers. GDPR forces penalties on retailers if a customer faces a 'legal effect' because of 'profiling'.

- **Security and data breach**

Unlike the previous data protection act, in GDPR, not only the retailers but also the data processors and suppliers have to comply with the security requirements and have direct liability on the data breach. In most cases, the data breach is internal and often results in severe consequences.

- **Cross border Data flows**

'Data sharing' is inevitable in retail industries. However, the retailers need to be GDPR compliant in sharing customer/employee data, whether internally within the company or with third parties like suppliers. Further, retailers operating cross-border sales should be complying with the additional rules on international data.

Non-compliance of GDPR may result in penalties up to 4% of global turnover or Euros 20 Million whichever is greater. Because of these severe penalties, there is a compelling demand for retailers to be data protection compliance. Besides, it is critical that the retailers should also maintain customer relationship while adhering to the regulations. One of the quick solutions that enable both is by incorporating data privacy or anonymisation techniques.

One of the primary obligations of the data controllers is 'Data protection by design and by default' which also states that the controllers must integrate organisational or technical measures of data anonymisation in the processing of personal data to be compliant with the Regulation (2016, p.48, Article 25). In the following sections we covered different diffpriv models alongside their possible real-time applications.

8.1. Logistic Regression with Differential Privacy

a) Feature Selection

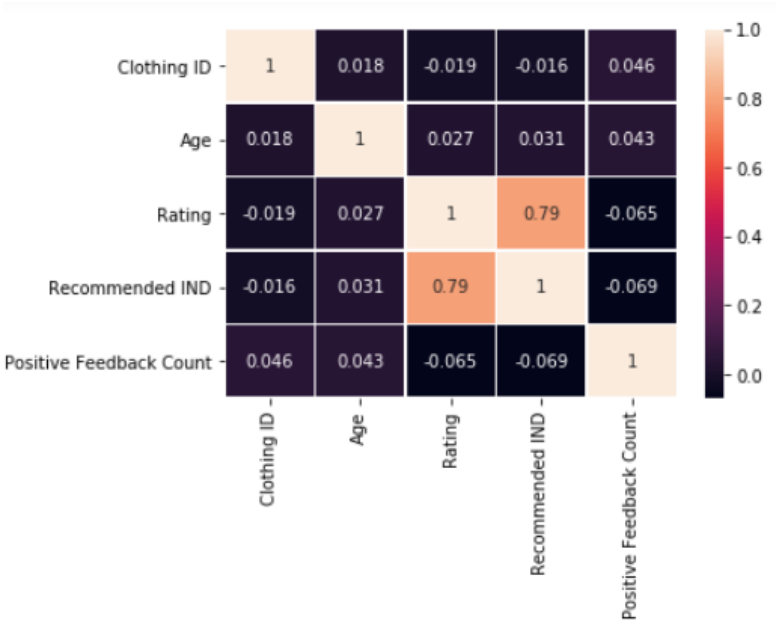


Figure 37: Correlation matrix for Feature selection

- **Rating**: Positive integer representing the product score granted by the customer. (1=Worst, 5= Best)
- **Recommended IND**: Binary variable stating where the customer recommends the product. (1 = Recommended, 0 = Not recommended).

b) Implementation

```
X = np.asarray(df[['Rating']])
y = np.asarray(df[['Recommended IND']])

from sklearn import preprocessing
X = preprocessing.StandardScaler().fit(X).transform(X)


from sklearn.model_selection import train_test_split
DP_X_train, DP_X_test, DP_y_train, DP_y_test = train_test_split( X, y, test_size=0.2, random_state=4)

from diffprivlib.models.logistic_regression import LogisticRegression
DP_LR = LogisticRegression(C=0.01, solver='liblinear',epsilon=1).fit(DP_X_train,DP_y_train)
DP_LR

DP_yhat = DP_LR.predict(DP_X_test)
DP_yhat_prob = DP_LR.predict_proba(DP_X_test)
```

Figure 38: Logistic Regression Code

c) Results

```
LogisticRegression(C=0.01, data_norm=2.8792894329397156, epsilon=1,
                   fit_intercept=True, max_iter=100, n_jobs=None, tol=0.0001,
                   verbose=0, warm_start=False)
```

Figure 39: Diffprivlib Logistic regression model

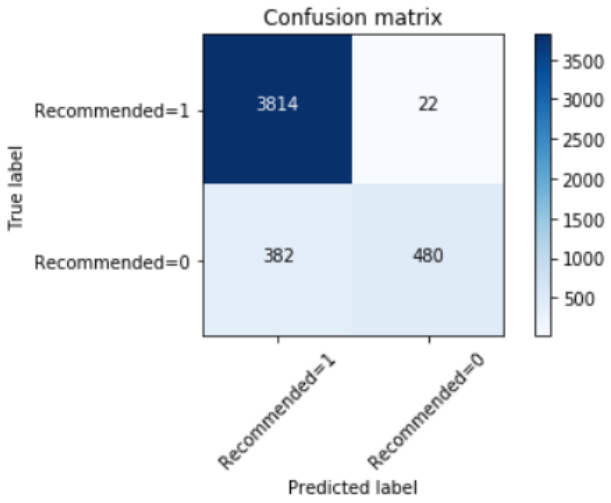


Figure 40: Confusion Matrix

	Algorithm	Accuracy Score	Jaccard Similarity Score	F1-score	Log_Loss
0	Logistic Regression	0.942316	0.942316	0.944645	0.149325
1	Logistic Regression with DP	0.914006	0.914006	0.904585	0.142780

Figure 41: Comparing the results of non-private and private regression models

d) Explanation

Given in Figure 38 is the python code for implementing logistic regression with diffprivlib library. In this model, we performed a regression between Rating and Recommendation IND. As a first step, we split the data into training and test sets. We then created a regression model using diffprivlib's built-in Logistic Regression model and trained the model using the training sets. Finally, we predicted values with the trained model using the test set. In order to access the model performance, we compute evaluation metrics such as confusion matrix, accuracy score, f1 score, log loss and Jaccard score.

Further, during the model creation, we did not explicitly specify any parameters except epsilon. However, from Figure 39, it is quite evident that the model itself calculated the remaining unspecified parameters during model fitting.

In addition to that, we compared the performance of the differentially private regression model with the non-private model (Figure 41) and found that both the model have almost **similar metrics** concerning performance. Therefore, differentially private regression models can be employed for real-time applications without being concerned about the performance. A python notebook with the before-mentioned code blocks is included herewith Logistic Regression

8.2. Linear Regression with Differential Privacy

a) Feature Selection

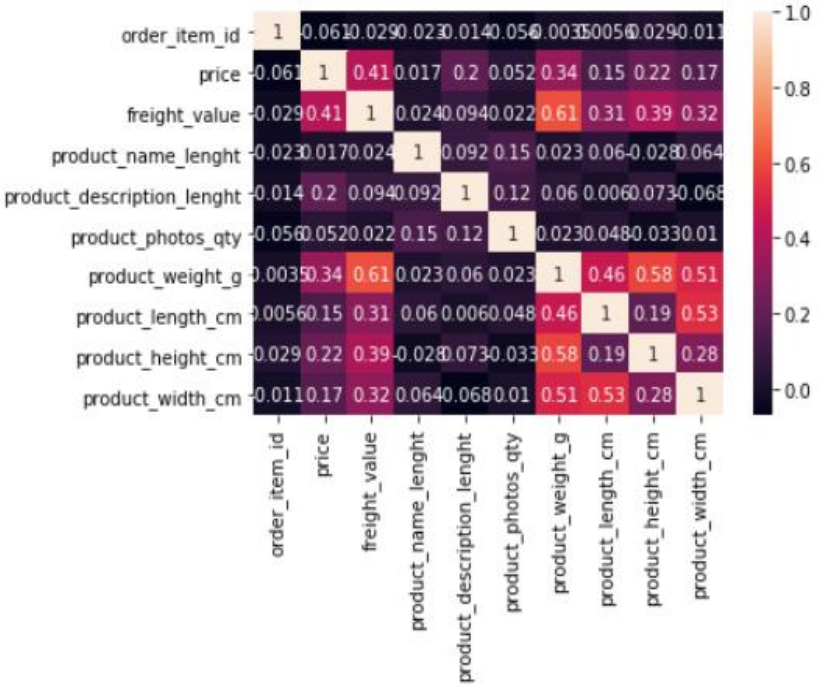


Figure 42: Correlation Matrix for Feature Selection

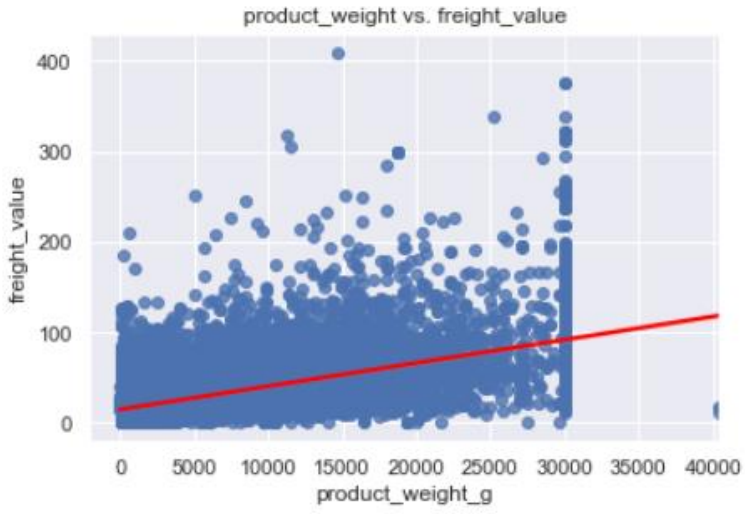


Figure 43: Scatterplot between freight_value and product_weight_g

b) Implementation

```
X = df['product_weight_g'].values.reshape(-1,1)
Y = df['freight_value'].values.reshape(-1,1)

from diffprivlib.models.linear_regression import LinearRegression
reg=LinearRegression(epsilon=1)
reg.fit(X, Y)

print('Differentially Private Linear Equation:')
print('freight_value='+str(reg.intercept_)+' + '+str(reg.coef_)+' * product_weight')
print('')

new_product_weight=np.array([30000.0]).reshape(-1,1)
print('Prediction:')
print('new_product_weight:'+str(new_product_weight))
print('Predicted freight_value:',reg.predict(new_product_weight))
```

Figure 44: Python code for Linear Regression

c) Results

```
LinearRegression(copy_X=True, data_norm=40425.0, epsilon=1, fit_intercept=True,
                 range_X=array([40425.]), range_y=array([409.68]))
```

