NOVA IMS Information Management School

MGI

Master Degree Program in

Information Management

Clustering in Risk Management Systems

Features that will help the Portuguese Tax Authority to identify and classify taxpayers

Ana Beatriz Silvestre Tavares

Project Work

presented as partial requirement for obtaining the Master's Degree Program in Information Management

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

NOVA Information Management School

Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

CLUSTERING IN RISK MANAGEMENT SYSTEMS

Features that will help the Portuguese Tax Authority to identify and classify taxpayers

by

Ana Beatriz Silvestre Tavares

Project Work presented as partial requirement for obtaining the Master's degree in Information Management, with a specialization in Knowledge Management and Business Intelligence

Supervised by

Prof. Nadine Côrte-Real, PhD, NOVA Information Management School

February, 2024

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledged the Rules of Conduct and Code of Honor from the NOVA Information Management School.

Lisbon, 25-02-2024

DEDICATION

To my supervisor, Professor Nadine Côrte-Real, for the knowledge and guidance she has given me and for the availability demonstrated for this project.

To my family, friends, and boyfriend for their support, incentive, and motivation that were fundamental for me to be able to write this thesis.

To my colleagues and a special thanks to my company colleagues that helped me in the discussion of ideas and supported me in working on this subject.

ABSTRACT

The payment of tax obligations by taxpayers is crucial to ensure that the State and Government Agencies have the financial resources to meet the public expenditure inherent in a state governed by the rule of law. And in the case of the Tax Authority (TA), it is also important to know its taxpayers and what characterizes them.

The focus of this work is to classify taxpayers in a way that allows the Tax Authority to act differently within various groups, aiming to prevent non-compliance situations. The Portuguese Tax Authority already detected 23251 million euros in uncollectible tax debts in the debt portfolio from 2011 to 2021.

Through this work, it was possible to characterize the taxpayer, not only the so-called "demographic" characteristics, but also characteristics based on his debts, both debts on consumption and income, and also tax debts. This allowed us to understand the behavior of the taxpayers who will have bad debts in order to anticipate them.

The metrics were identified and later the data extraction. A cluster analysis was conducted using the extracted variables in order to understand how taxpayers' group together based on their characteristics.

With this thesis, we found that there were some differences between clusters created in the 2017 fiscal year and those created in the 2021 fiscal year (individuals with individuals and business with business), which may contain important information for the TA.

KEYWORDS

Risk Management, Machine Learning model, Clustering, Unsupervised Learning, Portuguese Fiscal System, Taxpayers, Features.

Sustainable Development Goals (SGD):

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

VAT	Value-added tax
IRS	Personal income tax
IRC	Corporate income tax
DSRM	Design Science Research Methodology
TA	Tax Authorities
AI	Artificial Intelligence
ML	Machine Learning
CAE	Classification of Portuguese Economic Activities by Branch of Activity
NUTS	Nomenclature of Territorial Units for Statistical Purposes
DMR	Monthly Statement of Remuneration
IMI	Property Tax
IMT	Property Transfer Tax
IUC	Vehicle circulation tax
PCA	Principal Component Analysis

1. INTRODUCTION

1.1. PROBLEM IDENTIFICATION AND MOTIVATION

The payment of tax obligations by taxpayers is crucial to ensure that the State and Government Agencies have the financial resources to meet the public expenditure inherent in a state governed by the rule of law.

The focus of this work is to classify taxpayers in a way that allows the Tax Authority to act differently within various groups, aiming to prevent non-compliance situations. The Portuguese Tax Authority already detected 23,251 million euros in uncollectible tax debts in the debt portfolio from 2011 to 2021 (based on the *Report on the Fight against Tax and Customs Fraud and Evasion 2021*). At the end of 2021 were accounted 939 million euros of tax debt, which is something quite impacting. The purpose of this thesis is to apply methods and tools that will help the Portuguese Tax Authority identify ways to address relevant issues to prevent the increase of taxpayers' debt.

1.2. DEFINING THE OBJECTIVES FOR A SOLUTION

As we began to think about the scope that this thesis might take, a few questions came across, namely:

- How are taxpayers characterized by the Tax Authority?
- Is there a set of characteristics that could configure future non-compliance situations and allow the Tax Authority to act preventively within these groups?

It is important to study how these measures can impact the Tax Authority, as it already has preventive measures for some of the issues but not all.

The main idea of this thesis is to apply a clustering algorithm in order to understand how taxpayers are grouped together based on their characteristics and what information can we glean from our analysis.

1.3. RELEVANCE AND IMPORTANCE

According to Gabinete do Secretário de Estado dos Assuntos Fiscais (2022), there is a lack of "... methodologies to predict default, which allow the identification of companies at risk of becoming insolvent, in order to protect the State's interests in these situations.". This thesis provides a model for classifying taxpayers in the Portuguese tax system and will hopefully enable the Tax Authority to target its inspection actions more effectively and consistently to certain taxpayers.

2. LITERATURE REVIEW

Big data methods are fast turning into an essential tool for the detection of tax fraud around the world. In this context, it is noticeable the use of the Unsupervised Outlier Detection Method as a resourceful framework to explain the different phenomena that surround us (Adamov, 2019). The idea of combining clusters and training the models allow us to obtain relevant information for our research problem.

Taxation is a substantial source of government income for numerous countries (Adamov, 2019) and that's why tax evasion and tax avoidance pose significant challenges for authorities in every country around the globe. The estimated economic impact of these tax-related issues amounts to 3.2% of GDP in OECD countries (Buehn and Schneider., 2016).

The taxpayer clustering model is considered as a useful tool to support decision-making on credit risk by the financial regulator (Biryukov et al., 2022). Since the main purpose of this thesis is to classify taxpayers and group them according to their characteristics, we have decided to adopt this method.

The payment of taxes and duties by the taxpayers constitutes the main source of revenue for the State, to make public expenditures and guarantee the citizens' right to education, justice, health, and security.

In Portugal, taxes are classified into income, consumption, and wealth taxes. In the scope of this thesis, we will focus essentially on taxes on income (IRS, IRC) and consumption (VAT), because based on the *Report on the Fight against Tax and Customs Fraud and Evasion 2021*, out of a total of 939 million euros in tax debt paid by coercive collection, 29.6% refers to IRS, 27.6% to VAT, 17.5% to IRC and, finally, 25.2% to other taxes where IMT, IMI and IUC are included. As the latter refers to a set of taxes with less significant values, we have opted to use only the three taxes with the greatest impact. This information is very important because this large amount of debt is being paid after the end of the voluntary tax assessment period, which also leads to extra monitoring of resources by the Tax Authority, which still has to remind taxpayers (by email or letter) that they have missed a payment.

AI and ML are starting to be used, due to the complexity of some topics, where tax is known to be one of the most complicated business processes. There are more and more opinions that advanced tax analysis can be reached with machine learning and that is going to improve data quality, find missing support documents and details, and decrease human errors in tax processes. Never forgetting some concerns regarding the use of AI and ML for all businesses (where tax processes are included), like:

- Project design, execution, and investment planning;
- Worker-resource development and job disruption;
- Data access;

- Ethics and privacy;
- Regulations, law, and government policy[1].
- Machine learning is a section of artificial intelligence that enables machines to learn and develop over time despite their limited memory when it comes to process and storing data (Richardson, 2022).
- Again, considering Milner & Berg (2017) recent developments in tax technology indicate that AI assistance will be increasingly used, by various institutions like taxation authorities, tax advisors, etc.
- According to Savić et al., (2021), due to the absence of available tagged and grounded truth data that would allow "unbiased" supervised learning, tax evasion detection is usually addressed by unsupervised methods, based on anomaly detection algorithms. In comparison to supervised methods, unsupervised is less accurate: they don't only detect tax evasion cases but also identify business entities with irregular and suspicious tax behavior, also including dishonest taxpayers. Consequently, a tax evasion risk management system that is based on accurate, unsupervised learning can result in a better use of resources in the operational treatment of the identified risks (Sebtaoui et al., 2020).

Considering Vanhoeyveld et al. (2020), unsupervised anomaly detection shows a big predictive power for VAT fraud detection. It was already proposed to, a computational intelligence model to study the VAT fraud dynamics by Chica et al. (2021).

In this study, we use PCA and a clustering algorithm (K-Means), which is an unsupervised method algorithm of machine learning, where the objective is to identify groups of data with similar characteristics, in our case taxpayer's characterization, like Pinheiro et al. (2021). Due to is exploratory nature, it's usual to use this algorithm as the initial proposal in a new project (Milner & Berg, 2017).

The method that most effectively condenses the information from all input variables into orthogonal factors is Principal Components Analysis (Del Camino González Vasco et al., 2021) and this is the technique we initially use to reduce the dimension of our sample having in count to our economic and demographic variables. Followed by the K-Means algorithm which efficiently groups large data (it is not sensitive to the order of the data) based on their similarity (of the taxpayer behavior) and is simple to implement and interpret (Domingo Velasquez, 2007).

[1] Milner & Berg (2017).

2.1. TAXES

2.1.1. Personal income tax (IRS)

Personal income tax (IRS) applies to the income of citizens residing in the Portuguese territory and of non-residents who earn income in Portugal. It is a direct and progressive tax.

The IRS is calculated individually, but couples or unmarried couples may choose to file the IRS jointly. In this case, the tax is levied on the sum of the incomes of the persons comprising the household.

Tax base

Looking at *Código do Imposto Sobre o Rendimento das Pessoas Singulares (CIRS)*, the tax is based on the annual value of the income in the following categories, even when it derives from illegal acts after the corresponding deductions and rebates have been made, according to the bracket to which you belong, and considering the deductions provided for by law (for example, education or health expenses):

Category A - Income from dependent work;

Category B - Corporate and professional income;

Category E – Capital income;

Category F - Property income;

Category G – Asset increments;

Category H - Pensions.

2.1.2. Corporate income tax (IRC)

All companies that obtain income in Portugal are subject to the corporate income tax (IRC).

This tax is levied on income obtained, even when arising from illegal acts, during the tax period, by the respective taxpayers, under the terms of the legal framework of this tax.

IRC focuses on:

- The profit of commercial companies or civil companies under the commercial form, cooperatives, and public companies and that of other legal persons or entities that mainly carry out a commercial, industrial, or agricultural activity;
- The global income, corresponding to the algebraic sum of the income of the several categories considered for IRS purposes, as well as the asset increases obtained

gratuitously, of legal persons or entities that do not exercise, as the main activity, a commercial, industrial or agricultural activity;
- The profit attributable to a permanent establishment located in Portuguese territory;
- Income of the various categories, considered for IRS purposes, as well as asset increases obtained free of charge by entities, with or without a legal personality, which does not have a permanent establishment or, having one, are not attributable to it.

Profit consists of the difference between the net worth values at the end and the beginning of the tax period (based on Código do Imposto sobre o Rendimento das Pessoas Coletivas (CIRC)).

2.1.3. Value-added tax (VAT)

Value-added tax (VAT) is a tax applied to sales and services rendered in Portugal. By sales or services rendered, we mean:

- Transmissions of goods and services rendered in the national territory, for consideration;
- Imports of goods, considering as such, the entry of goods into the national territory from third countries or territories (extra-community);
- Intra-community transactions carried out in the national territory, as defined, and regulated in the VAT Regime on Intra-Community Transactions.

VAT is paid by the consumer at the time he pays for the good or service provided. The seller or service provider receives the VAT amount and subsequently hands it over to the Tax and Customs Authority (AT).

In the context of this thesis and based on *Código do Imposto sobre o Valor Acrescentado*, the analysis of information regarding the submission of periodic VAT declarations and the payment of the tax will help to identify anomalous patterns in these phases of the taxpayer's relationship with the Tax and Customs Authority.

2.1.4. Tax Justice

An important part of the assessment of the taxpayer's situation with the Tax Authority is the indication of the existence of current or past debt. Thus, information from tax infringements and tax litigation is essential to identify situations that may constitute a risk of non-compliance (based on *Código de Procedimento e de Processo Tributário*).

When a debt enters into tax foreclosure, several steps are followed to protect the legal rights or interests of taxpayers (and the Tax Administration).

Figure 2.1 – Tax Foreclosures Process

Source: (Autoridade Tributária e Aduaneira, Código de Procedimento do Processo Tributário)

The debts that can be collected through a tax execution procedure are those resulting from non-payment of taxes, social security contributions and quotes, fees, fines, penalties, other financial contributions, interest, and other debts to the state, finance, social security, or any other legal persons under public law.

The institution of tax execution proceedings is the responsibility of the tax administration services. When the debt process is initiated, the taxpayer still has a deadline to make a voluntary payment of the debt. After this deadline, the debt enters the enforced collection phase.

The tax execution process can only be suspended if there is an administrative claim, a court challenge, or a judicial appeal regarding the legality of the enforced debt. However, these processes alone do not determine the suspension of the tax execution.

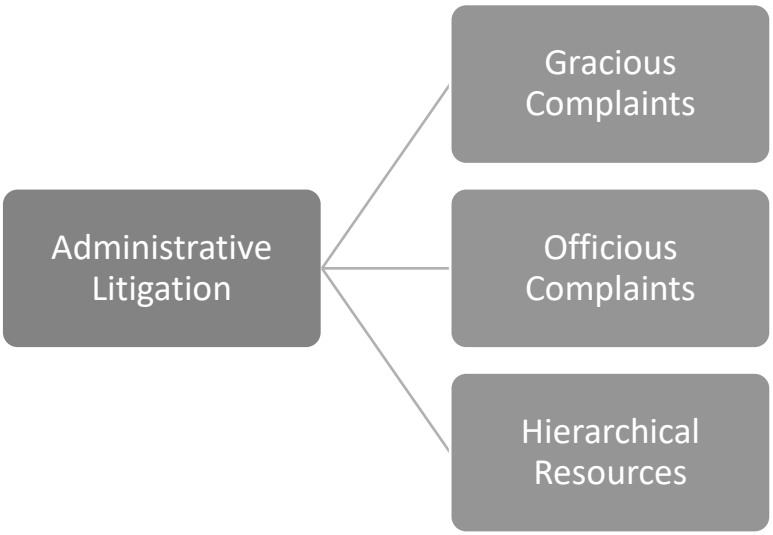

Figure 2.2 – Administrative Litigation

Source: (Autoridade Tributária e Aduaneira, Código de Procedimento do Processo Tributário)

After the deadline for voluntary payment, the measure that is most often used to ensure the realization of tax credits is the attachment of assets.

This whole process must be carried out as quickly as possible by the Tax Administration since the taxpayer may sell and dissipate their assets, which becomes an obstacle to the realization of tax credits. In order to prevent this type of behavior by the debtor, the Tax Administration can activate the arrest and/or attachment of assets. In addition to these measures, the Public Treasury also has the guarantees of credit privilege, legal mortgage, pledge, and right of retention.

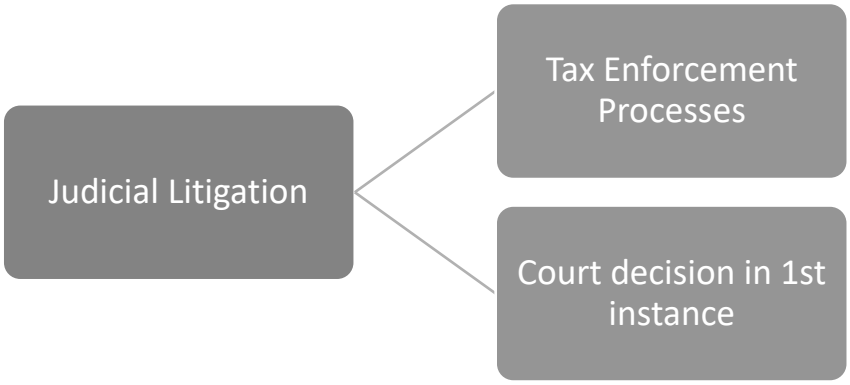

Figure 2.3 – Judicial Litigation

Source: (Autoridade Tributária e Aduaneira, Código de Procedimento do Processo Tributário)

One of the main purposes of identifying the most important taxes is to characterize the taxpayer, not only the so-called "demographic" characteristics, but also characteristics based on his debts of consumption, income, and tax debts.

2.2. UNSUPERVISED METHODS

According to Roux et al. (2018) consulting tax returns is a slow and expensive process, so consulting historical information is complicated and restricted.

Considering the paper of Dridi (2021) there are four types of Unsupervised Learning:

- Clustering;
- Association Rules;
- Anomaly Detection;
- Autoencoders.

2.2.1. Clustering

This algorithm will find groups in our data where the constituents are alike, it may be the most important unsupervised learning problem. It's possible to choose how many groups (clusters) the algorithms should identify (enables adjustment of the granularity of the clusters).

Figure 2.4 – Workflow of Clustering

Source: (Dridi, 2021)

There are five types of clustering algorithms:

- Hierarchical Clustering (composed of hierarchical data, like taxonomies) where the BIRCH algorithm is an example of a clustering technique used in big amounts of datasets;
- Centroid-based/Partitioning Clustering (efficient but sensitive to initial conditions and outliers) where k-means is the most algorithm used;
- Density-based Clustering (difficult with various densities of data and high dimensions) where DBSCAN is one of the algorithms;
- Distribution-based Clustering (this takes into account that data is formed by distributions) where Gaussian distribution an example of a used one;
- Fuzzy Clustering (enable data to be in more than one cluster);
- Grid-based Clustering (contrary to other methods, grid methods divide the space into cells, regardless of the distribution of objects).

2.2.2. Association Rules

Also known as frequent patterns. It's divided into:

- Frequent item sets (enables the detection of associations and correlations between items in datasets);
- Frequent subsequences (enables the detection of patterns through time or positions in a dataset);
- Frequent substructures (combined sets of common articles and common subgroups).

It's usually used in cross-selling, upselling, product, affinity promotion, and customer behavior. And not so typically used in fraud detection and medical treatment improvement of (Dridi, 2021).

2.2.3. Anomaly Detection

Anomaly detection is one of the most common applications of machine learning.

Based on Dridi (2021).Finding and identifying outliers helps to prevent fraud (fraud detection), adversary attacks (like military surveillance), and network intrusions (intrusion detection in companies, for example).

2.2.4. Autoencoders

The most well-known example of unsupervised algorithms is neural networks.

Autoencoders are a particular type of feedforward neural network where the input is exactly the same as the output. They compress the input to a lower-sized code and then rebuild the output from that representation.

It is composed of three parts:

- Encoder;
- Code;
- Decoder.

The encoder compacts the input and generates the code, while the decoder reconstructs the input using that code (Dridi, 2021).

Dias et al. (2016) classify taxpayers using financial indicators based on their risk of tax evasion. They made a study using a Cluster Analysis methodology to classify their observations into homogeneous groups that allowed them to identify tax evasion. Roux et al. (2018) detects tax fraud for under-Reporting declarations using clustering and estimating the distributions of declarations. Clustering algorithms are also used to characterize and detect those potential users of false invoices identifying groups of similar taxpayers' behavior (González & Velásquez, 2013).

In this thesis, we decide to use Clustering. We want to identify groups and, hopefully, some important characteristics of the taxpayers.

In Portugal, according to Gabinete do Secretário de Estado dos Assuntos Fiscais (2022), "... methodologies to predict default, which allows the identification of companies at risk of becoming insolvent, in order to protect the State's interests in these situations." Haven't been implemented yet. For that reason it is paramount to approach this topic and this work represents the first step to solve this problem.

3. METHODOLOGY

For this thesis, we choose Design Science Research Methodology (DSRM). According to Henver, et al. (2010), design science sustains a pragmatics-based research paradigm that calls for the creation of inventive artifacts to resolve real-world situations. Baskerville et al. (2018) defend that there is a relationship between science and technology. Science has its objective in the development of descriptive knowledge based on the world and human behavior through the implementation of scientific methods. Technology has as an objective the development of the prescriptive knowledge based on the purpose of designing artifacts to increase human capabilities (physically and mentally).

As for the objective of this thesis, we try to understand a real issues, which is the characterization of taxpayers based on their characteristics and their debts

Kuechler, B. et al. (2008) schematized the process in:

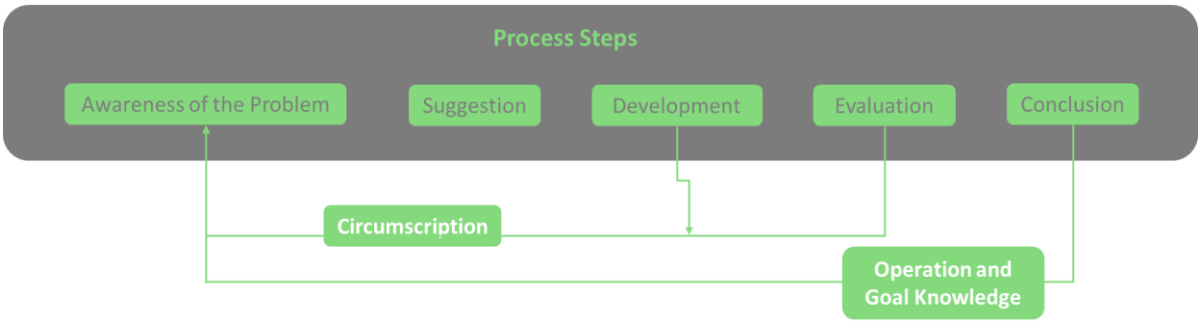

Figure 3.1 – Design Science Research Steps

Source: (Baskerville et al., 2018)

Awareness of the Problem: The initial motivation for the project emerged with the need to classify the taxpayers and analyzing which characteristics stand out. The data was structured in such a way that it is possible to build a base on the taxpayer. The lack of a tool that would allow to understand the characteristics of taxpayers was also a motivation for the project.

Suggestion: In this stage, a draft design of the final artifact was developed in which the new functionality or element is proposed through a prototype. Based on the existing literature it was suggested to use unsupervised methods that can help to identify, in groups, what can be a flag in the taxpayers' characterization.

Development: Deployment of the artifact proposed in the previous step. In this stage, the tasks that include data preparation, transformation, and modeling were implemented, as well as the selection of the method or methods for obtaining the best group of taxpayers' characteristics, considering the best methods from the literature review.

Evaluation: After the solution developed, it is necessary to evaluate whether the solution solves the initial problem and fulfills the objectives set for the thesis or not. It is also very important to understand the value added to the TA that will use this clustering method for future actions.

Conclusion: To conclude, the results of the study are presented with the aim of helping to solve the initial problem. Furthermore, the acquired knowledge is documented so that it can help to contribute to the creation of knowledge in the area and for upcoming studies like this.

3.1. SAMPLE

The selection of data for the sample went through several stages that will now be explained.

3.1.1. Definition of the criteria for obtaining the sample of taxpayers

The definition of the criteria to be used in obtaining the sample of taxpayers relevant to the study of the problem in question was the most challenging phase, given that within a universe of taxpayers, there are many variables and criteria that can be used to obtain a sample. Since the SR of this thesis focuses on predicting taxpayers that may have debts, obtaining taxpayers that have somehow incurred tax debts ended up being determinant in defining the criteria to be applied in obtaining the sample.

Understanding the characteristics of taxpayers is a determining factor in assessing the tax situation of taxpayers. So, we started by analyzing all the information on taxpayers with debt because it is part of some taxpayers' tax situation, and it is the most laborious sample. Given that the debt has an instauration phase, during which the debt may be contested by the taxpayer and may be canceled because of this contestation, we chose to focus our analysis on the data of taxpayers with confirmed debt. This option allowed us to focus the analysis on features that may contribute to a more precise characterization of the taxpayers with debt.

Additionally, we choose to focus the analysis on taxpayers with tax debts. This means that we analyzed taxpayers with debts in income, consumption, and property taxes.

The focus of the analysis was on debts occurring between 2017 and 2021 (regardless of the current state of debt) since the information for these years is already consolidated and does not show large fluctuations.

According to *Código de Procedimento e de Processo Tributário*, the tax debt can be in one of the following states:

Active:

- Active debt to date.

Suspended:

- Enforcement is suspended until the decision of the plea in the case of an administrative complaint, judicial review, or legal appeal regarding the legality of the enforceable debt;

- Enforcement also stays if, after the deadline for voluntary payment, a guarantee is provided before the filing of the corresponding administrative or judicial appeal;

- The execution is also suspended for a maximum period of 120 days, as of the end of the voluntary payment period.

Stated in Failures:

If it turns out:

- Lack of attachable assets on the part of the taxpayer, his successors, and jointly or severally liable persons;

- If the debtor is unknown and it is not possible to identify the building when the sizable debt is a real estate tax;

- The debtor of the seized credit is absent in an unknown place and has no other attachable assets. *(Article 272)*

Extinct:

- By enforced payment (SECTION X, SUBSECTION I);

- By voluntary payment (SECTION X, SUBSECTION II).

We also analyzed that are two types of NIFs:

1 - Individual

2 - Business

We independently analyzed the debt of individual taxpayers and business taxpayers.

Then we identified that our dimensions for building the basis of taxpayer activity are:

- Origin of debt (tax type);

- Type of NIF (individual or business);

- Status (stage of the debt);

- Year of filing.

The dimensions will be used to group the information in order to draw a uniform and complete sample of taxpayers.

Sample of taxpayers with debts:

We decided to make distributions (of the amount of debt) by type of NIF, state of debt, and tax (of which we only used IRS + DMR, IRC, and VAT, for the characterization of the taxpayer, the IMT and IMI data will only be used for our models and algorithms and will be taken based on the assets debts of the taxpayers that we took).

We performed an analysis where we used as base the amounts of taxpayers and the percentages they represented in each of the debt value intervals (these percentages will be used later to obtain the sample). This would allow us to have a representative sample of the taxpayers with debt, so we opted to sort them by ascending the amount of debt and dividing them into intervals of the amount owed. The intervals are used to ensure that we have a representative distribution and as homogeneous as possible of the universe of taxpayers, for example, children have NIFs but do not yet have debts, considering them would not be good for our sample.

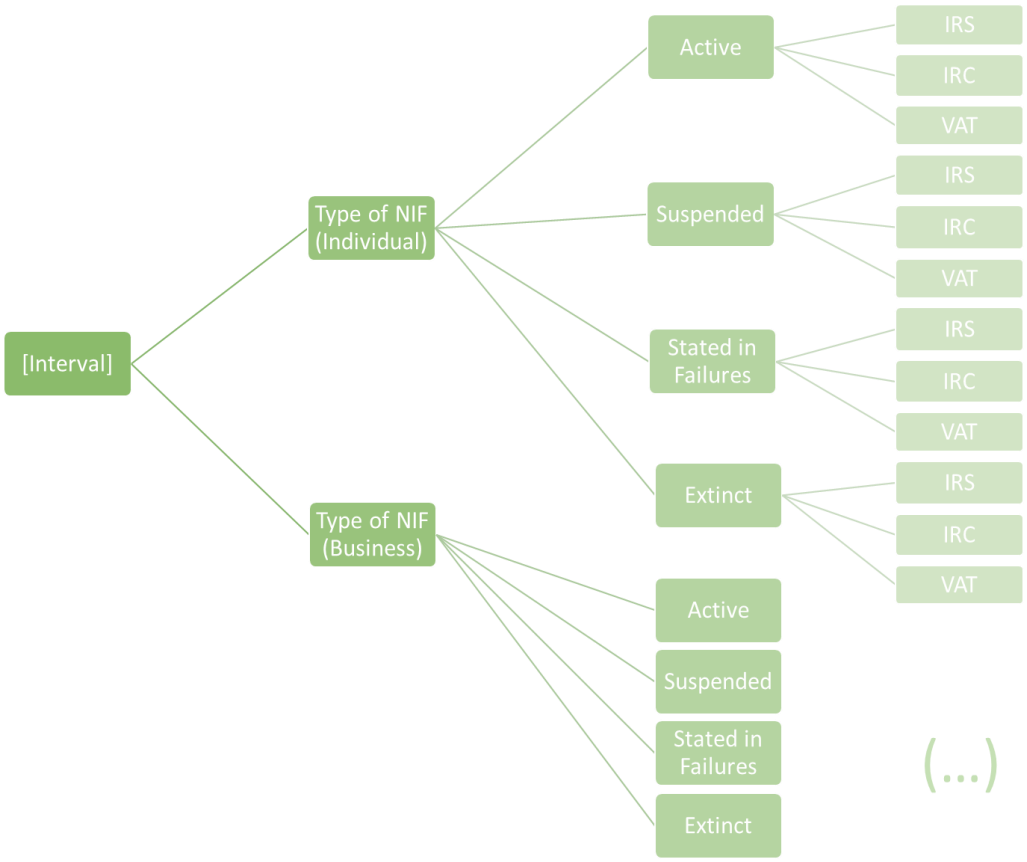

Figure 3.2 – How the percentages of the sample of taxpayers with debts were calculated

Sample of taxpayers without debts:

The sample was obtained by subtracting the taxpayers with debt from the total universe of taxpayers (of VAT, IRS, and IRC).

For the total universe we used the type of NIF (individual or business) and the origin of the debt (VAT, IRS, and IRC) as dimensions (they were already used before to obtain the sample of taxpayers with debt), but the intervals to be considered will have to be different because we will have no value of debt to guide us. We decided to apply different metrics for each tax because each one is measurable in a different way: for VAT we will use the total taxable base to define our ranges; for IRS total taxable income, where we will apply an autonomous taxpayer flag and for IRC we will use taxable income for business as a metric.

3.1.2. Defining the sample size

We selected about fifty thousand registers of taxpayers with debt and fifty thousand without debts, we tried to use a sample of one hundred thousand different NIFs with five years of information of each taxpayers' sample (50% of taxpayers with debt and 50% of taxpayers without debt).

3.1.3. Obtaining the data according to the defined criteria

They were obtained as mentioned in point 3.1.1. by the percentages that the taxpayers represent in each of the debt/income value intervals randomly, taking one record out of every ten or so.

3.2. CLUSTERING

3.2.1. PCA Algorithm

According with (Kurita, 2020) PCA (Principal Component Analysis) is a technique used to find a new set of variables, called principal components, that represent a given data set in a more meaningful way. The main objective of PCA is to reveal hidden patterns and structures in the data by identifying the most important features and filtering out the noise. It's used in different applications such as:

- Dimensionality reduction;
- Data compression;
- Feature extraction;
- Data visualization.

In our case, it's going to help us understand how many clusters we seem to have considering the percentage of the variance of the components.

3.2.2. K-Means Algorithm

K-means clustering is an Unsupervised Learning algorithm used to discover multiple groups or clusters that were not explicitly labeled in the data. The variable "K" represents the predetermined number of groups and can be given any value, for example, if K=3, there will be three clusters. This algorithm is iterative and divides the unmarked dataset into the K clusters that are chosen. According to (Dridi, 2021), Each dataset is assigned to only one group that has similar characteristics.

4. RESULTS AND DISCUSSION

This chapter presents the outputs of the analyses developed, as well as the results of the application of the algorithms and the comparison between the groups in order to understand the different characterization. All figures presented in this section were produced by me in Jupyter Notebook using the Python programming language.

In the first step, we made an analysis to understand our data. And after we performed clustering to identify groups of taxpayers with similar characteristics

Considering that we are analyzing data from two samples (taxpayers with and without debt, that were converged in one) we hope to have them as homogeneous as possible to classify the most similar groups of taxpayers and obtain the best result.

The most important variables we analyzed and selected for our analyses were:

Variables	Description
ANO_EXERCICIO	Fiscal year ranging from 2017 to 2021.
ANO_NASCIMENTO	Taxpayer's year of birth.
CODIGO_ATIVIDADE	Code of the subclass of the Activity in the reference period.
CODIGO_DIV_ATIVIDADE	Code of the division of the Activity in the reference period.
CODIGO_IRS	IRS activity code.
DISTRITO	Taxpayer's district. Divisions based on NUTS, each code corresponds to a district.
ESTADO_CIVIL	Taxpayer's Marital Status: S - Single, divorced or legally separated; C -Married; F - Separated in Fact; U - United in Fact; V - Widowed.
GENERO	Taxpayer gender: M – Male; F – Female.
ID _CONTRIBUINTE	Unique ID of the taxpayer.
INDICADOR_ATIVIDADE	Flag representing whether the taxpayer has open activity: V – Yes; F – No.
INDICADOR_DEV_ESTRATEG	Flag representing whether the taxpayer is a strategic debtor: V – Yes; F – No.

INDICADOR_DIV_IRC	Flag representing whether the taxpayer has debts in IRC: V – Yes; F – No.
INDICADOR_DIV_IRS	Flag representing whether the taxpayer has debts in IRS: V – Yes; F – No.
INDICADOR_DIV_IVA	Flag representing whether the taxpayer has debts in VAT: V – Yes; F – No.
INDICADOR_INSOLVENCIA	Flag representing whether the taxpayer has become insolvent or not: V – Yes; F – No.
INDICADOR_MOD3	Flag that represents whether the taxpayer has delivered Form 3: V – Yes; F – No.
INDICADOR_MOD3_PRAZO	Flag that represents whether the taxpayer has delivered Form 3 on time: V – Yes; F – No.
INDICADOR_SUJEITO_IVA	Flag representing whether the taxpayer is subject to the VAT regime: V – Yes; F – No.
INDICADOR_TRIB_AUTONOMA	Flag representing whether the taxpayer is subject to autonomous taxation: V – Yes; F – No.
INDICADOR_TRIB_CONJUNTA	Flag representing whether the taxpayer files his or her tax return jointly: V – Yes; F – No.
TIPO_NIF	Taxpayer type: S – Individual; C – Business.
TIPO_PERIODO_IVA	VAT Periodicity: M – Monthly; T – Quarterly.
VALOR_BASE_TRIB	Tax base amount (if any).
VALOR_DIVIDA_IRC	Amount of debt in IRC (if any).
VALOR_DIVIDA_IRS	Amount of debt in IRS (if any).
VALOR_DIVIDA_IVA	Amount of debt in VAT (if any).
VALOR_IRS_FINAL	Final IRS amount (if any).
VALOR_REND_BRUTO	Gross income amount (if any).
VALOR_REND_COLETAVEL	Taxable income amount (if any).
VALOR_VOLUME_NEG	Business volume value (if any).

Table 4.1 - Dataset Variables

4.1. ALGORITHM IMPLEMENTATION

4.1.1. Importation of the data

The implementation of the model starts with the load of an Excel file containing all the variables mentioned in a data frame for the taxpayers. The column “ID_CONTRIBUINTE” was defined as the index.

4.1.2. Data Understanding and Data Preparation

Despite our database having information for five years for each taxpayer, we decided to divide it between individuals and business, as these two groups have their own characteristics and only make sense to be applied to certain taxpayers. For example, the gender of a taxpayer is only applicable to individual taxpayers.

The second decision to be made was to characterize these two groups for two years: 2017 (the so-called "normal" year, before the pandemic) and 2021 (the "atypical" year, post-pandemic, and different rules). With this separation, our intention is to understand if there were any diverging points in the characterization of taxpayers between these two years, comparing, for example, the groups of individual/business taxpayers obtained in 2017 and the groups of individual/business taxpayers obtained in 2021.

After these divisions were made, we then started our analyses and preparation of the samples.

4.1.2.1. Univariate Analysis

In this chapter, we are going to present the frequency distribution for some of the variables. We also use the IQR method to detect outliers in the variables that seems to present false values in the first analysis of the count of distinct values that may exist for each variable.

Individual 2017

At this point, we will perform univariate analysis for individual taxpayers whose fiscal year was 2017, making a total of 71140 taxpayers from the initial sample.

Activity

For the variable that represents whether the taxpayers have open activity or not (INDICADOR_ATIVIDADE), as we can see in Figure 4.1, in our sample, the majority don’t have open activity (77.5%).

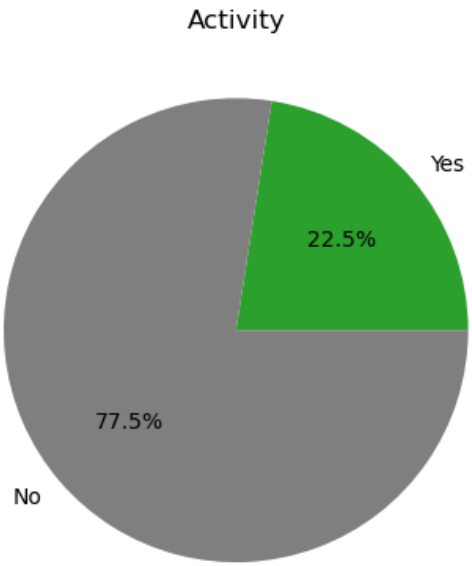

Figure 4.1- Univariate Analysis Activity of Individual taxpayers for 2017

Birth Year

The year range where most taxpayers are involved is between 1969 and 1979 (more than 14000 taxpayers), that contains ages between 48 and 58 (considering 2017 the last year that we are looking at). Two other intervals with high number of taxpayers are: [1959,1969] and [1979,1989].

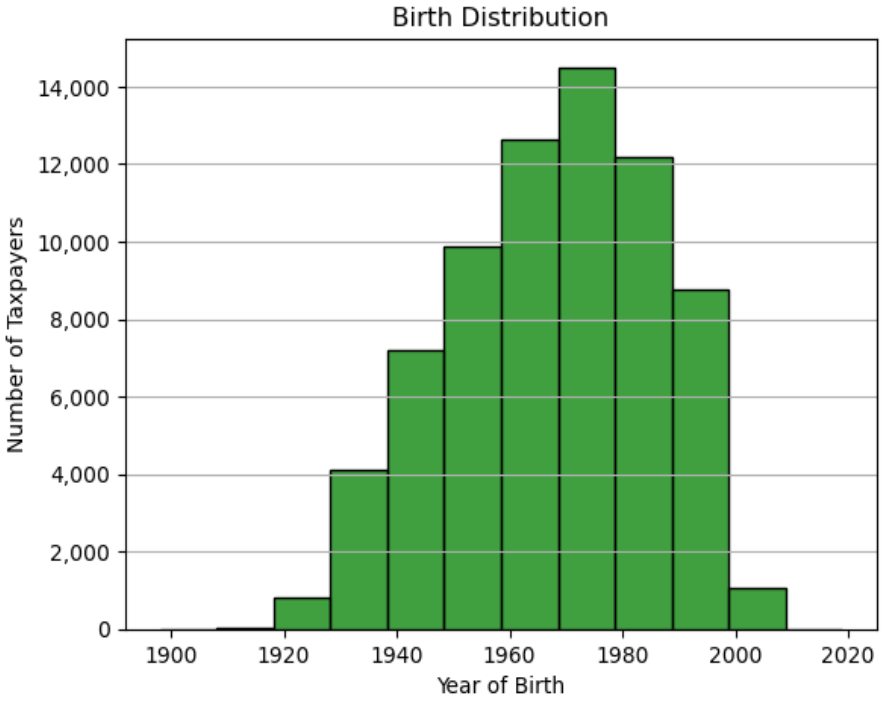

Figure 4.2 - Histogram of the Year of Birth of Individual taxpayers for 2017

We also created a Boxplot for this variable due to the outliers we identified after counting the different values, such as 1898. It does not make sense to consider a taxpayer who is 119 years old.

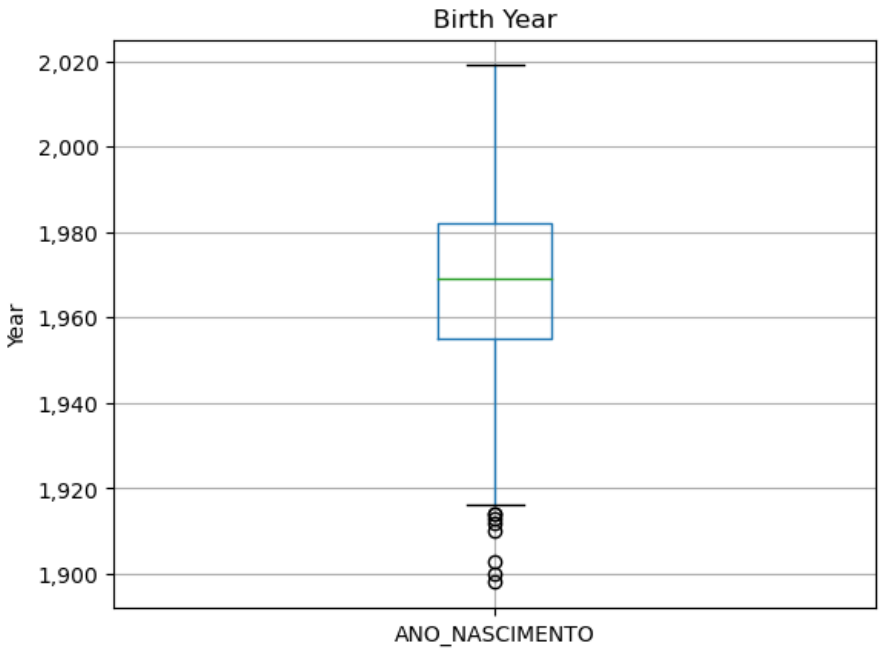

Figure 4.3 - Outlier Detection of the Year of Birth of Individual taxpayers for 2017

	Number of Outliers			**Valid Interval**	
Method	**Total Outliers**	**Lower Outliers**	**Upper Outliers**	**Lower Boundary**	**Upper Boundary**
IQR	8	8	0	1914.5	2022.5

Table 4.2 - – Interquartile Range Method for Outlier Detection of the Year of Birth

Marital Status

As for the Marital Status variable, the highest percentage is 43.6% related to taxpayers who are single, divorced or legally separated married taxpayers, but right after coming the married ones with 42.6%. On the other side, there are 6.6% widowed, 2.9% united in fact and 1.2% separated in fact. These three marital statuses don't even represent 11% of the sample.

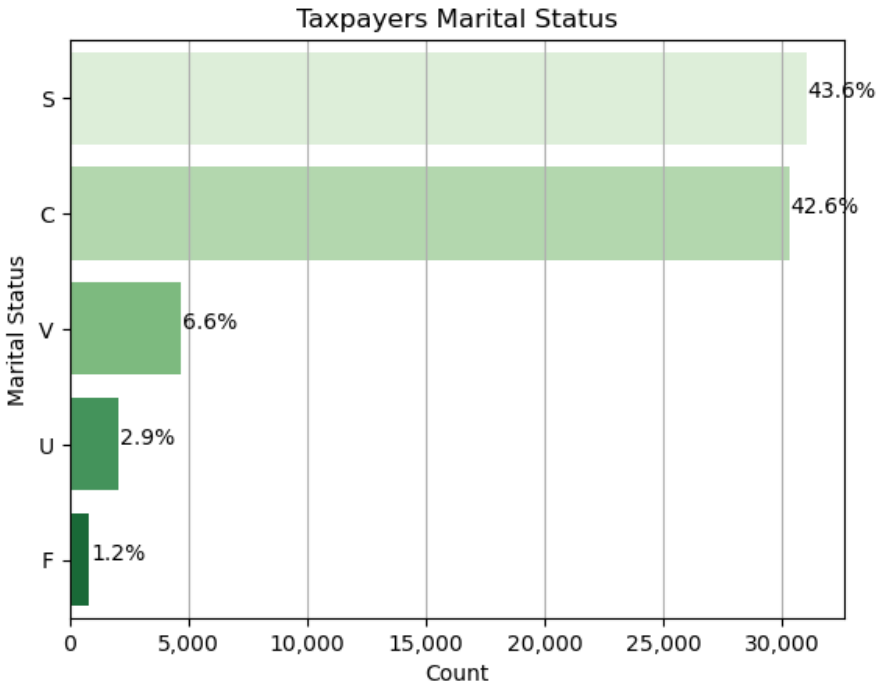

Figure 4.4 - Univariate Analysis Marital Status of Individual taxpayers for 2017

Tax Return Form 3

As we continue to study this sample, we also observe that 58721 taxpayers submitted Tax Return Form 3.

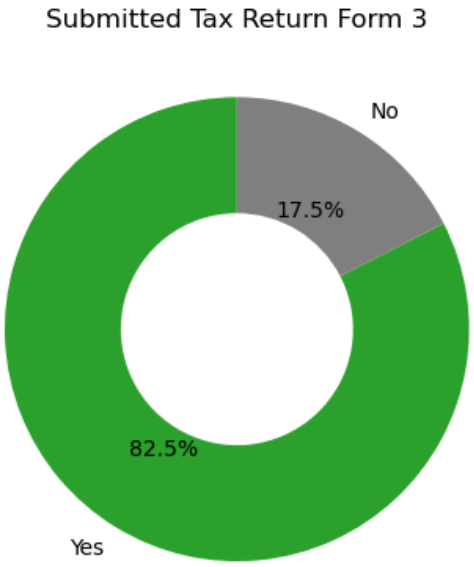

Figure 4.5 - Univariate Analysis Tax Return Form 3 of Individual taxpayers for 2017

VAT Period

Finally, the histogram below, indicates that taxpayers with an active business activity mostly opt for a quarterly VAT period, which is expected since individuals with business activities tend to have lower turnover.

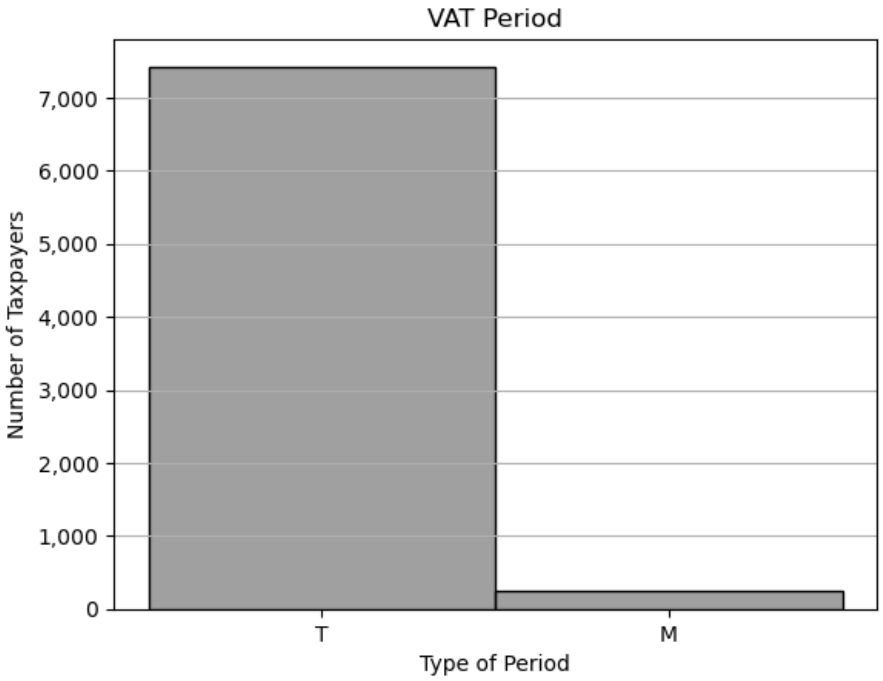

Figure 4.6 - Histogram of VAT Period of Individual taxpayers for 2017

Individual 2021

At this point, we will perform univariate analysis for individual taxpayers whose fiscal year was 2021, making a total of 71140 taxpayers from the initial sample.

Activity

For the variable that represents whether the taxpayers have open activity or not (INDICADOR_ATIVIDADE), despite being a lower percentage than in 2017, in this sample the majority don't have open activity to (76.9%).

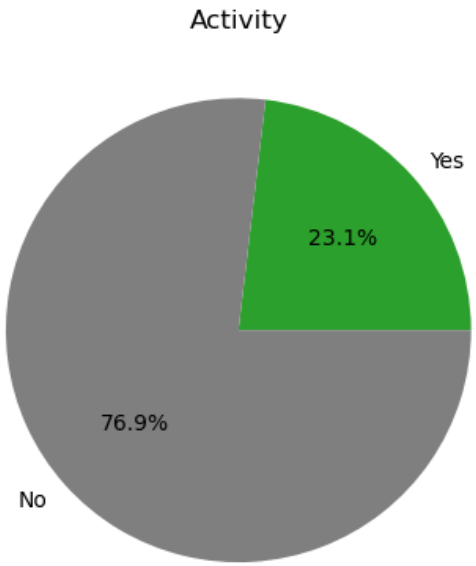

Figure 4.7 - Univariate Analysis Activity of Individual taxpayers for 2021

Birth Year

The analysis of the year of birth is the same as that of individual taxpayers in 2017 because when we constructed our initial sample, we extracted 5 years of information per taxpayer. For example, taxpayer 1 has a record for each year (2017, 2018, 2019, 2020, and 2021), but their year of birth remains the same.

We present the chart that demonstrate this:

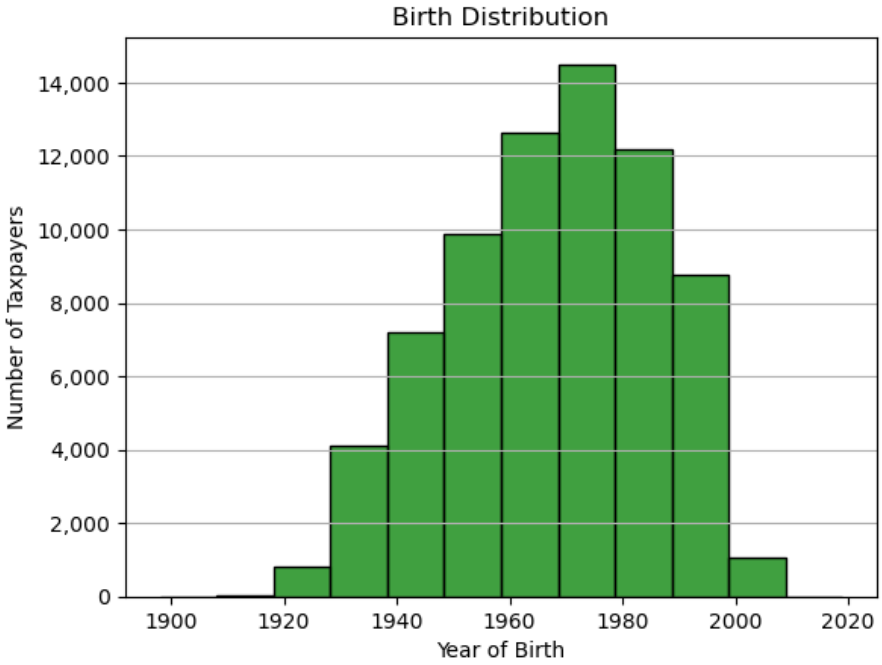

Figure 4.8 - Histogram of the Year of Birth of Individual taxpayers for 2021

The same applies to the boxplot that shows us outliers.

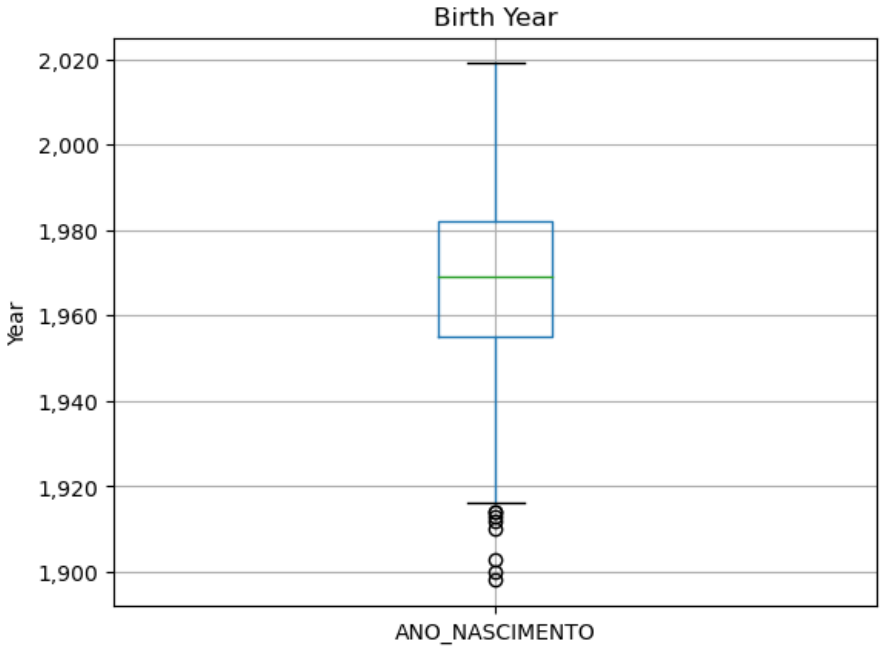

Figure 4.9 - Outlier Detection of the Year of Birth of Individual taxpayers for 2021

	Number of Outliers			**Valid Interval**	
Method	**Total Outliers**	**Lower Outliers**	**Upper Outliers**	**Lower Boundary**	**Upper Boundary**
IQR	8	8	0	1914.5	2022.5

Table 4.3– Interquartile Range Method for Outlier Detection of the Year of Birth

Marital Status

As for the Marital Status variable, the highest percentage is 46.8% related to taxpayers who are single, divorced or legally separated married taxpayers (a little more than 2017), and right after come the married ones to with 40.6%. The other three marital status represent 12.5% of the sample.

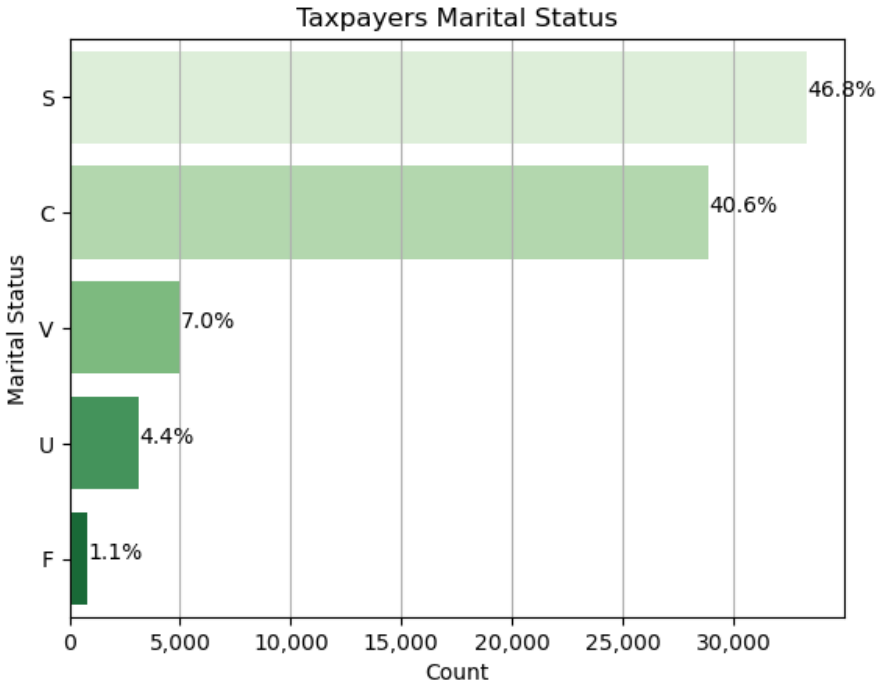

Figure 4.10 - Univariate Analysis Marital Status of Individual taxpayers for 2021

Tax Return Form 3

As we continue to study this sample, we also observe that 59786 taxpayers submitted Tax Return Form 3, more 1065 taxpayers than 2017.

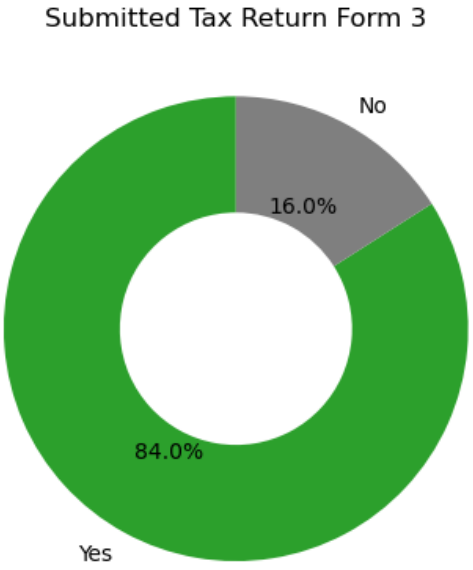

Figure 4.11 - Univariate Analysis Tax Return Form 3 of Individual taxpayers for 2021

VAT Period

Lastly, the histogram below, indicates that taxpayers with an active business activity mostly opt for a quarterly VAT period. It's a smaller number of taxpayers compared to 2017, but considering that there are also fewer individuals with business activity, it is normal.

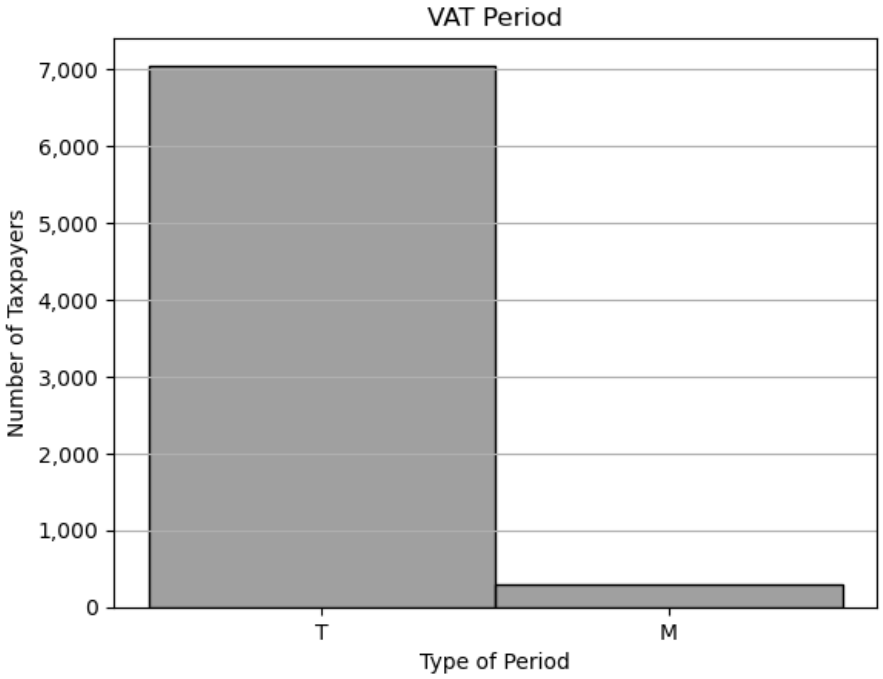

Figure 4.12 - - Histogram of VAT Period of Individual taxpayers for 2021

Business 2017

At this point, we will perform univariate analysis for business taxpayers whose fiscal year was 2017, making a total of 29923 taxpayers from the initial sample.

Activity

For the variable that represents whether the taxpayers have open activity or not (INDICADOR_ATIVIDADE), as we can see in Figure 4.13, in our sample, the majority have open activity (82.3%). What is expected, as we are analyzing a sample of business taxpayers.

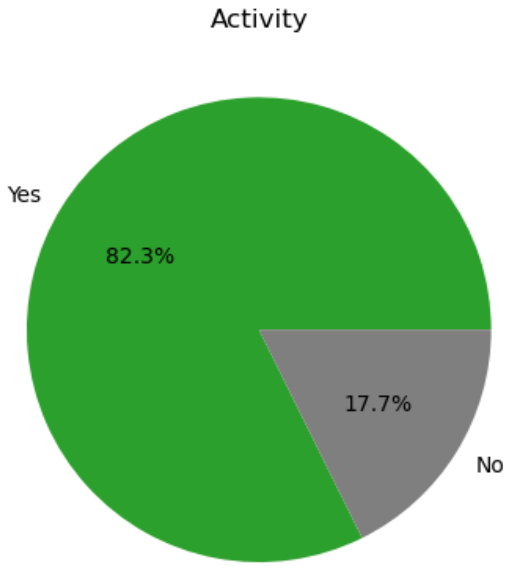

Figure 4.13 - Univariate Analysis Activity of Business taxpayers for 2017

Liable to VAT

In this donut chart we can observe that the majority of taxpayers are liable to VAT, which was expected since they are business entities.

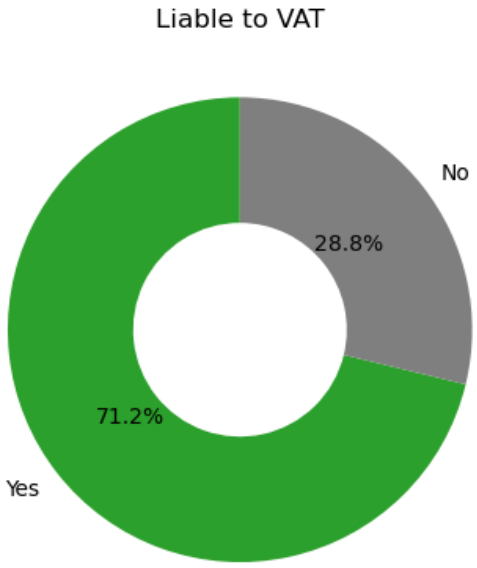

Figure 4.14 - Univariate Analysis Liable to VAT of Business taxpayers for 2017

VAT Period

Finally, the histogram below, indicates that taxpayers with an active business activity mostly opt for a quarterly VAT period. This indicates that perhaps the business taxpayers we are analyzing may also have lower turnover.

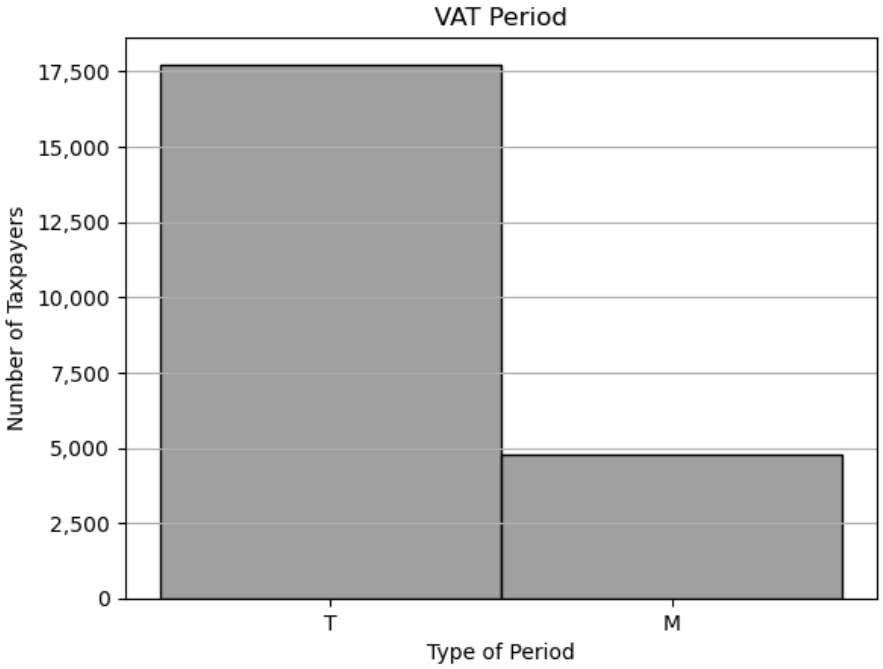

Figure 4.15 - Histogram of VAT Period of Business taxpayers for 2017

Business 2021

At this point, we will perform univariate analysis for business taxpayers whose fiscal year was 2021, making a total of 29923 taxpayers from the initial sample.