Figure 45: Diffprivlib Linear Regression Model

```
Linear Equation:
freight_value=[14.6096456] + [[0.0025791]] * product_weight

Prediction:
new_product_weight:[[30000.]]
Predicted freight_value: [[91.98255676]]
```

Figure 46: Prediction using a non-private regression model

```
Differentially Private Linear Equation:
freight_value=[14.28410477] + [[0.00273544]] * product_weight

Prediction:
new_product_weight:[[30000.]]
Predicted freight_value: [[96.34734475]]
```

Figure 47: Prediction with diffprivlib regression model

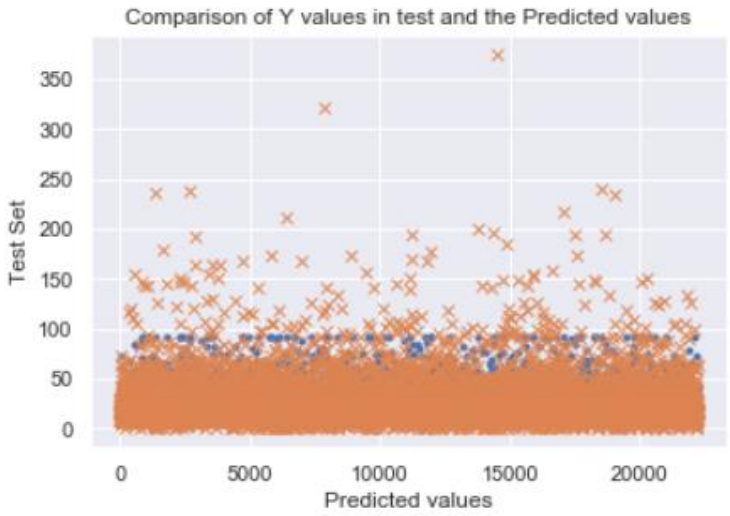


Figure 48: Model fitting - Actual and Predicted values (x - Actual, . – Predicted)

d) Explanation

Given in Figure 44, is the python code for implementing linear regression with diffprivlib library. In this model, we performed a regression between **freight value** and **product weight**. From the visual interpretation of the scatter plot, it is evident that both the predictor and target variables are positively correlated; in simple terms, the freight is calculated according to its weight.

As a first step, we split the data into training and test sets. We then created a regression model using diffprivlib's built-in Linear Regression model and trained the model using the training sets. Finally, we predicted values with the trained model using the test set. Further, from Figure 45, it is quite evident that the model itself calculated the remaining unspecified parameters during model fitting.

In addition to that, we compared the predicted values of the differentially private regression model with the non-private model (Figure 46 and Figure 47) and found that both the model has almost similar metrics concerning to performance. Therefore, differentially private linear regression models can be employed for real-time applications without being concerned about the performance. A python code file is included Linear Regression.

8.3. Customer Segmentation with Kmeans clustering

As mentioned earlier, profiling customers is a common practice in the retail sector as it serves as a foundation for analyzing customer behaviour or in predicting products that best interests the users. One of the conventional techniques of profiling is segmenting customers into similar groups based on a variety of factors. Further, customer segmentation allows retailers to promote a marketing strategy or provide customized offers to customers of similar interests.

a) Feature Selection

In this example, we are clustering customer segments based on factors like **Recency**, **Frequency** and **Monetary** values, which are computed using a simple query in MS SQL Server.

Exposure of accurate information on the before mentioned factors or information on the cluster may reveal critical information like how often a customer visits the store, the average sales value imposed by the customer or even the next possible visit. Therefore, it is crucial to provide an approximation rather than accurate information while profiling. In this case, we tried to anonymise the clustering information using the diffprivlib's **KMeans()** algorithm.

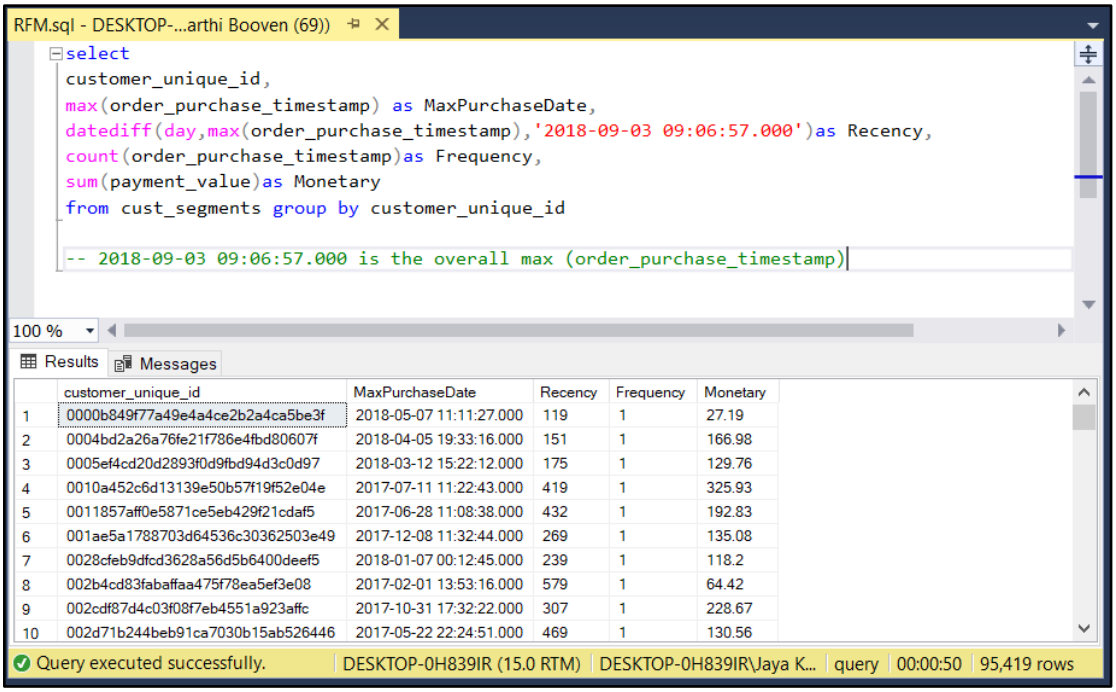


Figure 49: RFM Calculation using MS SQL Server

b) Implementation

Recency Cluster

Recency score represents the number of days a customer is inactive since the most recent purchase (Rouse 2005). In this example, we apply Kmeans clustering to segment customers based on the recency score.

- **Elbow Method**

Kmeans clustering is an unsupervised machine learning algorithm that sorts observations into similar clusters. One of the critical factors that influence the quality of a cluster solution is the number of groups or clusters (k). Therefore, it is essential to select the optimal value of k for clustering. One of the simplified methods for determining this optimal number of clusters is through an Elbow Method.

The Elbow Method plots variations of the model for different values of k and the 'elbow' reflects the cut-off point where adding another cluster does not improve the modelling of the data (Wikipedia 2020). Therefore, we pick the value at the elbow as the optimal value of k. In our case, performing Elbow Method on the Recency data returned a k=5 (Figure 51).

```
from sklearn.cluster import KMeans
df_recency = df[['Recency']]
sse={}
for k in range(1, 10):
    kmeans = KMeans(n_clusters=k, max_iter=1000).fit(df_recency)
    df_recency["clusters"] = kmeans.labels_
    sse[k] = kmeans.inertia_

plt.figure(figsize=(10, 5))
plt.plot(list(sse.keys()), list(sse.values()))
plt.xlabel("Number of Clusters")
plt.title("Elbow Method - Recency")
plt.show()
```

Figure 50: Code for Elbow Method – Recency

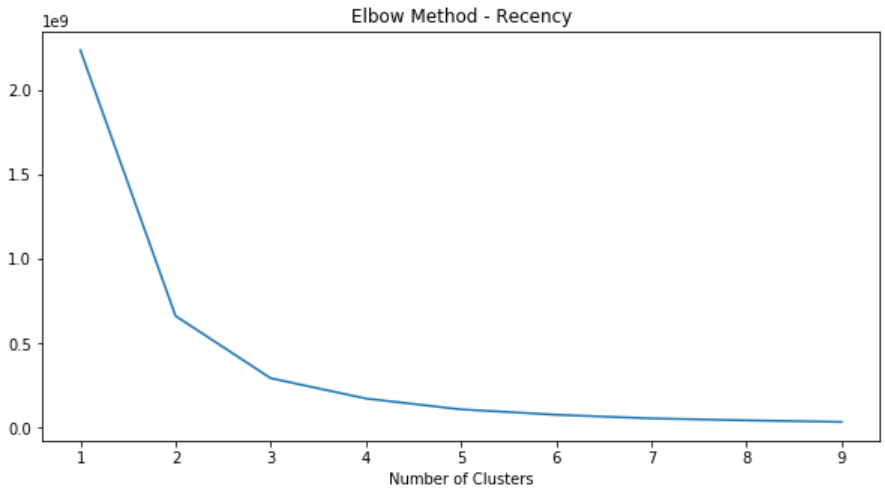


Figure 51: Output of Elbow Method (Optimal K=5)

- **Kmeans Clustering**

```
from diffprivlib.models.k_means import KMeans
kmeans = KMeans(n_clusters=5,epsilon=1,bounds=None)
kmeans.fit(df[['Recency']])
df['RecencyCluster'] = kmeans.predict(df[['Recency']])
df = order_cluster('RecencyCluster', 'Recency',df,False)
df.groupby('RecencyCluster')['Recency'].describe()
```

Figure 52: Code of Kmean clustering based on Recency

Implementing Kmeans clustering is as simple as just replacing the Scklearn import statement with diffprivlib import statement. After performing kmeans, we assign each cluster a cluster number (RecencyCluster). By comparing the mean values in Figure 53, the customers in Cluster 4 are very recent compared to Cluster 3 and 2. Therefore, Cluster 4 covers the most active customers, whereas cluster 0 covers the most inactive.

RecencyCluster	count	mean	std	min	25%	50%	75%	max
0	10550.0	524.867204	47.414371	463.0	486.0	519.0	552.0	729.0
1	15405.0	396.332944	35.971929	337.0	365.0	395.0	425.0	462.0
2	22631.0	274.381689	31.892820	221.0	244.0	277.0	297.0	336.0
3	25584.0	163.981473	32.090198	111.0	136.0	165.0	192.0	220.0
4	21249.0	55.837875	29.425348	0.0	30.0	52.0	81.0	110.0

Figure 53: Recency Cluster

Frequency Cluster

Frequency score represents the total number of orders a customer has placed over time. We then cluster the customers based on the Frequency score in the same way as the Recency (Rouse 2005). From the Elbow method, we identified that the optimal value of K for clustering is 5. In the Frequency cluster (Figure 55) **Cluster 4** represents the customers with higher frequency, where, higher frequency indicates potential target customers.