Activity

For the variable that represents whether the taxpayers have open activity or not (INDICADOR_ATIVIDADE), considering that the percentage has increased since 2017, in this sample the majority have open activity to (87.1%).

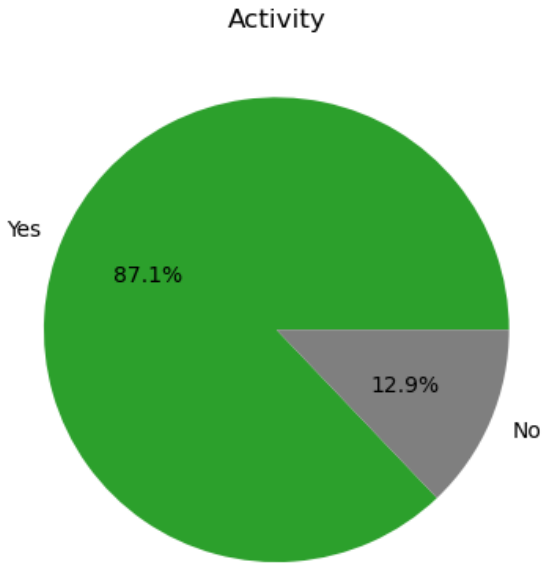

Figure 4.16 - Univariate Analysis Activity of Business taxpayers for 2021

Liable to VAT

In this donut chart, we coincidentally obtained the same percentage as in 2017, with the majority also being liable to VAT.

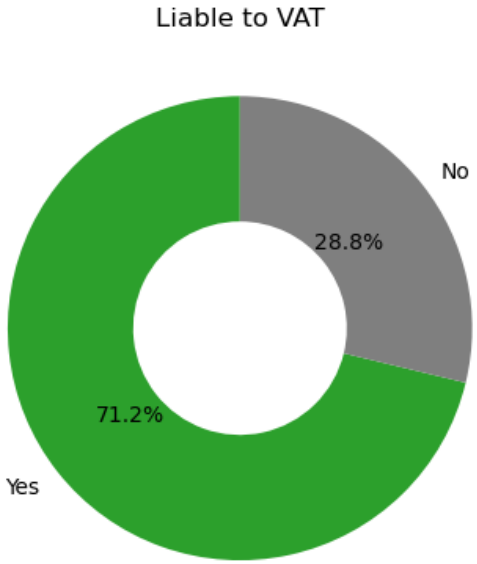

Figure 4.17 - Univariate Analysis Liable to VAT of Business taxpayers for 2021

VAT Period

Lastly, the histogram below, when compared to 2017, we observe that, although there are still more taxpayers choosing a quarterly VAT period, it's a lower number.

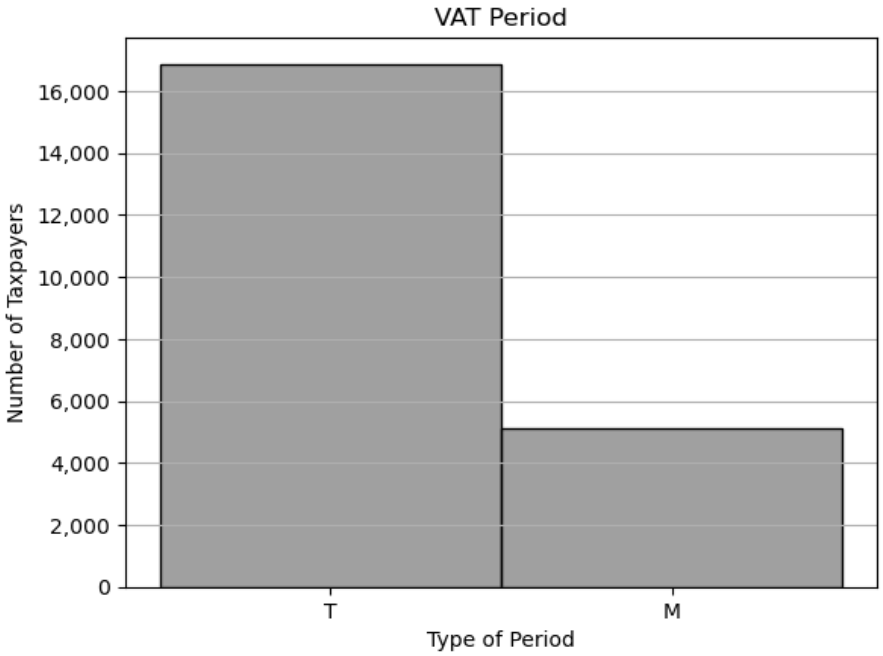

Figure 4.18 - Histogram of VAT Period of Business taxpayers for 2021

4.1.2.2. Bivariate Analysis

With the correlation matrix we see if there's any type of relation between two variables. Values nearing 1 indicates that there's a strong positive relation between those variables, the ones that are nearing -1 indicates strong negative relation between the two variables and the values that near 0 means that there is an absence of any kind of relationship between those variables.

Individual 2017

Looking at the variables that are included in the correlation matrix (see Appendix. Figure 4.31) we can see that there's a strong positive relation between some variables:

- INDICADOR_MOD3 and INDICADOR_MOD3_PRAZO (1), because the only difference between the variables is that one shows whether the taxpayer filed model 3 and the other, for those who filed model 3, whether they did on time;
- VALOR_REND_COLETAVEL and VALOR_REND_BRUTO (0.99), because both variables refer to the taxpayer's income.

It was considered the variables that the relation has correlation coefficient above the threshold of |x| > 0,7 resulting in multicollinearity, therefore we need to evaluate if any variable needs to be removed because each variable presents its own information.

Individual 2021

Observing the variables related to individual taxpayers from 2021 that are included in the correlation matrix (see Appendix. Figure 4.32) we can see that there's a strong positive relation between some variables:

- INDICADOR_ATIVIDADE and INDICADOR_SUJEITO_IVA (0.71), maybe because the taxpayers who have open activity are typically subject to VAT;
- INDICADOR_MOD3 and INDICADOR_MOD3_PRAZO (1), because the only difference between the variables is that one shows whether the taxpayer filed model 3 and the other, for those who filed model 3, whether they did on time;
- VALOR_REND_COLETAVEL and VALOR_REND_BRUTO (1), VALOR_REND_COLETAVEL and VALOR_IRS_FINAL (0.82), VALOR_IRS_FINAL and VALOR_REND_BRUTO (0.8), because all the variables refer to the taxpayer's income.

The relation between these variables has correlation coefficient above the threshold of |x| > 0,7 resulting in multicollinearity. For 2021, we obtain some different correlations from those of 2017.

Business 2017

Looking at the variables of business taxpayers for the fiscal year of 2017 (see Appendix. Figure 4.33) we can see that there's a strong positive relation between two variables:

- VALOR_VOLUME_NEG and VALOR_BASE_TRIB (0.96), maybe because those who have business will certainly have a VAT amount assigned.

Was considered the variables that the relation has correlation coefficient above the threshold of |x| > 0,7 resulting in multicollinearity, therefore we need to evaluate if any variable needs to be removed because each variable presents its own information.

Business 2021

Observing the correlation matrix (see Appendix. Figure 4.34), we can conclude that, similar to the matrix for business taxpayers for the fiscal year 2017, the only relationship that has correlation coefficient above the threshold of |x| > 0,7 is between VALOR_VOLUME_NEG and VALOR_BASE_TRIB (0.96).

4.1.2.3. Data Preparation

Individual 2017

After excluding the years from 2018 to 2021 and removing the business taxpayers, we started by creating the “IDADE” variable, subtracting the taxpayer's birth year from the current year (2017). Then we removed all taxpayers born before 1915 (following the boxplot) and after 2016 (since initially, the database covered the period from 2017 to 2021, it makes sense to remove taxpayers who are less than 1 year old). Next, we removed all variables that will not be used, such as “TIPO_NIF” as only individual taxpayers are being considered, “ANO_EXERCICIO” as only taxpayers from 2017 are being characterized, and “ANO_NASCIMENTO” as we created the age variable. We also removed variables that characterize business taxpayers, such as “VALOR_VOLUME_NEG”, “INDICADOR_INSOLVENCIA”, ”INDICADOR_DIV_IRC” e “VALOR_DIVIDA_IRC”.

The next step was to check for missing values. We found that there are 2122 taxpayers (2.98% of the sample) with missing marital status ("ESTADO_CIVIL”), which we decided to remove because marital status is an important characteristic for characterizing individual taxpayers. Additionally, there are 63455 taxpayers with the VAT period type (“TIPO_PERIODO_IVA”) missing, which is normal because only taxpayers with activity will have this field filled. In these cases, we filled the missing values with "Na" (Not Applicable). We also replaced commas with periods in the value fields to maintain the value format. Next, we defined the “ID_CONTRIBUINTE” as the index and began creating bins for the value variables (such as “VALOR_DIVIDA_IRS”, “VALOR_DIVIDA_IVA”, “VALOR_IRS_FINAL”, “VALOR_REND_COLETAVEL” and “VALOR_REND_BRUTO”) and age (“IDADE”) to facilitate the conversion of all categorical variables to dummy variables. We used one-hot encoding, where each column represents a category (depending on the variables) and has values of 0 or 1, with 0 indicating not belonging to that category and 1 indicating belonging (binary values). For the new bins variables and the remaining ones: “GENERO”, “ESTADO_CIVIL”, “DISTRITO”, “TIPO_PERIODO_IVA”, “CODIGO_ATIVIDADE”, “CODIGO_DIV_ATIVIDADE”, “CODIGO_IRS”, “INDICADOR_DIV_IRS”, “INDICADOR_DIV_IVA”, “INDICADOR_SUJEITO_IVA”, “INDICADOR_ATIVIDADE”, “INDICADOR_MOD3”, “INDICADOR_MOD3_PRAZO”, “INDICADOR_TRIB_CONJUNTA”, “INDICADOR_DEV_ESTRATEG”, “INDICADOR_TRIB_AUTONOMA”, we defined 7% as the minimum percentage of records that each category must have to be represented in a column. All categories that do not have at least 7% are considered "other". We chose 7% as the significant percentage because it is from this value that we have the breakdown of some variables that may be important in characterizing the taxpayers. Otherwise, the category would be only “<VARIABLE>_Other”. “CODIGO_IRS”, “CODIGO_ATIVIDADE” and “CODIGO_DIV_ATIVIDADE” were the three variables we decided to remove after splitting by categories because they did not present any "significant" categories.

The last step before applying the algorithm was to normalize our database to start modeling.

In the end, we obtained the following structure of variables in our sample of individual taxpayers with a fiscal year of 2017:

DISTRITO_11
DISTRITO_13
DISTRITO_15
DISTRITO_Other
ESTADO_CIVIL_C
ESTADO_CIVIL_Other
ESTADO_CIVIL_S
GENERO_F
GENERO_M
IDADE_BINS_<20
IDADE_BINS_>=92
IDADE_BINS_20-37
IDADE_BINS_38-55
IDADE_BINS_56-73
IDADE_BINS_74-91
INDICADOR_ATIVIDADE_0
INDICADOR_ATIVIDADE_1
INDICADOR_DEV_ESTRATEG_0
INDICADOR_DEV_ESTRATEG_1
INDICADOR_DIV_IRS_0
INDICADOR_DIV_IRS_1
INDICADOR_DIV_IVA_0
INDICADOR_DIV_IVA_1
INDICADOR_MOD3_0
INDICADOR_MOD3_1
INDICADOR_MOD3_PRAZO_0
INDICADOR_MOD3_PRAZO_1
INDICADOR_SUJEITO_IVA_0
INDICADOR_SUJEITO_IVA_1
INDICADOR_TRIB_AUTONOMA_0
INDICADOR_TRIB_AUTONOMA_1
INDICADOR_TRIB_CONJUNTA_0
INDICADOR_TRIB_CONJUNTA_1
TIPO_PERIODO_IVA_M
TIPO_PERIODO_IVA_Na
TIPO_PERIODO_IVA_T
V_BASE_TRIB_BINS_<392054428
V_BASE_TRIB_BINS_>=784108854.1
V_DIVIDA_IRS_BINS_<110680
V_DIVIDA_IRS_BINS_>=221358.1
V_DIVIDA_IRS_BINS_110680-221358

V_DIVIDA_IVA_BINS_<74115.42
V_DIVIDA_IVA_BINS_>=148230.83
V_DIVIDA_IVA_BINS_74115.42-148230.82
V_IRS_FINAL_BINS_<528121.18
V_IRS_FINAL_BINS_>=1092536.72
V_REND_BRUTO_BINS_<971921.87
V_REND_BRUTO_BINS_>=1943843.73
V_REND_BRUTO_BINS_971921.87-1943843.72
V_REND_COLETAVEL_BINS_<971426.76
V_REND_COLETAVEL_BINS_>=1942853.6
V_REND_COLETAVEL_BINS_971426.76-1942853.5

Table 4.4 - Individual taxpayers of 2017 variables

Individual 2021

For individual taxpayers with a fiscal year of 2021, the same data treatment was performed as for individual taxpayers with a fiscal year of 2017, except for excluding the years from 2017 to 2020. When creating the "IDADE" variable, we subtracted the taxpayer's birth year from the current year (2021).

In the end, we ended up with the following structure of variables in our sample of individual for 2021:

DISTRITO_11
DISTRITO_13
DISTRITO_15
DISTRITO_Other
ESTADO_CIVIL_C
ESTADO_CIVIL_Other
ESTADO_CIVIL_S
ESTADO_CIVIL_V
GENERO_F
GENERO_M
IDADE_BINS_<20
IDADE_BINS_>=92
IDADE_BINS_20-37
IDADE_BINS_38-55
IDADE_BINS_56-73
IDADE_BINS_74-91
INDICADOR_ATIVIDADE_0
INDICADOR_ATIVIDADE_1
INDICADOR_DEV_ESTRATEG_0
INDICADOR_DEV_ESTRATEG_1

INDICADOR_DIV_IRS_0
INDICADOR_DIV_IRS_1
INDICADOR_DIV_IVA_0
INDICADOR_DIV_IVA_1
INDICADOR_MOD3_0
INDICADOR_MOD3_1
INDICADOR_MOD3_PRAZO_0
INDICADOR_MOD3_PRAZO_1
INDICADOR_SUJEITO_IVA_0
INDICADOR_SUJEITO_IVA_1
INDICADOR_TRIB_AUTONOMA_0
INDICADOR_TRIB_AUTONOMA_1
INDICADOR_TRIB_CONJUNTA_0
INDICADOR_TRIB_CONJUNTA_1
TIPO_PERIODO_IVA_M
TIPO_PERIODO_IVA_Na
TIPO_PERIODO_IVA_T
V_BASE_TRIB_BINS_<4065040.66
V_BASE_TRIB_BINS_>=8130081.31
V_BASE_TRIB_BINS_4065040.66-8130081.3
V_DIVIDA_IRS_BINS_<164305.28
V_DIVIDA_IRS_BINS_>=328610.55
V_DIVIDA_IRS_BINS_164305.28-328610.54
V_DIVIDA_IVA_BINS_<34114.6
V_DIVIDA_IVA_BINS_>=68229.1
V_DIVIDA_IVA_BINS_34114.6-68229
V_IRS_FINAL_BINS_<924911.82
V_IRS_FINAL_BINS_>=1901051.1
V_REND_BRUTO_BINS_<1841402.21
V_REND_BRUTO_BINS_>=3682804.42
V_REND_COLETAVEL_BINS_<1840034.21
V_REND_COLETAVEL_BINS_>=3680068.42

Table 4.5 - Individual taxpayers of 2021 variables

Business 2017

After excluding the years from 2018 to 2021 and excluding individual taxpayers, we began by removing all variables that will not be used because they are unnecessary, such as “TIPO_NIF”, since we are only considering business taxpayers, and “ANO_EXERCICIO”, as we are only characterizing taxpayers from 2017, or because they are variables that characterize individual taxpayers, such as: “ANO_NASCIMENTO”, “GENERO”, “ESTADO_CIVIL”, “CODIGO_IRS”, “VALOR_REND_BRUTO”, “VALOR_IRS_FINAL”, “VALOR_REND_COLETAVEL”, “INDICADOR_MOD3”, “INDICADOR_MOD3_PRAZO”, “INDICADOR_TRIB_CONJUNTA”, “INDICADOR_TRIB_AUTONOMA”, “INDICADOR_DIV_IRS”, and “VALOR_DIVIDA_IRS”.

The next step was to check for missing values, from which we found 1 taxpayer with the district as null, which we decided to leave to see how the algorithm behaves, and 7414 taxpayers with the VAT period type to be filled, which is normal because there are taxpayers with activities that are not subject to VAT, and in these cases, we fill the nulls with “Na” (Not applicable).

Next, we replaced commas with periods in the value fields to maintain the value format. We then defined “ID_CONTRIBUINTE” as the index and started creating bins for the value variables (such as “VALOR_DIVIDA_IRC”, “VALOR_DIVIDA_IVA”, “VALOR_BASE_TRIB”, and “VALOR_VOLUME_NEG”) to facilitate the conversion of all categorical variables to dummy variables. We also used one hot encoding (the same method we used for individual taxpayers), where each column represents a category (according to the variables) and has values of 0 or 1, with 0 indicating it does not belong to that category and 1 indicating it does (binary values) for the new bins variables and remaining variables: “DISTRITO”, “TIPO_PERIODO_IVA”, “CODIGO_ATIVIDADE”, “CODIGO_DIV_ATIVIDADE”, “INDICADOR_DIV_IRC”, “INDICADOR_DIV_IVA”, “INDICADOR_SUJEITO_IVA”, “INDICADOR_ATIVIDADE”, “INDICADOR_DEV_ESTRATEG”, “INDICADOR_INSOLVENCIA”. We defined 7% as the minimum percentage of records that each category must contain to be represented in a column. All categories that do not have at least 7% are considered “other”. We chose 7% as the significant percentage because it is from this value that we have the breakdown of some variables that may be important in characterizing taxpayers. If not, the category would only be “(VARIABLE)_Other”. For example, since the “CODIGO_ATIVIDADE” does not present any “significant” category, we decided to eliminate the variable.

The last step before applying the algorithm was to normalize our database to then start modeling.

In the end, we obtained the following variable structure in our sample of business taxpayers for the exercise year of 2017:

CODIGO_DIV_ATIVIDADE_0
CODIGO_DIV_ATIVIDADE_47
CODIGO_DIV_ATIVIDADE_56
CODIGO_DIV_ATIVIDADE_Other
DISTRITO_11
DISTRITO_13
DISTRITO_3
DISTRITO_Other
INDICADOR_ATIVIDADE_0
INDICADOR_ATIVIDADE_1
INDICADOR_DEV_ESTRATEG_0
INDICADOR_DEV_ESTRATEG_1
INDICADOR_DIV_IRC_0
INDICADOR_DIV_IRC_1

INDICADOR_DIV_IVA_0
INDICADOR_DIV_IVA_1
INDICADOR_INSOLVENCIA_0
INDICADOR_INSOLVENCIA_1
INDICADOR_SUJEITO_IVA_0
INDICADOR_SUJEITO_IVA_1
TIPO_PERIODO_IVA_M
TIPO_PERIODO_IVA_Na
TIPO_PERIODO_IVA_T
V_BASE_TRIB_BINS_<1319063231
V_BASE_TRIB_BINS_>=2638126471
V_BASE_TRIB_BINS_1319063231-2638126470
V_DIVIDA_IRC_BINS_<1241666.37
V_DIVIDA_IRC_BINS_>=2483332.73
V_DIVIDA_IRC_BINS_1241666.37-2483332.72
V_DIVIDA_IVA_BINS_<2559061.32
V_DIVIDA_IVA_BINS_>=5118122.63
V_DIVIDA_IVA_BINS_2559061.32-5118122.62
V_VOLUME_NEG_BINS_<950591016
V_VOLUME_NEG_BINS_>=1901182031
V_VOLUME_NEG_BINS_950591016-1901182030

Table 4.6 - Business taxpayers of 2017 variables

Business 2021

For business taxpayers with a fiscal year of 2021, the same data treatment was performed as for business taxpayers with a fiscal year of 2017, except for excluding the years from 2017 to 2020.

In the end, we ended up with the following structure of variables in our sample of individual for 2021:

CODIGO_DIV_ATIVIDADE_0
CODIGO_DIV_ATIVIDADE_47
CODIGO_DIV_ATIVIDADE_56
CODIGO_DIV_ATIVIDADE_Other
DISTRITO_11
DISTRITO_13
DISTRITO_3
DISTRITO_Other
INDICADOR_ATIVIDADE_0
INDICADOR_ATIVIDADE_1
INDICADOR_DEV_ESTRATEG_0

INDICADOR_DEV_ESTRATEG_1
INDICADOR_DIV_IRC_0
INDICADOR_DIV_IRC_1
INDICADOR_DIV_IVA_0
INDICADOR_DIV_IVA_1
INDICADOR_INSOLVENCIA_0
INDICADOR_INSOLVENCIA_1
INDICADOR_SUJEITO_IVA_0
INDICADOR_SUJEITO_IVA_1
TIPO_PERIODO_IVA_M
TIPO_PERIODO_IVA_Na
TIPO_PERIODO_IVA_T
V_BASE_TRIB_BINS_<1201446680.1
V_BASE_TRIB_BINS_>=2402893360.1
V_BASE_TRIB_BINS_1201446680.1-2402893360
V_DIVIDA_IRC_BINS_<3657137.48
V_DIVIDA_IRC_BINS_>=7314274.95
V_DIVIDA_IVA_BINS_<662301.1
V_DIVIDA_IVA_BINS_>=1324602.1
V_DIVIDA_IVA_BINS_662301.1-1324602
V_VOLUME_NEG_BINS_<788982540.1
V_VOLUME_NEG_BINS_>=1577965080.1
V_VOLUME_NEG_BINS_788982540.1-1577965080

Table 4.7 - Business taxpayers of 2021 variables

4.1.3. Modeling

Individual 2017

We started with the PCA algorithm, that it is an algorithm that groups comparable objects into groups named clusters.

From the variance per component and the cumulative explained variance we can assume that 96.1% of the variance can be explained with fourteen components.

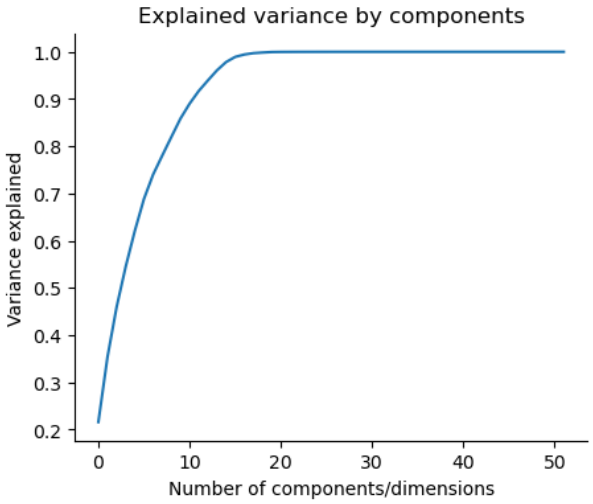

Figure 4.19 - Plot the cumulative explained variance for Individual taxpayers of 2017

The fourteen components will be used to the dimension reduction of the dataset for the K-Means Modeling.

The K-means is clustering algorithm used to discover clusters that were not explicitly labelled in the data. To define the number of clusters we used:

- The Elbow Method, which indicated four (see Appendix, Figure 4.35);
- The Silhouette Method, which indicated four too (see Appendix, Figure 4.36);
- And the Davies-Boulding Index, which indicated four too (see Appendix, Figure 4.37).

Observing the graphics of the methods we select K=4 (four clusters) and we test the cardinality and the magnitude where we pulled out that cardinality and magnitude are correlated, so no major anomalies seem to exist in clusters. The plot bellow can prove:

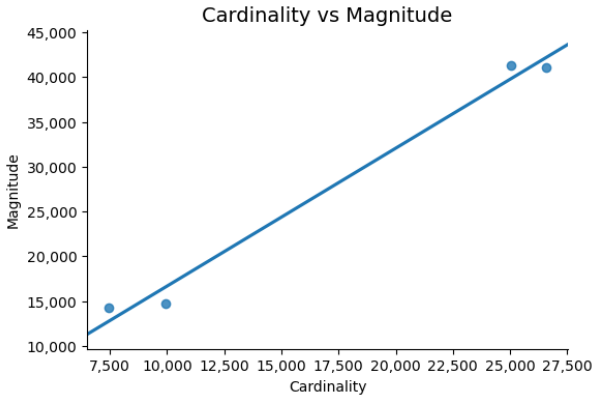

Figure 4.20 - Plot cardinality vs magnitude

After selecting the number of clusters, we graphically represented them with a Scatter plot of the two Principal Components by cluster with dimension reduction, to give us a visual understanding of their representation.

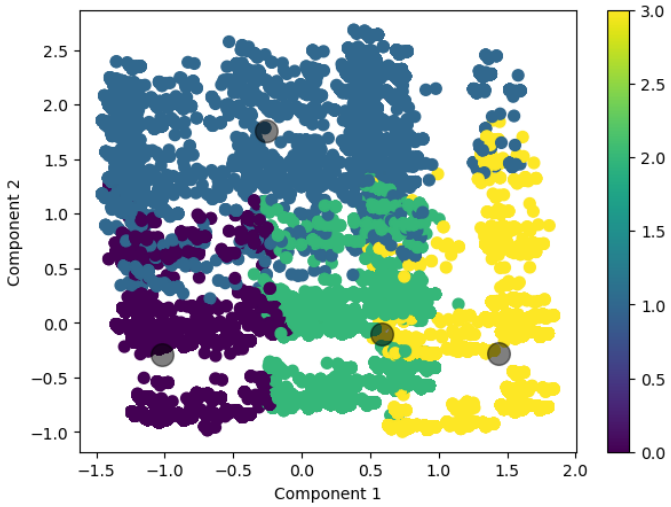

Figure 4.21 - Scatter plot of the two Principal Components by cluster with dimension reduction for Individual taxpayers of 2017

Before analyzing the variables within each cluster, we calculated their weights for component 1, from which we obtained 52 important features. These features were then used to characterize our clusters and understand their composition.

Individual 2021

We started with the PCA algorithm as well.

From the variance per component and the cumulative explained variance we can assume that 96.6% of the variance can be explained with fifteen components.

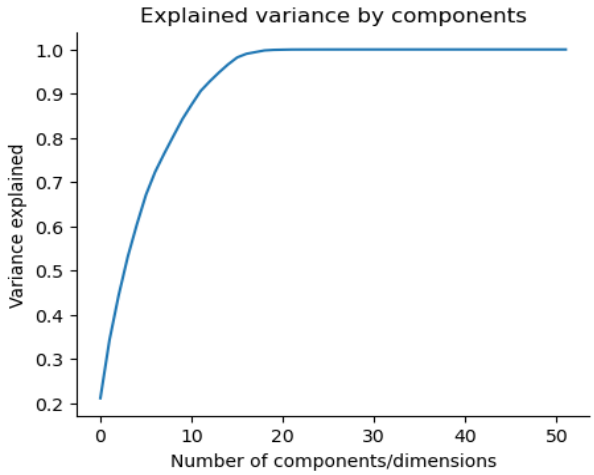

Figure 4.22 - Plot the cumulative explained variance for Individual taxpayers of 2021

The fifteen components will be used to the dimension reduction of the dataset for the K-Means Modeling. To define the number of clusters (the K) we used to:

- The Elbow Method, which indicated four (see Appendix, Figure 4.38);
- he Silhouette Method, which indicated four too (see Appendix, Figure 4.39);
- And the Davies-Boulding Index, which indicated five (see Appendix, Figure 4.40).

Observing the graphs, we can notice some discrepancy between the Elbow Method and Silhouette Method compared to the Davies-Bouldin Method, and after testing and visualizing, we decided to proceed with K=4. We also tested the cardinality and the magnitude, and they are correlated, so no major anomalies seem to exist in clusters. The plot bellow can show:

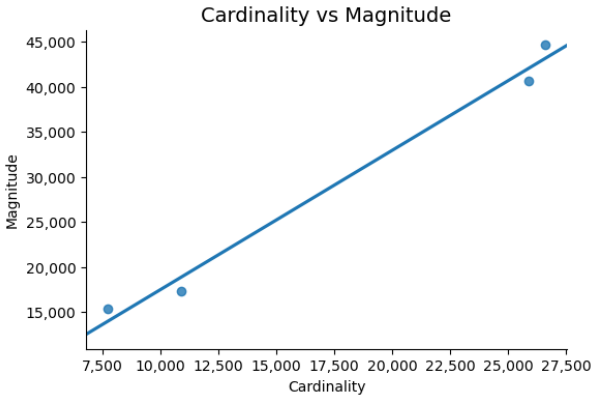

Figure 4.23 - Plot cardinality vs magnitude

After selecting the number of clusters, we graphically represented them with a Scatter plot of the two Principal Components to give us a visual understanding of the cluster presentation.

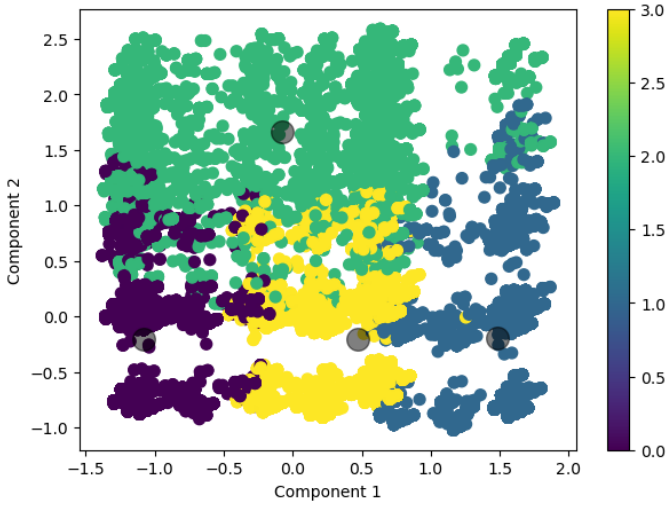

Figure 4.24 - Scatter plot of the two Principal Components by cluster with dimension reduction for Individual taxpayers of 2021

Before analyzing the variables within each cluster, we calculated their weights for component 1, from which we obtained 52 important features. These features were then used to characterize our clusters and understand their composition.

Business 2017

For the analysis of the business taxpayers, we started with the PCA algorithm too, that it is an algorithm that groups comparable objects into groups named clusters.

From the variance per component and the cumulative explained variance we can assume that 97% of the variance can be explained with ten components.

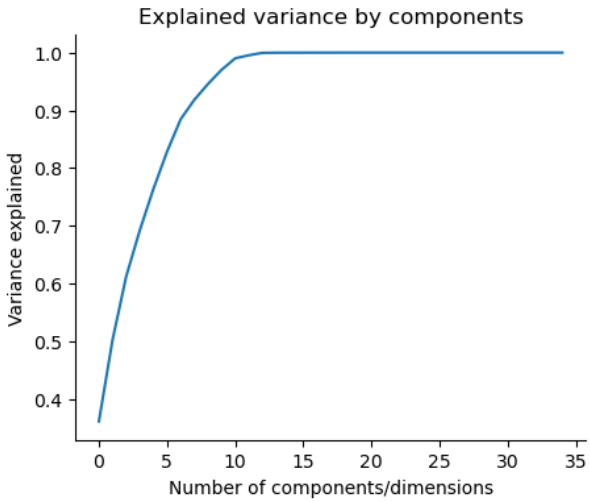

Figure 4.25 - Plot the cumulative explained variance for Business taxpayers of 2017

The ten components will be used to the dimension reduction of the dataset for the K-Means Modeling.

The K-means is clustering algorithm used to discover clusters that were not explicitly labelled in the data. To define the number of clusters we used:

- The Elbow Method, which indicated about seven (see Appendix, Figure 4.41);
- The Silhouette Method, which seems to indicate two (see Appendix, Figure 4.42);
- And the Davies-Boulding Index, which indicated two too (see Appendix, Figure 4.43).

Observing the graphs, we can notice some discrepancy between the Elbow Method compared to Silhouette Method and the Davies-Bouldin Method, and after testing and visualizing, we decided to proceed with K=2. We also tested the cardinality and the magnitude, and they are correlated, so no major anomalies seem to exist in clusters. The plot bellow can show:

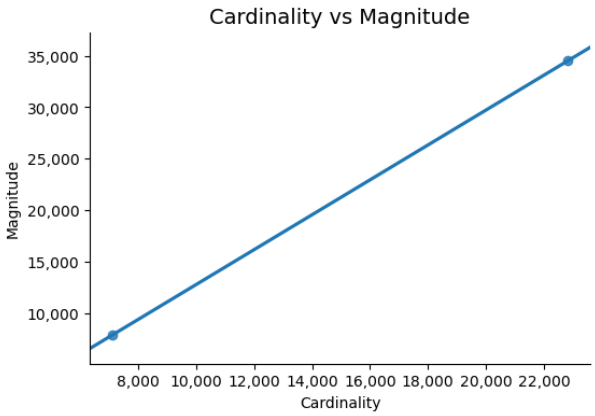

Figure 4.26 - Plot cardinality vs magnitude

After selecting the number of clusters, we graphically represented them with a Scatter plot of the two Principal Components by cluster with dimension reduction, to give us a visual understanding of their representation.

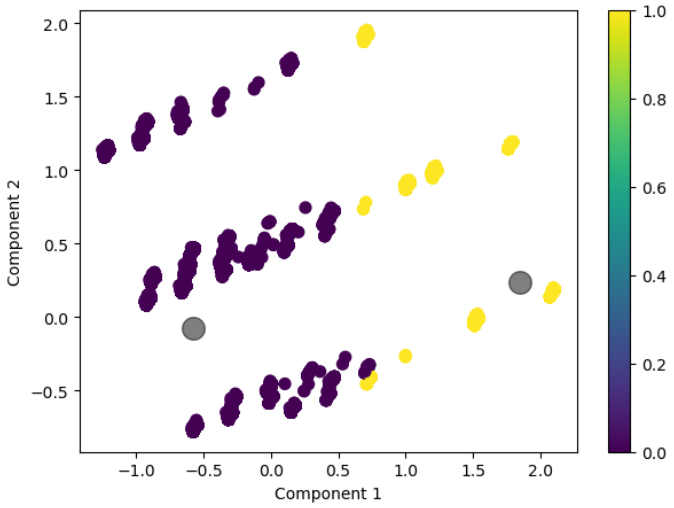

Figure 4.27 - Scatter plot of the two Principal Components by cluster with dimension reduction for Business taxpayers of 2017

Before analyzing the variables within each cluster, we calculated their weights for component 1, from which we obtained 35 important features which will help to understand clusters composition.

Business 2021

This analysis will be quite similar to that of business taxpayers for the fiscal year of 2017, we used the PCA algorithm too. From the variance per component and the cumulative explained variance we can assume that approximately 96.2% of the variance can be explained with ten components.

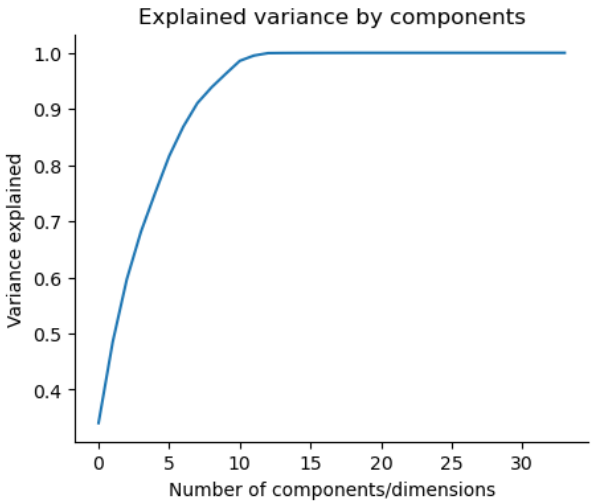

Figure 4.28 - Plot the cumulative explained variance for Business taxpayers of 2021

The ten components will be used to the dimension reduction of the dataset for the K-Means Modeling. To define the number of clusters we used:

- The Elbow Method, which indicated about four (see Appendix, Figure 4.44);
- The Silhouette Method, which seems to indicate two or nine (see Appendix, Figure 4.45);
- And the Davies-Boulding Index, which indicated two too (see Appendix, Figure 4.46).

Observing the graphs, we can notice some discrepancy between all of them, considering that the only agreement they show is for K=2 in the Silhouette Method and the Davies-Bouldin Method, and after testing and visualizing, we decided to proceed with K=2, too. We tested the cardinality and the magnitude, and they are correlated, so no major anomalies seem to exist in clusters. The next plot can show:

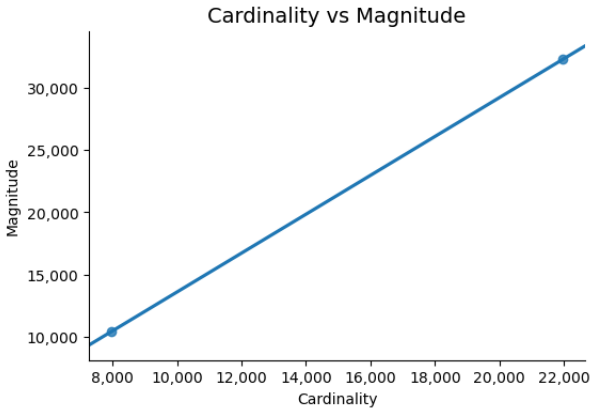

Figure 4.29 - Plot cardinality vs magnitude

Subsequently selecting the number of clusters, we graphically represented them with a Scatter plot of the two Principal Components by cluster with dimension reduction, to give us a visual understanding of their representation.

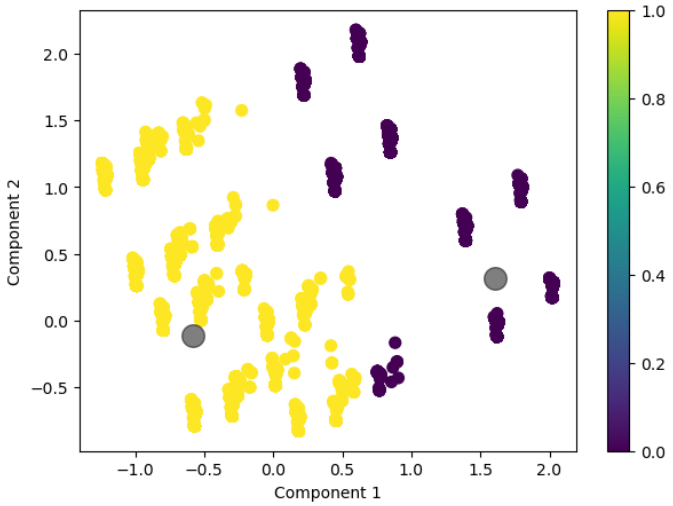

Figure 4.30 - Scatter plot of the two Principal Components by cluster with dimension reduction for Business taxpayers of 2021

Ahead of analyzing the variables within each cluster, we calculated their weights for component 1, from which we obtained 34 important features which will help to understand the characterization of the clusters and their variable composition.

4.1.4. Clusters Interpretation

In this subsection, we will interpret the clusters formed by the algorithm for the two types of taxpayers over the two years.

Individual 2017

Variables	Cluster 1	Cluster 2	Cluster 3	Cluster 4
DISTRITO_11	0.205828	0.229803	0.260031	0.265960
DISTRITO_13	0.174022	0.122604	0.128786	0.152718
DISTRITO_15	0.079920	0.083448	0.088662	0.095007
DISTRITO_Other	0.540230	0.564145	0.522521	0.486315
ESTADO_CIVIL_C	0.878729	0.089684	0.393347	0.079818
ESTADO_CIVIL_Other	0.095339	0.011554	0.062516	0.031807
ESTADO_CIVIL_S	0.025120	0.859972	0.509837	0.727085
ESTADO_CIVIL_V	0.000812	0.038790	0.034300	0.161290
GENERO_F	0.475962	0.431453	0.311416	0.527521

Variables	Cluster 1	Cluster 2	Cluster 3	Cluster 4
GENERO_M	0.524038	0.568547	0.688584	0.472479
IDADE_BINS_<20	0.000000	0.002201	0.000388	0.003647
IDADE_BINS_>=92	0.002589	0.059606	0.001035	0.014061
IDADE_BINS_20-37	0.117638	0.204860	0.222366	0.311527
IDADE_BINS_38-55	0.391482	0.290967	0.459099	0.308933
IDADE_BINS_56-73	0.350634	0.218982	0.266632	0.221483
IDADE_BINS_74-91	0.137657	0.223384	0.050479	0.140349
INDICADOR_ATIVIDADE_0	0.821418	0.846768	0.214600	0.845853
INDICADOR_ATIVIDADE_1	0.178582	0.153232	0.785400	0.154147
INDICADOR_DEV_ESTRATEG_0	0.999845	0.998533	0.997152	0.999286
INDICADOR_DEV_ESTRATEG_1	0.000155	0.001467	0.002848	0.000714
INDICADOR_DIV_IRS_0	0.976349	0.898212	0.771292	0.882059
INDICADOR_DIV_IRS_1	0.023651	0.101788	0.228708	0.117941
INDICADOR_DIV_IVA_0	0.994937	0.940303	0.541936	0.991766
INDICADOR_DIV_IVA_1	0.005063	0.059697	0.458064	0.008234
INDICADOR_MOD3_0	0.000000	1.000000	0.054232	0.000000
INDICADOR_MOD3_1	1.000000	0.000000	0.945768	1.000000
INDICADOR_MOD3_PRAZO_0	0.000039	1.000000	0.054362	0.000038
INDICADOR_MOD3_PRAZO_1	0.999961	0.000000	0.945638	0.999962
INDICADOR_SUJEITO_IVA_0	0.994474	0.939844	0.028734	0.991541
INDICADOR_SUJEITO_IVA_1	0.005526	0.060156	0.971266	0.008459
INDICADOR_TRIB_AUTONOMA_0	0.889666	0.999908	0.891017	0.917513
INDICADOR_TRIB_AUTONOMA_1	0.110334	0.000092	0.108983	0.082487
INDICADOR_TRIB_CONJUNTA_0	0.000000	1.000000	0.596687	0.986841
INDICADOR_TRIB_CONJUNTA_1	1.000000	0.000000	0.403313	0.013159
TIPO_PERIODO_IVA_M	0.000348	0.000092	0.034170	0.000526
TIPO_PERIODO_IVA_Na	0.991575	0.989638	0.109759	0.995338
TIPO_PERIODO_IVA_T	0.008077	0.010271	0.856070	0.004136
V_BASE_TRIB_BINS_<4065040.66	1.000000	1.000000	0.999353	1.000000
V_BASE_TRIB_BINS_>=8130081.31	0.000000	0.000000	0.000259	0.000000
V_BASE_TRIB_BINS_4065040.66-8130081.3	0.000000	0.000000	0.000388	0.000000
V_DIVIDA_IRS_BINS_<164305.28	0.999961	0.999725	1.000000	0.999850
V_DIVIDA_IRS_BINS_>=328610.55	0.000039	0.000000	0.000000	0.000075
V_DIVIDA_IRS_BINS_164305.28-328610.54	0.000000	0.000275	0.000000	0.000075
V_DIVIDA_IVA_BINS_<34114.6	1.000000	0.999908	0.998965	1.000000
V_DIVIDA_IVA_BINS_>=68229.1	0.000000	0.000000	0.000388	0.000000
V_DIVIDA_IVA_BINS_34114.6-68229	0.000000	0.000092	0.000647	0.000000
V_IRS_FINAL_BINS_<924911.82	1.000000	1.000000	0.999871	1.000000
V_IRS_FINAL_BINS_>=1901051.1	0.000000	0.000000	0.000129	0.000000
V_REND_BRUTO_BINS_<1841402.21	1.000000	1.000000	0.999871	1.000000
V_REND_BRUTO_BINS_>=3682804.42	0.000000	0.000000	0.000129	0.000000
V_REND_COLETAVEL_BINS_<1840034.21	1.000000	1.000000	0.999871	1.000000
V_REND_COLETAVEL_BINS_>=3680068.42	0.000000	0.000000	0.000129	0.000000

Table 4.8 - K-means Individual taxpayers for 2017 Value Cluster

Cluster 1: Majority married and totally filed joint taxation

- Over 87% of taxpayers are married and 100% filed joint taxation (highlight characteristics of the cluster);
- All taxpayers submitted IRS (filed form 3), have a taxable base value below 4,065,040.66€, have an VAT debt value below than 34,114.6€, have a final IRS value less than 924,911.82€, gross income value below 1,841,402.21€, and taxable income value below than 1,840,034.21€;
- It is the cluster (of the four) with the highest percentage of taxpayers who do not have VAT debts (99.4%) and IRS debts (97.6%). It also has the highest percentage of taxpayers who are not strategic debtors (99.98%) and taxpayers who are not subject to VAT (99.4%).

Cluster 2: Didn't submit form 3

- 100% did not submit IRS (did not file form 3, meaning that the indicator for on-time submission will also be 100% for submissions delivered late.
- All taxpayers have a taxable base value below than 4,065,040.66€, have a final IRS value less than 924,911.82€, gross income value below than 1,841,402.21€, and taxable income value below than 1,840,034.21€;
- Cluster with the highest percentage of taxpayers who are single, divorced, or legally separated (approximately 86%);
- Cluster with the highest percentage of taxpayers aged between 74 and 91 years (22.3%) and aged 92 years or older (about 6%).

Cluster 3: Majority have activity, subject to VAT, and opted for a quarterly VAT period

- Over 78.5% of taxpayers have activity, 97.1% are subject to VAT and 85.6% opted for a quarterly VAT period (highlight characteristics of the cluster);
- It is the only cluster that presents a percentage of taxpayers (although small, between 0.01% and 0.03%) for the variables: taxable base value greater than or equal to 8,130,081.31€ and between 4,065,040.66€ and 8,130,081.3€, VAT debt value greater than or equal to 68,229.1€, final IRS value greater than or equal to 1,901,051.1€, gross income value greater than or equal to 3,682,804.42€, and taxable income value greater than or equal to 3,680,068.42€;
- Cluster with the highest percentage of strategic debtors (although small, about 0.28%) and taxpayers with IRS and VAT debts (22.8% and 45.8%, respectively);
- Cluster with the highest percentage of taxpayers aged between 38 and 55 years (45.9%) and male gender (68.9%).

Cluster 4: Mix Cluster

- There is no characteristic that is solely distinctive of this cluster;
- All taxpayers submitted IRS (filed form 3), have a taxable base value below than 4,065,040.66€, have an VAT debt value below than 34,114.6€, have a final IRS value less than 924,911.82€, gross income value below than 1,841,402.21€, and taxable income value below than 1,840,034.21€;
- Cluster with the biggest percentage of female taxpayers (52.8%);
- Cluster with the highest percentage of taxpayers aged between 20 and 37 years (31.1%) and aged under 20 years (although small, 0.34%).
- Second cluster with the highest percentage of taxpayers who are single, divorced, or legally separated (72.7%);

Individual 2021

Variables	Cluster 1	Cluster 2	Cluster 3	Cluster 4
DISTRITO_11	0.195770	0.231058	0.272244	0.243393
DISTRITO_13	0.172249	0.137455	0.150226	0.131645
DISTRITO_15	0.077337	0.085960	0.096009	0.092051
DISTRITO_Other	0.554644	0.545528	0.481521	0.532911
ESTADO_CIVIL_C	0.911222	0.441733	0.036877	0.189529
ESTADO_CIVIL_Other	0.062133	0.079523	0.202725	0.033966
ESTADO_CIVIL_S	0.026645	0.478745	0.760398	0.776505
GENERO_F	0.473167	0.327880	0.529266	0.436841
GENERO_M	0.526833	0.672120	0.470734	0.563159
IDADE_BINS_<20	0.000640	0.001341	0.006073	0.132047
IDADE_BINS_>=92	0.001656	0.000671	0.010668	0.004120
IDADE_BINS_20-37	0.120503	0.221537	0.352871	0.388202
IDADE_BINS_38-55	0.401438	0.492423	0.301330	0.275148
IDADE_BINS_56-73	0.352928	0.237227	0.186304	0.152246
IDADE_BINS_74-91	0.122836	0.046802	0.142754	0.048236
INDICADOR_ATIVIDADE_0	0.821918	0.141478	0.842822	0.903025
INDICADOR_ATIVIDADE_1	0.178082	0.858522	0.157178	0.096975
INDICADOR_DEV_ESTRATEG_0	0.999925	0.997050	0.999600	0.999397
INDICADOR_DEV_ESTRATEG_1	0.000075	0.002950	0.000400	0.000603
INDICADOR_DIV_IRS_0	0.977796	0.797640	0.898797	0.959200
INDICADOR_DIV_IRS_1	0.022204	0.202360	0.101203	0.040800
INDICADOR_DIV_IVA_0	0.999210	0.675205	0.996724	0.982615
INDICADOR_DIV_IVA_1	0.000790	0.324795	0.003276	0.017385
INDICADOR_MOD3_0	0.000000	0.045997	0.000000	0.999900
INDICADOR_MOD3_1	1.000000	0.954003	1.000000	0.000100
INDICADOR_MOD3_PRAZO_0	0.000602	0.046131	0.000360	1.000000
INDICADOR_MOD3_PRAZO_1	0.999398	0.953869	0.999640	0.000000
INDICADOR_SUJEITO_IVA_0	0.998984	0.038487	0.996724	0.982514

Variables	Cluster 1	Cluster 2	Cluster 3	Cluster 4
INDICADOR_SUJEITO_IVA_1	0.001016	0.961513	0.003276	0.017486
INDICADOR_TRIB_AUTONOMA_0	0.884578	0.869787	0.934836	1.000000
INDICADOR_TRIB_AUTONOMA_1	0.115422	0.130213	0.065164	0.000000
INDICADOR_TRIB_CONJUNTA_0	0.024311	0.549551	0.977826	1.000000
INDICADOR_TRIB_CONJUNTA_1	0.975689	0.450449	0.022174	0.000000
TIPO_PERIODO_IVA_M	0.000263	0.031112	0.000280	0.000201
TIPO_PERIODO_IVA_Na	0.994694	0.015154	0.996524	0.989549
TIPO_PERIODO_IVA_T	0.005043	0.953735	0.003196	0.010250
V_BASE_TRIB_BINS_<392054428	1.000000	0.999866	1.000000	1.000000
V_BASE_TRIB_BINS_>=784108854.1	0.000000	0.000134	0.000000	0.000000
V_DIVIDA_IRS_BINS_<110680	1.000000	0.999866	0.999960	0.999598
V_DIVIDA_IRS_BINS_>=221358.1	0.000000	0.000134	0.000000	0.000201
V_DIVIDA_IRS_BINS_110680-221358	0.000000	0.000000	0.000040	0.000201
V_DIVIDA_IVA_BINS_<74115.42	1.000000	0.999464	1.000000	0.999297
V_DIVIDA_IVA_BINS_>=148230.83	0.000000	0.000268	0.000000	0.000402
V_DIVIDA_IVA_BINS_74115.42-148230.82	0.000000	0.000268	0.000000	0.000301
V_IRS_FINAL_BINS_<528121.18	1.000000	0.999866	1.000000	1.000000
V_IRS_FINAL_BINS_>=1092536.72	0.000000	0.000134	0.000000	0.000000
V_REND_BRUTO_BINS_<971921.87	1.000000	0.999732	1.000000	1.000000
V_REND_BRUTO_BINS_>=1943843.73	0.000000	0.000134	0.000000	0.000000
V_REND_BRUTO_BINS_971921.87-1943843.72	0.000000	0.000134	0.000000	0.000000
V_REND_COLETAVEL_BINS_<971426.76	1.000000	0.999732	1.000000	1.000000
V_REND_COLETAVEL_BINS_>=1942853.6	0.000000	0.000134	0.000000	0.000000
V_REND_COLETAVEL_BINS_971426.76-1942853.5	0.000000	0.000134	0.000000	0.000000

Table 4.9 - K-means Individual taxpayers for 2021 Value Cluster

Cluster 1: Majority married and filed joint taxation

- Over 90% of taxpayers are married and filed joint taxation (highlight characteristics of the cluster);
- All taxpayers submitted IRS (filed form 3), have a taxable base value below than 392,054,428€, have an IRS debt value below than 110,680€, have a vat debt value below than 741,15.42€, have a final IRS value less than 528,121.18€, gross income value below than 971,921.87€, and taxable income value below than 971,426.76€;
- It is the cluster (of the 4) with the highest percentage of taxpayers who do not have VAT debts (99.9%) and IRS debts (97.8%). It also has the highest percentage of taxpayers who are not strategic debtors (99.99%) and taxpayers who are not subject to VAT (99.9%).

Cluster 2: Majority have activity, subject to VAT, and opted for a quarterly VAT period

- Over 85% of taxpayers have activity, and over 95% are subject to VAT and opted for a quarterly VAT period (highlight characteristics of the cluster);
- It is the only cluster that presents a percentage of taxpayers (although small, about 0.01%) for the variables: taxable base value greater than or equal to 784,108,854.1€, IRS debt value greater than or equal to 221,358.1€, final IRS value greater than or equal to 1,092,536.72€, gross income value between 971,921.87€ and 194,843.72€ and greater than or equal to 1,943,843.73€, and taxable income value between 971,426.76€ and 1,942,853.5€ and greater than or equal to 1,942,853.6€;
- Cluster with the highest percentage of strategic debtors (although small, about 0.2%) and taxpayers with IRS and VAT debts (20.2% and 32.4%, respectively);
- Cluster with the highest percentage of taxpayers aged between 38 and 55 years (49.2%) and male gender (67.2%).

Cluster 3: Mix Cluster

- There is no characteristic that is solely distinctive of this cluster;
- All taxpayers submitted IRS (filed form 3), have a taxable base value below than 392,054,428€, have an VAT debt value below than 74,115.42€, have a final IRS value less than 528,121.18€, gross income value below than 971,921.87€, and taxable income value below than 971,426.76€;
- Cluster with the biggest percentage of female taxpayers (52.9%);
- Cluster with the highest percentage of taxpayers aged between 74 and 91 years (14.3%) and aged 92 years or older (although small, about 1.1%);
- Cluster with the biggest percentage of taxpayers who are separated in fact, united in fact, or widowed (20.2%).

Cluster 4: Majority did not submit form 3, and those who did, did it after the deadline

- 99% did not submit IRS (did not file form 3), and those who did, did so late (100%). Perhaps this is why both the indicator for those who opted for joint taxation and automatic taxation are at 0.
- All taxpayers have a taxable base value below than 392,054,428€, have a final IRS value less than 528,121.18€, gross income value below than 971,921.87€, and taxable income value below than 971,426.76€;
- Cluster with the highest percentage of taxpayers who are single, divorced, or legally separated (77.7%);
- Cluster with the highest percentage of taxpayers aged between 20 and 37 years (38.8%) and aged under 20 years (13.2%).

Business 2017

Variables	Cluster 1	Cluster 2
CODIGO_DIV_ATIVIDADE_0	0.014946	0.995920
CODIGO_DIV_ATIVIDADE_47	0.120359	0.000422
CODIGO_DIV_ATIVIDADE_56	0.097655	0.000281
CODIGO_DIV_ATIVIDADE_Other	0.767039	0.003376
DISTRITO_11	0.290949	0.316263
DISTRITO_13	0.181810	0.183877
DISTRITO_3	0.074907	0.076252
DISTRITO_Other	0.452334	0.423607
INDICADOR_ATIVIDADE_0	0.009687	0.712999
INDICADOR_ATIVIDADE_1	0.990313	0.287001
INDICADOR_DEV_ESTRATEG_0	0.989218	0.996624
INDICADOR_DEV_ESTRATEG_1	0.010782	0.003376
INDICADOR_DIV_IRC_0	0.665352	0.900394
INDICADOR_DIV_IRC_1	0.334648	0.099606
INDICADOR_DIV_IVA_0	0.688538	0.958357
INDICADOR_DIV_IVA_1	0.311462	0.041643
INDICADOR_INSOLVENCIA_0	0.990752	0.999297
INDICADOR_INSOLVENCIA_1	0.009248	0.000703
INDICADOR_SUJEITO_IVA_0	0.078852	0.958357
INDICADOR_SUJEITO_IVA_1	0.921148	0.041643
TIPO_PERIODO_IVA_M	0.209117	0.000000
TIPO_PERIODO_IVA_Na	0.014683	0.995920
TIPO_PERIODO_IVA_T	0.776200	0.004080
V_BASE_TRIB_BINS_<1319063231	0.999693	1.000000
V_BASE_TRIB_BINS_>=2638126471	0.000088	0.000000
V_BASE_TRIB_BINS_1319063231-2638126470	0.000219	0.000000
V_DIVIDA_IRC_BINS_<1241666.37	0.999825	1.000000
V_DIVIDA_IRC_BINS_>=2483332.73	0.000088	0.000000
V_DIVIDA_IRC_BINS_1241666.37-2483332.72	0.000088	0.000000
V_DIVIDA_IVA_BINS_<2559061.32	0.999912	1.000000
V_DIVIDA_IVA_BINS_>=5118122.63	0.000044	0.000000
V_DIVIDA_IVA_BINS_2559061.32-5118122.62	0.000044	0.000000
V_VOLUME_NEG_BINS_<950591016	0.999693	1.000000
V_VOLUME_NEG_BINS_>=1901182031	0.000131	0.000000
V_VOLUME_NEG_BINS_950591016-1901182030	0.000175	0.000000

Table 4.10 - K-means Business taxpayers for 2017 Value Cluster

Cluster 1: Majority have activity, subject to VAT, and opted for a quarterly VAT period

- Over 99% of taxpayers have activity,92.1% are subject to VAT and 77.6% opted for a quarterly VAT period. Excluding the division of activity involving retail trade, except for motor vehicles and motorcycles (47) and restaurants and similar (56), in this cluster, the aggregation of the remaining divisions of activity asserts itself as the majority, with 76.7% (highlight characteristics of the cluster);
- Cluster with the highest percentage of strategic debtors (although small, about 1%), taxpayers who entered insolvency (small to, 0.9%) and taxpayers with IRC and VAT debts (33.5% and 31.1%, respectively);
- Although the majority opted for the quarterly VAT period, this is the only cluster with a percentage of monthly VAT period (20.9%).
- The only cluster with a percentage for higher values of debt in Corporate Income Tax (IRC), VAT, and turnover.

Cluster 2: Majority don't have activity, aren't subject to VAT and don't have an applicable VAT period

- 72.3% of taxpayers don't have activity,95.8% aren't subject to VAT and 99.6% don't have an applicable VAT period. 99.5% of the taxpayers don't have the activity division code filled in (highlight characteristics of the cluster);
- All taxpayers have IRC debt value less than 1,241,666.37€, VAT debt value less than 2,559,061.32€, and turnover value less than 950,591,016€.

Business 2021

Variables	Cluster 1	Cluster 2
CODIGO_DIV_ATIVIDADE_0	0.994982	0.000182
CODIGO_DIV_ATIVIDADE_47	0.002133	0.112427
CODIGO_DIV_ATIVIDADE_56	0.002635	0.096529
CODIGO_DIV_ATIVIDADE_Other	0.000251	0.790862
DISTRITO_11	0.279513	0.303298
DISTRITO_13	0.200477	0.175702
DISTRITO_3	0.081797	0.072841
DISTRITO_Other	0.438214	0.448160
INDICADOR_ATIVIDADE_0	0.472964	0.003872
INDICADOR_ATIVIDADE_1	0.527036	0.996128
INDICADOR_DEV_ESTRATEG_0	0.978547	0.985605
INDICADOR_DEV_ESTRATEG_1	0.021453	0.014395
INDICADOR_DIV_IRC_0	0.894242	0.774098

Variables	Cluster 1	Cluster 2
INDICADOR_DIV_IRC_1	0.105758	0.225902
INDICADOR_DIV_IVA_0	0.853594	0.636297
INDICADOR_DIV_IVA_1	0.146406	0.363703
INDICADOR_INSOLVENCIA_0	0.996487	0.991436
INDICADOR_INSOLVENCIA_1	0.003513	0.008564
INDICADOR_SUJEITO_IVA_0	0.853594	0.082270
INDICADOR_SUJEITO_IVA_1	0.146406	0.917730
TIPO_PERIODO_IVA_M	0.004516	0.231460
TIPO_PERIODO_IVA_Na	0.994856	0.000000
TIPO_PERIODO_IVA_T	0.000627	0.768540
V_BASE_TRIB_BINS_<1201446680.1	1.000000	0.999590
V_BASE_TRIB_BINS_>=2402893360.1	0.000000	0.000228
V_BASE_TRIB_BINS_1201446680.1-2402893360	0.000000	0.000182
V_DIVIDA_IRC_BINS_<3657137.48	0.999875	0.999954
V_DIVIDA_IRC_BINS_>=7314274.95	0.000125	0.000046
V_DIVIDA_IVA_BINS_<662301.1	0.999749	0.999772
V_DIVIDA_IVA_BINS_>=1324602.1	0.000125	0.000000
V_DIVIDA_IVA_BINS_662301.1-1324602	0.000125	0.000228
V_VOLUME_NEG_BINS_<788982540.1	1.000000	0.999636
V_VOLUME_NEG_BINS_>=1577965080.1	0.000000	0.000182
V_VOLUME_NEG_BINS_788982540.1-1577965080	0.000000	0.000182
V_VOLUME_NEG_BINS_950591016-1901182030	0.000175	0.000000

Table 4.11 - K-means Business taxpayers for 2021 Value Cluster

Cluster 1: Majority don’t have the division code for the CAE, aren’t subject to VAT, and don't have an applicable VAT period

- 99.5% of taxpayers don’t have the division code for the Classification of Portuguese Economic Activities by Branch of Activity,85.3% aren’t subject to VAT and 99.5% don’t have an applicable VAT period (highlight characteristics of the cluster);
- All taxpayers have taxable base value less than 1,201,446,680.1€, and turnover value less than 788,982,540.1€.
- There is a higher percentage of taxpayers with debt than without debt (52.7% with debt), but the percentage of taxpayers without debt is much higher in this cluster (47.3%);
- Cluster with the highest percentage of strategic debtors (although small, about 2.1%).