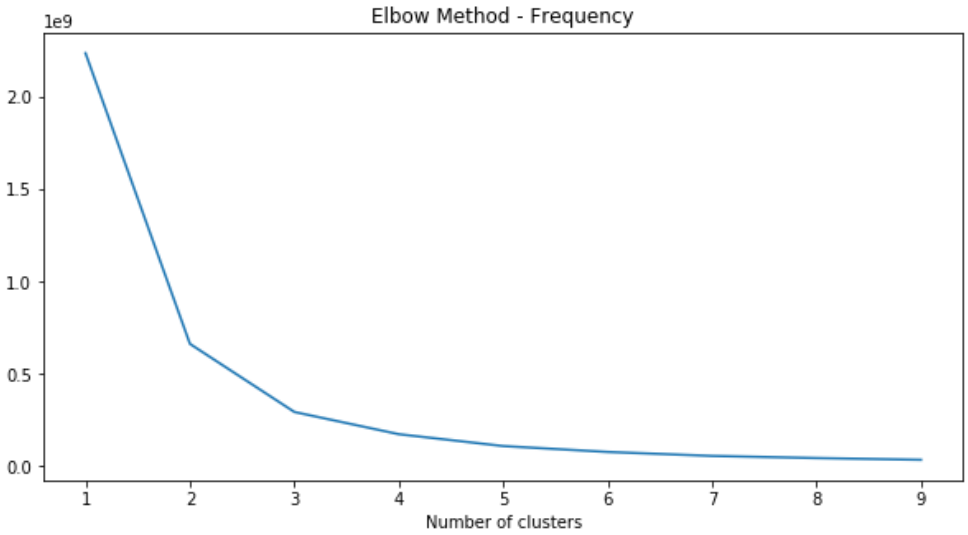


Figure 54: Elbow Method - Frequency (Optimal K=5)

FrequencyCluster	count	mean	std	min	25%	50%	75%	max
0	81088.0	1.000000	0.000000	1.0	1.0	1.0	1.0	1.0
1	12593.0	2.154133	0.361091	2.0	2.0	2.0	2.0	3.0
2	1223.0	4.255928	0.436560	4.0	4.0	4.0	5.0	5.0
3	512.0	7.876953	3.417249	6.0	6.0	6.0	8.0	29.0
4	3.0	49.333333	22.278540	35.0	36.5	38.0	56.5	75.0

Figure 55: Frequency Cluster

Monetary Cluster

Monetary or Revenue score represents the monetary sum obtained from the number of orders a customer has placed over time. We then cluster the customers based on the Revenue score in the same way as the Recency and Frequency (Rouse 2005). However, in this case, the Elbow method yielded **K=6**. In the Revenue cluster (Figure 57), **Cluster 4** represents the customers with a higher monetary score, where, the higher monetary value indicates potential target customers.

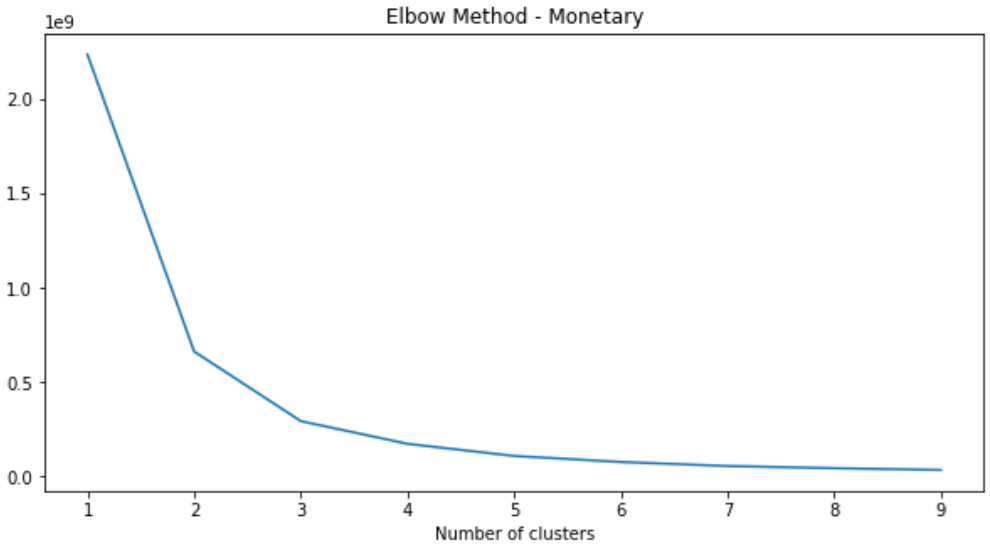


Figure 56: Elbow Method – Monetary (Optimal K=6)

RevenueCluster	count	mean	std	min	25%	50%
0	90524.0	144.056416	118.978710	9.59	62.010	106.780
1	3010.0	838.245023	135.600675	652.86	721.305	814.410
2	1880.0	2382.533165	2241.264354	1148.00	1351.510	1700.545
3	4.0	38994.810000	7038.234723	30186.00	34913.430	40268.620
4	1.0	109312.640000	NaN	109312.64	109312.640	109312.640

Figure 57: Revenue Cluster

Scoring Clusters:

As a next step, we compute OverallScore for clusters based on the mean values of the RecencyCluster, FrequencyCluster and RevenueCluster (Figure 58).

OverallScore	Recency	Frequency	Monetary
0	524.594852	1.000000	126.541658
1	408.912714	1.091186	143.955227
2	289.133736	1.111946	157.202039
3	182.548961	1.156991	169.560398
4	81.525884	1.221951	193.992631
5	101.805080	2.328824	578.894495
6	118.049841	3.080594	1325.680223
7	120.946341	4.685366	2392.553244
8	113.523810	6.395238	3734.427095
9	66.476923	7.738462	5719.751846

Figure 58: Overall Score (1=Low-value , 9= High-value customers)

The scoring mentioned above clearly shows us that customers with score 9 are the best customers, whereas 0 are the worst. To simplify, better we name these scores: 0 to 3 as **Low-value**, 4 to 6 as **Mid-value**, and 6+ as **High-value**.

```
df['Segment'] = 'Low-Value'
df.loc[df['OverallScore']>3,'Segment'] = 'Mid-Value'
df.loc[df['OverallScore']>6,'Segment'] = 'High-Value'
df.head()
```

Figure 59: Assigning Segments based on the Overall Score

	customer_unique_id	Recency	Frequency	Monetary	RecencyCluster	FrequencyCluster	RevenueCluster	OverallScore	Segment
0	0a0a92112bd4c708ca5fde585afaa872	339	8	109312.64	1	3	4	8	High-Value
1	20a5257c01689ac69410a14cb51bb447	356	10	17671.00	1	3	2	6	Mid-Value
2	93bc212addb25a5f5139fded3c2ee6b3	333	10	18667.00	2	3	2	7	High-Value
3	6d394722d5fc5e721aee6875a218d8db	303	8	16313.60	2	3	2	7	High-Value
4	d97b3cfb22b0d6b25ac9ed4e9c2d481b	321	13	9773.81	2	3	2	7	High-Value

Figure 60: Structure of final clusters

Plotting Customer Segments:

The customer segments are plotted based on the previously computed scores for different Recency, Frequency and Revenue buckets.

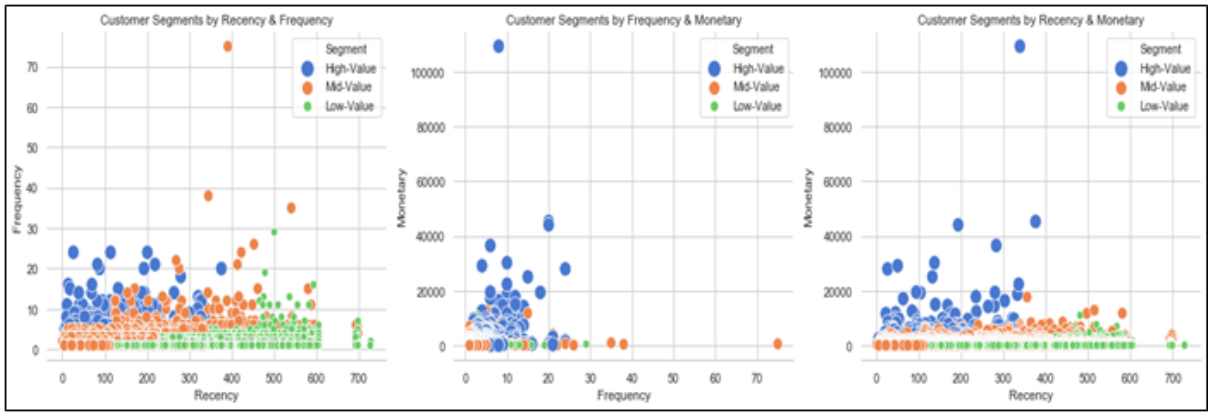


Figure 61: Customer Segmentation Output

The above example proves that diffprivlib can be effectively applied for customer analytics in the same way as other machine learning libraries. Further, a Jupyter notebook with the before-mention code blocks is included Kmeans Clustering

8.4. Natural Language Processing – Sentiment Analysis

Customer Experience is the critical factor that enables businesses to experience a competitive advantage in the market. From using chatbots for customer services to performing data analysis on customer questions, comments or product reviews, the applications of NLP have become one of the prominent techniques for enhancing the customer experience. In this section, we tried implementing sentiment analysis on customer reviews using the GaussianNB() machine learning model offered by the IBM differential privacy library.

The customer reviews in the Brazilian e-commerce dataset are in the Portuguese language; therefore, for demonstration purpose, we performed sentiment analysis on a similar e-commerce dataset with customer reviews in English. Given below is the dataset information.

- **Dataset name:** Women's clothing e-commerce reviews
- Available @ https://www.kaggle.com/nicapotato/womens-ecommerce-clothing-reviews

Column name	Description
Id	Unique ids
clothing_id	Unique ID of the product
age	Age of the reviewer
title	Title of the Review
review_text	Review
rating	Product Rating by reviewer
positive_feedback_count	Number of positive feedbacks on the review
division_name	Name of the division product is in
department_name	Name of the department product is in.

Figure 62: Key data points of the e-commerce reviews dataset

Initially, the concept of sensitive information in the retail sector is typically a customer's credit card details. However, after the amendment of the GDPR, companies have become considerate in processing or sharing any forms of customer data. For instance, sharing the data such as comments or customer reviews without anonymization have risks of exposing valuable information about a customer's purchase history, web history, topmost purchased products, even the exact number of purchases relating to a specific product. Further, the details mentioned earlier might help locate a particular customer; therefore, it is critical to consider these data as sensitive as well. In this section, we tried incorporating data anonymization into the sentiment analysis of customer reviews.