Cluster 2: Majority have activity, subject to VAT, and opted for a quarterly VAT period

- Over 99.6% of taxpayers have activity,91.8% are subject to VAT and 76.9% opted for a quarterly VAT period. Excluding the division of activity involving retail trade, except for motor vehicles and motorcycles (47) and restaurants and similar (56), in this cluster, the aggregation of the remaining divisions of activity asserts itself as the majority, with 76.7% (highlight characteristics of the cluster);
- Cluster with the highest percentage of taxpayers who entered insolvency (small to, 0.85%) and taxpayers with IRC and VAT debts (22.6% and 36.4%, respectively);
- Although the majority opted for the quarterly VAT period, this cluster also have the biggest percentage of monthly VAT period (23.1%).
- The only cluster with a percentage for higher values of debt in Corporate Income Tax (IRC) and turnover.

5. CONCLUSIONS

The main goal of this project was to apply a clustering algorithm in order to understand how taxpayers are grouped together based on their characteristics and what information can we glean from your analysis. This will hopefully enable the Tax Authority to target its inspection actions more effectively and consistently to certain taxpayers. The objectives initially defined were the following:

1. Constructing a database as homogeneous as possible, taking into account the characteristics of taxpayers, AT rules, and tax regulations. That's why we adjusted the percentages of the total taxpayer base to ensure that our database would have "a little bit of everything".

2. Identifying the main characteristics, such as the most important and interesting features, in our opinion, for our cluster analysis.

Comparing the results obtained in the clusters of individual taxpayers for 2017 and individual taxpayers for 2021, what stands out the most (considering the cluster that we consider to have the most relevant information for the Tax Authority) is cluster 2 "Didn’t submit form 3 (and out of deadline)" in the case of 2017 individuals and cluster 4 "Majority did not submit form 3, and those who did, did it late" in the case of 2021 individuals, as the dominant age intervals in the clusters could not be more different. For 2017, cluster 2 is the one where taxpayers aged between 74 and 91 years (22.3%) and aged 92 years or older (about 6%) have the highest percentage. For 2021, cluster 4 is the one where taxpayers aged between 20 and 37 years (38.8%) and aged under 20 years (13.2%) have the highest percentage. And what is the reason for this difference? This is a topic that could be interesting for the Tax Authority; it could be due to the lack of information that younger individuals have or even because young people are starting to work later and later.

Comparing the results obtained in the clusters of business taxpayers for 2017 and business taxpayers for 2021, as there are only two clusters for each sample (2017 and 2021), what stands out the most, apart from the differences in percentages, is that not having activity ceased. Is a characteristic that stands out in just one cluster in 2021. In 2017, in cluster 2, what stands out the most is "Majority don’t have activity, aren’t subject to VAT, and don’t have an applicable VAT period", while in 2021 what stands out is "Majority don’t have the division code for the CAE, aren’t subject to VAT, and don't have an applicable VAT period", having or not having activity both have a 50/50 percentage for this cluster. And what is the reason for this difference? This is a topic that could be interesting for the Tax Authority to analyze; there may be more businesses, or the support measures applied due to the COVID-19 pandemic may be having a positive impact.

5.1. LIMITATIONS

In this thesis, the biggest limitation we encountered was the sampling process considering the collection of external data. Also, we had to consider many variables to obtain the data as homogeneous as possible, since the construction part of the dataset was also done by us.

5.2. RECOMMENDATION FOR FUTURE WORKS

After the characterization it can be applied an anomaly detection Machine Learning algorithms to detect abnormal behaviors and try to anticipate the taxpayers who will have bad debts to the State.

Also, it may be interesting in future works to also involve the emotional/psychological variables, to try to understand if the taxpayers are aware that they may be contributing to the increase in the public debt, as well as if they do it on purpose or if it is pure forgetfulness.

BIBLIOGRAPHICAL REFERENCES

Adamov, A. (2019). Machine Learning and Advanced Analytics in Tax Fraud Detection. Center for Data Analytics Research (CeDAR), ADA University Baku, Azerbaijan.

Baskerville, R., Baiyere, A., Gregor, S., Hevner, A., & Rossi, M. (2018). Design science research contributions: Finding a balance between artifact and theory. Journal of the Association for Information Systems https://doi.org/10.17705/1jais.00495

Biryukov, A., Brezhneva, O., Altynbaeva, L., Schnayderman, A., & Efimova, N. (2022). Methods of Neural Network Modeling of Clusterization of Taxpayers to Determine Credit Risk by a Financial Regulator. In: Antipova, T. (eds) Comprehensible Science. ICCS 2021. Lecture Notes in Networks and Systems, vol 315. Springer, Cham. https://doi.org/10.1007/978-3-030-85799-8_1

Buehn, A. & Schneider, F. (2012). Size and Development of Tax Evasion in 38 OECD Coutries: What do we (not) know?. Journal of Economics and Political Economy. 3. 10.1453/jepe.v3i1.634.

Chica, M., Hernandez, J., Manrique-de-Lara-Peñate, C., & Chiong, R. (2021). An Evolutionary Game Model for Understanding Fraud in Consumption Taxes. http://arxiv.org/abs/2101.04424

Código do Imposto Sobre o Rendimento das Pessoas Singulares (CIRS) (portaldasfinancas.gov.pt) Last atualization: Lei nº24-D/2022, de 30 de dezembro

Código do Imposto sobre o Rendimento das Pessoas Coletivas (CIRC) (portaldasfinancas.gov.pt) Last atualization: Lei nº24-D/2022, de 30 de dezembro

Código do Imposto sobre o Valor Acrescentado (portaldasfinancas.gov.pt) Last atualization: Lei nº24-D/2022, de 30 de dezembro

Código de Procedimento e de Processo Tributário (portaldasfinancas.gov.pt) Last atualization: Lei nº12-D/2022, de 27 de junho

Decreto-Lei n.º 398/98 Lei Geral Tributária. (1998). Diário da República: I Série A, nº 290/98. https://diariodarepublica.pt/dr/legislacao-consolidada/decreto-lei/1998-34438775

Del Camino González Vasco, M., Jesús, M., Rodríguez, D., De, S., & Santos, L. (2021). Characterization and detection of potential fraud taxpayers in Personal Income Tax using data mining techniques. https://www.researchgate.net/publication/327981272

Dias, A., Pinto, C., Batista, J. ,& Neves, M., E. (2016), "Signaling Tax Evasion, Financial Ratios and Cluster Analysis", Working Paper 51, OBEGEF-Observatório de Economia e Gestão de Fraude, Coimbra.

Domingo Velasquez, J. (2007). Segmentación de los contribuyentes que declaran iva aplicando herramientas de clustering. https://www.researchgate.net/publication/292876147

Dridi S. (2021) Unsupervised Learning - A Systematic Literature Review (PDF) Unsupervised Learning - A Systematic Literature Review (researchgate.net)

Gabinete do Secretário de Estado dos Assuntos Fiscais (2022), Relatório Sobre o Combate à Fraude e Evasão Fiscais e Aduaneiras 2021

González, P., C., & Velásquez, J., D. (2013), "Characterization and detection of taxpayers with false invoices using data mining techniques", Expert Systems with Applications.

Hevner, A., & Chatterjee, S. (2010) Design Research in Information Systems, Integrated Series in Information Systems 22, 9-22

Imposto sobre o Rendimento das Pessoas Coletivas (IRC) em Portugal - ePortugal.gov.pt. (n.d.). Eportugal.gov.pt. https://eportugal.gov.pt/cidadaos-europeus-viajar-viver-e-fazer-negocios-em-portugal/impostos-para-atividades-economicas-em-portugal/imposto-sobre-o-rendimento-das-pessoas-coletivas-irc-em-portugal

Imposto sobre o Rendimento das Pessoas Singulares (IRS) em Portugal - ePortugal.gov.pt. (2024). Eportugal.gov.pt. https://eportugal.gov.pt/cidadaos-europeus-viajar-viver-e-fazer-negocios-em-portugal/trabalho-e-reforma-em-portugal/imposto-sobre-o-rendimento-das-pessoas-singulares-irs-em-portugal

Imposto sobre Valor Acrescentado (IVA) em Portugal - ePortugal.gov.pt. (n.d.). Eportugal.gov.pt. https://eportugal.gov.pt/cidadaos-europeus-viajar-viver-e-fazer-negocios-em-portugal/impostos-para-atividades-economicas-em-portugal/imposto-sobre-valor-acrescentado-iva-em-portugal

Instituto Nacional de Estatistica, Statistics Portugal (2007), Classificação Portuguesa das Atividades Económicas Rev.3. https://www.ine.pt/ine_novidades/semin/cae/CAE_REV_3.pdf

Joshi, S. (2022, August 5). What is Clustering in Machine Learning: Types and Methods. ANALYTIXLABS. https://www.analytixlabs.co.in/blog/types-of-clustering-algorithms/

Kuechler, B. , & Vaishnavi, V. (2008). On theory development in design science research: Anatomy of a research project. European Journal of Information Systems

Kurita, T. (2020). Principal Component Analysis (PCA) 1-4 https://doi.org/10.1007/978-3-030-03243-2_649-1

Milner, C. , & Berg, B. (2017). Tax Analytics Artificial Intelligence and Machine Learning–Level 5. PwC Advanced Tax Analytics & Innovation

Pinheiro, J. M., Diogo, T. A., & Samagaio, A. (2021). Tax Compliance: Factors that Influence Taxpayer Invoice Requests in Portugal. Revista Brasileira de Gestao de Negocios, 23(4), 619–634. https://doi.org/10.7819/rbgn.v23i4.4133

Richardson, D. (2022, November 11). What is AI/ML and why does it matter to your business? RedHat Blog https://www.redhat.com/en/blog/what-aiml-and-why-does-it-matter-your-business

Roux, D., Perez, B., Moreno, A., Villamil, M., & Figueroa, C. (2018) Tax Fraud Detection for Under-Reporting Declarations Using an Unsupervised Machine Learning Approach Tax Fraud Detection for Under-Reporting Declarations Using an Unsupervised Machine Learning Approach | Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining

Savić, M., Atanasijević, J., Jakovetić, D., & Krejić, N. (2021) Tax Evasion Risk Management Using a Hybrid. Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Serbia

Sebtaoui, FE., Adri, A., Rifai, S., & Sahaf, K. (2020) How will the risk management impact the success of just-in-time implementation? Journal of Industrial and Production Engineering

Vanhoeyveld, J., Martens, D., & Peeters, B. (2020) Value-added tax fraud detection with scalable anomaly detection techniques Value-added tax fraud detection with scalable anomaly detection techniques | Applied Soft Computing (acm.org)