Gaussian Naïve Bayes Model

Naïve Bayes is an algorithm for binary or multi-class classification. In this case, we used the algorithm to perform text classification on customer reviews; also, the model is trained to predict the sentiment of new customer reviews. To be precise, we classify the product reviews into positive and negative based on keywords present in the text. Here we used the GaussianNB() offered by diffprivlib to anonymize the text classification. The below figure shows the python statement for importing the differentially private GaussianNB model.

```
from diffprivlib.models.naive_bayes import GaussianNB
from diffprivlib.utils import global_seed, PrivacyLeakWarning, DiffprivlibCompatibilityWarning
```

As part of data preprocessing, we ignored the NA values and introduced a new field to represent the sentiment. Further, we created a word cloud of the top product names and reviews based on the densities of occurrence (Figure 63).

Figure 63. All words in the review

In addition to that, we assigned the initial values for sentiments based on the following assumptions.

- Reviews with higher rating (>4) is rated as positive. (True in the data frame)
- Reviews with lower rating (2<) is rated as negative. (False in the data frame)

These pre-assigned values act as the actual value while model training and the model accuracy is computed by comparing these assigned values with the predicted values.

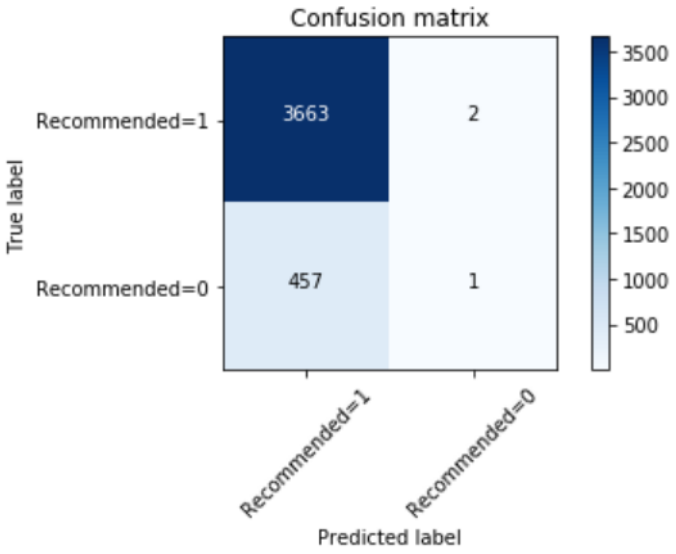
We then split that data into training and test sets and performed model fitting, followed by a prediction using the default epsilon value 1. Finally, as a part of the model evaluation, we computed the prediction accuracy using the confusion matrix (Figure 66).

```
DP_gb = GaussianNB()

DP_gb.fit(X_train.toarray(),y_train)

C:\Users\erayc\anaconda3\lib\site-packages\diffprivlib\models\naive_bayes.py:93: PrivacyLeakWarning: Bounds have not been specified and will be calculated
on the data provided. This will result in additional privacy leakage. To ensure differential privacy and no additional privacy leakage, specify bounds for
each dimension.
  "privacy leakage, specify bounds for each dimension.", PrivacyLeakWarning)

GaussianNB(bounds=[(0, 2), (0, 2), (0, 2), (0, 1), (0, 1), (0, 1), (0, 2),
                   (0, 1), (0, 1), (0, 1), (0, 3), (0, 2), (0, 1), (0, 1),
                   (0, 1), (0, 1), (0, 1), (0, 1), (0, 1), (0, 1), (0, 1),
                   (0, 1), (0, 1), (0, 1), (0, 1), (0, 1), (0, 1), (0, 1),
                   (0, 1), (0, 1), ...])
```

Figure 64. Gaussian NB implementation

```
DP_yhat = DP_gb.predict(X_test.toarray())
DP_yhat_prob=DP_gb.predict_proba(X_test.toarray())


cnf_matrix = confusion_matrix(y_test, DP_yhat, labels=[1,0])
np.set_printoptions(precision=2)
plt.figure()
plot_confusion_matrix(cnf_matrix, classes=['Recommended=1','Recommended=0'],normalize=False, title='Confusion matrix')
```

Figure 65. Gaussian NB Prediction



	Algorithm	Accuracy Score	F1-score
0	Gaussian Naive Bayes with DP	0.888673	0.836988

Figure 66: Gaussian NB Evaluation metrics

The differentially private Gaussian NB model showed a prediction accuracy of 88%. This high value of accuracy represents the model's capability of predicting most of the sentiments correctly. Moreover, the value of epsilon can be varied to offer low prediction accuracy and improved data anonymization. Apart from the before-mentioned models, the library also supports implementing **PCA**, **Standard Scalar** and **Logistic regression path** models.

8.5. Model Optimization

For the model to satisfy differential privacy and preserve privacy while predicting the training data, it is critical to **select appropriate epsilon (ε) value**. The following figure shows accuracy scores of a differentially private model for different values of ε over 30 iterations.

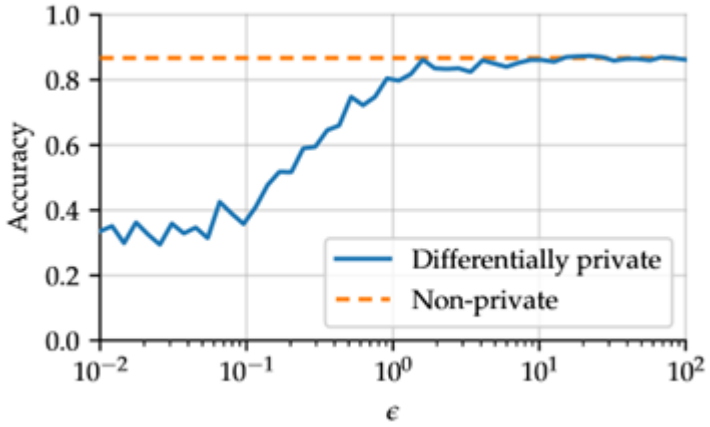


Figure 67: Comparing model accuracy for different values of epsilon

(Holohan et al. 2019)

Holohan et al. (2019) performed a similar analysis on two different datasets, one with 150 sample records and other with 48842 samples and proposed the following claims.

- The small dataset required a high threshold value of epsilon is needed to maintain an accuracy that is equivalent to that of a non-private model
- The larger dataset showed a model accuracy like 95% of the non-private accuracy at a comparatively low threshold of epsilon $\epsilon = 0.05$

9. Employing Exploratory analysis using Tableau

9.1. RFM Analysis for Customer

- **R- Recency** (How recently a customer has used our product / Finding the regular customer)
- **F- Frequency** (How many times a customer has used our product)
- **M- Monetary** (How much revenue generated by us from that customer)

Using Tableau, RFM Score is being calculated which span from 12 to 60. 12 is being the lowest score and 60 being the highest. Customer who have score less than 50 are the real concern for us. We need to give promotions/offers or engage with them so closely to bring up our business and their score.

We can further analyse the trend of the customer and their behaviour and try to drill down to understand that whether there is a shift of customer loyalty. For example: Customer suddenly disappeared from our system for 6 months, there may be many cases out of which probability that customer might be using another company SKUs are 72%. (Rouse 2005).

By getting the RFM score, we have segregated the customers into 3 categories

1. **Score above 50** – Valued customers for us in terms of monetary value generation, frequently using our product and becoming our regular customers. They have been using our Class A [10% of the products which gives 70% of the total revenue] and Vital [Very essential product of our business without which business is stagnant].

2. **Score between 30 to 50** – Second Priority customers but need to be focussed to convert them into excellent customers. They have been using our Class A and Class B [20% of the products which gives only 20% of the total revenue] products those are Vital and Essential [Products which are not much vital and an alternate of the same product can be taken if it is not present]

3. **Score below 30** – The much concerning customers who have not produced much revenue from us and are not regular with us. We will have to focus much on them to see that score increases for those customers. They have been using Class C [10% of the items which gives only 10% of the total revenue] and desirable items [Items which is seasonal for us and will not impact our business if they are not there]

We have segregated the three categories into colours to easily located them. Score above 50 in purple colour, 30-50 in green and below 30 in red colour ((Brial 2018) and (shawal 2018))

Background Technical Calculations:

Figure 68: Recency Calculation

Figure 69: Frequency Calculation

Figure 70: Monetary Calculation

Discussion:

Recency is found by taking the difference of date in days. The difference is between purchase order date (the date on which customer has placed an order by doing the proper payment) and today. Then, Rank Percentile is performed by giving minimum of date difference in ascending order.

- Rank percentile is a rank giving function where values are ranked between 0 to 1 where 0 - Poor and 1 – Best. Similarly, Frequency and Monetary is calculated with the help of rank percentile.
- Frequency – By calculating the count of number of records and putting the rank percentile in descending order.
- Monetary – By aggregating i.e. summing the payment generated and putting the rank percentile in descending order.

Figure 71: Recency Quintile Calculation

Discussion: When rank percentile is generated, then we try giving values from 0 to 0.2 as 1. 1 being the lowest as it fell into the lowest value range. Rank percentile values between 0.2 and 0.4 as 2, between 0.4 and 0.6 as 3, between 0.6 and 0.8 as 4 and above 0.8 as 5. 5 being the highest as it fell into the highest value range. Similar calculation is performed for frequency and monetary quintile.

Figure 72: RFM Calculation

Discussion: Finally, RFM score is calculated considering Monetary as highest priority as its quintile value is multiplied with 100, frequency as 2nd highest priority as its quintile value is multiplied with 1 and third is recency with least priority as its quintile value multiplied by 1.

Example: If a customer is a regular for us and buy only one-euro biscuit from our store, then it is not much of value to us as the revenue generation is not all great. As the customer buys a Class C and desirable product from us all the time. Therefore, Monetary can be considered as the highest priority.

If someone has generated highest revenue, then that would add much value to their RFM score. Revenue is the utmost important thing to any business. Better the RFM score, better is the customer for us. For example, If a customer is a regular customer for us and buy only one-euro biscuit from our store, then it is not much of value to us as the revenue generation is not much in number.

DP Calculation:

We have added noise to the RFM score so that the actual score of the customer remains hidden, instead the trend or customer classification remains the same and can be analysed.

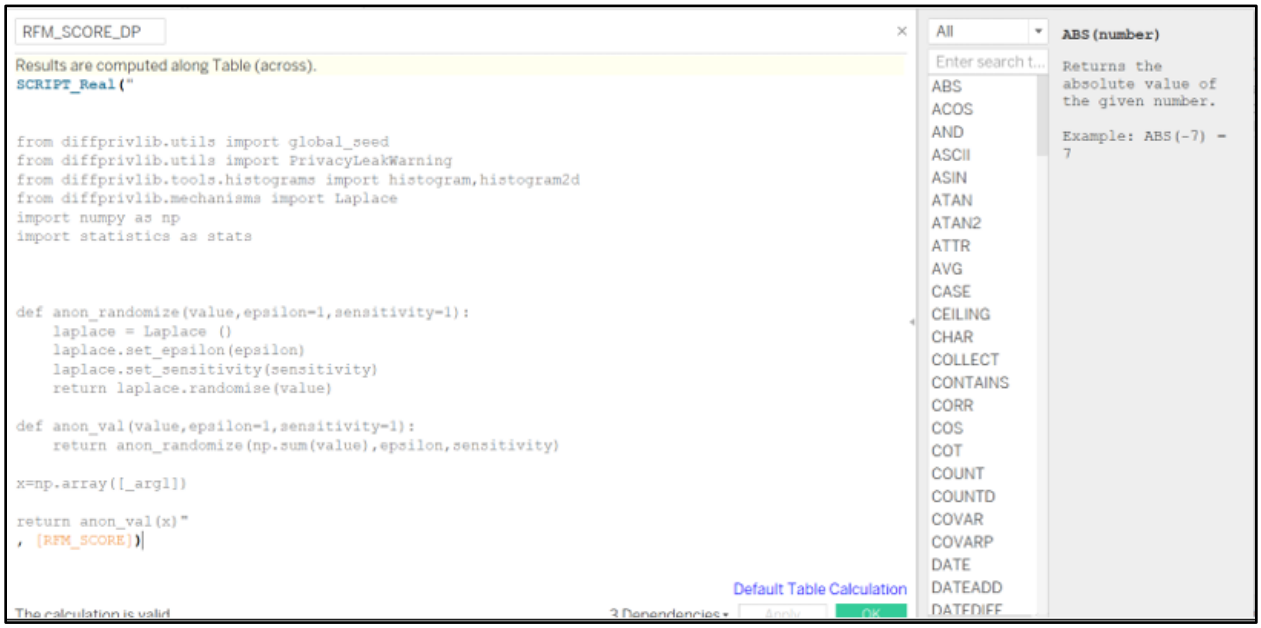


Figure 73: Differential Privacy Calculation

Explanation:

We have added noise or instil the concept of differential privacy using Python. As in the above case, once RFM score is being calculated, this calculation is somehow a private information of the customer depicting all the details of his purchases along with the sum of amount involved in the purchase. Somewhere, customer may feel that the private information of their purchase is being compromised. But, the retailer has to perform analysis without the customer's privacy being compromised. Here, the concept of Differential Privacy comes to picture. We tried adding noise to the RFM score so that the real value remains hidden and we can just able to differentiate which customer is better in RFM score. The comparision among the customer is carried but the exact value and number of Recency, frequency, and Monetary remains hidden by adding privacy budget. Therefore, the whole idea of differential privacy has been implemented for analysis.

Here, the Tableau and Python are integrated using TabPy. We have passed the data in form of argument where it is updated by some privacy budget (noise) and new value is being used in form of Tableau Calculation Field.

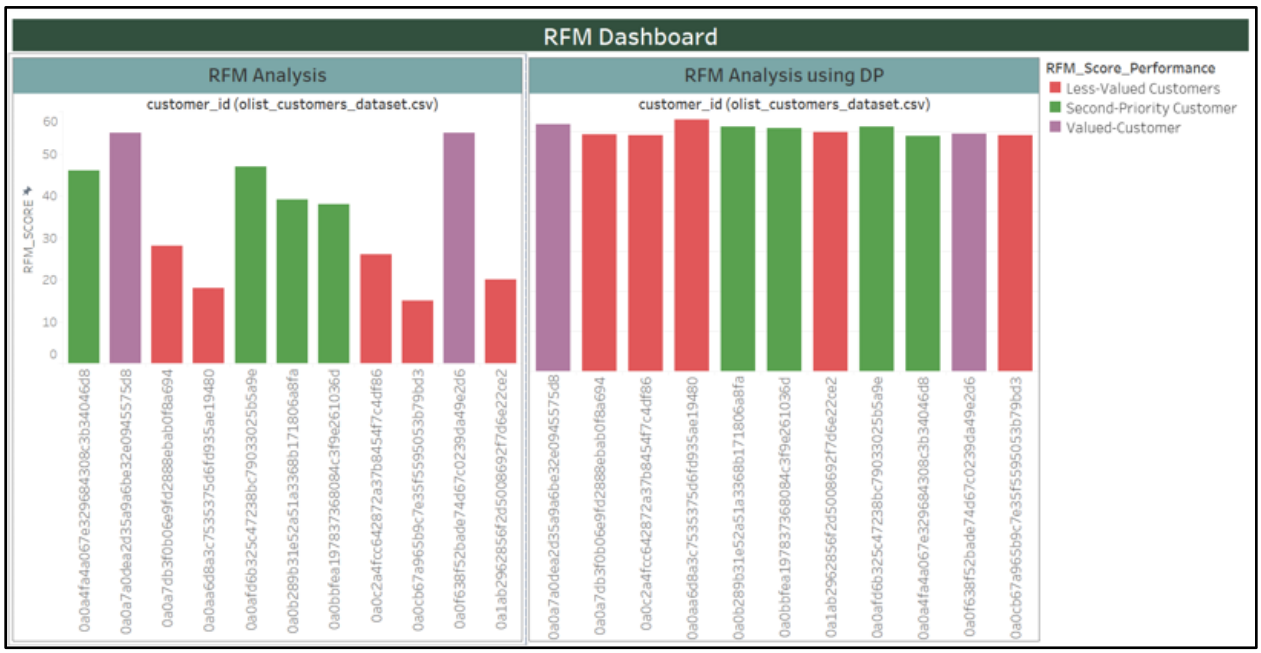


Figure 74: RFM Calculation using DP

Explanation:

From the dashboard, we can observe that the RFM value may not be the same with the normal calculation and DP as noise is being added but what is prudent is that the customer segregation as per the RFM value remains the same. From here, we can observe that the details of customer are not being compromised instead the purpose to perform analysis is done.

We have just used a limited number of customers with proper filter condition so that the real RFM value and RFM Value with noise can be easily identified in the dashboard. The target of identifying the customer and the bucket in which they fall is met although DP is being empowered.

9.2. Customer Churn Rate

Customer churn rate defines that the number of customers left using our product divided by the newly joined customers who have started using our product. Smaller the churn rate for a year, better is the customer relationship metrics.

It's a very healthy indication of the company – customer relationship as year on year the customer addition rate rises. But here in the dataset, the challenge is we can understand the new customers but old customer information is being hidden in the next year or given a new ID for the data privacy reason. (Stitch 2020)

Background Technical Calculations

We have used fixed level of details for customer id with minimum order date for a year. This will depict our journey with customers from the start of the customer using our product until the present year.

Figure 75: Customer Loyalty Calculation

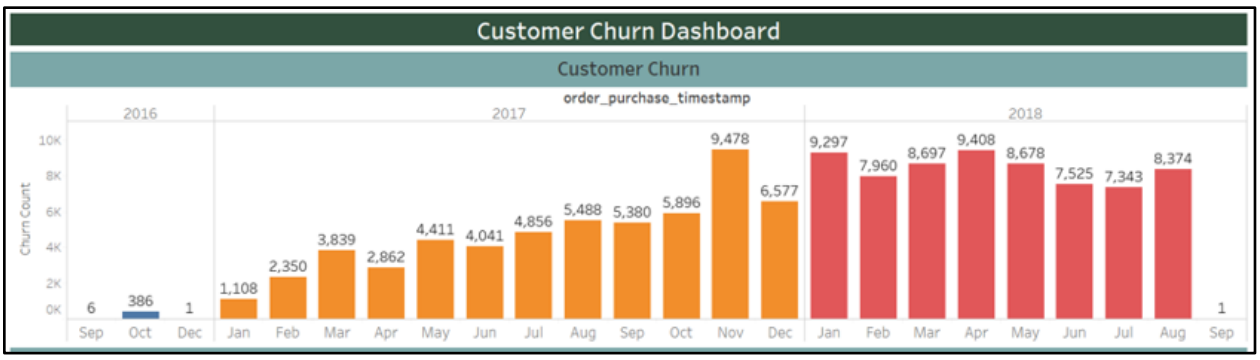


Figure 76: Customer Churn Dashboard

Explanation:

Here we have just performed exploratory analysis on customer churn rate every year to highlight the customer loyalty every year. It is depicted that overall customer addition was better in 2018 compared to 2017, a good sign of customer loyalty.

9.3. Payment Mode Dashboard

In this case, our intention is to explore the different payment modes, i.e. boleto, credit card, debit card and voucher. And the exploration would be using the normal and differential privacy methods.

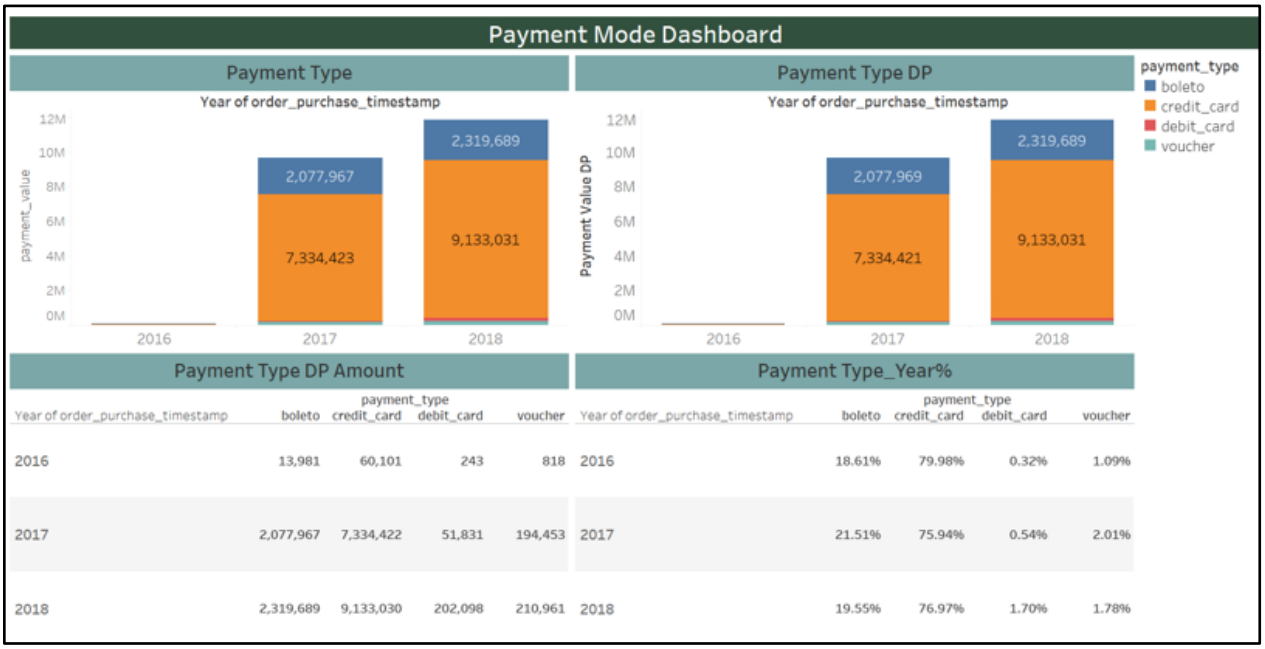


Figure 77: Payment Mode Dashboard

Explanation:

Here we have changed the worksheet into crosstab so that value of different payment modes across years can be judged. We have calculated the differential privacy like the ways used in the first case. Overall, the orders and payment in 2018 has been increased and mostly credit card and boleto are common method of payment. We have highlighted the amount and payment type using DP while the others are calculated normally. We don't find any difference in the values, but our intention is met. Although, the customer dependency on voucher utilization have been increased but the company needs to increase their voucher quantity in order to amplify their orders.