APPENDIX

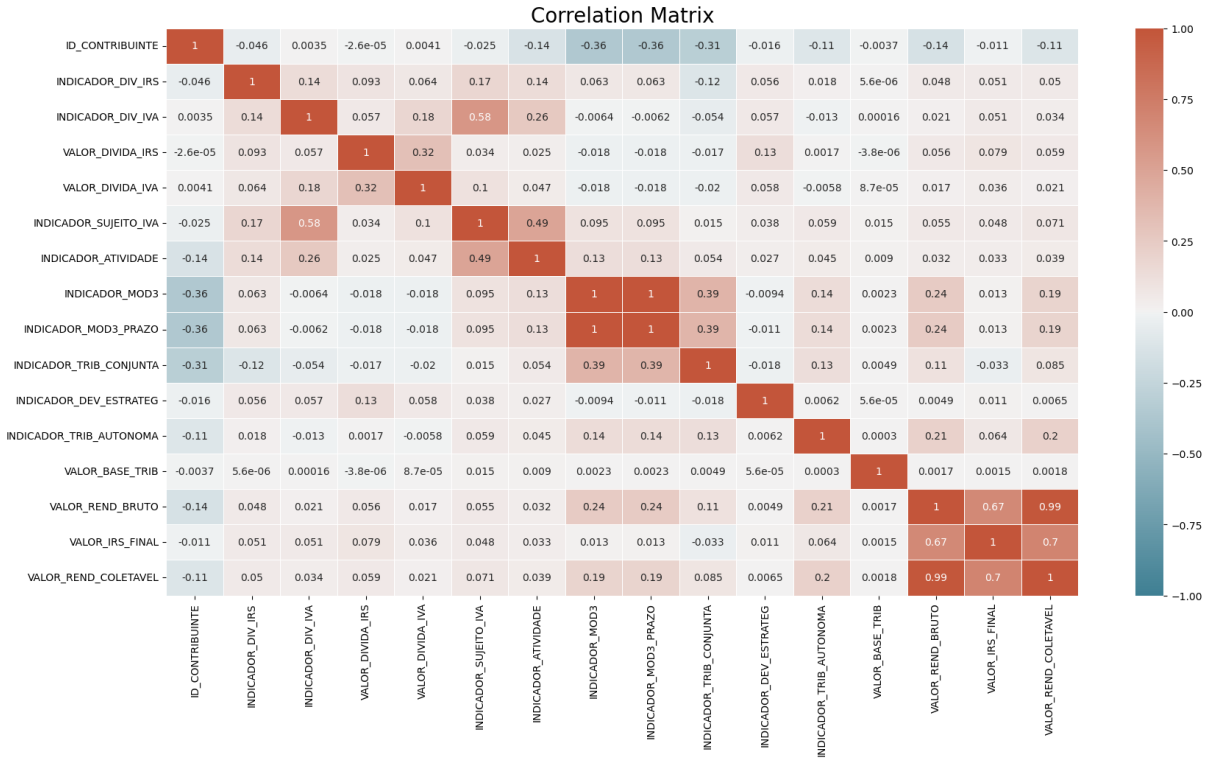

Figure 4.31 - Correlation Matrix of Individual taxpayers of 2017

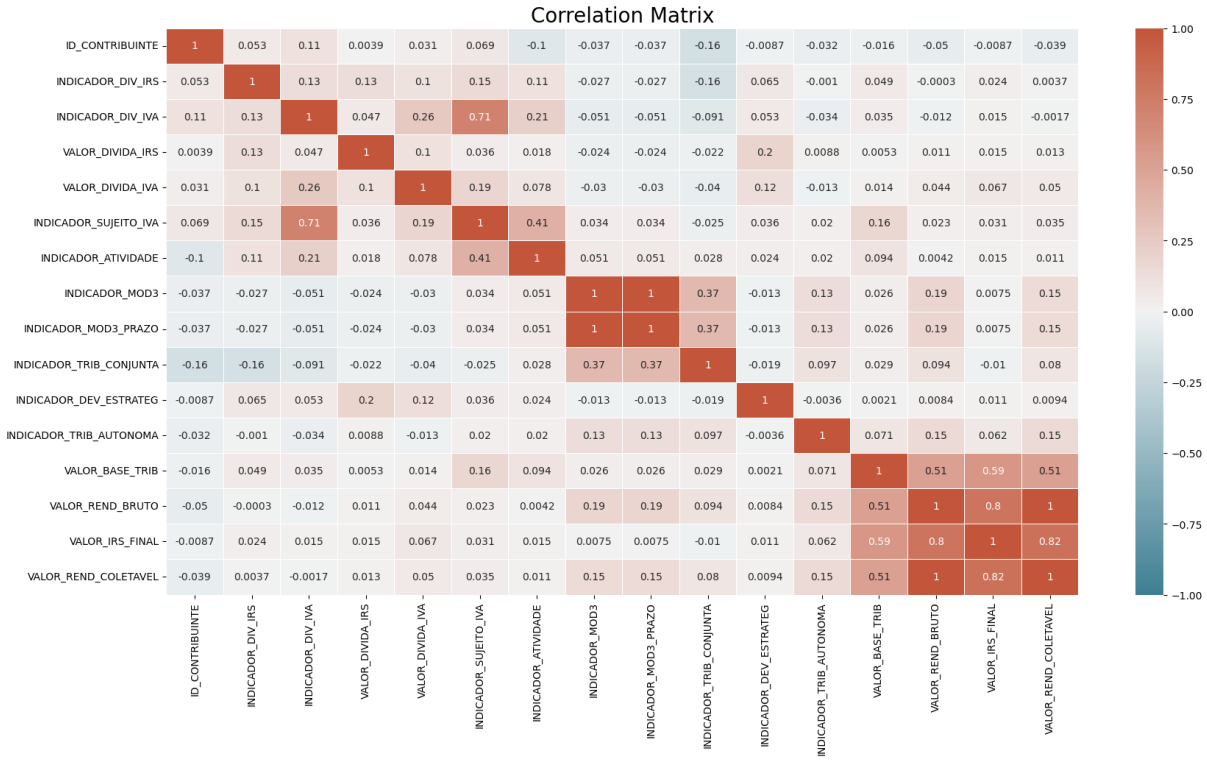

Figure 4.32 - Correlation Matrix of Individual taxpayers of 2021

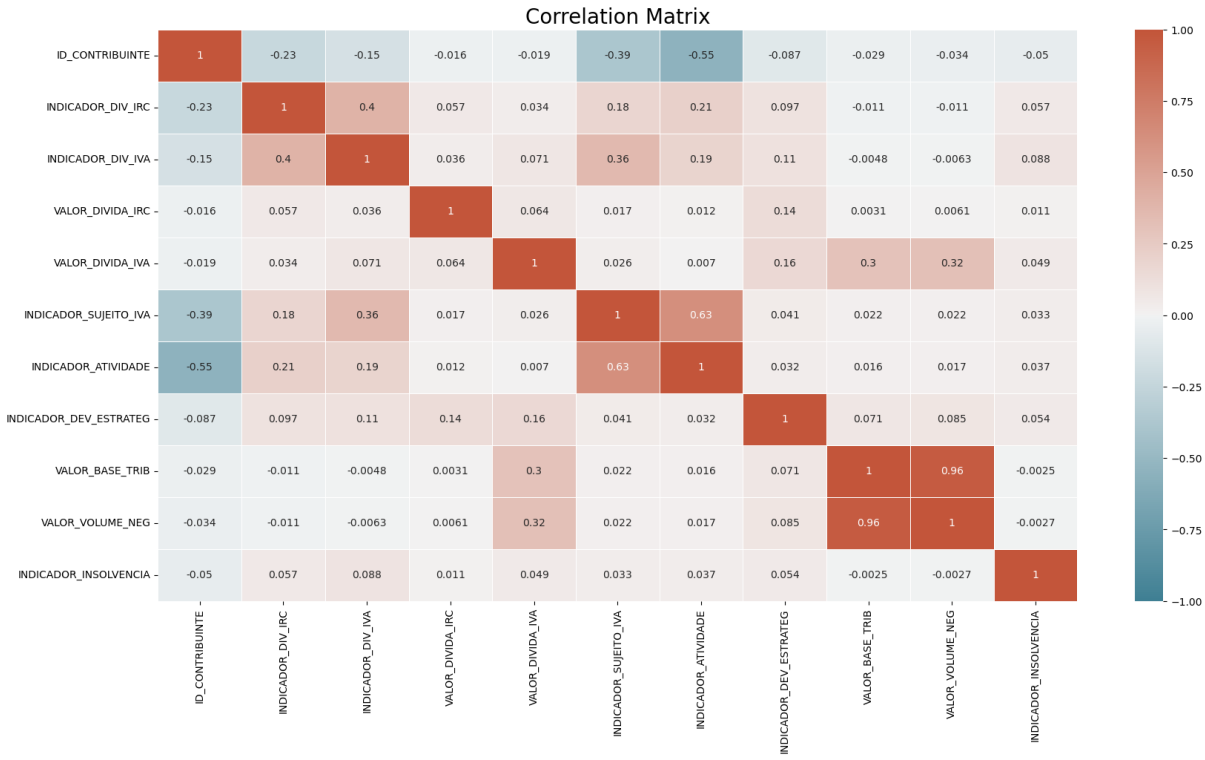

Figure 4.33 - Correlation Matrix of Business taxpayers of 2017

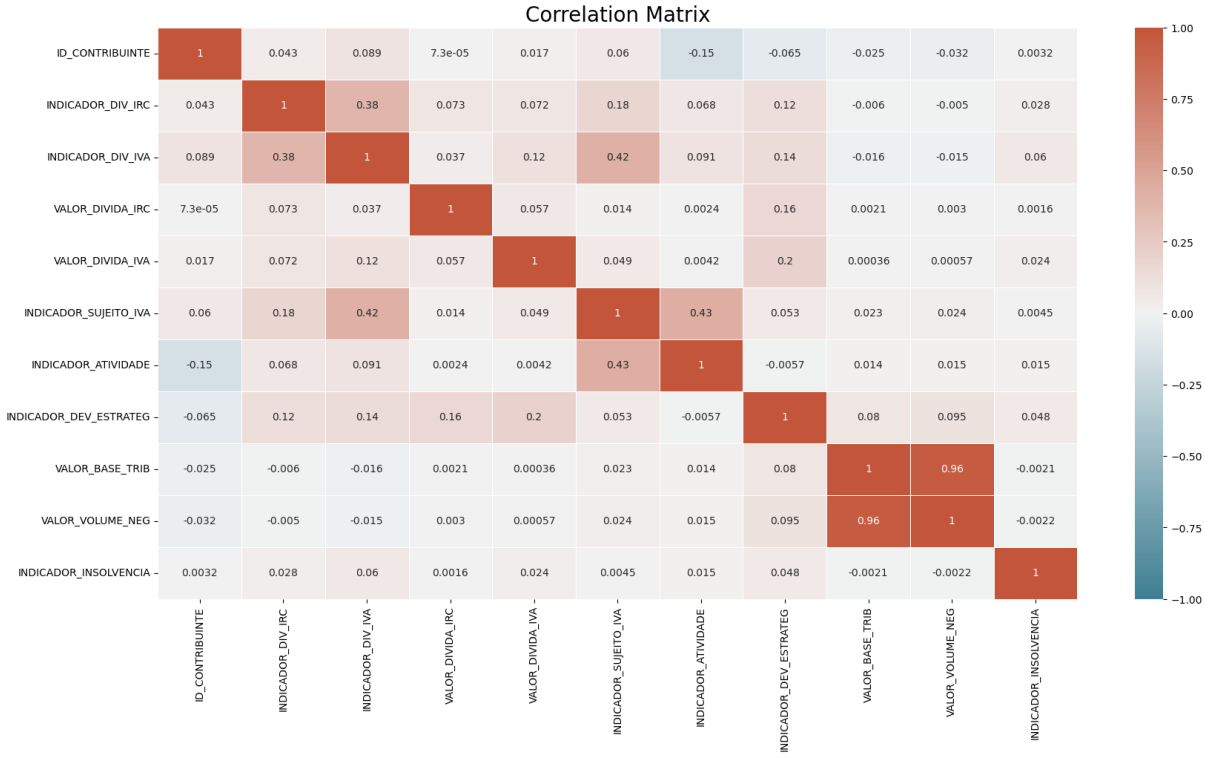

Figure 4.34 - Correlation Matrix of Business taxpayers of 2021

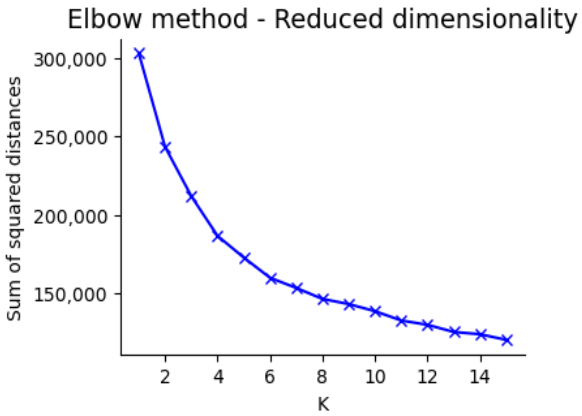

Figure 4.35 - Elbow method – Reduced dimensionality Individual 2017

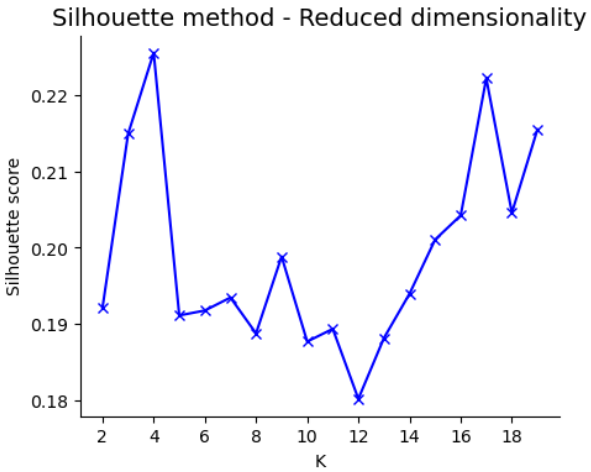

Figure 4.36 - Silhouette method – Reduced dimensionality Individual 2017

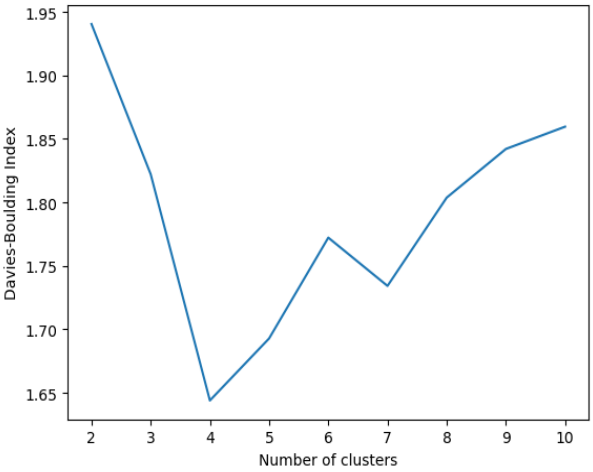

Figure 4.37 - Davies-Boulding Index – Reduced dimensionality Individual 2017

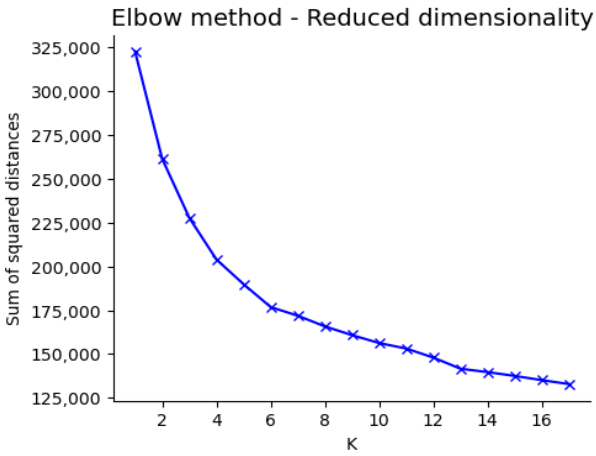

Figure 4.38 - Elbow method – Reduced dimensionality Individual 2021

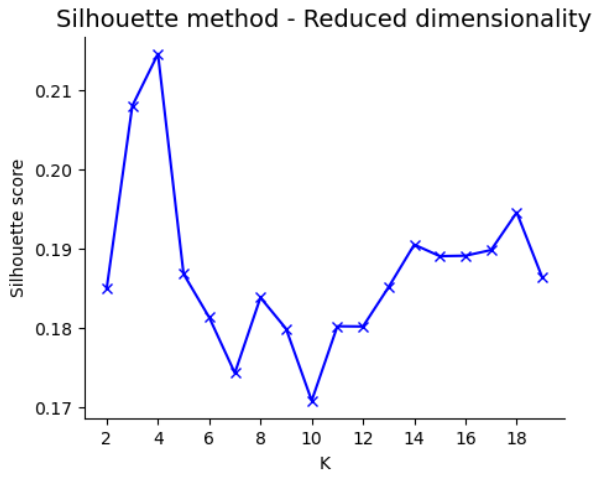

Figure 4.39 - Silhouette method – Reduced dimensionality Individual 2021

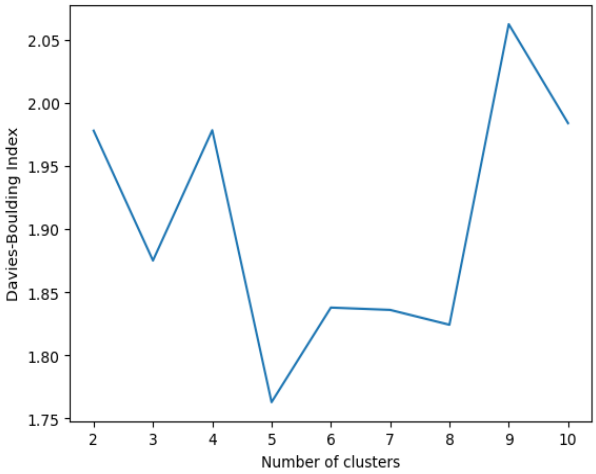

Figure 4.40- Davies-Boulding Index – Reduced dimensionality Individual 2021

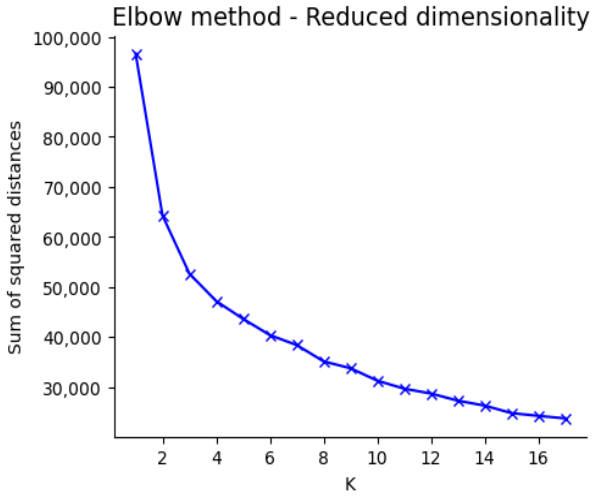

Figure 4.41 - Elbow method – Reduced dimensionality Business 2017

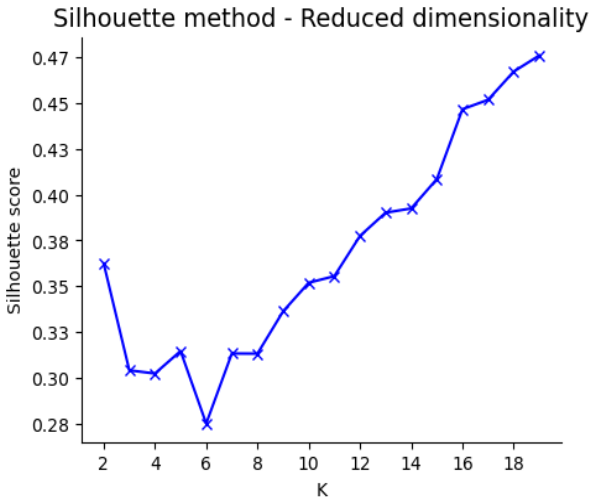

Figure 4.42 - Silhouette method – Reduced dimensionality Business 2017

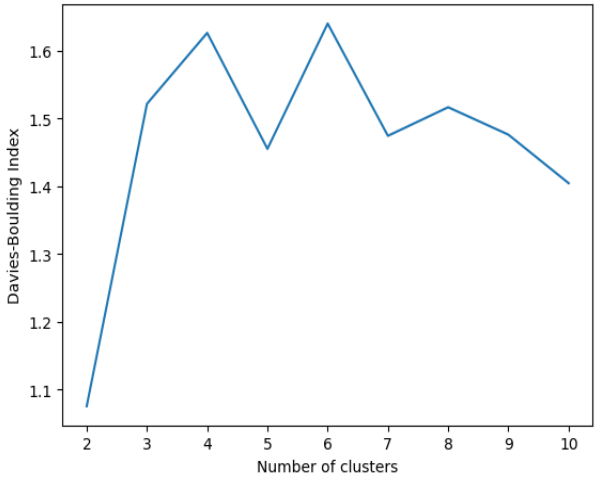

Figure 4.43 - Davies-Boulding Index – Reduced dimensionality Business 2017

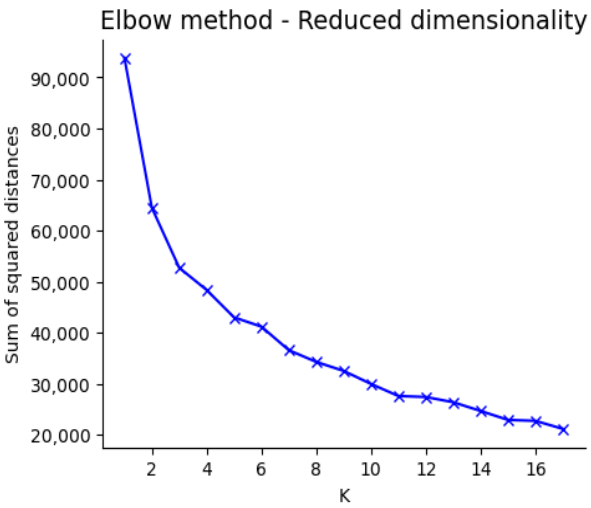

Figure 4.44 - Elbow method – Reduced dimensionality Business 2021

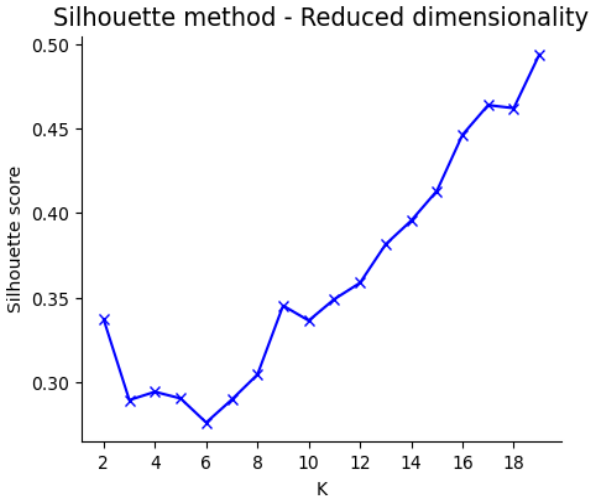

Figure 4.45 - Silhouette method – Reduced dimensionality Business 2021

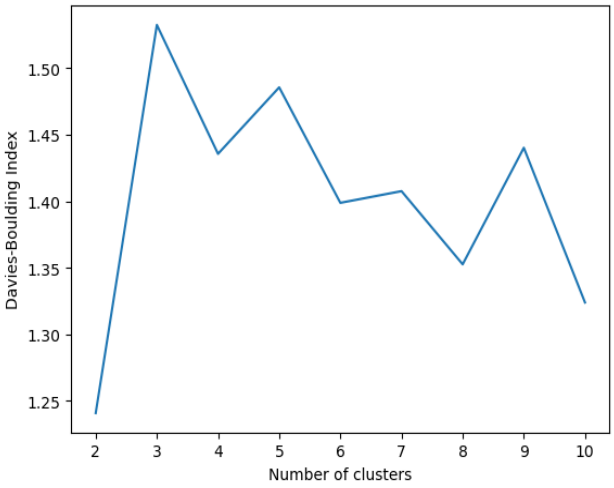

Figure 4.46 - Davies-Boulding Index – Reduced dimensionality Business 2021

PROJECT CODE

Project

```
# Loading packages
import os
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.patches as patches
import seaborn as sns
import numpy as np
import matplotlib.ticker as tick
from matplotlib import ticker
from math import ceil
from datetime import date
import missingno as msno
from sklearn import linear_model
from scipy.cluster.hierarchy import dendrogram, linkage, fcluster
from sklearn.cluster import AgglomerativeClustering
import seaborn as sns
import joypy
import category_encoders as ce
import collections
from sklearn.impute import SimpleImputer
from sklearn import preprocessing
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score
from sklearn.preprocessing import MinMaxScaler
from sklearn.datasets import load_iris
from sklearn.metrics import davies_bouldin_score

pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
```

```
# Global definitions
baseFolder = os.getcwd()
exportsFolder = baseFolder + os.sep +'Exports' + os.sep

subPlots_Title_fontSize = 12
subPlots_xAxis_fontSize = 10
subPlots_yAxis_fontSize = 10
subPlots_label_fontSize = 10
heatmaps_text_fontSize = 8

plots_Title_fontSize = 14
plots_Title_textColour = 'black'

plots_Legend_fontSize = 12
plots_Legend_textColour = 'black'

plots_barTexts_fontSize = 8

# Loading the dataset and visualizing summary statistics
data = pd.read_csv('Taxpayers_Data.csv', sep=";",low_memory=False)

data.info()
```

Individual 2017

```
dataS17 = data.copy()
```

```
dataS17.head()
```

```
#Remove the business
dataS17.drop(dataS17.index[(dataS17["TIPO_NIF"] == 'C')],axis=0,inplace=True)
```

```
#Remove years that are not used
dataS17.drop(dataS17.index[(dataS17["ANO_EXERCICIO"] == 2018)],axis=0,inplace=True)
dataS17.drop(dataS17.index[(dataS17["ANO_EXERCICIO"] == 2019)],axis=0,inplace=True)
dataS17.drop(dataS17.index[(dataS17["ANO_EXERCICIO"] == 2020)],axis=0,inplace=True)
dataS17.drop(dataS17.index[(dataS17["ANO_EXERCICIO"] == 2021)],axis=0,inplace=True)
```

DATA ANALYSIS

```
dataS17_1 = dataS17.copy()
```

```
#Check the size of the database
dataS17_1.shape
```

```
#Check for false values
dataS17_1['ANO_NASCIMENTO'].value_counts()
```

```
#Check for false values
dataS17_1['INDICADOR_DIV_IRS'].value_counts()
```

```
#Check for false values
dataS17_1['INDICADOR_DIV_IVA'].value_counts()
```

```
#Check for false values
dataS17_1['TIPO_PERIODO_IVA'].value_counts()
```

```
#Check for false values
dataS17_1['INDICADOR_SUJEITO_IVA'].value_counts()
```

```
#Check for false values
dataS17_1['INDICADOR_ATIVIDADE'].value_counts()
```

```
#Check for false values
dataS17_1['INDICADOR_MOD3'].value_counts()
```

```
#Check for false values
dataS17_1['INDICADOR_MOD3_PRAZO'].value_counts()
```

```
#Check for false values
dataS17_1['INDICADOR_TRIB_CONJUNTA'].value_counts()
```

```
#Check for false values
dataS17_1['INDICADOR_TRIB_AUTONOMA'].value_counts()
```

```
#Check for false values
dataS17_1['DISTRITO'].value_counts()
```

```
dataS17_1.head(20)
```

```
#Marital Status

ed = sns.countplot(y="ESTADO_CIVIL", data=dataS17_1, palette = 'Greens', order=['S','C','V','U','F'])
plt.title('Taxpayers Marital Status')
plt.xlabel('Count')
plt.ylabel('Marital Status')
#plt.xticks(np.arange(0,20,2))
ed.get_xaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))
ed.yaxis.grid(False)
ed.xaxis.grid(True)

total = len(dataS17_1['ESTADO_CIVIL'])
for p in ed.patches:
        percentage = '{:.1f}%'.format(100 * p.get_width()/total)
        x = p.get_x() + p.get_width() + 50
        y = p.get_y() + p.get_height()/2
        ed.annotate(percentage, (x, y))

plt.show()
```

```
# Activity
#0    55106
#1    16034

x = [16034, 55106]
labels = ['Yes', 'No']
colors = ['tab:green', 'tab:grey']

fig, ax = plt.subplots()
ax.pie(x, labels = labels, colors = colors, autopct='%1.1f%%')
ax.set_title('Activity')
plt.show()
```

```
# VAT Period

by = sns.histplot(dataS17_1["TIPO_PERIODO_IVA"] , color="grey", kde = False, bins = 12)
plt.title('VAT Period')
plt.ylabel('Number of Taxpayers')
by.set_xlabel('Type of Period')
by.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))
by.yaxis.grid(True)
by.xaxis.grid(False)

plt.show()
```

```
# Birth Year

by = sns.histplot(dataS17_1["ANO_NASCIMENTO"] , color="green", kde = False, bins = 12)
plt.title('Birth Distribution')
plt.ylabel('Number of Taxpayers')
by.set_xlabel('Year of Birth')
by.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))
by.yaxis.grid(True)
by.xaxis.grid(False)

plt.show()
```

```
#Submitted Tax Return Form 3
labels = ['Yes', 'No']
sizes = [58721, 12419]
colors = ['tab:green', 'tab:grey']

#1    58721
#0    12419


plt.pie(sizes, labels=labels, colors = colors, autopct='%1.1f%%', startangle=90, wedgeprops=dict(width=0.5))
plt.title('Submitted Tax Return Form 3')
plt.show()
```

```
#IQR
Q1 = dataS17_1['ANO_NASCIMENTO'].quantile(0.25)
Q3 = dataS17_1['ANO_NASCIMENTO'].quantile(0.75)
IQR = Q3 - Q1

print('IQR method')
print('Lower outliers: ',((dataS17_1['ANO_NASCIMENTO'] < (Q1 - 1.5 * IQR))).sum())
print('Upper outliers: ',((dataS17_1['ANO_NASCIMENTO'] > (Q3 + 1.5 * IQR))).sum())
print('Total outliers: ',((dataS17_1['ANO_NASCIMENTO'] < (Q1 - 1.5 * IQR)) | (dataS17_1['ANO_NASCIMENTO'] > (Q3 + 1.5 * IQR)
print('Lower boundary: ',Q1 - 1.5 * IQR)
print('Upper boundary: ',Q3 + 1.5 * IQR)

print('\n')

dataS17_1[['ANO_NASCIMENTO']].boxplot()
axes = plt.gca()
axes.set_title('Birth Year')
axes.set_ylabel('Year')
axes.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))
plt.show()
```

DATA PREPARATION

```
#Age

now=2017

dataS17_1['IDADE'] = (now - dataS17_1['ANO_NASCIMENTO'])
dataS17_1.head()
```

```
# When we decided to delete the values where the age of the taxpayers is bigger than 102.5 (Considering the minimum
#as 1914.5 and the maximum as the current year to remove outliers)

dataS17_1 = dataS17_1.drop(dataS17_1.index[dataS17_1["ANO_NASCIMENTO"] < 1915])
dataS17_1 = dataS17_1.drop(dataS17_1.index[dataS17_1["ANO_NASCIMENTO"] > 2016])
```

```
dataS17_1.shape
```

```
#IQR
Q1 = dataS17_1['ANO_NASCIMENTO'].quantile(0.25)
Q3 = dataS17_1['ANO_NASCIMENTO'].quantile(0.75)
IQR = Q3 - Q1

print('IQR method')
print('Lower outliers: ',((dataS17_1['ANO_NASCIMENTO'] < (Q1 - 1.5 * IQR))).sum())
print('Upper outliers: ',((dataS17_1['ANO_NASCIMENTO'] > (Q3 + 1.5 * IQR))).sum())
print('Total outliers: ',((dataS17_1['ANO_NASCIMENTO'] < (Q1 - 1.5 * IQR)) | (dataS17_1['ANO_NASCIMENTO'] > (Q3 + 1.5 * IQR)
print('Lower boundary: ',Q1 - 1.5 * IQR)
print('Upper boundary: ',Q3 + 1.5 * IQR)

print('\n')

dataS17_1[['ANO_NASCIMENTO']].boxplot()
axes = plt.gca()
axes.set_title('Birth Year')
axes.set_ylabel('Year')
axes.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))
plt.show()
```

```
dataS17_2 = dataS17_1.copy()
```

```
# Remove the columNS that should not be used because are business variables or are not necessary
dataS17_2.drop(columns=['VALOR_VOLUME_NEG','TIPO_NIF', 'ANO_EXERCICIO','INDICADOR_INSOLVENCIA','INDICADOR_DIV_IRC',
                        'VALOR_DIVIDA_IRC','ANO_NASCIMENTO'], inplace=True)
```

```
dataS17_2.head()
```

```
#Check the size of the database
dataS17_2.shape
```

```
#Correlation matrix

plt.figure(figsize = (20, 10))
custom_cmap = sns.diverging_palette(220, 20, as_cmap=True)
sns.heatmap(dataS17_2.corr(), annot = True, linewidths=.5, cmap = custom_cmap, vmin = -1)
plt.title('Correlation Matrix', fontsize = 20)
plt.show()
```

```
# Check missing values
print(dataS17_2.isnull().sum())
```

```
#Drop column if they only contain missing values

dataS17_2 = dataS17_2.dropna(how='all')
dataS17_2.shape
```

```
#Since it doesn't make sense to have individual taxpayers without marital status and they only represent 2.98% of the
#individual taxpayer data for 2017, let's remove them from our database.

dataS17_2 = dataS17_2.dropna(subset=['ESTADO_CIVIL'])
dataS17_2.shape
```

```
#We will replace null values with 0 for numerical values and 'Na' for categorical values, as it makes sense to represent
#them this way since they are non-existent
dataS17_2['VALOR_BASE_TRIB'] = dataS17_2['VALOR_BASE_TRIB'].replace(np.nan, 0)
dataS17_2['VALOR_REND_BRUTO'] = dataS17_2['VALOR_REND_BRUTO'].replace(np.nan, 0)
dataS17_2['VALOR_IRS_FINAL'] = dataS17_2['VALOR_IRS_FINAL'].replace(np.nan, 0)
dataS17_2['VALOR_REND_COLETAVEL'] = dataS17_2['VALOR_REND_COLETAVEL'].replace(np.nan, 0)
dataS17_2['VALOR_DIVIDA_IVA'] = dataS17_2['VALOR_DIVIDA_IVA'].replace(np.nan, 0)
dataS17_2['VALOR_DIVIDA_IRS'] = dataS17_2['VALOR_DIVIDA_IRS'].replace(np.nan, 0)
dataS17_2['TIPO_PERIODO_IVA'] = dataS17_2['TIPO_PERIODO_IVA'].fillna('Na')
```

```
#Replace ',' with '.' to use the value format
dataS17_2['VALOR_BASE_TRIB'] = dataS17_2['VALOR_BASE_TRIB'].replace({',': '.'}, regex=True)
dataS17_2['VALOR_REND_BRUTO'] = dataS17_2['VALOR_REND_BRUTO'].replace({',': '.'}, regex=True)
dataS17_2['VALOR_IRS_FINAL'] = dataS17_2['VALOR_IRS_FINAL'].replace({',': '.'}, regex=True)
dataS17_2['VALOR_REND_COLETAVEL'] = dataS17_2['VALOR_REND_COLETAVEL'].replace({',': '.'}, regex=True)
dataS17_2['VALOR_DIVIDA_IVA'] = dataS17_2['VALOR_DIVIDA_IVA'].replace({',': '.'}, regex=True)
dataS17_2['VALOR_DIVIDA_IRS'] = dataS17_2['VALOR_DIVIDA_IRS'].replace({',': '.'}, regex=True)
```

```
# Check missing values aigain
print(dataS17_2.isnull().sum())
```

```
# Summary statistics for all variables
dataS17_2.describe(include='all').T
```

```
#Set index based on the taxpayers info
dataS17_2.set_index(['ID_CONTRIBUINTE'], inplace=True)
```

```
dataS17_2.head()
```

```
dataS17_3 = dataS17_2.copy()
```

```
# Bin the age

dataS17_3['IDADE_BINS'] = pd.cut(x=dataS17_3['IDADE'], bins=[0, 19, 37, 55, 73, 91, 109],
                                 labels=['<20','20-37','38-55','56-73','74-91','>=92'])

# Drop the age column
dataS17_3.drop(columns='IDADE', inplace=True)
```

```
dataS17_3.head()
```

```
# Bin the montant values. We set -0.00001 in all bins so that values at 0 also enter

bins_2 = np.linspace(-0.00001,332037,4)
bins_2

dataS17_3['VALOR_DIVIDA_IRS'] = dataS17_3['VALOR_DIVIDA_IRS'].astype(float)

dataS17_3['V_DIVIDA_IRS_BINS'] = pd.cut(x=dataS17_3['VALOR_DIVIDA_IRS'], bins=bins_2,
                                        labels=['<110680','110680-221358','>=221358.1'])


dataS17_3.drop(columns='VALOR_DIVIDA_IRS', inplace=True)
```

```
# Bin the montant values

bins_3 = np.linspace(-0.00001,222346.23,4)
bins_3

dataS17_3['VALOR_DIVIDA_IVA'] = dataS17_3['VALOR_DIVIDA_IVA'].astype(float)

dataS17_3['V_DIVIDA_IVA_BINS'] = pd.cut(x=dataS17_3['VALOR_DIVIDA_IVA'], bins=bins_3,
                                        labels=['<74115.42','74115.42-148230.82','>=148230.83'])


dataS17_3.drop(columns='VALOR_DIVIDA_IVA', inplace=True)
```

```
# Bin the montant values

bins_4 = np.linspace(-0.00001,1176163281,4)
bins_4

dataS17_3['VALOR_BASE_TRIB'] = dataS17_3['VALOR_BASE_TRIB'].astype(float)

dataS17_3['V_BASE_TRIB_BINS'] = pd.cut(x=dataS17_3['VALOR_BASE_TRIB'], bins=bins_4,
                                       labels=['<392054428','392054428-784108854','>=784108854.1'])


dataS17_3.drop(columns='VALOR_BASE_TRIB', inplace=True)
```

```
# Bin the montant values

bins_5 = np.linspace(-36294.3700001,1656952.26,4)
bins_5

dataS17_3['VALOR_IRS_FINAL'] = dataS17_3['VALOR_IRS_FINAL'].astype(float)

dataS17_3['V_IRS_FINAL_BINS'] = pd.cut(x=dataS17_3['VALOR_IRS_FINAL'], bins=bins_5,
                                       labels=['<528121.18','528121.18-1092536.71','>=1092536.72'])

dataS17_3.drop(columns='VALOR_IRS_FINAL', inplace=True)
```

```
# Bin the montant values

bins_6 = np.linspace(-0.00001,2914280.25,4)
bins_6

dataS17_3['VALOR_REND_COLETAVEL'] = dataS17_3['VALOR_REND_COLETAVEL'].astype(float)

dataS17_3['V_REND_COLETAVEL_BINS'] = pd.cut(x=dataS17_3['VALOR_REND_COLETAVEL'], bins=bins_6,
                                            labels=['<971426.76','971426.76- 1942853.5','>=1942853.6'])

dataS17_3.drop(columns='VALOR_REND_COLETAVEL', inplace=True)
```

```
# Bin the montant values

bins_7 = np.linspace(-0.00001,2915765.58,4)
bins_7

dataS17_3['VALOR_REND_BRUTO'] = dataS17_3['VALOR_REND_BRUTO'].astype(float)

dataS17_3['V_REND_BRUTO_BINS'] = pd.cut(x=dataS17_3['VALOR_REND_BRUTO'], bins=bins_7,
                                        labels=['<971921.87','971921.87-1943843.72','>=1943843.73'])

dataS17_3.drop(columns='VALOR_REND_BRUTO', inplace=True)
```

```
# Encode categorical variables to dummy variables

cols = ['IDADE_BINS', 'V_DIVIDA_IRS_BINS', 'V_DIVIDA_IVA_BINS', 'V_BASE_TRIB_BINS', 'V_IRS_FINAL_BINS',
        'V_REND_COLETAVEL_BINS', 'V_REND_BRUTO_BINS']
ce_one_hot = ce.OneHotEncoder(cols = cols, use_cat_names=True)
dataS17_3 = ce_one_hot.fit_transform(dataS17_3)
```

```
print(dataS17_3.shape)
dataS17_3.describe(include='all').T
```

```
dataS17_4= dataS17_3.copy()
```

```
def one_hot(s, threshold,prefix):
    """Do one-hot encoding for categories above a threshold and create a dummy column for all others named "Other"

    Args:
        s (series): series to apply the transformation
        threshold (numerical): threshold above which a dummy column should be created (from 0 to 1, corresponding from 0
        to 100%)
        prefix (string): prefix to assign to the created columns

    Returns:
        dataframe: dataframe with the applied transformations
    """
    # Check dummies for the column
    d = pd.get_dummies(s)

    # Check if percentage is below threshold and act accordingly
    f = pd.value_counts(s, sort=False, normalize=True) < threshold
    if f.sum() == 0:
        newSeries=d
    else:
        newSeries=d.loc[:, ~f].join(d.loc[:, f].sum(1).rename('Other'))
    # Add the prefix and return the result
    newSeries=newSeries.add_prefix(prefix)
    return newSeries
```

```
# One hot encoding of all categories above 7% - all others categories to "Other"
cols = ['GENERO', 'ESTADO_CIVIL', 'DISTRITO', 'TIPO_PERIODO_IVA', 'CODIGO_ATIVIDADE', 'CODIGO_DIV_ATIVIDADE', 'CODIGO_IRS',
        'INDICADOR_DIV_IRS','INDICADOR_DIV_IVA','INDICADOR_SUJEITO_IVA','INDICADOR_ATIVIDADE','INDICADOR_MOD3',
        'INDICADOR_MOD3_PRAZO','INDICADOR_TRIB_CONJUNTA','INDICADOR_DEV_ESTRATEG','INDICADOR_TRIB_AUTONOMA']
for i in range(len(cols)):
    colToProcess = dataS17_4[cols[i]].astype(str)
    procDF = one_hot(colToProcess, .07,cols[i]+'_')
    dataS17_4.drop(columns=cols[i], inplace=True)
    dataS17_4 = pd.concat([dataS17_4, procDF], axis=1)
```

```
print(dataS17_4.shape)
dataS17_4.describe(include='all').T
```

```
#Drop not significant variables
dataS17_4 .drop(columns='CODIGO_IRS_0', inplace=True)
dataS17_4 .drop(columns='CODIGO_IRS_Other',inplace=True)
dataS17_4 .drop(columns='CODIGO_ATIVIDADE_0',inplace=True)
dataS17_4 .drop(columns='CODIGO_ATIVIDADE_Other',inplace=True)
dataS17_4 .drop(columns='CODIGO_DIV_ATIVIDADE_0',inplace=True)
dataS17_4 .drop(columns='CODIGO_DIV_ATIVIDADE_Other', inplace=True)
```

```
# Get a copy of the dataframe before normalization for results' analysis
dataS17_4_beforeNorm = dataS17_4.copy(deep=True)
```

```
# Normalize all columns
# MinMax scaler returns an array, so the dataframe must be recreated
dataS17_4 = pd.DataFrame(preprocessing.MinMaxScaler().fit_transform(dataS17_4.values),
                         columns=dataS17_4.columns,index=dataS17_4.index)
```

```
# Check dataframe structure and statistics after all transformations

print(dataS17_4.shape)
dataS17_4.describe(include='all').T
```

```
#Rename the column from 'TIPO_PERIODO_IVA_Other' to 'TIPO_PERIODO_IVA_M', as there are only 2 types of VAT periods,
#monthly and quarterly
#Same for INDICADOR_DEV_ESTRATEG and INDICADOR_DIV_IVA that only can have 2 results

dataS17_4 = dataS17_4.rename(columns={'TIPO_PERIODO_IVA_Other': 'TIPO_PERIODO_IVA_M'})
dataS17_4 = dataS17_4.rename(columns={'INDICADOR_DEV_ESTRATEG_Other': 'INDICADOR_DEV_ESTRATEG_1'})
dataS17_4 = dataS17_4.rename(columns={'INDICADOR_DIV_IVA_Other': 'INDICADOR_DIV_IVA_1'})
```

```
# Check dataframe structure and statistics after all transformations

print(dataS17_4.shape)
dataS17_4.describe(include='all').T
```

```
finalS17 = dataS17_4.copy()
```

```
# Fit the PCA algorithm to data
pca = PCA().fit(finalS17)

# Show the variance per component
pcaevr = ['{:f}'.format(item) for item in pca.explained_variance_ratio_]
pcaDF = pd.DataFrame({'Component': range(1, len(finalS17.columns)+1),
                      'Variance explained': pcaevr,
                      'Cumulative variance explained': np.cumsum(pca.explained_variance_ratio_)})
pcaDF
```

```
# Plot the cumulative explained variance

# Draw
fig , ax = plt.subplots(figsize=(5, 4))
plt.plot(np.cumsum(pca.explained_variance_ratio_))

# Decoration
sns.despine()
plt.xlabel('Number of components/dimensions')
plt.ylabel('Variance explained')
plt.rc('axes', labelsize=10)
plt.title('Explained variance by components', fontsize=12)
```

Modeling - K-Means - 14 Components

With 14 components is possible to explain 96.1% of the variance.

```
# Apply the dimension reduction to the dataset (for 14 components)
pca_reduced = PCA(n_components = 14)
pca_reduced.fit(finalS17)
finalS17_pca_reduced = pca_reduced.transform(finalS17)
print(finalS17_pca_reduced.shape)
```

```
# Select K based on the sum of squared distances - Elbow method
ssd = []
K = range(1,16)
for k in K:
    km = KMeans(n_clusters=k, random_state=123)
    km = km.fit(finalS17_pca_reduced)
    ssd.append(km.inertia_)

# Show Results
# Draw
fig , ax = plt.subplots(figsize=(4, 3))
plt.plot(K, ssd, 'bx-')
# Decoration
sns.despine()
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
plt.xlabel('K')
plt.ylabel('Sum of squared distances')
plt.rc('axes', labelsize=subPlots_label_fontSize)
ax.xaxis.set_major_locator(ticker.MaxNLocator(integer=True))
plt.title('Elbow method - Reduced dimensionality', fontsize=plots_Title_fontSize);

plt.show()
```

```
# Select K based on the Silhouette method
sil = []

# Dissimilarity can only be measured in more than 1 partition, therefore it starts on K=2
for k in range(2,20):
  km = KMeans(n_clusters = k, random_state=123).fit(finalS17)
  labels = km.labels_
  sil.append(silhouette_score(finalS17, labels, metric = 'euclidean'))


# Show Results
# Draw
fig , ax = plt.subplots(figsize=(5, 4))
plt.plot(range(2, 20), sil, 'bx-')
# Decoration
sns.despine()
fmt = "{x:,.2f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
plt.xlabel('K')
plt.ylabel('Silhouette score')
plt.rc('axes', labelsize=subPlots_label_fontSize)
ax.xaxis.set_major_locator(ticker.MaxNLocator(integer=True))
plt.title('Silhouette method - Reduced dimensionality', fontsize=plots_Title_fontSize);

plt.show()
```

```
# Select K based on the Davies-Boulding method
results = {}

for i in range(2,11):
    kmeans = KMeans(n_clusters=i, random_state=30)
    labels = kmeans.fit_predict(finalS17_pca_reduced)
    db_index = davies_bouldin_score(finalS17_pca_reduced, labels)
    results.update({i: db_index})

plt.plot(list(results.keys()), list(results.values()))
plt.xlabel("Number of clusters")
plt.ylabel("Davies-Boulding Index")
plt.show()
```

```
K=4
kmeans = KMeans(n_clusters=K, random_state=123)
allDistances = kmeans.fit_transform(finalS17_pca_reduced)
y_kmeans = kmeans.predict(finalS17_pca_reduced)
```

```
# Plot clusters cardinality
# Count observations per cluster
freqByCluster = dataS17_4_beforeNorm.groupby(y_kmeans).size()

# Draw
fig, ax = plt.subplots(figsize=(7,5))
g = sns.countplot(x=y_kmeans, color='grey')

# Decoration
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
for index,data in enumerate(freqByCluster):
    plt.text(x=index-0.2 , y=data+50 , s=f"{data}" , fontdict=dict(fontsize=plots_barTexts_fontSize))
sns.despine()
plt.title("Cluster cardinality", fontsize=plots_Title_fontSize)
plt.xlabel("Cluster")
plt.ylabel("Frequency in cluster")
plt.rc('axes', labelsize=subPlots_label_fontSize)
plt.show();
```

```
# Plot clusters magnitude
# Compute Magnitude
finalS17['distanceToCentroid'] = np.min(allDistances,axis=1)
magnitude = finalS17['distanceToCentroid'].groupby(y_kmeans).sum()
finalS17 = finalS17.drop(columns=['distanceToCentroid'])

# Draw
fig, ax = plt.subplots(figsize=(7,5))
g = sns.barplot(x=magnitude.index, y=magnitude.values, color='grey')

# Decoration
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
for index,data in enumerate(magnitude):
    plt.text(x=index-0.2 , y=data+50 , s=f"{data:,.0f}" , fontdict=dict(fontsize=plots_barTexts_fontSize))
sns.despine()
plt.title("Cluster magnitude", fontsize=plots_Title_fontSize)
plt.xlabel("Cluster")
plt.ylabel("Sum of distances to centroid")
plt.rc('axes', labelsize=subPlots_label_fontSize)
plt.show();
```

```
# Plot cardinality vs magnitude
# Draw
fig, ax = plt.subplots(figsize=(6,4))
g = sns.regplot(x=freqByCluster, y=magnitude, scatter=True, seed=123,truncate=False, ci=None)

# Decoration
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.xaxis.set_major_formatter(tick)
ax.yaxis.set_major_formatter(tick)
sns.despine()
plt.title("Cardinality vs Magnitude", fontsize=plots_Title_fontSize)
plt.xlabel("Cardinality")
plt.ylabel("Magnitude")
plt.rc('axes', labelsize=subPlots_label_fontSize)
plt.show();
```