9.4. Association Rule Mining

Generally, for every business customer purchase pattern is utmost important as with consumer buying behaviour reveals a lot of secrets of customers. To create this, we have implemented an unsupervised learning in the retail which uncovers the interesting relationship between the products hidden in the large dataset.

We have used here for Market Basket Analysis to locate which product is being brought more frequently with the other products. By this retailer can plan what product needs to be kept beside what other product, just like there was a popular Walmart case. When Walmart combined Loyalty card system with point of sales data, they discover that Young Americans who buy Diapers in Friday Afternoon also buys Beer along with it. Here, we will perform exploratory basket segment analysis on a product as a whole. But when the situation arises, we want to link the customer buying habits, we have to add noise to their purchase amount and then perform the analysis. Therefore, if we are trying to perform analysis on individual customer level, we have to introduce differential privacy (Whitehorn, M., 2006.) (Wagle, M., 2020.)

Figure 78: Association Rule Mapping

Background Technical Calculations

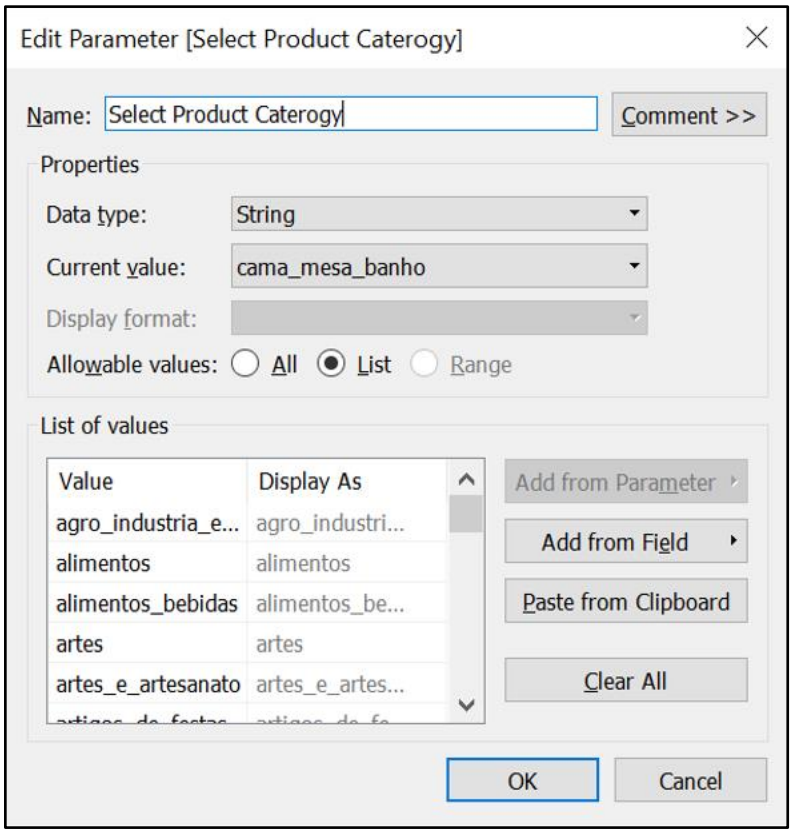


Figure 79: Association Rule Parameter Creation

Figure 80: Association Rule Count Creation

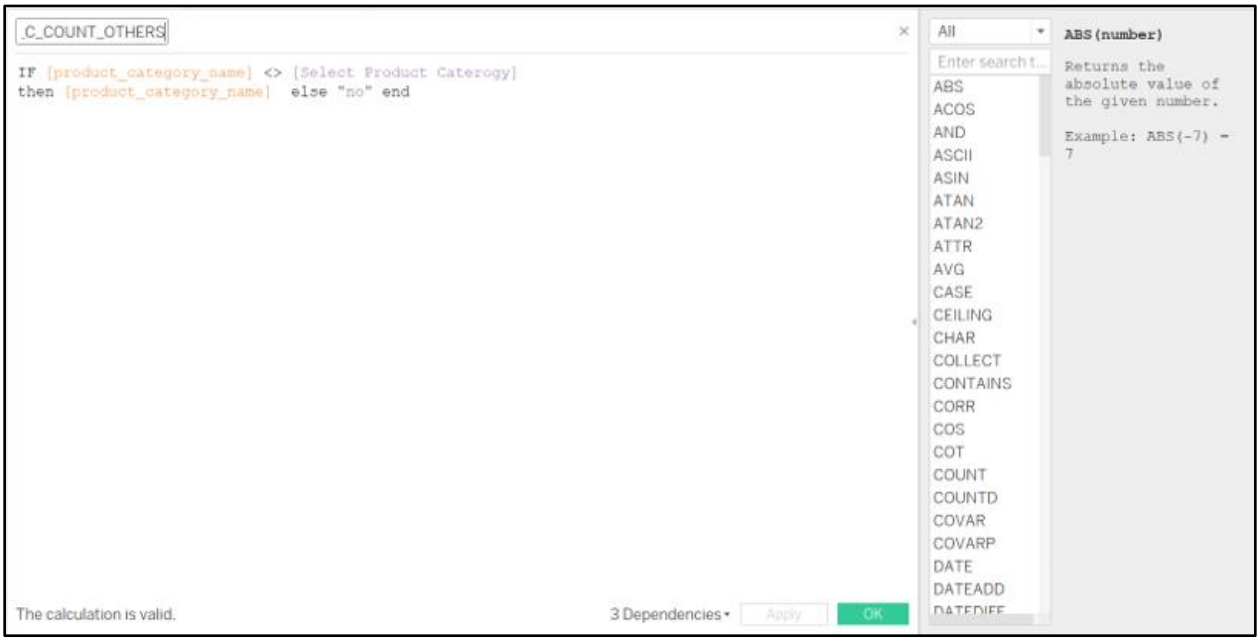


Figure 81: Association Rule Count Creation – Other than the Selected Product

Discussion:

At the onset, we have created a parameter taking all the product category values into the parameter. Thus, allowing the user to choose the product of their own choose so that they can observe the association of that product along with other products.

The second count calculation take the count of the product along with the selected product. If there is no match, it is assigned with a value 0.

In the third calculation, it highlights the product category for all the product category except the selected one. Else it is highlighted as “No” which is later filtered out in the process.

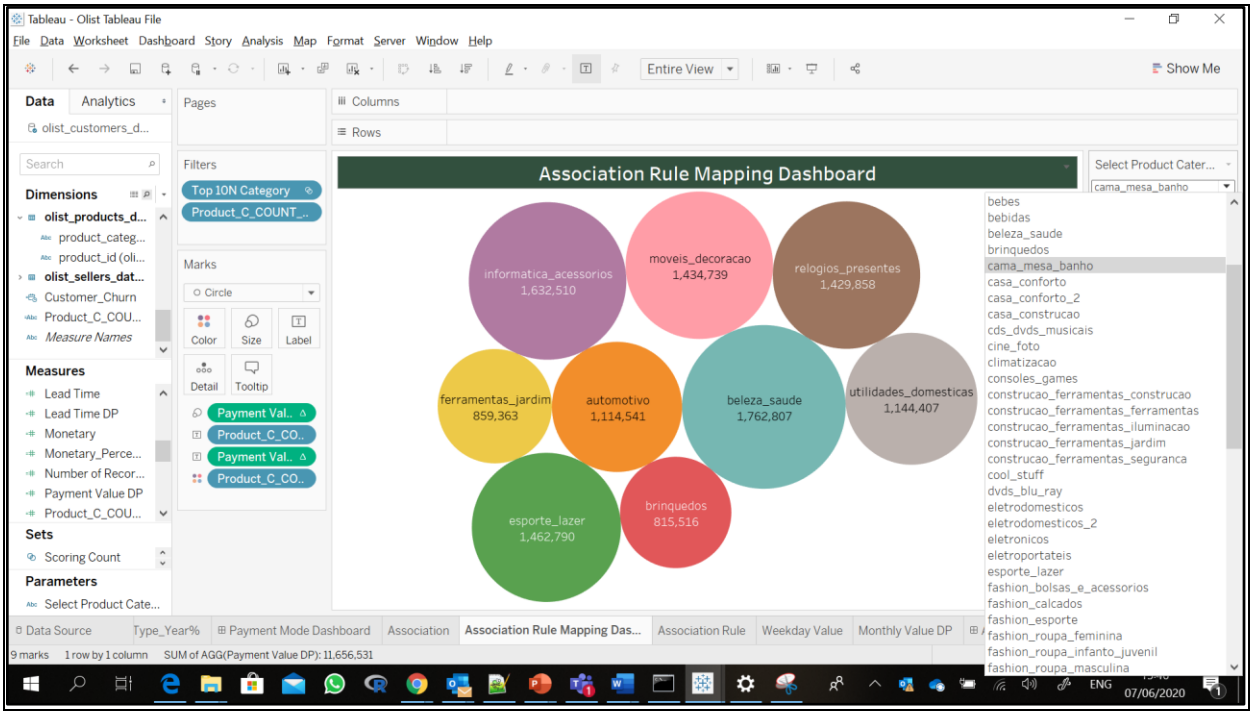


Figure 82: Association Rule Mining Dashboard1

Explanation:

The Association of product as per the selection is cama_mesa_banho. The payment value used here is powered by differential privacy, just in case we are taking a particular customer as a filter, then we are exploiting their buying habits and purchase amount. In this case, to implement differential privacy just to make a sense out of a Class A and vital customers buying pattern, noise is being added to the purchase order. We can see the top 10 products which goes hand in hand with the selected product and found that beleza_saude goes well with cama_mesa_banho as the number is quite high in the dashboard. That means retailer can plan to place beleza_saude and cama_mesa_banho on the same shelf so that customer can buy both of them all the time. Or else, they can give a association offer with a promotion to make both the products as Class A and Vital ones.

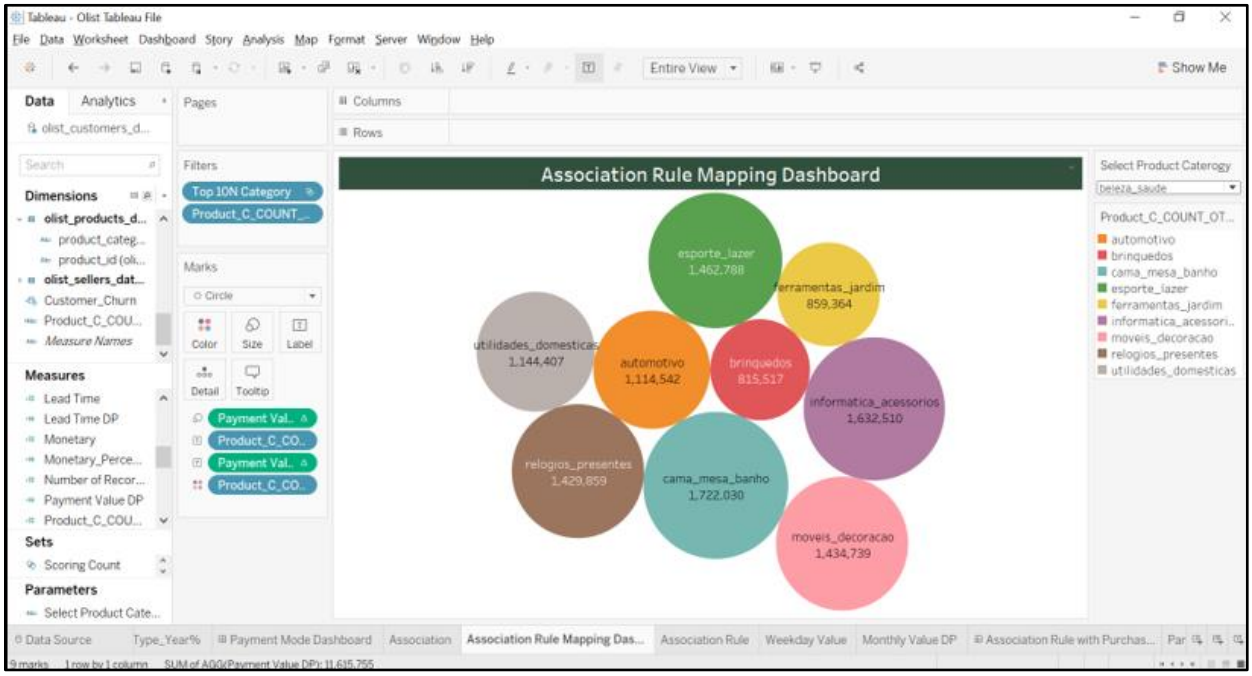


Figure 83: Association Rule Mining Dashboard2

Here in this case, we observe that when we select beleza_saude, cama_mesa_banho goes well with that. Hence, it is proved that they have very close association with each other, and retailer needs to plan on this strategy. Another solid idea could be placing the beleza_saude and cama_mesa_banho along with other products highlighted as they go well with these 2 products. The above data mining using differential privacy helps to fetch the product association information to understand the consumer buying pattern and further, helps the retailer to maximise their operations and sales provided if we are targeting a few customers who are Class A and Vital to us. Otherwise, adding noise to implement differential privacy do not make any sense.

Sometimes, it happens that the association products are placed far from each other and out of the sight of the customer. It just removes the possibility of maximising the sales. Therefore, Association creation would be a boon for retailers.

9.5. Pareto Analysis

Pareto Analysis is a statistical technique used in decision making which follows Pareto Principle, and sometimes also known as 80/20 rule. It states that 80% of the major problems are resolved by 20% of the major causes. In short, vital few and the trivial many concept. It is to perform exploratory analysis on the products. Further, Pareto Analysis is one of the popular tool used in Quality Control and Six Sigma.

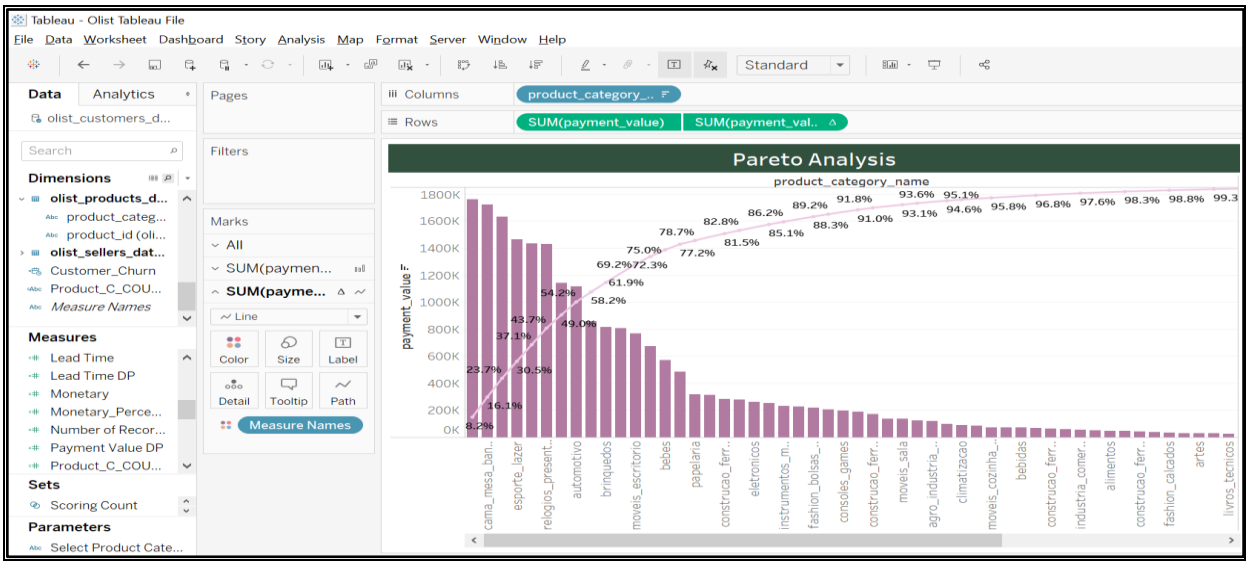


Figure 84: Pareto Analysis

Explanation:

From the above diagram, it clearly shows that the first 10-15 items which constitute nearly 20% of the items makes almost 70-80% of the sales. In short, if the retailer thinks of concentrating only on this top vital item, they will be solving 80% of the sales issue. Thus, identifying the major products is the essential need of any business. (Haughey, D., 2016.)

9.6. Lead Time Analysis

Lead Time is the actual time latency taken from the initiation of the order to the final delivery of the order.

There are two types of lead time:

i) Internal Lead Time: The total time consumed for buying process internal operation to progress from the identification of the requirement to the internal issue of the order placed.
ii) External Lead Time: The time consumed for external process which includes development work, manufacture, dispatch as well as delivery of the item.

The smaller the Lead Time, better is the service to the customer and indirectly enhances the customer satisfaction level.(Wikipedia, 2020. *Lead Time*.)

Background Technical Calculations

Figure 85: Lead Time Calculation

Figure 86: Lead Time Calculation using Differential Privacy

Discussion:

We have calculated the Lead Time by calculating the difference in days between the order purchase day and order delivered date. Then, after calculating the lead time, we wished to add noise by implementing the concept of Differential Privacy and analyse further in any level, let say on the customer, organization, or State.

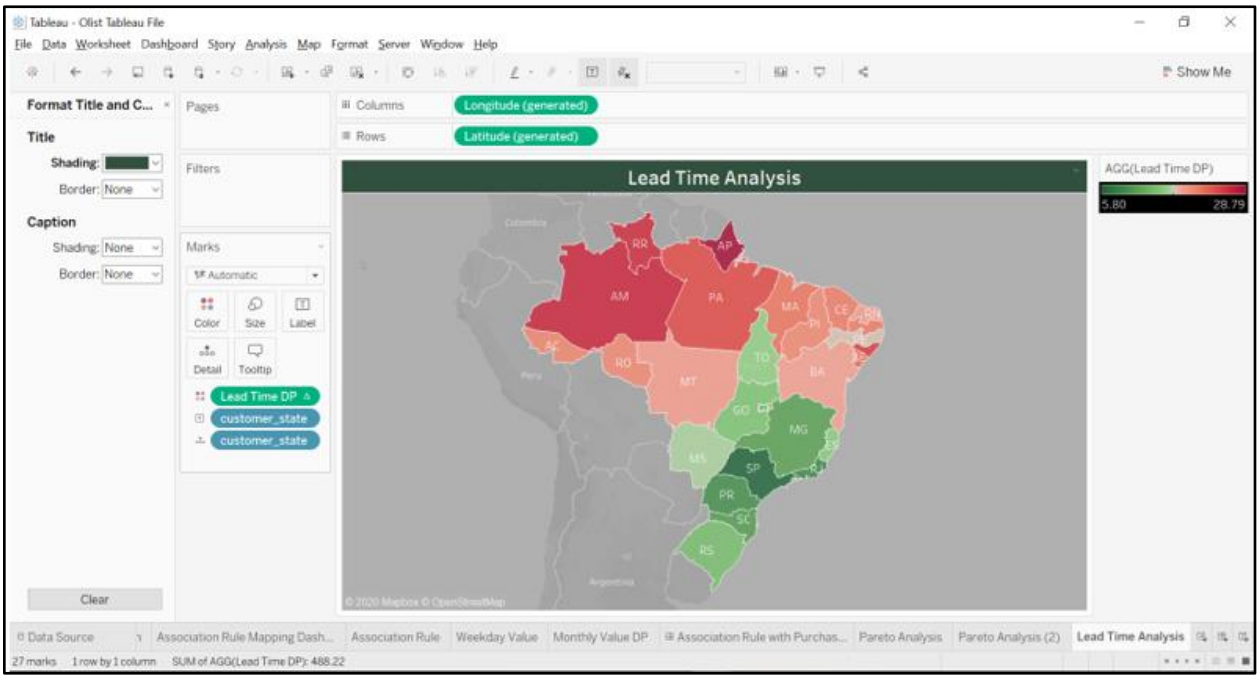


Figure 87: Lead Time Analysis

Explanation:

We have taken the average of the Lead Time with privacy budget added for differential privacy implementation. Then, we have assigned that value to every state of the country. This figure will show how was the delivery system in each state. The lower the Lead Time, better is the delivery system. In the above case, State with Code SP is the darkest green suggesting that it took the least Lead Time and the best state in terms of delivery system. But if we investigate state code AP it is dark red suggesting the worst state in term of delivery operations. Organization must look at state code AP for proper delivery operation by working upon all the possible ways to make it better.

10. Conclusion

In a modern era, de-anonymizing data is much more comfortable with the availability of advanced tools and techniques that aid hackers or malicious insiders to reassemble user identities by matching different datasets. Through our research, we would like to strengthen the fact that differential privacy is a straight-forward solution to the problem of de-anonymization. In addition to that, we have identified different scenarios where differential privacy can be beneficial to the retail sector. Following are a few recommendations we conclude from the study.

S. No.	Challenge / Opportunity in Retail Sector	DP Recommendation
1.	**Secured Data Sharing**	Data sharing is essential and unavoidable in the retail industries, be it with stakeholders, marketing teams, data analysts or with third parties like the suppliers. Besides, data sharing is the only way that could help businesses reach decisions through collaborative efforts. However, it brings significant privacy risks both for individuals and for the organisations. Differential privacy mechanisms such as the Laplace or Gaussian offers an effective way to protect individual privacy by injecting noise into the data or to the output of a machine learning model. Another way to preserve an individual's information is through aggregating the data. Statistical functions like anon_avg, anon_sum, anon_count and anon_std offer a way to incorporate aggregations into the results of an analysis.

		Since a random amount of noise is injected into the data, the attackers cannot backtrack and access the original data or identify a specific individual with full certainty. Therefore, these randomised methods facilitate secured data sharing.
2.	**Data Governance**	Data Governance generally decides the level of accessibility/visibility of the data to different personas. The data management techniques allow certain IT and Business stewards to view /edit limited data depending upon the regulatory compliance, risk management and accessibility defined by data governance managers. Now, with the implementation of differential privacy technique, it will outpace and override the data governance principle completely. Any user with any qualification in an industry can access the data with differential privacy implemented and use it for purposes they desire. The reason is that the data has been corrupted with noise by adding the privacy budget and the real data remains hidden behind the noise. Therefore, with the advent of Differential Privacy, we necessarily do not have to think of Data Governance and accessibility.

3.	**Cost-effective Privacy Solutions**	In a world where the risks and costs associated with privacy are on the rise, differential privacy offers a cost-effective solution. In the path to becoming GDPR compliant, many retailers are compelled to seek out privacy solutions that they cannot even afford. The differential privacy solution that we propose is based on python programming language and is derived from IBM's differential privacy library. Python is an open-source scripting language; similarly, the IBM differential privacy library is also readily available; therefore, there are no additional costs incurred on technology requirements. Furthermore, because of its active user community python stands as the primary choice for data analytics operations in most retail companies. Besides, the language also supports integration with a variety of BI tools such as Tableau. Therefore, the proposed solution can not only be implemented easily with a team of analysts but also can be accommodated effectively into the existing Business Intelligence (BI) systems.

4.	**Data Leakage in Machine learning models**	Machine learning models serve as the backbone for crucial retail-based analytics such as customer segmentation, market basket analysis, and advanced predictive analytics like automated pricing and recommendation systems. Differentially private machine learning algorithms help in reducing data leakage using minimal privacy budget to produce accurate insights. It reduces the overall privacy loss on explanation data, by adaptively reusing past differentially private explanations. It also amplifies the privacy guarantees concerning the training data. Furthermore, differentially private aggregation on historical data allows building predictive models that could benefit new customers rather than waiting for months to collect the data of the new customer (Georgianpartners.com 2020).

Summing up, retail businesses should consider differentially private solutions to avoid data leakage from their deployed machine learning models. Besides, differential privacy also helps organizations to build trust with its customers by guaranteeing the privacy of an individual's data mathematically. This trust eventually paves the way for aggregating even more customer data in the future, resulting in the ability to offer better AI solutions and product capabilities to customers efficiently. Even though differential privacy is an emerging technology, when appropriately applied, can provide beneficial results while safeguarding customer information. Therefore, every business dealing with data needs to be investing in understanding and adopting differential privacy.

11. Appendix

These are some of the providers with the privacy enhancing technique used by them:

Provider	Privacy Enhancing Technique in use:
Collibra	Differential privacy
Privitar	Encryption, Masking (Pseudonymization & Tokenization)
Infosum	Differential privacy
Ethyca	Automated data privacy
CloverDx	Deep natural Anonymization technique.
STATICE	Pseudonymization and K-anonymization techniques
Tonic	Differential privacy and data masking
Ekobit (BIzdataX)	Data masking
Microsoft and Harvard's Institute for Quantitative Social Science Collaboration	Open data differential privacy platform (https://www.linkedin.com/pulse/microsoft-harvards-institute-quantitative-social-science-john-kahan/)

Figure 88: Providers and their privacy enhancing technique

Video Links : Initial Research with Query Layer

1. https://drive.google.com/file/d/10uWsj6nLP1VeFg0J0QEUUfKM5-HLEb98
2. https://drive.google.com/drive/folders/1jUPnqS3Eh7j3i2Y0uuHL8hua2mFSxmS1

12. References

- Anon. (2017). 'Differential Privacy,' or How Apple Finds the Most Popular Emojis Without Reading Your Texts. Available at: https://medium.com/penn-engineering/differential privacy-or-how-apple-s-find-the-most-popular-emojis-without-reading-your-texts 7e9aad8a3ece. [Accessed on 29 May 2020]

- Brial, P.-O. (2018). What are the differences between Class A, B and C purchases? - Infographic. [online]. Available from: https://www.manutan.com/blog/en/procurement-strategy/what are-the-differences-between-class-a-b-and-c-purchases-infographic. [Accessed on 29 May 2020]

- Choi, J. (2018). It's easier than ever to travel with Project Fi. Available at: https://blog.google/products/project-fi/its-easier-ever-travel-project-fi/. [Accessed on 29 May 2020]

- Columbia Business School (2015)," What Is The Future Of Data Sharing?" Available athttps://www8.gsb.columbia.edu/globalbrands/sites/globalbrands/files/images/The_Future_of_Data_Sharing_Columbia-Aimia_October_2015.pdf [Accessed 29 June 2020]

- Georgianpartners.com (2020)," CEO's Guide to Differential Privacy", [Online] Available at: <https://georgianpartners.com/wp-content/uploads/2018/04/CEOs-Guide-to-Differential-Privacy-April-2018.pdf> [Accessed 4 July 2020].

- Google et al. (2018),"Differential Privacy", Available at https://github.com/google/differential-privacy [Accessed on 29 May 2020]

- Guru99, 2020. Star And Snowflake Schema In Data Warehouse. [online] Guru99.com. Available at: <https://www.guru99.com/star-snowflake-data-warehousing.html#:~:text=SNOWFLAKE%20SCHEMA%20is%20a%20logical,splits%20data%20into%20additional%20tables.> [Accessed 2 June 2020].

- Haughey, D., 2016. Pareto Analysis Step By Step. [online] Project Smart. Available at: <https://www.projectsmart.co.uk/pareto-analysis-step-by-step.php#:~:text=Pareto%20Analysis%20is%20a%20statistical,of%20doing%20the%20entire%20job.> [Accessed 2 June 2020].

- Holohan N., Aonghusa P. A., Levachar K., Braghin S. (2019), ”Diffprivlib: The IBM Differential Privacy Library”, Available at https://arxiv.org/abs/1907.02444v1 [Accessed on 29 May 2020]

- IBM et al. (2019), ”IBM Differential Privacy Library”, Available at https://github.com/IBM/differential-privacy-library/ .[Accessed on 30 May 2020]

- Kaggle (2018), Brazilian E-Commerce Public Dataset by Olist, Available at https://www.kaggle.com/olistbr/brazilian-ecommerce [Accessed 24 June 2020]

- Khalid, A. (2018). Google is making its differential privacy tool available to all developers. Available at: https://www.engadget.com/2019-09-05-google-is-making-its-differential-privacy-tool-available all.html? [Accessed on 29 May 2020]

- Kumar, sumit. (2019). Meet CRISP-DM: An approach to answer all your questions. Available at: https://medium.com/@sumit.yg/analyzing-seattle-s-airbnb-listings-data-49abdc0977c8. [Accessed on 29 May 2020]

- Marr, B. (2018). How Much Data Do We Create Every Day? The Mind-Blowing Stats Everyone Should Read. Available at: https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/#ad79efb60ba9.

- Regulation (2016), Regulation (EU) 2016/679 Of the European Parliament and of the Council, Official Journal of the European Union, Available at https://eur-lex.europa.eu/legal-content or https://www.privacy-regulation.eu/en/ [Accessed 28 June 2020]

- Rouse, M. (2005). What is RFM analysis (recency, frequency, monetary)? - Definition from WhatIs.com. Available at: https://searchdatamanagement.techtarget.com/definition/RFM-analysis#:~:text=RFM%20(recency%2C%20frequency%2C%20monetary,the%20customer%20spends%20(monetary). [Accessed on 1 June 2020]

- Savage, Olivia (2018)," Top 5 GDPR issues for retailers", Available at https://www.lewissilkin.com/api/download/downloadattachment?id=241c4c17-6fc3-4f34-9c6c-a1f5d8687c1f [Accessed 28 June 2020]

- Shawal, M. (2018). VED Analysis, SDE Analysis and FSN Analysis. [online]. Available from: https://www.yourarticlelibrary.com/business-management/marketing-management-business-management/ved-analysis-sde-analysis-and-fsn-analysis/69363. [Accessed on 1 June 2020]

- Stitch. 2020. Churn Rate 101. Available at: <https://www.stitchdata.com/churn-rate/> [Accessed 1 June 2020].

- Wagle, M., 2020. Association Rules: Unsupervised Learning In Retail. Medium. Available at: <https://medium.com/@manilwagle/association-rules-unsupervised-learning-in-retail-69791aef99a> [Accessed 1 June 2020].

- Whitehorn, M., 2006. The Parable Of The Beer And Diapers. Theregister.com. Available at: <https://www.theregister.com/2006/08/15/beer_diapers/> [Accessed 2 June 2020].

- Wikipedia, 2020. Lead Time. En.wikipedia.org. Available at: <https://en.wikipedia.org/wiki/Lead_time> [Accessed 2 June 2020].

- Wikipedia (2020), "Elbow method (clustering)", Available at https://en.wikipedia.org/wiki/Elbow_method_(clustering) [Accessed 24 June 2020]