Let's use PCA to visualize with only two components

```
# Scatter plot of the two Principal Components by cluster to see if there is any pattern
# Draw
fig, ax = plt.subplots(figsize=(7,5))
plt.scatter(finalS17_pca_reduced[:, 0], finalS17_pca_reduced[:, 1], c=y_kmeans, s=50, cmap='viridis')
centers = kmeans.cluster_centers_
ax.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5)

# Decoration
plt.xlabel('Component 1')
plt.ylabel('Component 2')
plt.colorbar();
plt.show()
```

```
# Create a dataframe with the weights per component
wheightsDFS17 = pd.DataFrame(pca.components_,columns=finalS17.columns)
wheightsDFS17
```

```
# Study the weights of component 1
compS17 = abs(wheightsDFS17.iloc[0])  ## index 0 is component 1
compS17 = compS17.sort_values(ascending=False)
compS17
```

```
## Check the mean values of each cluster for top 52 relevant features
top_features = compS17[:53].index
clustersMeanDFS17 = pd.DataFrame(finalS17[top_features].groupby(y_kmeans).mean(),columns=top_features)
clustersMeanDFS17.transpose()
```

Business 2017

```
dataC17 = data.copy()
```

```
#Remove Individual
dataC17.drop(dataC17.index[(dataC17["TIPO_NIF"] == 'S')],axis=0,inplace=True)
```

```
#Drop the years that are not used
dataC17.drop(dataC17.index[(dataC17["ANO_EXERCICIO"] == 2018)],axis=0,inplace=True)
dataC17.drop(dataC17.index[(dataC17["ANO_EXERCICIO"] == 2019)],axis=0,inplace=True)
dataC17.drop(dataC17.index[(dataC17["ANO_EXERCICIO"] == 2020)],axis=0,inplace=True)
dataC17.drop(dataC17.index[(dataC17["ANO_EXERCICIO"] == 2021)],axis=0,inplace=True)
```

DATA ANALYSIS

```
dataC17_1 = dataC17.copy()
```

```
#Check the size of the database
dataC17_1.shape
```

```
#Check for false values
dataC17_1['INDICADOR_DIV_IRC'].value_counts()
```

```
#Check for false values
dataC17_1['INDICADOR_DIV_IVA'].value_counts()
```

```
#Check for false values
dataC17_1['INDICADOR_ATIVIDADE'].value_counts()
```

```
#Check for false values
dataC17_1['TIPO_PERIODO_IVA'].value_counts()
```

```
#Check for false values
dataC17_1['INDICADOR_SUJEITO_IVA'].value_counts()
```

```
#Check for false values
dataC17_1['CODIGO_DIV_ATIVIDADE'].value_counts()
```

```
#Check for false values
dataC17_1['DISTRITO'].value_counts()
```

```
#Check for false values
dataC17_1['INDICADOR_INSOLVENCIA'].value_counts()
```

```
#Check for false values
dataC17_1['INDICADOR_DEV_ESTRATEG'].value_counts()
```

```
# Activity
#1    24634
#0     5289

x = [24634, 5289]
labels = ['Yes', 'No']
colors = ['tab:green', 'tab:grey']

fig, ax = plt.subplots()
ax.pie(x, labels = labels, colors = colors, autopct='%1.1f%%')
ax.set_title('Activity')
plt.show()
```

```
# VAT Period

by = sns.histplot(dataC17_1["TIPO_PERIODO_IVA"] , color="grey", kde = False, bins = 12)
plt.title('VAT Period')
plt.ylabel('Number of Taxpayers')
by.set_xlabel('Type of Period')
by.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))
by.yaxis.grid(True)
by.xaxis.grid(False)

plt.show()
```

```
#Liable to VAT
labels = ['Yes', 'No']
sizes = [21312, 8611]
colors = ['tab:green', 'tab:grey']


plt.pie(sizes, labels=labels, colors = colors, autopct='%1.1f%%', startangle=90, wedgeprops=dict(width=0.54))
plt.title('Liable to VAT')
plt.show()
```

```
dataC17_1.head(20)
```

DATA PREPARATION

```
dataC17_2 = dataC17_1.copy()
```

```
# Remove the columNS that should not be used because are business variables
dataC17_2.drop(columns=['GENERO','ESTADO_CIVIL','ANO_NASCIMENTO','TIPO_NIF','ANO_EXERCICIO','CODIGO_IRS',
                        'VALOR_REND_BRUTO','VALOR_IRS_FINAL','VALOR_REND_COLETAVEL','INDICADOR_MOD3',
                        'INDICADOR_MOD3_PRAZO','INDICADOR_TRIB_CONJUNTA','INDICADOR_TRIB_AUTONOMA','INDICADOR_DIV_IRS',
                        'VALOR_DIVIDA_IRS'], inplace=True)
```

```
#Correlation matrix

plt.figure(figsize = (20, 10))
custom_cmap = sns.diverging_palette(220, 20, as_cmap=True)
sns.heatmap(dataC17_2.corr(), annot = True, linewidths=.5, cmap = custom_cmap, vmin = -1)
plt.title('Correlation Matrix', fontsize = 20)
plt.show()
```

```
# Check missing values
print(dataC17_2.isnull().sum())
```

```
#Drop column if they only contain missing values

dataC17_2 = dataC17_2.dropna(how='all')
dataC17_2.shape
```

```
#We will replace null values with 0 for numerical values and 'Na' for categorical values, as it makes sense to represent
#them this way since they are non-existent
dataC17_2['VALOR_BASE_TRIB'] = dataC17_2['VALOR_BASE_TRIB'].replace(np.nan, 0)
dataC17_2['VALOR_VOLUME_NEG'] = dataC17_2['VALOR_VOLUME_NEG'].replace(np.nan, 0)
dataC17_2['VALOR_DIVIDA_IRC'] = dataC17_2['VALOR_DIVIDA_IRC'].replace(np.nan, 0)
dataC17_2['VALOR_DIVIDA_IVA'] = dataC17_2['VALOR_DIVIDA_IVA'].replace(np.nan, 0)
dataC17_2['CODIGO_DIV_ATIVIDADE'] = dataC17_2['CODIGO_DIV_ATIVIDADE'].replace(np.nan, 0)
dataC17_2['TIPO_PERIODO_IVA'] = dataC17_2['TIPO_PERIODO_IVA'].fillna('Na')
```

```
#Replace ',' with '.' to use the value format
dataC17_2['VALOR_BASE_TRIB'] = dataC17_2['VALOR_BASE_TRIB'].replace({',': '.'}, regex=True)
dataC17_2['VALOR_VOLUME_NEG'] = dataC17_2['VALOR_VOLUME_NEG'].replace({',': '.'}, regex=True)
dataC17_2['VALOR_DIVIDA_IRC'] = dataC17_2['VALOR_DIVIDA_IRC'].replace({',': '.'}, regex=True)
dataC17_2['VALOR_DIVIDA_IVA'] = dataC17_2['VALOR_DIVIDA_IVA'].replace({',': '.'}, regex=True)
```

```
# Check missing values again
print(dataC17_2.isnull().sum())
```

```
#Set index based on the taxpayers info
dataC17_2.set_index(['ID_CONTRIBUINTE'], inplace=True)
```

```
dataC17_3 = dataC17_2.copy()
```

```
# Bin the montant values
bins_1 = np.linspace(-0.00001,3724999.09,4)
bins_1

dataC17_3['VALOR_DIVIDA_IRC'] = dataC17_3['VALOR_DIVIDA_IRC'].astype(float)

dataC17_3['V_DIVIDA_IRC_BINS'] = pd.cut(x=dataC17_3['VALOR_DIVIDA_IRC'], bins=bins_1,
                                        labels=['<1241666.37','1241666.37-2483332.72','>=2483332.73'])


dataC17_3.drop(columns='VALOR_DIVIDA_IRC', inplace=True)
```

```
# Bin the montant values

bins_2 = np.linspace(-0.00001,7677183.94,4)
bins_2

dataC17_3['VALOR_DIVIDA_IVA'] = dataC17_3['VALOR_DIVIDA_IVA'].astype(float)

dataC17_3['V_DIVIDA_IVA_BINS'] = pd.cut(x=dataC17_3['VALOR_DIVIDA_IVA'], bins=bins_2,
                                        labels=['<2559061.32','2559061.32-5118122.62','>=5118122.63'])


dataC17_3.drop(columns='VALOR_DIVIDA_IVA', inplace=True)
```

```
# Bin the montant values

bins_3 = np.linspace(-0.00001,3957189698,4)
bins_3

dataC17_3['VALOR_BASE_TRIB'] = dataC17_3['VALOR_BASE_TRIB'].astype(float)

dataC17_3['V_BASE_TRIB_BINS'] = pd.cut(x=dataC17_3['VALOR_BASE_TRIB'], bins=bins_3,
                                       labels=['<1319063231','1319063231-2638126470','>=2638126471'])


dataC17_3.drop(columns='VALOR_BASE_TRIB', inplace=True)
```

```
# Bin the montant values

bins_4 = np.linspace(-0.00001,2851773045,4)
bins_4

dataC17_3['VALOR_VOLUME_NEG'] = dataC17_3['VALOR_VOLUME_NEG'].astype(float)

dataC17_3['V_VOLUME_NEG_BINS'] = pd.cut(x=dataC17_3['VALOR_VOLUME_NEG'], bins=bins_4,
                                        labels=['<950591016','950591016-1901182030','>=1901182031'])

dataC17_3.drop(columns='VALOR_VOLUME_NEG', inplace=True)
```

```
# Encode categorical variables to dummy variables

cols = ['V_DIVIDA_IRC_BINS', 'V_DIVIDA_IVA_BINS', 'V_BASE_TRIB_BINS','V_VOLUME_NEG_BINS']
ce_one_hot = ce.OneHotEncoder(cols = cols, use_cat_names=True)
dataC17_3 = ce_one_hot.fit_transform(dataC17_3)
```

```
print(dataC17_3.shape)
dataC17_3.describe(include='all').T
```

```
dataC17_4= dataC17_3.copy()
```

```
def one_hot(s, threshold,prefix):
    """Do one-hot encoding for categories above a threshold and create a dummy column for all others named "Other"

    Args:
        s (series): series to apply the transformation
        threshold (numerical): threshold above which a dummy column should be created (from 0 to 1, corresponding
        from 0 to 100%)
        prefix (string): prefix to assign to the created columns

    Returns:
        dataframe: dataframe with the applied transformations
    """
    # Check dummies for the column
    d = pd.get_dummies(s)

    # Check if percentage is below threshold and act accordingly
    f = pd.value_counts(s, sort=False, normalize=True) < threshold
    if f.sum() == 0:
        newSeries=d
    else:
        newSeries=d.loc[:, ~f].join(d.loc[:, f].sum(1).rename('Other'))
    # Add the prefix and return the result
    newSeries=newSeries.add_prefix(prefix)
    return newSeries
```

```
# One hot encoding of all categories above 7% - all others categories to "Other"
cols = ['DISTRITO', 'TIPO_PERIODO_IVA', 'CODIGO_ATIVIDADE', 'CODIGO_DIV_ATIVIDADE','INDICADOR_DIV_IRC','INDICADOR_DIV_IVA',
'INDICADOR_SUJEITO_IVA', 'INDICADOR_ATIVIDADE', 'INDICADOR_DEV_ESTRATEG', 'INDICADOR_INSOLVENCIA']
for i in range(len(cols)):
    colToProcess = dataC17_4[cols[i]].astype(str)
    procDF = one_hot(colToProcess, .07,cols[i]+'_')
    dataC17_4.drop(columns=cols[i], inplace=True)
    dataC17_4 = pd.concat([dataC17_4, procDF], axis=1)
```

```
print(dataC17_4.shape)
dataC17_4.describe(include='all').T
```

```
#Drop not significant variables
dataC17_4 .drop(columns='CODIGO_ATIVIDADE_0',inplace=True)
dataC17_4 .drop(columns='CODIGO_ATIVIDADE_Other',inplace=True)
```

```
# Get a copy of the dataframe before normalization for results' analysis
dataC17_4_beforeNorm = dataC17_4.copy(deep=True)
```

```
# Normalize all columns
# MinMax scaler returns an array, so the dataframe must be recreated
dataC17_4 = pd.DataFrame(preprocessing.MinMaxScaler().fit_transform(dataC17_4.values),
                         columns=dataC17_4.columns,index=dataC17_4.index)
```

```
# Check dataframe structure and statistics after all transformations

print(dataC17_4.shape)
dataC17_4.describe(include='all').T
```

```
#Rename the INDICADOR_DEV_ESTRATEG_Other column to INDICADOR_DEV_ESTRATEG_1, as there are only 2 types of options
#Same for INDICADOR_INSOLVENCIA
dataC17_4 = dataC17_4.rename(columns={'INDICADOR_DEV_ESTRATEG_Other': 'INDICADOR_DEV_ESTRATEG_1'})
dataC17_4 = dataC17_4.rename(columns={'INDICADOR_INSOLVENCIA_Other': 'INDICADOR_INSOLVENCIA_1'})
```

```
finalC17 = dataC17_4.copy()
```

```
# Fit the PCA algorithm to data
pca = PCA().fit(finalC17)

# Show the variance per component
pcaevr = ['{:f}'.format(item) for item in pca.explained_variance_ratio_]
pcaDF = pd.DataFrame({'Component': range(1, len(finalC17.columns)+1),
                      'Variance explained': pcaevr,
                      'Cumulative variance explained': np.cumsum(pca.explained_variance_ratio_)})
pcaDF
```

```
# Plot the cumulative explained variance

# Draw
fig , ax = plt.subplots(figsize=(5, 4))
plt.plot(np.cumsum(pca.explained_variance_ratio_))

# Decoration
sns.despine()
plt.xlabel('Number of components/dimensions')
plt.ylabel('Variance explained')
plt.rc('axes', labelsize=10)
plt.title('Explained variance by components', fontsize=12)
```

Modeling - K-Means - 10 Components

With 10 components is possible to explain 97% of the variance.

```
# Apply the dimension reduction to the dataset (for 10 components)
pca_reduced = PCA(n_components = 10)
pca_reduced.fit(finalC17)
finalC17_pca_reduced = pca_reduced.transform(finalC17)
print(finalC17_pca_reduced.shape)
```

```
# Select K based on the sum of squared distances - Elbow method
ssd = []
K = range(1,18)
for k in K:
    km = KMeans(n_clusters=k, random_state=123)
    km = km.fit(finalC17_pca_reduced)
    ssd.append(km.inertia_)

# Show Results
# Draw
fig , ax = plt.subplots(figsize=(5, 4))
plt.plot(K, ssd, 'bx-')
# Decoration
sns.despine()
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
plt.xlabel('K')
plt.ylabel('Sum of squared distances')
plt.rc('axes', labelsize=subPlots_label_fontSize)
ax.xaxis.set_major_locator(ticker.MaxNLocator(integer=True))
plt.title('Elbow method - Reduced dimensionality', fontsize=plots_Title_fontSize);

plt.show()
```

```
# Select K based on the Silhouette method
sil = []

# Dissimilarity can only be measured in more than 1 partition, therefore it starts on K=2
for k in range(2,20):
  km = KMeans(n_clusters = k, random_state=123).fit(finalC17)
  labels = km.labels_
  sil.append(silhouette_score(finalC17, labels, metric = 'euclidean'))


# Show Results
# Draw
fig , ax = plt.subplots(figsize=(5, 4))
plt.plot(range(2, 20), sil, 'bx-')
# Decoration
sns.despine()
fmt = "{x:,.2f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
plt.xlabel('K')
plt.ylabel('Silhouette score')
plt.rc('axes', labelsize=subPlots_label_fontSize)
ax.xaxis.set_major_locator(ticker.MaxNLocator(integer=True))
plt.title('Silhouette method - Reduced dimensionality', fontsize=plots_Title_fontSize);

plt.show()
```

```
#Davies-Boulding Index
results = {}

for i in range(2,11):
    kmeans = KMeans(n_clusters=i, random_state=30)
    labels = kmeans.fit_predict(finalC17_pca_reduced)
    db_index = davies_bouldin_score(finalC17_pca_reduced, labels)
    results.update({i: db_index})

plt.plot(list(results.keys()), list(results.values()))
plt.xlabel("Number of clusters")
plt.ylabel("Davies-Boulding Index")
plt.show()
```

```
K=2
kmeans = KMeans(n_clusters=K, random_state=123)
allDistances = kmeans.fit_transform(finalC17_pca_reduced)
y_kmeans = kmeans.predict(finalC17_pca_reduced)
```

```
# Plot clusters cardinality
# Count observations per cluster
freqByCluster = dataC17_4_beforeNorm.groupby(y_kmeans).size()

# Draw
fig, ax = plt.subplots(figsize=(7,5))
g = sns.countplot(x=y_kmeans, color='grey')

# Decoration
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
for index,data in enumerate(freqByCluster):
    plt.text(x=index-0.2 , y=data+50 , s=f"{data}" , fontdict=dict(fontsize=plots_barTexts_fontSize))
sns.despine()
plt.title("Cluster cardinality", fontsize=plots_Title_fontSize)
plt.xlabel("Cluster")
plt.ylabel("Frequency in cluster")
plt.rc('axes', labelsize=subPlots_label_fontSize)
plt.show();
```

```
# Plot clusters magnitude
# Compute Magnitude
finalC17['distanceToCentroid'] = np.min(allDistances,axis=1)
magnitude = finalC17['distanceToCentroid'].groupby(y_kmeans).sum()
finalC17 = finalC17.drop(columns=['distanceToCentroid'])

# Draw
fig, ax = plt.subplots(figsize=(7,5))
g = sns.barplot(x=magnitude.index, y=magnitude.values, color='grey')

# Decoration
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
for index,data in enumerate(magnitude):
    plt.text(x=index-0.2 , y=data+50 , s=f"{data:,.0f}" , fontdict=dict(fontsize=plots_barTexts_fontSize))
sns.despine()
plt.title("Cluster magnitude", fontsize=plots_Title_fontSize)
plt.xlabel("Cluster")
plt.ylabel("Sum of distances to centroid")
plt.rc('axes', labelsize=subPlots_label_fontSize)
plt.show();
```

```
# Plot cardinality vs magnitude
# Draw
fig, ax = plt.subplots(figsize=(6,4))
g = sns.regplot(x=freqByCluster, y=magnitude, scatter=True, seed=123,truncate=False, ci=None)

# Decoration
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.xaxis.set_major_formatter(tick)
ax.yaxis.set_major_formatter(tick)
sns.despine()
plt.title("Cardinality vs Magnitude", fontsize=plots_Title_fontSize)
plt.xlabel("Cardinality")
plt.ylabel("Magnitude")
plt.rc('axes', labelsize=subPlots_label_fontSize)
plt.show();
```

Let's use PCA to visualize with only two components

```
# Scatter plot of the two Principal Components by cluster to see if there is any pattern
# Draw
fig, ax = plt.subplots(figsize=(7,5))
plt.scatter(finalC17_pca_reduced[:, 0], finalC17_pca_reduced[:, 1], c=y_kmeans, s=50, cmap='viridis')
centers = kmeans.cluster_centers_
ax.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5)

# Decoration
plt.xlabel('Component 1')
plt.ylabel('Component 2')
plt.colorbar();
plt.show()
```

```
finalC17.head()
```

```
# Create a dataframe with the weights per component
wheightsDFC17 = pd.DataFrame(pca.components_,columns=finalC17.columns)
wheightsDFC17
```

```
# Study the weights of component 1
compC17 = abs(wheightsDFC17.iloc[0])  ## index 0 is component 1
compC17 = compC17.sort_values(ascending=False)
compC17
```

```
## Check the mean values of each cluster for top 35 relevant features
top_features = compC17[:36].index
clustersMeanDFC17 = pd.DataFrame(finalC17[top_features].groupby(y_kmeans).mean(),columns=top_features)
clustersMeanDFC17.transpose()
```

Individual 2021

```
dataS21 = data.copy()
```

```
#Remove the business
dataS21.drop(dataS21.index[(dataS21["TIPO_NIF"] == 'C')],axis=0,inplace=True)
```

```
#Drop the years that are not used
dataS21.drop(dataS21.index[(dataS21["ANO_EXERCICIO"] == 2018)],axis=0,inplace=True)
dataS21.drop(dataS21.index[(dataS21["ANO_EXERCICIO"] == 2019)],axis=0,inplace=True)
dataS21.drop(dataS21.index[(dataS21["ANO_EXERCICIO"] == 2020)],axis=0,inplace=True)
dataS21.drop(dataS21.index[(dataS21["ANO_EXERCICIO"] == 2017)],axis=0,inplace=True)
```

```
dataS21.shape
```

DATA ANALYSIS

```
#Check for false values
dataS21['ANO_NASCIMENTO'].value_counts()
```

```
#Check for false values
dataS21['INDICADOR_DIV_IRS'].value_counts()
```

```
#Check for false values
dataS21['INDICADOR_DIV_IVA'].value_counts()
```

```
#Check for false values
dataS21['TIPO_PERIODO_IVA'].value_counts()
```

```
#Check for false values
dataS21['INDICADOR_SUJEITO_IVA'].value_counts()
```

```
#Check for false values
dataS21['INDICADOR_ATIVIDADE'].value_counts()
```

```
#Check for false values
dataS21['INDICADOR_MOD3'].value_counts()
```

```
#Check for false values
dataS21['INDICADOR_MOD3_PRAZO'].value_counts()
```

```
#Check for false values
dataS21['INDICADOR_TRIB_CONJUNTA'].value_counts()
```

```
#Check for false values
dataS21['INDICADOR_TRIB_AUTONOMA'].value_counts()
```

```
#Check for false values
dataS21['DISTRITO'].value_counts()
```

```
#Marital Status

ed = sns.countplot(y="ESTADO_CIVIL", data=dataS21, palette = 'Greens', order=['S','C','V','U','F'])
plt.title('Taxpayers Marital Status')
plt.xlabel('Count')
plt.ylabel('Marital Status')
#plt.xticks(np.arange(0,20,2))
ed.get_xaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))
ed.yaxis.grid(False)
ed.xaxis.grid(True)

total = len(dataS21['ESTADO_CIVIL'])
for p in ed.patches:
        percentage = '{:.1f}%'.format(100 * p.get_width()/total)
        x = p.get_x() + p.get_width() + 50
        y = p.get_y() + p.get_height()/2
        ed.annotate(percentage, (x, y))

plt.show()
```

```
# Activity
#0    54678
#1    16462

x = [16462, 54678]
labels = ['Yes', 'No']
colors = ['tab:green', 'tab:grey']

fig, ax = plt.subplots()
ax.pie(x, labels = labels, colors = colors, autopct='%1.1f%%')
ax.set_title('Activity')
plt.show()
```

```
# VAT Period

by = sns.histplot(dataS21["TIPO_PERIODO_IVA"] , color="grey", kde = False, bins = 12)
plt.title('VAT Period')
plt.ylabel('Number of Taxpayers')
by.set_xlabel('Type of Period')
by.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))
by.yaxis.grid(True)
by.xaxis.grid(False)

plt.show()
```

```
# Birth Year

by = sns.histplot(dataS21["ANO_NASCIMENTO"] , color="green", kde = False, bins = 12)
plt.title('Birth Distribution')
plt.ylabel('Number of Taxpayers')
by.set_xlabel('Year of Birth')
by.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))
by.yaxis.grid(True)
by.xaxis.grid(False)

plt.show()
```

```
#Submitted Tax Return Form 3

labels = ['Yes', 'No']
sizes = [59786, 11354]
colors = ['tab:green', 'tab:grey']

plt.pie(sizes, labels=labels, colors = colors, autopct='%1.1f%%', startangle=90, wedgeprops=dict(width=0.5))
plt.title('Submitted Tax Return Form 3')
plt.show()
```

```
#IQR
Q1 = dataS21['ANO_NASCIMENTO'].quantile(0.25)
Q3 = dataS21['ANO_NASCIMENTO'].quantile(0.75)
IQR = Q3 - Q1

print('IQR method')
print('Lower outliers: ',((dataS21['ANO_NASCIMENTO'] < (Q1 - 1.5 * IQR))).sum())
print('Upper outliers: ',((dataS21['ANO_NASCIMENTO'] > (Q3 + 1.5 * IQR))).sum())
print('Total outliers: ',((dataS21['ANO_NASCIMENTO'] < (Q1 - 1.5 * IQR)) | (dataS21['ANO_NASCIMENTO'] > (Q3 + 1.5 * IQR))).s
print('Lower boundary: ',Q1 - 1.5 * IQR)
print('Upper boundary: ',Q3 + 1.5 * IQR)

print('\n')

dataS21[['ANO_NASCIMENTO']].boxplot()
axes = plt.gca()
axes.set_title('Birth Year')
axes.set_ylabel('Year')
axes.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))

plt.show()
```

DATA PREPARATION

```
dataS21_1 = dataS21.copy()
```

```
#Age

now=2021

dataS21_1['IDADE'] = (now - dataS21_1['ANO_NASCIMENTO'])
dataS21_1.head()
```

```
# When we decided to delete the values where the age of the taxpayers is bigger than 102.5 (Considering the minimum
#as 1914.5 and the maximum as the current year to remove outliers)

dataS21_1 = dataS21_1.drop(dataS21_1.index[dataS21_1["ANO_NASCIMENTO"] < 1915])
```

```
dataS21_2 = dataS21_1.copy()
```

```
# Remove the columNS that should not be used because are business variables
dataS21_2.drop(columns=['VALOR_VOLUME_NEG','TIPO_NIF', 'ANO_EXERCICIO','INDICADOR_INSOLVENCIA','INDICADOR_DIV_IRC',
                        'VALOR_DIVIDA_IRC','ANO_NASCIMENTO'], inplace=True)
```

```
#VCheck the size of the database
dataS21_2.shape
```

```
#Correlation matrix

plt.figure(figsize = (20, 10))
custom_cmap = sns.diverging_palette(220, 20, as_cmap=True)
sns.heatmap(dataS21_2.corr(), annot = True, linewidths=.5, cmap = custom_cmap, vmin = -1)
plt.title('Correlation Matrix', fontsize = 20)
plt.show()
```

```
# Check missing values
print(dataS21_2.isnull().sum())
```

```
#Drop column if they only contain missing values

dataS21_2 = dataS21_2.dropna(how='all')
dataS21_2.shape
```

```
#Since it doesn't make sense to have individual taxpayers without marital status and they only represent 0.037% of the
#individual taxpayer data for 2017, let's remove them from our database.

dataS21_2 = dataS21_2.dropna(subset=['ESTADO_CIVIL'])
dataS21_2.shape
```

```
#We will replace null values with 0 for numerical values and 'Na' for categorical values, as it makes sense to represent
#them this way since they are non-existent
Na para categóricos
dataS21_2['VALOR_BASE_TRIB'] = dataS21_2['VALOR_BASE_TRIB'].replace(np.nan, 0)
dataS21_2['VALOR_REND_BRUTO'] = dataS21_2['VALOR_REND_BRUTO'].replace(np.nan, 0)
dataS21_2['VALOR_IRS_FINAL'] = dataS21_2['VALOR_IRS_FINAL'].replace(np.nan, 0)
dataS21_2['VALOR_REND_COLETAVEL'] = dataS21_2['VALOR_REND_COLETAVEL'].replace(np.nan, 0)
dataS21_2['VALOR_DIVIDA_IVA'] = dataS21_2['VALOR_DIVIDA_IVA'].replace(np.nan, 0)
dataS21_2['VALOR_DIVIDA_IRS'] = dataS21_2['VALOR_DIVIDA_IRS'].replace(np.nan, 0)
dataS21_2['TIPO_PERIODO_IVA'] = dataS21_2['TIPO_PERIODO_IVA'].fillna('Na')
```

```
#Replace ',' with '.' to use the value format
dataS21_2['VALOR_BASE_TRIB'] = dataS21_2['VALOR_BASE_TRIB'].replace({',': '.'}, regex=True)
dataS21_2['VALOR_REND_BRUTO'] = dataS21_2['VALOR_REND_BRUTO'].replace({',': '.'}, regex=True)
dataS21_2['VALOR_IRS_FINAL'] = dataS21_2['VALOR_IRS_FINAL'].replace({',': '.'}, regex=True)
dataS21_2['VALOR_REND_COLETAVEL'] = dataS21_2['VALOR_REND_COLETAVEL'].replace({',': '.'}, regex=True)
dataS21_2['VALOR_DIVIDA_IVA'] = dataS21_2['VALOR_DIVIDA_IVA'].replace({',': '.'}, regex=True)
dataS21_2['VALOR_DIVIDA_IRS'] = dataS21_2['VALOR_DIVIDA_IRS'].replace({',': '.'}, regex=True)
```

```
# Check missing values again
print(dataS21_2.isnull().sum())
```

```
#Set index based on the taxpayers info
dataS21_2.set_index(['ID_CONTRIBUINTE'], inplace=True)
```

```
dataS21_2.head()
```

```
dataS21_3 = dataS21_2.copy()
```

```
# Bin the age

dataS21_3['IDADE_BINS'] = pd.cut(x=dataS21_3['IDADE'], bins=[0, 19, 37, 55, 73, 91, 109],
                                 labels=['<20','20-37','38-55','56-73','74-91','>=92'])

# Drop the age column
dataS21_3.drop(columns='IDADE', inplace=True)
```

```
# Bin the montant values

bins_2 = np.linspace(-0.00001,492915.82,4)
bins_2

dataS21_3['VALOR_DIVIDA_IRS'] = dataS21_3['VALOR_DIVIDA_IRS'].astype(float)

dataS21_3['V_DIVIDA_IRS_BINS'] = pd.cut(x=dataS21_3['VALOR_DIVIDA_IRS'], bins=bins_2,
                                        labels=['<164305.28','164305.28-328610.54','>=328610.55'])


dataS21_3.drop(columns='VALOR_DIVIDA_IRS', inplace=True)
```

```
# Bin the montant values

bins_3 = np.linspace(-0.00001,102343.51,4)
bins_3

dataS21_3['VALOR_DIVIDA_IVA'] = dataS21_3['VALOR_DIVIDA_IVA'].astype(float)

dataS21_3['V_DIVIDA_IVA_BINS'] = pd.cut(x=dataS21_3['VALOR_DIVIDA_IVA'], bins=bins_3,
                                        labels=['<34114.6','34114.6-68229','>=68229.1'])


dataS21_3.drop(columns='VALOR_DIVIDA_IVA', inplace=True)
```

```
# Bin the montant values

bins_4 = np.linspace(-0.00001,12195121.96,4)
bins_4

dataS21_3['VALOR_BASE_TRIB'] = dataS21_3['VALOR_BASE_TRIB'].astype(float)

dataS21_3['V_BASE_TRIB_BINS'] = pd.cut(x=dataS21_3['VALOR_BASE_TRIB'], bins=bins_4,
                                       labels=['<4065040.66','4065040.66-8130081.3','>=8130081.31'])


dataS21_3.drop(columns='VALOR_BASE_TRIB', inplace=True)
```

```
# Bin the montant values

bins_5 = np.linspace(-51227.4100001,2877190.255,4)
bins_5

dataS21_3['VALOR_IRS_FINAL'] = dataS21_3['VALOR_IRS_FINAL'].astype(float)

dataS21_3['V_IRS_FINAL_BINS'] = pd.cut(x=dataS21_3['VALOR_IRS_FINAL'], bins=bins_5,
                                       labels=['<924911.82','924911.82-1901051','>=1901051.1'])

dataS21_3.drop(columns='VALOR_IRS_FINAL', inplace=True)
```

```
# Bin the montant values

bins_6 = np.linspace(-0.00001,5520102.625,4)
bins_6

dataS21_3['VALOR_REND_COLETAVEL'] = dataS21_3['VALOR_REND_COLETAVEL'].astype(float)

dataS21_3['V_REND_COLETAVEL_BINS'] = pd.cut(x=dataS21_3['VALOR_REND_COLETAVEL'], bins=bins_6,
                                            labels=['<1840034.21','1840034.21- 3680068.41','>=3680068.42'])

dataS21_3.drop(columns='VALOR_REND_COLETAVEL', inplace=True)
```

```
# Bin the montant values

bins_7 = np.linspace(-0.00001,5524206.625,4)
bins_7

dataS21_3['VALOR_REND_BRUTO'] = dataS21_3['VALOR_REND_BRUTO'].astype(float)

dataS21_3['V_REND_BRUTO_BINS'] = pd.cut(x=dataS21_3['VALOR_REND_BRUTO'], bins=bins_7,
                                        labels=['<1841402.21','1841402.21-3682804.41','>=3682804.42'])

dataS21_3.drop(columns='VALOR_REND_BRUTO', inplace=True)
```

```
dataS21_3.head()
```

```
# Encode categorical variables to dummy variables

cols = ['IDADE_BINS', 'V_DIVIDA_IRS_BINS', 'V_DIVIDA_IVA_BINS', 'V_BASE_TRIB_BINS', 'V_IRS_FINAL_BINS',
        'V_REND_COLETAVEL_BINS', 'V_REND_BRUTO_BINS']
ce_one_hot = ce.OneHotEncoder(cols = cols, use_cat_names=True)
dataS21_3 = ce_one_hot.fit_transform(dataS21_3)
```

```
print(dataS21_3.shape)
dataS21_3.describe(include='all').T
```

```
dataS21_4= dataS21_3.copy()
```

```
def one_hot(s, threshold,prefix):
    """Do one-hot encoding for categories above a threshold and create a dummy column for all others named "Other"

    Args:
        s (series): series to apply the transformation
        threshold (numerical): threshold above which a dummy column should be created (from 0 to 1, corresponding from 0
        to 100%)
        prefix (string): prefix to assign to the created columns

    Returns:
        dataframe: dataframe with the applied transformations
    """
    # Check dummies for the column
    d = pd.get_dummies(s)

    # Check if percentage is below threshold and act accordingly
    f = pd.value_counts(s, sort=False, normalize=True) < threshold
    if f.sum() == 0:
        newSeries=d
    else:
        newSeries=d.loc[:, ~f].join(d.loc[:, f].sum(1).rename('Other'))
    # Add the prefix and return the result
    newSeries=newSeries.add_prefix(prefix)
    return newSeries
```

```
# One hot encoding of all categories above 7% - all others categories to "Other"
cols = ['GENERO', 'ESTADO_CIVIL', 'DISTRITO', 'TIPO_PERIODO_IVA', 'CODIGO_ATIVIDADE', 'CODIGO_DIV_ATIVIDADE', 'CODIGO_IRS',
'INDICADOR_DIV_IRS','INDICADOR_DIV_IVA','INDICADOR_SUJEITO_IVA','INDICADOR_ATIVIDADE','INDICADOR_MOD3',
'INDICADOR_MOD3_PRAZO','INDICADOR_TRIB_CONJUNTA','INDICADOR_DEV_ESTRATEG','INDICADOR_TRIB_AUTONOMA']
for i in range(len(cols)):
    colToProcess = dataS21_4[cols[i]].astype(str)
    procDF = one_hot(colToProcess, .07,cols[i]+'_')
    dataS21_4.drop(columns=cols[i], inplace=True)
    dataS21_4 = pd.concat([dataS21_4, procDF], axis=1)
```

```
print(dataS21_4.shape)
dataS21_4.describe(include='all').T
```

```
dataS21_4.drop(columns='CODIGO_IRS_0', inplace=True)
dataS21_4.drop(columns='CODIGO_IRS_Other',inplace=True)
dataS21_4.drop(columns='CODIGO_ATIVIDADE_0',inplace=True)
dataS21_4.drop(columns='CODIGO_ATIVIDADE_Other',inplace=True)
dataS21_4.drop(columns='CODIGO_DIV_ATIVIDADE_0',inplace=True)
dataS21_4.drop(columns='CODIGO_DIV_ATIVIDADE_Other', inplace=True)
```

```
# Get a copy of the dataframe before normalization for results' analysis
dataS21_4_beforeNorm = dataS21_4.copy(deep=True)
```

```
# Normalize all columns
# MinMax scaler returns an array, so the dataframe must be recreated
dataS21_4 = pd.DataFrame(preprocessing.MinMaxScaler().fit_transform(dataS21_4.values),
                         columns=dataS21_4.columns,index=dataS21_4.index)
```

```
# Check dataframe structure and statistics after all transformations
print(dataS21_4.shape)
dataS21_4.describe(include='all').T
```

```
#Rename the column from 'TIPO_PERIODO_IVA_Other' to 'TIPO_PERIODO_IVA_M', as there are only 2 types of VAT periods,
#monthly and quarterly
#Same for INDICADOR_DEV_ESTRATEG and INDICADOR_DIV_IVA that only can have 2 results

dataS21_4 = dataS21_4.rename(columns={'TIPO_PERIODO_IVA_Other': 'TIPO_PERIODO_IVA_M'})
dataS21_4 = dataS21_4.rename(columns={'INDICADOR_DEV_ESTRATEG_Other': 'INDICADOR_DEV_ESTRATEG_1'})
dataS21_4 = dataS21_4.rename(columns={'INDICADOR_DIV_IVA_Other': 'INDICADOR_DIV_IVA_1'})
```

```
# Check dataframe structure and statistics after all transformations
print(dataS21_4.shape)
dataS21_4.describe(include='all').T
```

```
finalS21 = dataS21_4.copy()
```

```
# Fit the PCA algorithm to data
pca = PCA().fit(finalS21)

# Show the variance per component
pcaevr = ['{:f}'.format(item) for item in pca.explained_variance_ratio_]
pcaDF = pd.DataFrame({'Component': range(1, len(finalS21.columns)+1),
                      'Variance explained': pcaevr,
                      'Cumulative variance explained': np.cumsum(pca.explained_variance_ratio_)})
pcaDF
```

```
# Plot the cumulative explained variance

# Draw
fig , ax = plt.subplots(figsize=(5, 4))
plt.plot(np.cumsum(pca.explained_variance_ratio_))

# Decoration
sns.despine()
plt.xlabel('Number of components/dimensions')
plt.ylabel('Variance explained')
plt.rc('axes', labelsize=10)
plt.title('Explained variance by components', fontsize=12)
```

Modeling - K-Means - 15 Components

With 15 components is possible to explain 96.6% of the variance.

```
# Apply the dimension reduction to the dataset (for 15 components)
pca_reduced = PCA(n_components = 15)
pca_reduced.fit(finalS21)
finalS21_pca_reduced = pca_reduced.transform(finalS21)
print(finalS21_pca_reduced.shape)
```

```
# Select K based on the sum of squared distances - Elbow method
ssd = []
K = range(1,18)
for k in K:
    km = KMeans(n_clusters=k, random_state=123)
    km = km.fit(finalS21_pca_reduced)
    ssd.append(km.inertia_)

# Show Results
# Draw
fig , ax = plt.subplots(figsize=(5, 4))
plt.plot(K, ssd, 'bx-')
# Decoration
sns.despine()
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
plt.xlabel('K')
plt.ylabel('Sum of squared distances')
plt.rc('axes', labelsize=subPlots_label_fontSize)
ax.xaxis.set_major_locator(ticker.MaxNLocator(integer=True))
plt.title('Elbow method - Reduced dimensionality', fontsize=plots_Title_fontSize);

plt.show()
```

```
# Select K based on the Silhouette method
sil = []

# Dissimilarity can only be measured in more than 1 partition, therefore it starts on K=2
for k in range(2,20):
  km = KMeans(n_clusters = k, random_state=123).fit(finalS21)
  labels = km.labels_
  sil.append(silhouette_score(finalS21, labels, metric = 'euclidean'))


# Show Results
# Draw
fig , ax = plt.subplots(figsize=(5, 4))
plt.plot(range(2, 20), sil, 'bx-')
# Decoration
sns.despine()
fmt = "{x:,.2f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
plt.xlabel('K')
plt.ylabel('Silhouette score')
plt.rc('axes', labelsize=subPlots_label_fontSize)
ax.xaxis.set_major_locator(ticker.MaxNLocator(integer=True))
plt.title('Silhouette method - Reduced dimensionality', fontsize=plots_Title_fontSize);

plt.show()
```

```
#Davies-Boulding Index
results = {}

for i in range(2,11):
    kmeans = KMeans(n_clusters=i, random_state=30)
    labels = kmeans.fit_predict(finalS21_pca_reduced)
    db_index = davies_bouldin_score(finalS21_pca_reduced, labels)
    results.update({i: db_index})

plt.plot(list(results.keys()), list(results.values()))
plt.xlabel("Number of clusters")
plt.ylabel("Davies-Boulding Index")
plt.show()
```

```
K=4
kmeans = KMeans(n_clusters=K, random_state=123)
allDistances = kmeans.fit_transform(finalS21_pca_reduced)
y_kmeans = kmeans.predict(finalS21_pca_reduced)
```

```
# Plot clusters cardinality
# Count observations per cluster
freqByCluster = dataS21_4_beforeNorm.groupby(y_kmeans).size()

# Draw
fig, ax = plt.subplots(figsize=(7,5))
g = sns.countplot(x=y_kmeans, color='grey')

# Decoration
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
for index,data in enumerate(freqByCluster):
    plt.text(x=index-0.2 , y=data+50 , s=f"{data}" , fontdict=dict(fontsize=plots_barTexts_fontSize))
sns.despine()
plt.title("Cluster cardinality", fontsize=plots_Title_fontSize)
plt.xlabel("Cluster")
plt.ylabel("Frequency in cluster")
plt.rc('axes', labelsize=subPlots_label_fontSize)
plt.show()
```

```
# Plot clusters magnitude
# Compute Magnitude
finalS21['distanceToCentroid'] = np.min(allDistances,axis=1)
magnitude = finalS21['distanceToCentroid'].groupby(y_kmeans).sum()
finalS21 = finalS21.drop(columns=['distanceToCentroid'])

# Draw
fig, ax = plt.subplots(figsize=(7,5))
g = sns.barplot(x=magnitude.index, y=magnitude.values, color='grey')

# Decoration
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
for index,data in enumerate(magnitude):
    plt.text(x=index-0.2 , y=data+50 , s=f"{data:,.0f}" , fontdict=dict(fontsize=plots_barTexts_fontSize))
sns.despine()
plt.title("Cluster magnitude", fontsize=plots_Title_fontSize)
plt.xlabel("Cluster")
plt.ylabel("Sum of distances to centroid")
plt.rc('axes', labelsize=subPlots_label_fontSize)
plt.show()
```

```
# Plot cardinality vs magnitude

# Draw
fig, ax = plt.subplots(figsize=(6,4))
g = sns.regplot(x=freqByCluster, y=magnitude, scatter=True, seed=123,truncate=False, ci=None)

# Decoration
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.xaxis.set_major_formatter(tick)
ax.yaxis.set_major_formatter(tick)
sns.despine()
plt.title("Cardinality vs Magnitude", fontsize=plots_Title_fontSize)
plt.xlabel("Cardinality")
plt.ylabel("Magnitude")
plt.rc('axes', labelsize=subPlots_label_fontSize)
plt.show()
```

Let's use PCA to visualize with only two components

```
# Scatter plot of the two Principal Components by cluster to see if there is any pattern

# Draw
fig, ax = plt.subplots(figsize=(7,5))
plt.scatter(finalS21_pca_reduced[:, 0], finalS21_pca_reduced[:, 1], c=y_kmeans, s=50, cmap='viridis')
centers = kmeans.cluster_centers_
ax.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5)

# Decoration
plt.xlabel('Component 1')
plt.ylabel('Component 2')
plt.colorbar();

plt.show()
```

```
# Create a dataframe with the weights per component
wheightsDFS21 = pd.DataFrame(pca.components_,columns=finalS21.columns)
wheightsDFS21
```

```
# Study the weights of component 1
compS21 = abs(wheightsDFS21.iloc[0])  ## index 0 is component 1
compS21 = compS21.sort_values(ascending=False)
compS21
```

```
## Check the mean values of each cluster for top 52 relevant features
top_features = compS21[:53].index
clustersMeanDFS21 = pd.DataFrame(finalS21[top_features].groupby(y_kmeans).mean(),columns=top_features)
clustersMeanDFS21.transpose()
```

Business 2021

```
dataC21 = data.copy()
```

```
#Drop individual
dataC21.drop(dataC21.index[(dataC21["TIPO_NIF"] == 'S')],axis=0,inplace=True)
```

```
#Drop the years that are not used
dataC21.drop(dataC21.index[(dataC21["ANO_EXERCICIO"] == 2018)],axis=0,inplace=True)
dataC21.drop(dataC21.index[(dataC21["ANO_EXERCICIO"] == 2019)],axis=0,inplace=True)
dataC21.drop(dataC21.index[(dataC21["ANO_EXERCICIO"] == 2020)],axis=0,inplace=True)
dataC21.drop(dataC21.index[(dataC21["ANO_EXERCICIO"] == 2017)],axis=0,inplace=True)
```

DATA ANALYSIS

```
dataC21_1 = dataC21.copy()
```

```
#Check the size of the database
dataC21_1.shape
```

```
#Check for false values
dataC21_1['INDICADOR_DIV_IRC'].value_counts()
```

```
#Check for false values
dataC21_1['INDICADOR_DIV_IVA'].value_counts()
```

```
#Check for false values
dataC21_1['INDICADOR_ATIVIDADE'].value_counts()
```

```
#Check for false values
dataC21_1['TIPO_PERIODO_IVA'].value_counts()
```

```
#Check for false values
dataC21_1['INDICADOR_SUJEITO_IVA'].value_counts()
```

```
#Check for false values
dataC21_1['CODIGO_DIV_ATIVIDADE'].value_counts()
```

```
#Check for false values
dataC21_1['DISTRITO'].value_counts()
```

```
#Check for false values
dataC21_1['INDICADOR_INSOLVENCIA'].value_counts()
```

```
#Check for false values
dataC21_1['INDICADOR_DEV_ESTRATEG'].value_counts()
```

```
# Activity
#1    26068
#0     3855

x = [26068, 3855]
labels = ['Yes', 'No']
colors = ['tab:green', 'tab:grey']

fig, ax = plt.subplots()
ax.pie(x, labels = labels, colors = colors, autopct='%1.1f%%')
ax.set_title('Activity')
plt.show()
```

```
# VAT Period

by = sns.histplot(dataC21_1["TIPO_PERIODO_IVA"] , color="grey", kde = False, bins = 12)
plt.title('VAT Period')
plt.ylabel('Number of Taxpayers')
by.set_xlabel('Type of Period')
by.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))
by.yaxis.grid(True)
by.xaxis.grid(False)

plt.show()
```

```
#Liable to VAT

labels = ['Yes', 'No']
sizes = [21313, 8610]
colors = ['tab:green', 'tab:grey']

plt.pie(sizes, labels=labels, colors = colors, autopct='%1.1f%%', startangle=90, wedgeprops=dict(width=0.54))
plt.title('Liable to VAT')
plt.show()
```

DATA PREPARATION

```
dataC21_2 = dataC21_1.copy()
```

```
# Remove the columNS that should not be used because are business variables
dataC21_2.drop(columns=['GENERO','ESTADO_CIVIL','ANO_NASCIMENTO','TIPO_NIF','ANO_EXERCICIO','CODIGO_IRS',
                        'VALOR_REND_BRUTO','VALOR_IRS_FINAL','VALOR_REND_COLETAVEL','INDICADOR_MOD3',
                        'INDICADOR_MOD3_PRAZO','INDICADOR_TRIB_CONJUNTA','INDICADOR_TRIB_AUTONOMA',
                        'INDICADOR_DIV_IRS','VALOR_DIVIDA_IRS'], inplace=True)
```

```
#Correlation matrix

plt.figure(figsize = (20, 10))
custom_cmap = sns.diverging_palette(220, 20, as_cmap=True)
sns.heatmap(dataC21_2.corr(), annot = True, linewidths=.5, cmap = custom_cmap, vmin = -1)
plt.title('Correlation Matrix', fontsize = 20)
plt.show()
```

```
# Check missing values
print(dataC21_2.isnull().sum())
```

```
#Drop column if they only contain missing values
dataC21_2 = dataC21_2.dropna(how='all')
dataC21_2.shape
```

```
#We will replace null values with 0 for numerical values and 'Na' for categorical values, as it makes sense to represent
#them this way since they are non-existent
dataC21_2['VALOR_BASE_TRIB'] = dataC21_2['VALOR_BASE_TRIB'].replace(np.nan, 0)
dataC21_2['VALOR_VOLUME_NEG'] = dataC21_2['VALOR_VOLUME_NEG'].replace(np.nan, 0)
dataC21_2['VALOR_DIVIDA_IRC'] = dataC21_2['VALOR_DIVIDA_IRC'].replace(np.nan, 0)
dataC21_2['VALOR_DIVIDA_IVA'] = dataC21_2['VALOR_DIVIDA_IVA'].replace(np.nan, 0)
dataC21_2['CODIGO_DIV_ATIVIDADE'] = dataC21_2['CODIGO_DIV_ATIVIDADE'].replace(np.nan, 0)
dataC21_2['TIPO_PERIODO_IVA'] = dataC21_2['TIPO_PERIODO_IVA'].fillna('Na')
```

```
#Replace ',' with '.' to use the value format
dataC21_2['VALOR_BASE_TRIB'] = dataC21_2['VALOR_BASE_TRIB'].replace({',': '.'}, regex=True)
dataC21_2['VALOR_VOLUME_NEG'] = dataC21_2['VALOR_VOLUME_NEG'].replace({',': '.'}, regex=True)
dataC21_2['VALOR_DIVIDA_IRC'] = dataC21_2['VALOR_DIVIDA_IRC'].replace({',': '.'}, regex=True)
dataC21_2['VALOR_DIVIDA_IVA'] = dataC21_2['VALOR_DIVIDA_IVA'].replace({',': '.'}, regex=True)
```

```
# Check missing values
print(dataC21_2.isnull().sum())
```

```
#Set index based on the taxpayers info
dataC21_2.set_index(['ID_CONTRIBUINTE'], inplace=True)
```

```
dataC21_3 = dataC21_2.copy()
```

```
# Bin the montant values
bins_1 = np.linspace(-0.00001,10971412.42,4)
bins_1

dataC21_3['VALOR_DIVIDA_IRC'] = dataC21_3['VALOR_DIVIDA_IRC'].astype(float)

dataC21_3['V_DIVIDA_IRC_BINS'] = pd.cut(x=dataC21_3['VALOR_DIVIDA_IRC'], bins=bins_1,
                                        labels=['<3657137.48','3657137.48-7314274.94','>=7314274.95'])


dataC21_3.drop(columns='VALOR_DIVIDA_IRC', inplace=True)
```

```
# Bin the montant values

bins_2 = np.linspace(-0.00001,1986903.08,4)
bins_2

dataC21_3['VALOR_DIVIDA_IVA'] = dataC21_3['VALOR_DIVIDA_IVA'].astype(float)

dataC21_3['V_DIVIDA_IVA_BINS'] = pd.cut(x=dataC21_3['VALOR_DIVIDA_IVA'], bins=bins_2,
                                        labels=['<662301.1','662301.1-1324602','>=1324602.1'])


dataC21_3.drop(columns='VALOR_DIVIDA_IVA', inplace=True)
```

```
# Bin the montant values

bins_3 = np.linspace(-0.00001,3604340044,4)
bins_3

dataC21_3['VALOR_BASE_TRIB'] = dataC21_3['VALOR_BASE_TRIB'].astype(float)

dataC21_3['V_BASE_TRIB_BINS'] = pd.cut(x=dataC21_3['VALOR_BASE_TRIB'], bins=bins_3,
                                       labels=['<1201446680.1','1201446680.1-2402893360','>=2402893360.1'])


dataC21_3.drop(columns='VALOR_BASE_TRIB', inplace=True)
```

```
# Bin the montant values

bins_4 = np.linspace(-0.00001,2366947621,4)
bins_4

#bins_5 = bins_5.astype(str)
dataC21_3['VALOR_VOLUME_NEG'] = dataC21_3['VALOR_VOLUME_NEG'].astype(float)

dataC21_3['V_VOLUME_NEG_BINS'] = pd.cut(x=dataC21_3['VALOR_VOLUME_NEG'], bins=bins_4,
                                        labels=['<788982540.1','788982540.1-1577965080','>=1577965080.1'])

dataC21_3.drop(columns='VALOR_VOLUME_NEG', inplace=True)
```

```
# Encode categorical variables to dummy variables

cols = ['V_DIVIDA_IRC_BINS', 'V_DIVIDA_IVA_BINS', 'V_BASE_TRIB_BINS','V_VOLUME_NEG_BINS']
ce_one_hot = ce.OneHotEncoder(cols = cols, use_cat_names=True)
dataC21_3 = ce_one_hot.fit_transform(dataC21_3)
```

```
print(dataC21_3.shape)
dataC21_3.describe(include='all').T
```

```
dataC21_4= dataC21_3.copy()
```

```
def one_hot(s, threshold,prefix):
    """Do one-hot encoding for categories above a threshold and create a dummy column for all others named "Other"

    Args:
        s (series): series to apply the transformation
        threshold (numerical): threshold above which a dummy column should be created (from 0 to 1, corresponding from 0
        to 100%)
        prefix (string): prefix to assign to the created columns

    Returns:
        dataframe: dataframe with the applied transformations
    """
    # Check dummies for the column
    d = pd.get_dummies(s)

    # Check if percentage is below threshold and act accordingly
    f = pd.value_counts(s, sort=False, normalize=True) < threshold
    if f.sum() == 0:
        newSeries=d
    else:
        newSeries=d.loc[:, ~f].join(d.loc[:, f].sum(1).rename('Other'))
    # Add the prefix and return the result
    newSeries=newSeries.add_prefix(prefix)
    return newSeries
```

```
# One hot encoding of all categories above 7% - all others categories to "Other"
cols = ['DISTRITO', 'TIPO_PERIODO_IVA', 'CODIGO_ATIVIDADE', 'CODIGO_DIV_ATIVIDADE','INDICADOR_DIV_IRC','INDICADOR_DIV_IVA',
'INDICADOR_SUJEITO_IVA','INDICADOR_ATIVIDADE','INDICADOR_DEV_ESTRATEG','INDICADOR_INSOLVENCIA']
for i in range(len(cols)):
    colToProcess = dataC21_4[cols[i]].astype(str)
    procDF = one_hot(colToProcess, .07,cols[i]+'_')
    dataC21_4.drop(columns=cols[i], inplace=True)
    dataC21_4 = pd.concat([dataC21_4, procDF], axis=1)
```

```
print(dataC21_4.shape)
dataC21_4.describe(include='all').T
```

```
dataC21_4 .drop(columns='CODIGO_ATIVIDADE_0',inplace=True)
dataC21_4 .drop(columns='CODIGO_ATIVIDADE_Other',inplace=True)
```

```
# Get a copy of the dataframe before normalization for results' analysis
dataC21_4_beforeNorm = dataC21_4.copy(deep=True)
```

```
# Normalize all columns
# MinMax scaler returns an array, so the dataframe must be recreated
dataC21_4 = pd.DataFrame(preprocessing.MinMaxScaler().fit_transform(dataC21_4.values),
                         columns=dataC21_4.columns,index=dataC21_4.index)
```

```
# Check dataframe structure and statistics after all transformations

print(dataC21_4.shape)
dataC21_4.describe(include='all').T
```

```
#Rename the INDICADOR_DEV_ESTRATEG_Other column to INDICADOR_DEV_ESTRATEG_1, as there are only 2 types of options
#Same for INDICADOR_INSOLVENCIA

dataC21_4 = dataC21_4.rename(columns={'INDICADOR_DEV_ESTRATEG_Other': 'INDICADOR_DEV_ESTRATEG_1'})
dataC21_4 = dataC21_4.rename(columns={'INDICADOR_INSOLVENCIA_Other': 'INDICADOR_INSOLVENCIA_1'})
```

```
dataC21_4.head()
```

```
finalC21 = dataC21_4.copy()
```

```
# Fit the PCA algorithm to data
pca = PCA().fit(finalC21)

# Show the variance per component
pcaevr = ['{:f}'.format(item) for item in pca.explained_variance_ratio_]
pcaDF = pd.DataFrame({'Component': range(1, len(finalC21.columns)+1),
                      'Variance explained': pcaevr,
                      'Cumulative variance explained': np.cumsum(pca.explained_variance_ratio_)})
pcaDF
```

```
# Plot the cumulative explained variance

# Draw
fig , ax = plt.subplots(figsize=(5, 4))
plt.plot(np.cumsum(pca.explained_variance_ratio_))

# Decoration
sns.despine()
plt.xlabel('Number of components/dimensions')
plt.ylabel('Variance explained')
plt.rc('axes', labelsize=10)
plt.title('Explained variance by components', fontsize=12)
```

Modeling - K-Means - 10 Components

With 10 components is possible to explain 96.2% of the variance.

```
# Apply the dimension reduction to the dataset (for 10 components)
pca_reduced = PCA(n_components = 10)
pca_reduced.fit(finalC21)
finalC21_pca_reduced = pca_reduced.transform(finalC21)
print(finalC21_pca_reduced.shape)
```

```
# Select K based on the sum of squared distances - Elbow method
ssd = []
K = range(1,18)
for k in K:
    km = KMeans(n_clusters=k, random_state=123)
    km = km.fit(finalC21_pca_reduced)
    ssd.append(km.inertia_)

# Show Results
# Draw
fig , ax = plt.subplots(figsize=(5, 4))
plt.plot(K, ssd, 'bx-')
# Decoration
sns.despine()
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
plt.xlabel('K')
plt.ylabel('Sum of squared distances')
plt.rc('axes', labelsize=subPlots_label_fontSize)
ax.xaxis.set_major_locator(ticker.MaxNLocator(integer=True))
plt.title('Elbow method - Reduced dimensionality', fontsize=plots_Title_fontSize);

plt.show()
```

```
# Select K based on the Silhouette method
sil = []

# Dissimilarity can only be measured in more than 1 partition, therefore it starts on K=2
for k in range(2,20):
  km = KMeans(n_clusters = k, random_state=123).fit(finalC21)
  labels = km.labels_
  sil.append(silhouette_score(finalC21, labels, metric = 'euclidean'))


# Show Results
# Draw
fig , ax = plt.subplots(figsize=(5, 4))
plt.plot(range(2, 20), sil, 'bx-')
# Decoration
sns.despine()
fmt = "{x:,.2f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
plt.xlabel('K')
plt.ylabel('Silhouette score')
plt.rc('axes', labelsize=subPlots_label_fontSize)
ax.xaxis.set_major_locator(ticker.MaxNLocator(integer=True))
plt.title('Silhouette method - Reduced dimensionality', fontsize=plots_Title_fontSize);

plt.show()
```

```
#Davies-Boulding Index
results = {}

for i in range(2,11):
    kmeans = KMeans(n_clusters=i, random_state=30)
    labels = kmeans.fit_predict(finalC21_pca_reduced)
    db_index = davies_bouldin_score(finalC21_pca_reduced, labels)
    results.update({i: db_index})

plt.plot(list(results.keys()), list(results.values()))
plt.xlabel("Number of clusters")
plt.ylabel("Davies-Boulding Index")
plt.show()
```

```
K=2
kmeans = KMeans(n_clusters=K, random_state=123)
allDistances = kmeans.fit_transform(finalC21_pca_reduced)
y_kmeans = kmeans.predict(finalC21_pca_reduced)
```

```
# Plot clusters cardinality
# Count observations per cluster
freqByCluster = dataC21_4_beforeNorm.groupby(y_kmeans).size()

# Draw
fig, ax = plt.subplots(figsize=(7,5))
g = sns.countplot(x=y_kmeans, color='grey')

# Decoration
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
for index,data in enumerate(freqByCluster):
    plt.text(x=index-0.2 , y=data+50 , s=f"{data}" , fontdict=dict(fontsize=plots_barTexts_fontSize))
sns.despine()
plt.title("Cluster cardinality", fontsize=plots_Title_fontSize)
plt.xlabel("Cluster")
plt.ylabel("Frequency in cluster")
plt.rc('axes', labelsize=subPlots_label_fontSize)
plt.show();
```

```
# Plot clusters magnitude
# Compute Magnitude
finalC21['distanceToCentroid'] = np.min(allDistances,axis=1)
magnitude = finalC21['distanceToCentroid'].groupby(y_kmeans).sum()
finalC21 = finalC21.drop(columns=['distanceToCentroid'])

# Draw
fig, ax = plt.subplots(figsize=(7,5))
g = sns.barplot(x=magnitude.index, y=magnitude.values, color='grey')

# Decoration
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.yaxis.set_major_formatter(tick)
for index,data in enumerate(magnitude):
    plt.text(x=index-0.2 , y=data+50 , s=f"{data:,.0f}" , fontdict=dict(fontsize=plots_barTexts_fontSize))
sns.despine()
plt.title("Cluster magnitude", fontsize=plots_Title_fontSize)
plt.xlabel("Cluster")
plt.ylabel("Sum of distances to centroid")
plt.rc('axes', labelsize=subPlots_label_fontSize)
plt.show()
```

```
# Plot cardinality vs magnitude

# Draw
fig, ax = plt.subplots(figsize=(6,4))
g = sns.regplot(x=freqByCluster, y=magnitude, scatter=True, seed=123,truncate=False, ci=None)

# Decoration
fmt = "{x:,.0f}"
tick = ticker.StrMethodFormatter(fmt)
ax.xaxis.set_major_formatter(tick)
ax.yaxis.set_major_formatter(tick)
sns.despine()
plt.title("Cardinality vs Magnitude", fontsize=plots_Title_fontSize)
plt.xlabel("Cardinality")
plt.ylabel("Magnitude")
plt.rc('axes', labelsize=subPlots_label_fontSize)
plt.show()
```

Let's use PCA to visualize with only two components

```
# Scatter plot of the two Principal Components by cluster to see if there is any pattern

# Draw
fig, ax = plt.subplots(figsize=(7,5))
plt.scatter(finalC21_pca_reduced[:, 0], finalC21_pca_reduced[:, 1], c=y_kmeans, s=50, cmap='viridis')
centers = kmeans.cluster_centers_
ax.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5)

# Decoration
plt.xlabel('Component 1')
plt.ylabel('Component 2')
plt.colorbar();
plt.show()
```

```
# Create a dataframe with the weights per component
wheightsDFC21 = pd.DataFrame(pca.components_,columns=finalC21.columns)
wheightsDFC21
```

```
# Study the weights of component 1
compC21 = abs(wheightsDFC21.iloc[0])  ## index 0 is component 1
compC21 = compC21.sort_values(ascending=False)
compC21
```

```
## Check the mean values of each cluster for top 34 relevant features
top_features = compC21[:35].index
clustersMeanDFC21 = pd.DataFrame(finalC21[top_features].groupby(y_kmeans).mean(),columns=top_features)
clustersMeanDFC21.transpose()
```

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa