FACULDADE
DE ECONOMIA
UNIVERSIDADE
DO PORTO

Automating Output Checking in Official Statistics with Machine Learning and Large Language Models

Ricardo Dias de Carvalho

Dissertation

Master in Modelling, Data Analysis and Decision Support Systems

Supervised by:

PhD Afshin Ashofteh and PhD Pedro Campos
https://orcid.org/0000-0001-5075-9822
https://orcid.org/0000-0001-5495-9434

2025

Acknowledgments

I would first like to express my deepest gratitude to my advisors, PhD Afshin Ashofteh and PhD Pedro Campos, for their invaluable guidance, patience, and support throughout this journey. Their insight and encouragement pushed me to grow both academically and personally, and this dissertation would not have been possible without them.

I would also like to thank Statistics Portugal (INE) for the opportunity to carry out my internship within the institution. I am especially grateful to Maria Ferreira, Anabela Silva, Teresa Fragoso, and Sónia Quaresma, whose support and collaboration were essential throughout this work. I am particularly indebted to my advisor Pedro Campos and to Inês Rodrigues de Sá, whose guidance, valuable insights, and constant availability - from providing access to documents to clarifying doubts and sharing their expertise - were fundamental in helping me fully understand the problem and move this project forward.

My gratitude also goes to the professors of the Master's program and the Faculty of Economics, University of Porto, for their dedication, knowledge, and for contributing significantly to my academic and professional development.

To my colleagues and friends in the program, thank you for the discussions, encouragement, and late-night study sessions that made this path lighter and more enjoyable.

Finally, to my family, partner and closest friends, who have always believed in me even when I doubted myself - thank you for your unconditional love, patience, and encouragement. This achievement is as much yours as it is mine.

Abstract

This thesis focuses on improving how Statistical Offices, such as Statistics Portugal (INE), check whether research outputs based on sensitive microdata can be safely released without revealing confidential information. Currently, this process is largely manual, time-consuming, and dependent on expert reviewers. The goal was to design a semi-automated system that reduces workload, accelerates reviews, and preserves confidentiality by combining deterministic rules, machine learning (ML), and large language models (LLMs). The proposed system operates in three layers. First, a rule-based component applies strict disclosure rules (e.g., minimum sample sizes, dominance thresholds) to flag risky outputs, providing a transparent baseline. Second, ML models (XGBoost, LightGBM) learn from historical institutional decisions to capture contextual patterns and *principles-based* reasoning that go beyond rigid thresholds. Third, LLMs play a dual role: they assist in cleaning and standardising unstructured researcher requests (extracting variables, filters, and metadata into structured features), and they provide contextual explanations of automated classifications, increasing transparency and usability. To train and test the system, historical outputs and requests from INE were processed, standardised, and enriched with features from outputs, metadata, and researcher-provided documentation. Rule-based checks proved highly effective for clear-cut cases. ML models achieved high predictive performance, especially when combining rule-based signals with structural and request-derived features. While LLMs were not the most accurate classifiers, they substantially improved robustness by handling messy, irregular inputs, supported feature engineering through automated cleaning, and produced interpretable justifications that bridged communication between researchers and disclosure reviewers. Together, these layers form a hybrid system that demonstrates how combining rules, ML, and LLMs can make statistical disclosure control more efficient, transparent, and scalable. Rules ensure compliance, ML captures exceptions and contextual nuance, and LLMs enhance both preprocessing (through data cleaning and feature extraction) and postprocessing (through explanations and transparency). While some extensions - such as computing features directly from the underlying microdata - remain to be implemented, this work highlights how hybrid AI-driven systems can pave the way toward faster, more accurate, and transparent output checking, supporting statistical offices as they adapt disclosure control to the realities of modern data-intensive research.

Keywords: Statistical Disclosure Control; Output Checking; Machine Learning; Large Language Models; Official Statistics; Confidentiality.

Resumo

Esta dissertação centra-se na melhoria do processo através do qual os Institutos de Estatística, como o Instituto Nacional de Estatística (INE), verificam se os resultados de investigação baseados em microdados sensíveis podem ser divulgados em segurança sem revelar informação confidencial. Atualmente, este processo é em grande parte manual, demorado e dependente de revisores especializados. O objetivo foi conceber um sistema semi-automatizado que reduza a carga de trabalho, acelere a revisão e preserve a confidencialidade, combinando regras determinísticas, Machine Learning (ML) e Large Language Models (LLMs). O sistema proposto integra três camadas. Em primeiro lugar, um componente baseado em regras aplica critérios estritos de confidencialidade (por exemplo, tamanhos mínimos de amostra, limiares de dominância). Em segundo lugar, modelos de ML (XGBoost, LightGBM) aprendem com decisões institucionais históricas para capturar padrões contextuais e raciocínio, indo além dos limiares rígidos. Em terceiro lugar, os LLMs desempenham um papel duplo: apoiam a limpeza e normalização de pedidos não estruturados (extraindo variáveis, filtros e metadados) e fornecem explicações contextuais das classificações automáticas, aumentando a transparência e a usabilidade. Para treinar e testar o sistema, pedidos e resultados históricos do INE foram processados, normalizados e enriquecidos com variáveis derivadas dos outputs, metadados e documentação fornecida pelos investigadores. As verificações baseadas em regras mostraram-se altamente eficazes nos casos mais claros. Os modelos de ML alcançaram elevado desempenho preditivo, sobretudo quando combinaram o resultado da verificação baseada em regras com características estruturais extraídas dos pedidos. Embora os LLMs não tenham sido os classificadores mais precisos, contribuíram significativamente para a robustez ao lidar com inputs irregulares ou incompletos, apoiaram a engenharia de variáveis através da limpeza automática e produziram justificações interpretáveis que reforçaram a comunicação entre investigadores e revisores de confidencialidade. Em conjunto, estas três camadas formam um sistema híbrido que demonstra como a combinação de regras, ML e LLMs pode tornar o controlo de divulgação estatística mais eficiente, transparente e escalável. As regras asseguram a conformidade, a ML capta exceções e nuances contextuais, e os LLMs reforçam tanto o pré-processamento (através da limpeza de dados e extração de características) como o pós-processamento (através de explicações e transparência). Embora algumas extensões - como o cálculo de indicadores diretamente a partir dos microdados - permaneçam por implementar, este trabalho demonstra um caminho claro para transformar a verificação de outputs de um processo manual e lento num procedimento colaborativo, adaptativo e auditável, que melhor serve tanto os institutos de estatística como os investigadores.

Palavras-chave: Controlo de Segredo Estatístico; Verificação de Outputs; Machine Learning; Large Language Models; Estatísticas Oficiais; Confidencialidade.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In an era of data-driven decision-making, researchers rely on *microdata* provided by national statistical institutes and other organizations to conduct empirical analyses for their research. At the same time, ensuring *data confidentiality* and *privacy* has become increasingly challenging due to the enforcement of strict data protection regulations such as the *General Data Protection Regulation* (GDPR) [5]. To meet these requirements, *Statistical Disclosure Control* (SDC) mechanisms are essential to ensure that research outputs do not compromise sensitive or personally identifiable information for individuals or groups [19].

A critical component of SDC is *output checking*, a process through which research results are reviewed before being released. At institutions such as *Statistics Portugal* (INE), researchers access microdata within *safe centers*, under tightly controlled conditions [28]. Before any output can leave the center, it must undergo a manual output checking procedure to assess potential disclosure risks and determine whether it can be approved for release [15]. Although effective, this process is labor-intensive, time-consuming, and highly dependent on expert knowledge, creating significant challenges for scalability and efficiency and making it an ideal candidate for automation using advanced *artificial intelligence* (AI) techniques.

The motivation for this research arises from several factors that hinder the current manual output checking process at Statistics Portugal and similar institutions [19].

Manual checking requires substantial time and resources, as expert reviewers must carefully examine each output, leading to delays in data access and slowing the entire research workflow. While a submission template, known as the output request document, exists to guide researchers, outputs are often submitted in heterogeneous and inconsistent formats, lacking the standardized structure needed to enable automated checks [4]. Moreover, although rule-based approaches can address straightforward disclosure risks, more complex, context-dependent risks remain difficult to capture using rigid, predefined rules alone. On top of these challenges, there is the necessity for full compliance with legal and ethical frameworks such as the GDPR and the emerging AI Act [5], which impose strict requirements on any system handling confidential data.

This research asks: *How can Large Language Models (LLMs), other machine learning techniques, and deterministic methods be combined to automate and optimize the output checking process in official statistics, while ensuring efficiency, scalability, accuracy, explainability, and compliance with data confidentiality regulations such as the GDPR and the AI Act?*

To address this question, this dissertation investigates a hybrid approach that integrates deterministic rule-based methods, machine learning (ML), and LLMs, complemented by hu-

man oversight in borderline cases that automation cannot fully resolve. Specifically, LLMs are employed for data standardization, feature extraction, and contextual risk assessment, while ML models capture complex disclosure risks beyond rigid rules. Together, these components form a system designed to enhance efficiency, accuracy, scalability, and transparency, while ensuring compliance with relevant regulatory frameworks, including the GDPR and the AI Act.

The ultimate goal is to develop a scalable framework for disclosure safety assessment that (i) standardizes output request documents with LLM-assisted guidance; (ii) incorporates ML and hybrid SDC techniques for risk evaluation; and (iii) provides transparent, explainable decisions. A prototype has been created, demonstrating a practical solution to the limitations of manual output checking and offering a model that can be adapted by other national statistical institutes facing similar confidentiality and efficiency challenges. This research was conducted as part of a thesis internship at Statistics Portugal (INE), providing access to real historical outputs and ensuring that the prototype was designed in close alignment with institutional practices.

Following this introduction, Chapter 2 provides a comprehensive literature review. Chapter 3 outlines the research methodology. Chapter 4 presents the implementation of the proposed system and discusses the experimental results. Finally, Chapter 5 concludes the thesis by summarizing the main findings.

Chapter 2

Literature Review

2.1 Statistical Disclosure Control

Statistical Disclosure Control (SDC) techniques refer to the set of methods designed to prevent individuals or groups from being identified in the process or results of data analysis [26]. At official statistics offices, SDC is typically divided into two stages. The first is *input SDC*, which involves removing or modifying direct identifiers from the original microdata before researchers gain access. The second is *output SDC*, which corresponds to the review of analytical results before dissemination [19].

At Statistics Portugal, input SDC is applied prior to researcher access. The internal team produces safe-use files (anonymised microdata) by removing or encoding direct identifiers in each observation. Output SDC, on the other hand, corresponds to the process of output checking [15].

2.1.1 Types of disclosure

While most outputs carry relatively low disclosure risk - since data are usually sampled, transformed, or aggregated - risks can still arise depending on the statistical methods used and the context of dissemination. These risks are typically divided into *primary* and *secondary disclosure* [19].

Primary disclosure refers to when a statistic directly allows inferences to be made about an individual unit, e.g., extreme values in a dataset or small frequency counts if, for instance, one single unit falls in a specific category. Most of the time, this type of disclosure can be analyzed by simple rules such as minimum thresholds, maximum percentage-rule or 'N, K' rule, in which the largest N observations must contribute less than K% of the total. In primary disclosure, many risks lend themselves to deterministic, policy-codified checks (e.g., minimum counts, dominance) [19]. However, recent guidance cautions that automation should sit within a principled classification and policy context rather than rely on ad-hoc rules alone [16].

Class disclosure (or group disclosure) is a subtype of primary disclosure in which all individuals fall into a single category. For instance, in a survey of salaries within a small village, if all respondents report salaries within a narrow range, it becomes possible to approximate any individual's salary. Class disclosure can be managed by suppressing outputs in which 0% or 100% (or a sufficiently high value) of the population falls within the same cell. However, structural zeros (i.e., values that are expected by definition, such as zero prevalence of

driving licences among people under 18) complicate this rule [19]. Detecting class disclosure is therefore one of the most challenging areas of primary disclosure control, particularly for automated systems, which may struggle to distinguish between sensitive disclosures and values that inherently carry no risk.

Secondary disclosure refers to cases in which identities or values can be revealed by a combination of different statistics (e.g., differencing two tables), and it usually relates to several outputs. Secondary disclosure within a single output is straightforward to deal with, as the reconstruction of suppressed values in tables can be dealt with by secondary suppression [19]. The major problem for secondary disclosure is differencing, which refers to comparing two outputs with two similar statistics that only differ in one way, and the difference between these two statistics can be used to compute both frequencies and exact values [19].

2.1.2 Classification of outputs

Beyond the input/output distinction, a key strand of recent work emphasises the importance of classifying research outputs into categories that reflect their disclosure risk. Such classifications provide a structured basis for guidance, standardisation, and semi-automation of output checking [16].

One influential approach is that of Ritchie [42], who introduced the notion of *safe* and *unsafe* outputs. Outputs can be grouped into a limited number of types (e.g., frequency tables, regressions), which are labelled according to their expected disclosure risk (Table 2.1). This classification is based solely on the *form* of the output rather than on the data or context in which it was produced, providing a practical rule-of-thumb for output checkers. Safe outputs, such as regression coefficients or correlation measures, carry little meaningful disclosure risk, whereas unsafe outputs, such as frequency tables or percentiles, should not be released unless additional context demonstrates they are non-disclosive. In both cases, the ultimate decision remains with the output checker [4].

Building on this, the *Statbarn* framework [20] groups outputs not by form but by their shared *disclosure signatures*. For example, frequency tables are characterised by small-cell and dominance risks, while aggregations are subject to denominator floors and contributor dominance. Regression coefficients may raise concerns about saturation or residual degrees of freedom. Each class of outputs is then linked to a standard set of tests, exceptions, and remedies. This moves guidance away from method-specific checklists and towards class-specific policies, improving consistency, signalling maturity (well-understood vs. provisional checks), and providing a clean hook for semi-automation. See Table A.1 in Appendix A for a comparison of typologies.

A complementary perspective is provided by the RRSA framework (*Runners, Repeaters, Strangers, Aliens*) [2]. Unlike typologies based on statistical form or disclosure signatures, RRSA classifies requests by their operational characteristics, such as predictability, frequency, and discretion required. Routine, rule-compliant requests are treated as *runners* that can be cleared quickly, borderline cases are *repeaters* requiring contextual judgement, novel cases are *strangers* requiring detailed review, and exceptional cases are *aliens*, which fall outside existing policies. This triage-based classification emphasises efficiency by accelerating routine approvals while focusing expert resources on unusual or high-risk requests.

Taken together, these frameworks illustrate complementary approaches to classification. Ritchie's typology provides a pragmatic safe/unsafe baseline, Statbarn reframe outputs in

Table 2.1: Classification of outputs by type of statistic [42]

Type of Statistics	Type of Output General	Classification
Descriptive statistics	Frequency tables	Unsafe
	Magnitude tables	Unsafe
	Maxima, minima and percentiles (incl. median)	Unsafe
	Mode	Safe
	Means, indices, ratios, indicators	Unsafe
	Concentration ratios	Safe
	Higher moments of distributions (incl. variance, covariance, kurtosis, skewness)	Safe
	Graphs: pictorial representations of actual data	Unsafe
Correlation and regression type analysis	Linear regression coefficients	Safe
	Non-linear regression coefficients	Safe
	Estimation residuals	Unsafe
	Summary and test statistics from estimates (R^2, χ^2, etc.)	Safe
	Correlation coefficients	Safe
	Factor analysis	Safe
	Correspondence analysis	Safe

terms of disclosure signatures, enabling semi-automation, and RRSA adds an operational triage dimension, ensuring expert attention is directed where it is most needed.

2.2 Output checking

The goal of the output control is to maximize the use of the datasets while minimizing the risk of disclosure, guaranteeing that no individual or group can be identified. It is not possible to tell ahead of the research if an output is going to be released or not. Even though frameworks such as the *safe/unsafe* typology or the *runners, repeaters, strangers, and aliens* (RRSA) classification provide useful guidance for anticipating which outputs are likely to be released, a more detailed examination of the output checking process is still required.

When performing output checking, there are two types of possible errors, with different consequences. On the one hand, confidentiality errors refer to releasing an output that should not have been released, breaching confidentiality. On the other hand, inefficiency errors refer to not releasing an output that was actually safe. The complexity of this problem led to the adoption of a hybrid approach that aims to minimize the errors, while trying to reduce the time it takes to perform the output checking [4].

2.2.1 PBOSDC vs RBOSDC

The two-model approach includes a rules-based output SDC (RBOSDC) (or rule-of-thumb) and a principles-based output SDC (PBOSDC). Even though the two models could be used independently from each other, the general idea is to combine the two models trying to take advantage of each model's strength. For example, the rule-of-thumb model is faster and can be used to make a first selection of easily classifiable outputs. After this step, the principle-based model can be applied to the remaining output to analyze it in more detail and take context into consideration [19].

The hybrid model spreads the responsibility for clearing an output. In a rules-based model, the responsibility lies with the people that design the rules. In this principles-based model, the responsibility lies with each individual output checker and the researcher [19].

The rules-based model focuses on preventing confidentiality errors, accepting the possibility of inefficiency errors and uses strict rules. This model requires less interaction and expertise on the checker's side. In this case, the checker uses simple rules-of-thumb to label an output as safe or unsafe. It is important to point out that even though the rules are very strict, there is a small chance that it may lead to wrong labelling, mainly because context is not taken into consideration [19].

The main advantage of the rule of thumb principle is that rules can be applied more or less automatically by both researchers and staff members with only limited knowledge of disclosure control, reducing the burden of the output checking process [19].

The principles-based model (PBOSDC) takes context into consideration and appears because simple rules for output checking are insufficient, as simple rules can never take into account the full complexity of research output. For instance, a table that contains very small cell counts (maybe even some cell counts of 1) is not necessarily unsafe if, for example, the data are not confidential. Moreover, when checking output for disclosure, one should take into account the fact that the checked result may be related to another (previously) released output. This combination with other outputs can increase the risk of disclosure of the results [19].

Although this model demands significant time and resources, it minimizes the probabilities of *false positives* and *false negatives*. The main disadvantages of the principles-based model are the fact that it requires more time from the statistical offices teams and that it relies on serious training of the data provider's staff and of researchers [19].

2.2.2 Disclosure Rules and Output Classes

There are extensive protocols with rules that the statistical offices teams apply when performing the output check. These rules include, for instance, that aggregated results (e.g., descriptive statistics such as counts, averages, totals, proportions, etc.) must not refer to a small number of statistical units (fewer than 3 or 10 units, depending on the data in question) or that there must be no concentration of observations in the same cell of a given row/column of a frequency table, nor a dominance of the aggregated value by a small number of units in a magnitude table.

When doing their research, users must be informed about the main rules applied during the verification of the outputs they intend to extract, including that only aggregated data can be provided, under no circumstances is it permitted to make individual-level data (microdata) available and that it should not be possible to identify individual statistical units or obtain

information on previously unknown attributes related to individual statistical units from the outputs [4].

The set of rules can be found in *Guidelines for the Checking of Output Based on Microdata Research* [4].

RBOSDC Considerations

For RBOSDC Considerations, the rules are grouped below according to their disclosure control logic. Rules were numbered according to the order in which they appear in table 2.2.

Sample Size and Structural Thresholds These rules ensure that outputs are based on enough observations to protect confidentiality.

- **Rule 1**: Each cell/value must be based on at least 10 unweighted units
- **Rule 5**: Outputs derived from statistical models (e.g., higher moments, R^2, t-tests) are safe if the model has at least 10 degrees of freedom
- **Rule 9**: Multivariate outputs (e.g., factor and correspondence analysis) are safe if based on at least two variables and a sufficient number of observations

Group Disclosure / Dominance Rules These rules prevent identification through concentration of values or units in one group or respondent.

- **Rule 2**: No cell should contain more than 90% of units in any row or column (group disclosure)
- **Rule 3**: The largest contributor must not exceed 50% of the cell total
- **Rule 10**: Correlation coefficients must not be exactly -1, 0, or 1

Disclosure by Identity or Extreme Values These rules address outputs that directly or indirectly reveal specific individuals or sensitive data points.

- **Rule 4**: Outputs that directly or indirectly reflect individual data points (e.g., maxima, percentiles, residuals) must not be released

Graphical and Model-Based Outputs Rules for ensuring models and visual outputs don't expose sensitive patterns.

- **Rule 6**: Graphs are not permitted; only underlying cleared data may be used
- **Rule 7**: Regression outputs (linear or non-linear) must withhold at least one coefficient (e.g., intercept)
- **Rule 8**: No residuals and no plots of residuals should be released.

Table 2.2: Disclosure control rules-of-thumb applied to each output class

Output Class	Applied Rules
Class 1: Frequency tables	Rule 1; Rule 2
Class 2: Magnitude tables	Rule 1; Rule 2; Rule 3
Class 3: Maxima, minima, percentiles	Rule 1; Rule 4
Class 4: Modes	Rule 1; Rule 2
Class 5: Means, indices, ratios, indicators	Rule 1; Rule 3
Class 6: Concentration ratios	Rule 1; Rule 3
Class 7: Higher moments of distributions (variance, covariance, kurtosis, skewness)	Rule 1; Rule 5
Class 8: Graphs (descriptive or fitted values)	Rule 6
Class 9: Linear regression coefficients	Rule 7
Class 10: Non-linear regression coefficients	Rule 7
Class 11: Estimation residuals	Rule 8
Class 12: Summary and test statistics	Rule 5
Class 13: Factor analysis	Rule 9
Class 14: Correlation coefficients	Rule 1; Rule 10
Class 15: Correspondence analysis	Rule 9

PBOSDC Considerations

The following principles-based considerations guide output checking when deterministic rules are insufficient. These are not strict thresholds but contextual criteria applied by trained checkers to assess disclosure risk in more nuanced scenarios. For Sample Size and Contextual Adequacy, these checks go beyond raw counts to assess whether small groups are being indirectly highlighted.

- Under PBOSDC, counts between 2 and 9 may be released only when the residual identification risk is judged negligible after considering: the availability of auxiliary information that could aid re-identification, the age of the data, sample/design features that limit credible back-calculation, whether contributors' rank ordering or membership is publicly known, and the utility/reliability of the estimate. If any doubt remains the rule-of-thumb should be applied.

- Even if the sample size exceeds the minimum, does the output expose narrow slices of the population (e.g., 11 women in a specific occupation in a small municipality)?

In Risk of Re-identification by Combination, these considerations address situations where outputs appear safe in isolation but could lead to disclosure when combined with previous outputs or public information.

- Can this output be cross-referenced with previously released tables to isolate individuals or small groups?
- Are unique combinations of attributes (e.g., geography + rare job + age) presented that increase re-identification risk?
- Are statistical models (e.g., regression, correlation) based solely on categorical variables, effectively revealing unique category combinations?
- For percentiles, could external knowledge of rank ordering (e.g., league tables of firms) make the percentile respondent identifiable? If so, the percentile should not be released.

In Sensitivity of Attributes or Outliers, these considerations address whether the attributes shown carry intrinsic sensitivity or reflect individuals with extreme values.

- Does the output refer to sensitive characteristics (e.g., health, income, sexual orientation)?
- Are any extreme values shown that might be attributable to a known individual (e.g., the only billionaire in a region)?
- For minima, maxima and percentiles, the researcher will always have to provide additional information to assure that the risk of disclosure is negligible.
- What is the variance (dispersion) of values around the reported percentile? If it is very low or zero (many identical values), there is a risk of group disclosure at that percentile.
- Is the variance around the percentile extremely high with a single value near that cut point (e.g., a distribution with many high and low values but one central value, so the 50th percentile/median reveals that unit)? If so, the percentile may pinpoint an individual and should not be released.

In Visual or Graphical Interpretability Risks, visual representations can pose additional risks by revealing patterns not obvious in raw numbers.

- Do plotted values (e.g., scatterplots, maps) visually isolate individuals or rare categories?
- Are graphical elements (e.g., fitted regression lines) revealing extreme influence or residual patterns?

Case of Statistics Portugal (INE)

Consistent with this methodology, Statistics Portugal emphasises RBOSDC for *very well understood/confirmed* classes (e.g., frequencies, magnitudes) and PBOSDC for *provisional/no knowledge* classes (e.g., quantiles, selected diagnostics/graphics) [15, 16].

INE's most recent guide articulates the hybrid design (RBOSDC + PBOSDC), sets default thresholds for common outputs, and specifies reviewer–researcher responsibilities. In particular, it (i) codifies baseline rules for well-understood classes (e.g., minimum cell counts, dominance limits, residual degrees of freedom for models); (ii) directs contextual review for provisional classes (e.g., quantiles, certain diagnostics and graphics); and (iii) documents

researcher-facing requirements (only aggregated outputs, prohibition of microdata, justification for borderline cases). The most recent version also consolidates the access workflow, roles, and disclosure policy, formalizing the hybrid rules-based/principles-based approach and clarifying researcher obligations (output request form, aggregation restrictions, and documentation). See Appendix D for INE baseline thresholds at a glance [15].

2.2.3 Automating output checking

Desirable characteristics

In *Understanding Output Checking*, Eurostat listed the desirable characteristics of an automated output checking process in order for it to be effective and successful, labelling them as either essential, important and desirable and dividing them into three categories: perceptions and usability, statistical characteristics and implementation [19].

Among the essential characteristics, in the perceptions and usability the authors named the importance of user and data center manager acceptance. In statistical characteristics they consider as essential the need for the model to deal with different types of outputs and to adapt to the statistical offices protocols. They also listed that primary and secondary disclosure must be addressed. Regarding the implementation they classified as essential that the system is scalable over users and outputs, that the tool should not require regular operational adjustments and that it should be able to handle auxiliary information [19].

To know that at which stage should it be applied, in the same document, they suggest different stages in which the system could be applied:

- *End-of-Activity Review* (EoAR): Checking after research is complete, which is the current framework but it is usually slow.
- *Reproductive Review (RR)*: Running outputs through predefined validation models before approval.
- *Seamless Within-Activity Review* (SWAR): Integrating checks directly into researcher workflows.
- *Discretionary Within-Activity Review* (DWAR): Allowing researchers to selectively check outputs before submission after applying SWAR.

Beyond theoretical guidance, concrete tools have been proposed to implement automated output checking in practice. In Automatic Checking of Research Outputs (ACRO) - a tool for dynamic disclosure checks [43], the authors introduce a tool developed in *STATA* for dynamic disclosure checks. The purpose of the tool is to distinguish between safe and unsafe outputs as well as the ones that require further analysis based on the disclosure risk. In their findings they concluded that ACRO has good potential to reduce manual output checking. The main challenges were the initial adoption of the tool by the team and the need for the tool to support additional programming languages in order to be useful in secure data centers.

Machine Learning-Assisted Output Checking

In *Towards Machine Learning-Assisted Output Checking for Statistical Disclosure Control* [17], the authors explore the use of machine learning to partially automate output checking, following the rule-of-thumb approach.

In their experiment, they took 14 rules and rewrote them in the following attributes: AnalysisType (e.g. Frequency Table), Output, Confidential (whether the original data is confidential), Context (used variables and filters) and Decision (either Yes for safe or No for unsafe outputs) (see 2.3). They then created synthetic datasets based on different rules and attributes, for both the cases in which Decision is Yes or No. Finally, they trained a feed-forward neural network with 2 hidden layers of 64 units each and obtained a 94.08% accuracy, a 4.2% false positive rate and a 7.4% false negative rate. Furthermore, they conducted a series of experiments to find out if a neural network is able to generalize when exposed to samples generated from rules it has not been exposed to during training.

Table 2.3: Examples of rules encoding used in *Towards Machine Learning-Assisted Output Checking for Statistical Disclosure Control* [17]

	Rule 1	**Rules 3a/3b/3c**
Analysis Type	Frequency Table	Maximum / Minimum / Percentile
Output	Number of units in each cell	Value of Maximum / Minimum / Percentile
Confidential	YES/NO (YES means the data on which the frequency table is computed are confidential)	YES/NO
Decision	YES/NO. The decision is NO if: – the output is confidential *and* – some cell contains fewer than 10 units *or* – a single cell contains more than 90% of the total number of units in a row or column.	YES/NO. The decision is NO if data are confidential.

One of the main limitations of the research was that it did not use real log files from the natural output request process, because they are not public. Furthermore, the research did not address the principle-based model. However, the results were promising as the system showed very good results when trained with the whole data set and was able to generalize when dealing with rules not present in the training data.

Additionally, *COACH* (*COmputer-Assisted output CHecking with Human-in-the-Loop*) extends the ML-assisted line of research by reproducing the simulated-data experiments of *Towards Machine Learning-Assisted Output Checking for Statistical Disclosure Control*, benchmarking multiple algorithms (e.g., feed-forward neural networks, LightGBM), and introducing pre-processing techniques to bridge gaps between simulated and real data. A key innovation is its Human-in-the-Loop (HITL) interface, where human checkers review predictions (safe/unsafe), provide feedback for retraining, and inspect explanations via SHAP. This integration of human expertise significantly improved results: on real test data, HITL feedback reduced false positives from 38 to just 3, thereby mitigating the most critical risk in disclosure control (incorrectly releasing unsafe outputs) while improving model adaptability to real-world cases [46].

Building on this, *COACH+* enhances the framework in two main directions. First, it extends coverage to graphical outputs by incorporating convolutional neural networks (CNNs)

alongside tabular models. Second, it introduces a streamlined front-end where checkers can directly upload Excel files and figures, removing the need for manual value entry. In experiments, the combination of CNNs with tabular ML models outperformed random baselines for image classification, while the refined HITL workflow further reduced false positives and improved reviewer efficiency. Overall, *COACH+* strengthens semi-automation within Trusted Research Environments (TREs), showing how HITL-based feedback loops can enhance both performance and usability in operational disclosure control settings [45].

In a related line of work, Rigaud et al. (CASD) describe a production-ready, ML-assisted export-level checker trained on a large corpus of past manual decisions, using feature engineering over diverse file types (including OCR and regex for documents) to generate structured representations of outputs. The system integrates a Human-in-the-Loop (HITL) loop that routes high-risk outputs for manual review, while low-risk ones can pass with reduced oversight. In their evaluation, XGBoost provided the best trade-off between recall and manual-control rates, outperforming random baselines. Importantly, the authors outline a strategy of continuous training and file-level modeling to adapt the checker over time. This work complements rule-of-thumb tools (ACRO/SACRO) and experimental ML prototypes (COACH/-COACH+) by demonstrating how a semi-automated pipeline can perform on real, heterogeneous export logs rather than on synthetic tasks [41].

2.2.4 Other Statistical Offices

There are researches and changes happening now at some Statistical Offices in terms of automating the output checking processes. However, this section provides an overview of the current official protocols at these institutes.

At the Netherlands National Statistics Institute, the user can access part of the data at their website by agreeing to conform to the legal protections surrounding the datasets. The allowed researchers can also access the microdata in a controlled setting in two locations (the Hague and Heerlen) in which they can apply their own software to perform their analysis. The researcher has to pay a fee for the service and a tariff for using the facilities. There are also remote facilities that aim to replicate the setting of these controlled environments in which researchers can access microdata at specific universities or other research institutes, providing the confidentiality requirements are guaranteed [8]. The output checking is done manually by Statistics Netherlands [7].

Similarly, in Germany there are also two options for accessing microdata: the Research Data Centres (RDC) and other institutes. At the RDC the researchers can do their analysis in controlled environments at the premises of the statistical offices that are equipped with the most common statistical programs or languages (*R, SPSS, Stata, SAS*). On remote execution settings, data structures files that resemble the original datasets are made available, in which the researchers can prepare the code in the mentioned languages that will then be applied to the original datasets by the staff of the statistical officers. In both cases, the output checking is then done by the RDC staff [18].

On the other hand, Statistics Denmark has a system in which users need to run the tabular outputs through tau-Argus before release. Similarly, in France, the statistical offices allow for reproductive review but as a way to ensure the validity of outputs and not for checking the disclosure risk of outputs [19].

In Australia, the Bureau of Statistics created a tool, TableBuilder, in which the research-

ers can do their analysis and create their own tables based on microdata. These tables are automatically processed to protect privacy and confidentiality before the output is provided to the researchers [37].

In Japan, following a 2019 revision of the Statistics Act, the National Statistics Center operates a nationwide on-site secure microdata service (21 facilities as of July 2023, mostly at universities) using thin-client access over SINET; users analyse on virtual desktops and cannot export files directly - outputs are retrieved and checked by Center staff. Current rules include a "≥10 units" principle for statistics, "≥10 residual degrees of freedom" for models, dominance rules for establishment surveys, and optional protections against group disclosure. Practice has highlighted issues such as hidden attributes in composite tables, recalculation risks when totals and breakdowns are jointly released, and converting decision-tree outputs to frequency-table form for checking. Ongoing work explores quantile disclosure control (rounding-suppression) and proposes thresholds such as groups with ≥40 units and an interquartile range ≥30% of the median [1].

2.3 Machine learning for classification tasks

In the output-checking context, tabular features are heterogeneous, often sparse, and mix categorical encodings with engineered indicators. Gradient-boosted decision trees (GBDT) are frequently highlighted for such settings because they capture nonlinear interactions without extensive preprocessing, and handle missing values naturally. Among modern GBDT implementations, *XGBoost* and *LightGBM* are widely cited due to scalability and mature tooling [11, 31].

XGBoost is a gradient boosting method that combines decision trees with regularisation to improve both accuracy and generalisation. It handles missing values efficiently and is well suited for datasets with many sparse or one-hot encoded features, which are common in microdata [11]. *LightGBM*, on the other hand, introduces optimisations such as Gradient-based One-Side Sampling (GOSS), which focuses on the most informative data points, and Exclusive Feature Bundling (EFB), which reduces dimensionality by grouping features that rarely appear together. These techniques make it faster to train while maintaining high accuracy [31]. Together, these properties make XGBoost and LightGBM highly practical for output checking, as they can efficiently handle mixed tabular inputs, deal with missing data, and provide probability scores that can be adapted to meet disclosure control policies.

We also have tree ensembles vs. deep neural networks on tabular data. Comparative studies on diverse benchmarks consistently show that tree ensembles (including XGBoost and LightGBM) typically match or outperform contemporary deep neural architectures while requiring less hyperparameter tuning. Gains from deep models tend to appear only with very large samples or when combined with learned representations from text or images [44]. Importantly, the recent studies applying machine learning to output checking (*COACH*, *COACH+*) also adopted LightGBM as part of their pipelines, further reinforcing its suitability for this task [46, 45].

2.3.1 Evaluation metrics

Output checking is typically cast as a binary classifier (unsafe vs. safe) whose scores are mapped to three operational bands: *safe*, *review* (HITL), and *unsafe*. Given the class imbalance

(unsafe $\ll$ safe) and the higher cost of false positives, recent work emphasises precision and low false-positive rates.

Primary metrics for triage are (i) **Precision**: measuring the proportion of correctly identified unsafe outputs among those flagged, to limit unnecessary reviews or blocks; (ii) **False-Positive Rate (FPR)** and **specificity** (TNR $= 1 -$ FPR) for the safe class, with operating points chosen to keep FPR low, especially for the *block* band; (iii) **Calibration** (Brier score, reliability curves): ensuring that predicted probabilities correspond to stable precision over time. PR AUC is often reported, but thresholds are typically selected by precision/FPR rather than recall.

For point comparisons, the **Matthews Correlation Coefficient (MCC)** is widely recommended because it remains reliable under imbalance and accounts for all cells:

$$\text{MCC} = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}},$$

with range $[-1, 1]$ (best $= 1$) [12].

While many operational triage systems prioritise precision and FPR, this dissertation adopts unsafe-as-positive with recall/FNR as primary, reflecting a conservative stance against missed unsafe outputs (see Section 3.6.3).

2.4 Large Language Models

Large Language Models (LLMs) are artificial intelligence models designed to understand and generate natural human language. These models are based on neural networks with billions of parameters and are trained on vast amounts of data. LLMs are capable of performing several natural language processing tasks, including text generation, code generation, translation, classification, and prediction [40].

2.4.1 Architecture

LLMs are based mainly on *Transformers*, a deep learning architecture introduced in 2017 in *Attention is All You Need* [50] that is designed to process sequential data by modeling the relationships between elements in a sequence. This technology is based on the attention mechanism, which allows the model to focus on different parts of the input, eliminating the need for recurrence and making natural language processing more efficient [40].

The transformer architecture consists of an encoder, which is responsible for understanding and representing the input sequence, and a decoder that focuses on generating output sequences. There are different model designs in LLMs based on whether their architecture is encoder-only, decoder-only or encoder-decoder that determine which specific kinds of task the model is best suited for:

- *Encoder-only* models - focus on understanding and representing the input sequence without generating an output. These models are adequate for tasks that require understanding, classification or extraction from the input, such as text classification or sentiment analysis. Some examples are *BERT* (Bidirectional Encoder Representations from Transformers) and *RoBERTa*, an optimised *BERT* variant [40].

- *Decoder-only* models – focus on generating sequences in an autoregressive manner by processing inputs step by step and predicting the next token based on prior context. These models are best suited to perform tasks that require sequential output generation, such as text and code generation or chatbots. Examples include the GPT series (e.g., *GPT-3*, *GPT-4*), *LLaMA*, and the *Mistral* models (e.g., open-mixtral-8x7b, open-mixtral-8x22b, mistral-medium-latest) [40].

- *Encoder-decoder* - these models incorporate both the encoder and decoder, making them well-suited to perform tasks that require transformation from one sequence to another, such as text-to-text generation tasks, text summarization or text-to-code generation. Some examples are *T5* (Text-to-Text Transfer Transformer) and *BART* (Bidirectional and Auto-Regressive Transformers) [40].

2.4.2 Prompt engineering

Prompt engineering refers to the practice of designing and optimizing the input text (or *prompt*) provided to a LLM to obtain specific and desired results. This process involves designing clear instructions, providing context and/or examples to help the model better understand the tasks without altering the models parameters [36]. There are several approaches to prompt engineering some of which are listed below.

Zero-shot prompting refers to a prompt engineering technique that asks the LLM to answer queries or perform tasks without providing any prior examples, relying solely on its pre-trained knowledge and the instructions given in the prompt [36].

Few-shot learning also known as *In-context learning* involves providing a LLM with some examples (input-output pairs) directly within the prompt. These examples demonstrate the task, helping the model understand the pattern and generalize it to produce the desired responses. The selection of examples can significantly influence the model's behaviour and introduce biases [36].

Single-turn instructions refers to the technique of giving a single, complete prompt to the LLM that includes all the relevant information (such as instructions, examples, and context) to perform a task. This approach prioritizes efficiency and clarity by providing everything the model needs to produce a response without the need for multiple iterations [36].

Multi-turn instructions involves multiple iterations with the LLM, by providing feedback, refining the task, or guiding the model incrementally. Each turn builds upon the previous one, for which reason this approach is especially useful for solving complex tasks that benefit from back-and-forth interaction [36].

Reasoning in LLMs refers to the ability of these models to generate answers to logical problems, task planning or apply critical thinking by providing reasoning [36]. For instance, this might include asking the LLM to explain the reasoning behind a given answer. Reasoning can be achieved by combining carefully designed prompting styles which structure the input to encourage logical and structured responses. Additionally, reasoning can be further enhanced by training the models on specialized reasoning datasets that include examples that require logical thinking, structured reasoning or problem-solving skills. Some of the prompting techniques for reasonings are:

- *Chain-of-thought* (CoT) - refers to proving the model with examples that contain not only the inputs and outputs but also the reasoning process that connects them, showing how

to break down a problem into logical step-by-step reasoning steps. By including the reasoning information in the examples, the model is encouraged to generate responses that follow the same step-by-step approach improving the ability to perform complex tasks that require logical thinking [36]. Auto-CoT extends this idea by automatically generating diverse reasoning paths and demonstrations through sampling techniques [48].

- *Self-consistency* - based on the idea that the correct answer to a given query is more likely to appear in different answers and, therefore, includes generating several responses to the same prompt and choosing the most frequent answer, reducing the impact of random errors or inconsistencies in reasoning [36].

- *Tree-of-Thought* (ToT) improves the LLM reasoning capability by allowing the model to explore multiple reasoning paths for a given query and reconsider some steps. It involves looking ahead to evaluate the possible outcomes of a specific reasoning process and backtracking to rethink the steps that might have led to undesired outcomes [36].

Contextual Prompting [9] refers to embedding external content such as metadata or domain-specific rules directly into the input prompt to enhance the LLM's understanding of the task. This technique is particularly useful for structured data tasks where metadata improves accuracy.

Automated prompt engineering refers to using an agent or similar automated system that interacts with LLM and that provides some feedback to improve the prompt. It may include approaches such as generating instruction candidates and then searching for candidate solutions to maximize a selected score function introduced in *Large language models are human-level prompt engineers* [52]

2.4.3 Fine-tuning

LLMs perform well with several tasks, but struggle, sometimes, with specific requests. *Fine-tuning* is the process by which a pre-trained LLM is further trained on a specific dataset to adapt it to specific tasks or domains. By updating the model's parameters using task-specific or domain-specific data, fine-tuning enhances the model's performance for specific tasks, making it accurate and specialized when compared to a solely pre-trained model. Some of the different techniques are presented below [36].

Transfer learning takes a pre-trained model and further trains it on a new dataset specific for a task [36].

Supervised Fine-Tuning (SFT) is the most common and straightforward fine-tuning method, where the model is trained on input–output pairs to directly imitate the desired behaviour [38]. In this setting, the LLM learns to generate the correct response when presented with a structured prompt, without requiring preference data or reinforcement learning. SFT is widely used to adapt general-purpose models to domain-specific applications, such as legal text processing or biomedical question answering.

Instruction-tuning refers to the process of fine-tuning LLMs using datasets that include instructions alongside input-output pairs. These instructions usually include multi-task data in natural language that will guide the LLM in understanding and responding to user queries by following the provided instructions in the prompt [36].

Alignment-tuning refers to the process of ensuring LLMs generate responses align with human values and expectations, addressing issues such as false, biased and harmful outputs sometimes generated by LLMs. A model is considered aligned if it performs well according to the "HHH" criteria: helpful, honest and harmless [36].

A key element in aligning models is *Reinforcement Learning from Human Feedback* (RLHF), a technique that incorporates human feedback into the fine-tuning process of large language models. RLHF is an iterative process that involves two main steps: *reward modeling* (RM) and *reinforcement learning* (RL) [36]. In reward modeling, a separate classifier (the reward model) is trained on human-labeled data according to the "HHH" criteria (helpful, honest, harmless), ranking LLM-generated responses into preferred and non-preferred. The reward model is then used to guide reinforcement learning, where the LLM's parameters are adjusted to maximize the likelihood of producing preferred responses.

An alternative approach, *Direct Preference Optimization* (DPO), introduced in *Direct Preference Optimization: Your Language Model is Secretly a Reward Model* [39], removes the need for training a separate reward model. Instead, DPO directly updates the LLM by learning from paired preference data, where one response is marked as preferred over another. Compared to RLHF, DPO is computationally more efficient, but it relies heavily on well-curated datasets and may struggle with tasks where preference rankings are subjective or inconsistent.

Parameter-Efficient Fine-Tuning (PEFT) is a collection of methods designed to fine-tune models for specific tasks, updating only a portion of the model's parameters and therefore reducing computational cost and memory requirements. The selection of parameters is based on the idea that not all model parameters contribute equally to learning task-specific features. Some of the most used techniques are *prompt tuning*, *LoRA*, and *adapters*. While prompt tuning adds learnable embeddings (vectors) to the model's input that are stored independently from the model, keeping the original weights and architecture unchanged, LoRA introduces trainable low-rank matrices into specific layers, adjusting some parameters but keeping the original model's architecture. Lastly, adapters introduce small neural modules between the model layers, modifying its architecture and adding some parameters [32].

2.4.4 LLMs and Official Statistics

In June 2023, the European Parliament approved a regulation on artificial intelligence (the AI Act) that aims at ensuring that AI applications are safe, transparent, accountable, nondiscriminatory, and environmentally sustainable. This act states that human supervision should prevail in the creation of new tools, outlines specific responsibilities for AI providers and users, and categorizes AI applications according to the risk they pose to human safety [5].

The lowest level, "limited risk", stated that AI systems must adhere to baseline transparency guidelines and that users should be able to opt for or against continuing to use the tools. Among the high risk AI technologies are the ones that could compromise safety or violate basic rights. The prohibited risk ones include applications that pose a severe threat to human safety and are forbidden, including social scoring systems, real-time remote biometric identification tools, with possible exceptions for serious criminal investigations after receiving judicial authorization [5].

The *AI Act* further stipulates that Generative AI systems, including large language models (LLMs), must adhere to specific transparency obligations. These include providing summar-

ies of copyrighted material used during training and ensuring that outputs are clearly acknowledged as AI-generated content [5].

In parallel, in May 2023 the European Commission published internal guidelines governing the use of Generative AI by its staff. These guidelines emphasise strict caution in the handling of sensitive or non-public information, prohibiting its disclosure to online AI systems. They also instruct staff to critically assess AI-generated responses for potential biases, factual inaccuracies, and violations of intellectual property rights, as well as to refrain from directly reproducing such outputs in official publications. Furthermore, staff are explicitly advised not to rely on publicly available generative AI models for time-sensitive tasks [5].

Complementing these legal and procedural safeguards, Eurostat's *An Introduction to Large Language Models and Their Relevance for Statistical Offices – 2024 Edition* sets out a framework for the safe and robust application of LLMs in official statistics. Recommended practices include the use of clean and curated training datasets drawn from trustworthy sources to reduce the risk of bias and toxic content, the systematic testing of models with prompts designed to reveal potential biases, toxic behaviour, or personal data leaks, and the continuous monitoring of outputs to identify problems and address them through fine-tuning. These measures are presented as essential to ensure that the deployment of LLMs in statistical offices remains both reliable and compliant with ethical and legal standards.

2.4.5 LLMs and data cleaning and transformation

One of the challenges of automating output checking at Statistics Portugal is resolving inconsistencies in historical output and output request documents. LLMs and regex-based scripts offer promising solutions for the processes of data cleaning and standardization. The challenges are related to differences in input structure (structured or semi-structured data), formats and languages.

In *Data Integration and Transformation using Large Language Models* [34], the authors used LLM to assist in *data cleaning and standardization*, performing tasks such as standardizing variable names, handling misformed data, converting data types, schema alignment or error detections. Metadata was used to provide context in prompt engineering techniques.

Similarly, in *LLMs with User-defined Prompts as Generic Data Operators for Reliable Data Processing* [33], a framework for data cleaning, transformation and modelling is introduced that replaces user-defined functions (UDFs) with user-defined prompts. Among the techniques they applied, they used LLMs to map ambiguous or inconsistent values to standardized values in the metadata.

The article *LLMClean: Context-Aware Tabular Data Cleaning via LLM-Generated OFDs* [3] introduces an approach for context-aware tabular data cleaning that uses *Ontological Functional Dependencies* (OFDs) - functional dependencies that ensure that data relationships are consistent with domain-specific meanings and hierarchical structures. The methodology involves introducing domain knowledge and constraints in the prompt design and refinement process, which allows the LLM to identify errors in datasets effectively. The authors use models like *GPT* or *LLaMA-2* to analyze column headers, metadata and samples of the data in order to map each column to a concept, enhancing the models ability to standardize the datasets values.

In *LLMs, with User-defined Prompts as Generic Data Operators for Reliable Data Processing,* LLMs were also used for rule-of-thumb anomaly detection by prompting the model to identify data

patterns and detect anomalies based on implicit or explicit rules in the dataset. The authors used few-shot learning, providing the LLM with examples, and applied iterative prompt optimization based on the feedback from the outputs [33].

Addressing semi-structured and unstructured data is also important. In *Redefining Health Care Data Interoperability* [51], the authors focused on cleaning and transforming various types of health data, including structured, semi-structured, and unstructured text as well as multilingual data, and incorporating metadata in the process. They addressed datasets or notes with free-text inputs from which LLMs extracted specific terms, such as medications, with the associated values. Metadata was used to guide the LLM in interpreting these ambiguous or inconsistent data, by prompting the LLM to align the extracted values with metadata rules or standards (e.g., mapping values to a vocabulary or standard terminologies). The authors addressed the different languages using LLMs to translate the data into a target language that preserves the domain-specific terminology.

Furthermore, in *Relationalizing Tables with Large Language Models: The Promise and Challenges* [25], the authors focused on using LLMs to transform semi-structured and unstructured data. The process breaks down the tasks into smaller steps and employs *chain-of-thought* prompting to guide the model. By providing the model with the reasoning for each of the required transformations, this system is capable of isolating and finding solutions to tasks such as values identification, mapping, and standardization.

Application of Large Language Models in Chemistry Reaction Data Extraction and Cleaning [24] presents a workflow for converting unstructured data into a structured format, using a combination of fine-tuning, prompt crafting, and verification. They used models such as *GPT-3.5* and *LLaMA-2* and tested different levels of prompt design (from no prompt to expert-designed prompts) and concluded that the results improved as the prompts became more detailed. Regarding fine-tuning, their findings show that the model's performance can improve up to 200% when trained with domain-specific data.

For Fine-tuning, in *RetClean: Retrieval-Based Data Cleaning Using LLMs and Data Lakes* [35], the authors explored how retrieval augmented generation and fine-tuning can improve cleaning processes. Their findings reveal that fine-tuning can improve up to 30% the accuracy of the models.

In *Automated LOINC Standardization Using Pre-trained Large Language Models* [49], the authors focused on mapping lab test orders and results to their corresponding standardized concepts in Logical Observation Identifiers Names and Codes (LOINC) - a standard ontology for clinical laboratory tests. This process involves taking the textual descriptions of lab tests often inconsistent, ambiguous, or varied in terminology and standardizing them by associating them with LOINC concepts. In their approach they fine-tuned a pre-trained LLM (*T5*) first on generic healthcare-related data and then specifically on the task of mapping lab tests to LOINC, applying *contrastive learning* to help the model distinguish between similar or ambiguous descriptions.

2.4.6 LLMs and feature engineering

In *Leveraging LLMs for Optimised Feature Selection and Embedding in Structured Data* [22], the authors introduced a methodology for combining feature selection techniques with text embeddings, using the *BERT* model to improve the accuracy in the prediction of student employability. Initially, the authors applied traditional feature selection methods to identify the

most relevant features in the prediction task. After selecting the features, they converted the tabular data into text using three variations of *BERT*, with the aim of leveraging the LLM's contextual understanding to generate richer representation of the data and capture relationships and patterns not easily identified with traditional methods. Finally, they applied feature selection on the textual features.

They used *Artificial Neural Networks* (ANN), *CatBoost* and a *BERT-based* classifiers. In the results without applying feature selection, ANN achieved the highest accuracy of 79%. After applying the embeddings, the *BERT-based* classifier had the highest accuracy of 85%. Their findings show that the best accuracies came from applying feature selection both before and after embedding.

On the other hand, in *Enhancing Traffic Incident Management with Large Language Models- A Hybrid Machine Learning Approach for Severity Classification* [21], the authors explored how to use LLMs to extract features from unstructured textual data (accident narratives) in order to combine them with structured data for accident severity classification. These new features allowed the model to capture contextual data from the accident, which would otherwise be difficult to obtain from traditional features. They used supervised fine-tuning on task-specific data to help ensure embeddings capture the relevant details for the classification purposes. In their findings, they concluded that accuracy significantly improved when the accident description features were included. From the several analyzed LLMs, their results show that the *BERT-based* models had the best performance, while in terms of the prediction tasks, *XGBoost* has the highest classification accuracy.

Similarly, in *Explainable Cognitive Decline Detection in Free Dialogues with a Machine Learning Approach Based on Pre-Trained Large Language Models* [14], the authors used LLMs to derive meaningful features from unstructured textual data (free dialogue), combining it with machine learning models for predicting cognitive decline. In their approach, they tried to capture subtle linguistic, semantic and reasoning patterns that indicate cognitive decline, but contrary to the accident severity classification article, they didn't do fine-tuning, focusing on prompt engineering. The authors broke the analysis into smaller tasks introducing instructions with domain-specific context and implementing zero and few-shot prompting.

The CAAFE [23] framework adds to the discussion, by focusing on the use of dataset descriptions, metadata and domain knowledge to generate and validate new features. The LLMs generate *python* code design to create new features based on chain-of-though and contextual prompting (using the provided data) and the result is evaluated based on metrics such as *mutual information* or *feature importance* scores and accuracy for a specific prediction task. The LLM iteratively excludes the non useful features based on these results, automating the process to determine and choose the features that contribute the most to the model performance. In their findings, they found that the accuracy increased up to 15% compared to baseline models and reduced the time and effort required for feature engineering.

In *Leveraging Large Language Models through Natural Language Processing to Provide Interpretable Machine Learning Predictions of Mental Deterioration in Real Time* [13], the authors generated a framework that uses LLMs to extract important features from doctor-patient dialogue to predict mental deterioration and that prioritizes explainability. In the output of their predictive model, they added information about which features were the most important to the prediction, something extremely important when dealing with sensitive decisions that need to be transparent and justifiable.

LLMs features and Class Probability Features are discussed in *Product Helpfulness Detection*

With Novel Transformer-Based BERT Embedding and Class Probability Features [29], the authors introduced a new method, BERT-RF, in which they combined LLM generated with *class probability features* to classify the helpfulness of product reviews. They used a *BERT-based* model to extract new features from text reviews and used them to predict the class probabilities of whether the review was classified as helpful or not. These probabilities were then added to the dataset as new features. Afterwards, they used machine learning models such as *Light Gradient Boosting Machine* (LGBM) to predict the helpfulness of the review, achieving an accuracy of 98%, significantly higher than using only the *BERT* generated features.

For LLMs and feature engineering without training data, in *LLM-Select - Feature Selection with Large Language Models* [30], the authors explore how LLMs are capable of selecting the most important features in a prediction task from the metadata, without accessing the training data. They used *chain-of-thought, few-shot, contextual prompting* and added an *iterative loop* that refines the prompts based on the accuracy results. In their research, they found that models such as *GPT-4* and *LLaMA-2*, when appropriately prompted, can rival traditional feature selection techniques without accessing training data, with the *LLaMa* models having the best performance.

2.4.7 LLMs for Compliance Rule Extraction and Checking

In *Automated Building Information Modeling Compliance Check through a Large Language Model Combined with Deep Learning and Ontology* [10], the authors used LLMs to extract the compliance rules from unstructured regulatory texts and transform them into structured, machine-readable formats in the form of *ontologies*, representing the features and their relationships. The LLM would then compare each of the building's features with the rules in the ontology, such as verifying whether the door's height aligns with the prescribed standards.

In their finding, they concluded that the LLM successfully extracted the compliance rules from the documents and translated them into a structured ontology. The model was also accurate in matching the building features with the corresponding compliance rules. The final results demonstrated high accuracy when validated against a manually curated benchmark dataset, achieving over 90% accuracy, depending on the complexity or ambiguity of the rule. The LLM was also effective in resolving some ambiguous rules and inferring missing thresholds or conditions by leveraging its pre-trained knowledge and the provided context.

Chapter 3

Methodology

3.1 Problem framing

At Statistics Portugal (INE), there are four *safe centers* located in the headquarters building in Lisbon, in its Regional Delegations in Coimbra and Porto, and a remote safe center in the Regional Directorate of Statistics of Madeira (DREM). In these centers, researchers can access 71 microdata datasets to perform their analysis [28]. The idea behind the safe centers is to give researchers the maximum freedom in analyzing the data files, so they will produce outputs of varying sizes and forms (e.g., *tables, plots*). These centers do not have access to the Internet, and researchers are forbidden to extract data without going through the *output checking* protocol [15].

The process researchers go through to access the data at INE (Statistics Portugal) starts by requesting access to the safe centers and datasets to the Dissemination Service, after which, if the request is approved, the Methodology Service of the Department of Methodology and Information Systems prepares the files. Once at the safe centers, the researchers receive the files, usually in *SPSS (.sav)* or *plain text format*, and they can do their research using whatever programming language or software they want [15].

After creating their outputs, the researchers give them to the Dissemination Service, along with a structured description, the *output request* documents. The output check is then performed by the Methodology Service of the Department of Methodology and Information Systems. First, a *rule-of-thumb model* and then a *principles-based model* (discussed in Section 2.2.1) are applied to check the output for possible data confidentiality problems. If the output is deemed safe for disclosure, it is delivered to the researchers who can then use and disseminate it without restrictions [15].

Currently, both these models are applied fully manually, which is highly time-consuming and difficult to scale [19]. Furthermore, this process currently heavily depends on specialized and trained reviewers who must learn the rules and nuances to perform the output checking task effectively.

This project proposes an approach to automating the output checking process as much as possible by integrating machine learning and large language models (LLMs) with complementary techniques for statistical disclosure control. The current section describes the methodology used, outlining the approaches, techniques, and models considered at each stage, as well as the key metrics used for evaluation.

3.1.1 Relation to Prior Work

A preliminary version of this research was presented at the 20th Iberian Conference on Information Systems and Technologies (CISTI 2025) as *AI-Driven Output Checking for Official Statistics: Leveraging LLMs and Workflow Automation* [6]. That paper introduced a prototype implemented in the n8n workflow orchestration framework, demonstrating how large language models could be integrated with deterministic rule checks to provide an automated end-to-end output checking pipeline. The prototype relied primarily on synthetic test cases and focused on feasibility rather than large-scale evaluation. With this prototype we were able to validate the capacity of LLMs to assist in the extraction of features from user queries, as well as to interpret the context of outputs and support disclosure control.

The present dissertation builds directly on that work while extending it in several important directions. First, it incorporates historical output requests and associated outputs from Statistics Portugal, enabling model training and evaluation on real data. Second, it introduces systematic metadata retrieval and feature engineering to standardise outputs and requests into machine-readable form. Third, the dissertation explores the use of machine learning models (XGBoost and LightGBM) for disclosure risk prediction, alongside LLM-based assessments, allowing for a comparative evaluation of rule-of-thumb, ML, and LLM approaches. Finally, the system design integrates these components into a hybrid framework that addresses the limitations of the earlier prototype, with a particular emphasis on reproducibility, calibration, and scalability for operational deployment.

These advances set the stage for the system architecture presented in the next section, where the rule-of-thumb, machine learning, and LLM components are integrated into a unified output checking pipeline.

3.2 System Architecture

Building on the preliminary prototype and the extensions described in this dissertation, the proposed system implements a semi-automated workflow that integrates deterministic rules, machine learning models, and large language models (LLMs) for output checking in official statistics. The aim is to reduce manual effort while maintaining transparency, reproducibility, and compliance with disclosure control requirements. The following subsections detail the main components of this architecture and how they interact within the hybrid framework.

Figure 3.1 provides an overview of the end-to-end architecture. The process begins with the researcher's submission of outputs and corresponding output request forms, and ends with a checked output accompanied by an explanation that can be reviewed by disclosure control experts.

The full system was developed in Python, with all components (rule-of-thumb scripts, feature extraction, ML models, and LLM integration) integrated into a single, reproducible codebase.

After receiving the output, the system goes through the *output preparation* in which the required features will be extracted and the output will be formed in a way so that the following disclosure safety assessments can be performed.

Third, a *rule-of-thumb checking* module applies a set of deterministic scripts to evaluate disclosure risks under established rule-of-thumb criteria. Fourth, the results are passed to a

machine learning disclosure classification model (ML-SDC), trained on historical labeled outputs. This model predicts whether the output is *safe* or *unsafe* for release by combining the extracted structured features with the results of the rule-of-thumb checks.

Finally, a *large language model agent* integrates all available information - including the researcher's description, metadata, rule-of-thumb outcomes, ML predictions, and qualitative guidelines - to produce a final classification. The output of the LLM contains the decision label (*SAFE*, *Unsafe*, or *Needs_Info*), as well as an explanation that documents the reasoning behind the decision. This explanation supports both researchers and disclosure checkers by improving transparency and making potential risks more explicit.

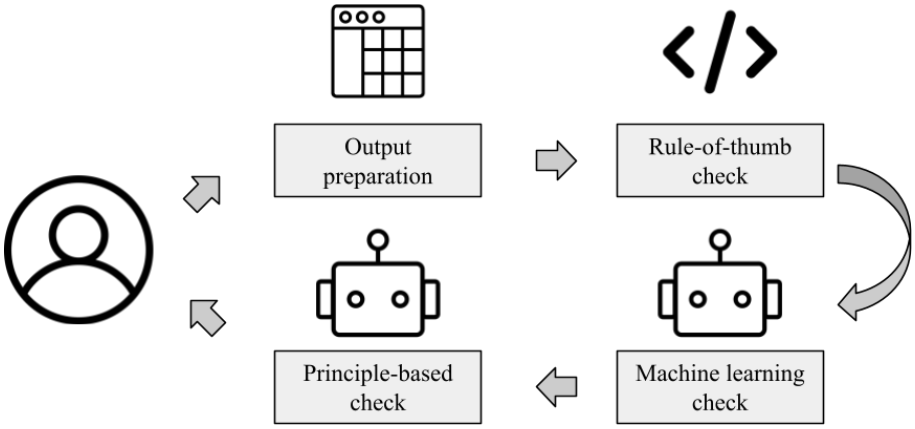

Figure 3.1: Overview of the automated output checking workflow. The process begins with output preparation, then undergoes rule-of-thumb, machine learning, and LLM checks, before returning a checked output with an accompanying explanation.

3.3 From Rules to Features

Building on the system architecture outlined in Section 3.2, the next step is to translate disclosure control guidelines into concrete features that the automated system can evaluate. In practice, this means converting the principles and rules described in the *Guidelines for the Checking of Output Based on Microdata Research* [4], and summarized in Section 2.2.2, into measurable variables. These features serve as the basis for both the rule-of-thumb and principles-based models, and their definition depends on the source of information available (e.g., output request, output, metadata, or original dataset). The remainder of this section details the required features and their sources, concluding with a structured summary.

First, the rule-of-thumb disclosure checks. Rule-of-thumb disclosure checks provide a deterministic safeguard to ensure that outputs derived from microdata do not compromise confidentiality. In this project, eight *checks* were derived from the rules, each addressing a specific aspect of disclosure safety. The complete list of required checks is presented in Appendix B, while Table 3.1 summarizes the applicability of each check to different output types. Below, we outline the key checks:

1. **Cell count (Check 1):** Each output (e.g., table cell, summary statistic) must be based on at least 10 unweighted units.

2. **Row and column proportions (Check 2):** No single cell should represent more than 90% of the units in any row or column.

3. **Maxima, percentiles, residuals (Check 3):** Outputs that directly or indirectly reflect individual data points cannot be released.

4. **Largest contributor (Check 4):** The value of the largest contributor in a cell should not exceed 50% of the cell total.

5. **Perfect or null correlations (Check 5):** Correlation coefficients equal to -1, 0, or 1 are flagged as disclosive.

6. **Model degrees of freedom (Check 6):** Model-based statistics (e.g., R^2, F-tests) must be estimated with at least 10 degrees of freedom.

7. **Variable count in multivariate outputs (Check 7):** Multivariate methods such as factor or correspondence analysis must include at least two variables and a sufficient number of units.

8. **Coefficient withholding in regression (Check 8):** At least one regression coefficient (e.g., the intercept) must be withheld from publication in both linear and non-linear regression models.

Table 3.1: Rule-of-thumb checks applicable by output type.

Output Type	C1	C2	C3	C4	C5	C6	C7	C8
Frequency Tables	x	x						
Magnitude Tables (Means, Totals)	x	x		x				
Maxima / Minima / Percentiles[a]	x		x					
Modes	x	x						
Means / Ratios / Indicators	x			x				
Concentration Ratios	x	x		x				
Higher Moments (e.g., Variance)	x					x		
Graphs / Visuals[b]								
Regression Coefficients	x					x		x
Residuals[c]								
Summary / Test Statistics	x					x		
Factor / Correspondence Analysis	x						x	
Correlations	x				x			

[a] Maxima/minima generally not released; percentiles are often withheld due to individual-level disclosure.
[b] Graphs are not released; however, cleared data may allow external reconstruction.
[c] Residuals and residual plots are not released.

It should be noted that some checks, particularly those related to dominance, are difficult to apply directly. For example, researchers are not expected to provide the value of the largest respondent, since doing so would itself reveal confidential information. In such cases, the

output should be retained until additional assurances can be obtained from the researcher or validated through secure data access.

Second, features for Principles-Based Disclosure safety assessment. In addition to deterministic rule checks, certain outputs must also be assessed against *principles-based* disclosure control considerations. Table C.1 in Appendix C provides the detailed mapping of risk considerations and key characteristics by output type. For clarity, Table 3.2 below summarizes the main principles-based risks, grouped according to the four categories defined in Section 2.2.2.

Table 3.2: Summary of principles-based risk considerations by output type.

Output Type	Sample Size Adequacy	Combination Risk	Sensitive Attributes	Graphical / Visual Risk
Frequency Tables	x	x	x	
Magnitude Tables	x	x	x	
Regression Outputs	x	x[a]	x	x
Percentiles / Max / Min	x	x	x	
Graphs / Maps		x	x	x
Residuals / Diagnostics	x		x	x
Time Series / Trends	x	x		x

[a] Includes regressions (linear or non-linear) and correlation coefficients based solely on categorical variables, which may reveal unique category combinations.

At present, these contextual features are evaluated manually by the output checking team, who interpret them in light of Eurostat and national guidelines. In the proposed system, however, they are assessed not through fixed thresholds but through machine learning and LLM-based agents, which can interpret context and identify nuanced disclosure risks that deterministic rules alone cannot capture.

3.3.1 Feature Sources

When the output alone is sufficient to determine release eligibility, it only needs to be normalised into the format required by the checking system. However, the more information or calculations are required, the more complex the whole system becomes.

Some of these calculations can be computed directly from the released output (e.g., row/-column group proportions in frequency tables). Others, however, require further information that can come from metadata, the output request document or even from the underlying dataset.

For example, when assessing the disclosure safety of a *frequency table*, cell counts and row/-column group proportions can usually be checked directly from the output. By contrast, for *magnitude tables* the same checks often require microdata: to compute the cell count the system must recover the exact variables used and replicate any filters; only after reconstructing the analysis population can it compute each cell's count. As a practical workaround, users are sometimes asked to provide the corresponding frequency table built with the same groupings and filters. However, to check the largest-contributor proportion, the original dataset is still required.

For this reason, features were labeled based on their source(s): Output, Output request

document, Metadata, Dataset or External source. This information is summarized in Appendix E.

Practical issues, such as mismatches in variable or dataset names, incomplete or ambiguous output requests, missing metadata, or restricted access to the original dataset, may prevent the computation of required features and, consequently, hinder a full assessment of disclosure risk.

The process of feature extraction, therefore, follows a priority order based on dependencies and data availability, ensuring that easily computable features (e.g., from outputs) are retrieved first, before resorting to more complex sources (metadata or microdata).

From the output (first and preferred): Whenever possible, the system reads directly from the submitted result (e.g., table, model printout, CSV). From these files, it extracts the reported statistics (for example, counts, means, percentiles, and coefficients), the cell values, and totals. For frequency and cross-tab outputs, it also derives row/column shares to assess group proportions. When provided, it records the number of observations used, the number of coefficients or variables, the positions of any percentiles, the presence and granularity of graphs or residuals, and the reported degrees of freedom.

From the output request: Next, the system parses the researcher's request to capture intent and scope. This includes identifying the dataset names and variables referenced, the filter conditions that define the analysis population, the intended statistic or output type (e.g., frequency table, regression, mean, percentile), the aggregation level (such as sex or NUTS 3), and any temporal references (for example, "2021" or "last five years"). This layer helps reconcile what was asked with what appears in the output.

From metadata: The request is then matched to dataset metadata to retrieve structural and sensitivity characteristics that qualify disclosure risk. Relevant details include the sample design, dataset size, the types of variables, the available geographic granularity, and reference period. Even when microdata are not accessible, these metadata provide an initial basis for disclosure control by flagging sensitive variables or confidential sources.

From the original dataset (conditional): When microdata access is permitted (for example, within a safe centre), the system can compute features that cannot be recovered from the output alone. These include unweighted cell counts when they are not printed, organisational dominance metrics, and the largest-contributor share. It can also derive degrees of freedom where not reported and perform variance or outlier checks, as well as aggregate residual diagnostics. Although this step was included in the system design, it was not fully implemented in this internship due to time constraints. Nevertheless, it remains a critical part of a complete disclosure control pipeline and provides a clear avenue for future work.

External data. To operationalise PBOSDC considerations (Section 2.2.2), the system could consult non-confidential external sources - e.g., prior official releases (to detect cross-release linkage/differencing), public rank lists. For this internship, no external features were used; results rely only on the output, the output request, metadata, and, where available, the underlying dataset.

3.4 Metadata Retrieval and standardisation

A consistent layer of *dataset metadata* was required to contextualize both output requests and extracted outputs (e.g., dataset contents, available years, variable names and labels, and auxiliary documentation). Although Statistics Portugal provides a catalog document with links to structured metadata for each dataset [27], the information is fragmented and not standardized, which makes it difficult to use directly in automated systems.

To address this, we carried out a process of metadata retrieval and standardisation, producing a single consolidated file that harmonizes all available metadata (including dataset names, descriptions, and variable dictionaries). The workflow followed six steps: (i) harvesting official links from the INE catalog PDF; (ii) parsing the PDF for dataset codes, names, frequency, and notes; (iii) bulk-downloading the linked files; (iv) post-processing container formats; (v) extracting variable dictionaries from Excel, PDF, and HTML sources; and (vi) merging the outputs into a unified, analysis-ready table. These steps are described in detail in Section 4.1.

3.5 Output standardisation and Feature Extraction

This section describes how outputs and output requests are standardized, processed, and encoded for use in rule-of-thumb, machine learning, and LLM-based disclosure checks. A unified logic ensures consistency across real-time evaluation and the processing of historical outputs used for training and validation.

Beyond normalisation, LLMs were used to propose candidate features from request text (variables, filters, group keys), which were then validated deterministically and binarised for the ML model. This hybrid pattern - LLM-assisted proposal, deterministic validation - follows recent work showing gains from leveraging language models for feature creation/selection while keeping the final representation tabular and auditable [22, 21, 14, 23, 29, 30].

3.5.1 Output Ingestion and Preprocessing

In the proposed system, multiple types of researcher-submitted outputs must be supported, including tabular results (e.g., frequency tables, cross-tabulations), statistical summaries (e.g., means, percentiles), and model outputs (e.g., regressions). Each output is transformed into a structured format (typically CSV) to allow for automated analysis. Where possible, metadata identifiers - such as dataset names or variable references - are also extracted from the output or accompanying documentation and linked to the standardized metadata layer described in Section 3.4.

For disclosure control, outputs are normalised into a tabular structure in which each row corresponds to a single reported item, enriched with contextual features such as dataset name, variables used, applied filters, and output values. This representation follows a similar approach to that described in *Towards Machine Learning-Assisted Output Checking for Statistical Disclosure Control* [17].

In some cases, preprocessing requires computing auxiliary statistics, such as row and column distributions in frequency tables. These enable the evaluation of disclosure risks, for example, verifying that no single cell accounts for more than 90% of a row or column total (group disclosure) [4]. Outputs that cannot be easily parsed - such as graphs, free-text

summaries, or unsupported formats - are flagged for manual review. Further details on feature computation are provided in Section 4.3.1.

3.5.2 Output Request Documents

As explained in Section 3.1, the output request is a form provided by the Dissemination Service that captures a structured description of the researcher's analyses. Statistics Portugal is developing a new version of this form. However, for this study, we use the previous version, since most historical requests are in that format and the new one is still under development.

The form is a table, where each row corresponds to one requested output. For each output, the researchers must provide the variables used, the filters applied, the result type (e.g., table, chart, correlation, regression model), the physical file name, and the folder where the output is located. Document-level metadata (applying to all outputs) should also be supplied: researcher name, submission date, project title, and dataset(s) used.

Some output request forms are completed clearly and allow the main characteristics of each output to be captured. However, much of the time, the information is missing, unstructured, or misspelled. Common problems include: (i) missing information (fields left blank); (ii) vague variables, filters, or aggregations (entries such as "All" or "Not applicable"); (iii) multiple outputs described in a single row; and (iv) references to other outputs either in the same document or in past documents.

Table 3.3: Illustrative output request

Field	Entry
Dataset name	1.8–IRS; 4.4–IMT; 6.3–QP; 2.4–SCIE
Variables used	The computation is based mainly on the variables *Total income* and *Consignations* (earmarked donations), taken from the IRS assessment notices and the front pages of the IRS returns (1.8–IRS).
Analysis rules (filters)	The filters remain the same, and the same process of aggregating consecutive years is applied to ensure that each observation corresponds to at least three units.
Result type	Parish–year–level table with: mean, median, and standard deviation of incomes (levels and logs); income shares of the top 1%, top 5%, and bottom 50%; the percentage of individuals in each income percentile; and the percentage who allocated 0.5% of IRS to charities.

Table 3.3 illustrates an output request that presents several issues for automation: (i) *multiple outputs per row*, as the "Result type" mixes several distinct outputs; (ii) *cross-references*, such as "filters are the same as before," which are not self-contained; (iii) *ambiguous datasets*, where multiple sources are listed but not clearly matched to specific outputs; and (iv) *vague variables*, such as "based mainly on," which do not specify the exact inputs.

To address these challenges, a stage was added in which relevant features are extracted and encoded from the request documents. Specifically, the output request is parsed to identify and standardise the *type of output* (e.g., frequency table, correlation), *filters or rules applied* (e.g., geographic restrictions, age thresholds), *grouping level* (e.g., freguesia, municipality, country), *sensitive content indicators* (e.g., income, top 1%), and *metadata elements*. These features are standardized using a combination of regular expressions, keyword mappings, and large language models (LLMs). This step allows the system to capture essential characteristics of the outputs without requiring access to either the outputs themselves or the underlying microdata.

This methodological choice is supported by recent studies showing that LLMs can effectively assist in data cleaning and transformation, particularly for tasks such as variable name standardisation, schema alignment, and handling ambiguous or inconsistent values [34, 33]. In line with the findings of *LLMClean: Context-Aware Tabular Data Cleaning via LLM-Generated OFDs* [3], metadata and domain knowledge were incorporated into prompts to improve contextual accuracy in mappings. Furthermore, since many requests contained free-text in multiple languages, the approach also draws on insights from multilingual cleaning frameworks, ensuring that terminology remains consistent across Portuguese and English entries [51].

3.5.3 Processing Historical Outputs

In addition to preparing current researcher submissions, this study required building a dataset of historical outputs for training and evaluation. These outputs, archived in multiple formats and folders, were retrieved and reformatted into the standardized tabular structure. A dedicated pipeline was developed to traverse directories, extract outputs and output requests, and harmonize their representation, ensuring consistency with the features used in the automated system. The details of this retrieval and processing workflow are provided in Section 4.2.

3.6 Disclosure safety Evaluation

The disclosure safety evaluation is a complex task that involves complementary reasoning paths. On the one hand, the strict rules (rule-of-thumb) allow for a quick labelling of either safe or unsafe outputs. While these rules are relatively straightforward, they may still require some contextual awareness [19]. For this reason, a hybrid system was developed that goes through three steps:

- Rule-of-thumb Output Statistical Disclosure Control (RBOSDC) - the first step in which the deterministic rule-of-thumb model is applied.
- Machine learning (ML) - in the second step, a machine learning model is applied that will predict the output disclosure safety based on historical data.
- Principles-Based Output Statistical Disclosure Control (PBOSDC) - in the last step, an AI-Agent analyses the output with the corresponding RBOSDC and ML labels, making a more complete analysis of the output disclosure safety.

3.6.1 Rule-of-thumb Output Statistical Disclosure Control (RBOSDC)

The rule-of-thumb layer constitutes the first stage of automated statistical disclosure control, providing a conservative baseline classification of each parsed output. By the time this stage is applied, the system has already identified the output type and computed the relevant descriptive statistics. The RBOSDC procedure then applies a set of predefined thresholds, derived from international guidelines, to these statistics in order to classify each output as *SAFE*, *UNSAFE*, or *NO_DATA*. In this way, every output undergoes a fast and transparent initial screening, which can subsequently be refined by the more nuanced principles-based review.

For each output, the RBOSDC layer produces a structured record that includes the overall rule-of-thumb label, the detailed results of individual rules, and supporting diagnostics. The overall label, stored in the field rule_based, can take one of three values: *SAFE*, *SAFE*, or *NO_DATA*. Alongside this, the system records per-rule Boolean indicators, to register information such as violations of the minimum cell-size threshold or breaches of the dominance rule. In addition, summary diagnostics are reported to quantify the extent of violations, for example, the number and share of cells failing Rule 1, or the residual degrees of freedom in regression models. These indicators provide a transparent explanation of why an output has been flagged and enable downstream modules to incorporate not only the final label but also the underlying evidence.

Special consideration was given to the treatment of maxima and minima, which are subject to a general "not for release" policy due to their high disclosure risk. Since these statistics are often embedded within broader descriptive tables rather than appearing as standalone outputs, the RBOSDC layer includes explicit flags that detect the presence of maxima or minima in the submitted result. This additional measure ensures that outputs containing such statistics are reliably captured and subjected to stricter review.

Implementation details of these checks are presented in Section 4.4.

3.6.2 Constructing Output-Level Labels

A key methodological challenge in preparing data for supervised models was the definition of reliable target labels. At Statistics Portugal, approval decisions are issued at the folder level: a submission is either cleared or rejected in its entirety. While this guarantees that no unsafe outputs are ever released, it also introduces ambiguity when training at the individual-output level, since safe outputs may appear within folders rejected due to other disclosive elements.

To mitigate this, we combined the institutional folder labels with the deterministic rule-of-thumb checks, using the latter to filter cases where the folder label was a poor proxy for per-output safety. The resulting curated labels preserve all genuinely safe outputs, retain cases of principles-based overrides (where rules flagged unsafe but the final decision was safe), and anchor deterministically unsafe outputs. This procedure ensures that the target variable for machine learning reflects institutional practice while remaining consistent with the output-level ground truth.

3.6.3 Machine Learning–Based Statistical Disclosure Control (ML–SDC)

To complement the deterministic rule-of-thumb layer, a supervised learning component was introduced. The purpose of this layer is to capture patterns of disclosure safety that cannot be fully addressed by rigid thresholds, especially in cases where principles-based considerations apply (see Section 2.2.2). In particular, the ML layer is useful for identifying borderline cases in which rule-of-thumb checks would suggest suppression, yet expert reviewers approved the output after contextual judgement. It also contributes additional sensitivity for magnitude tables, where contributor-level details are often missing from the researcher's submission.

Unlike the RBOSDC layer, which operates at the granularity of individual table cells, the ML classifier treats the entire output as a single observation. This representation aligns more

closely with how disclosure decisions are taken in practice and allows the model to exploit dependencies across features.

Training Data Preparation

Following the construction of the output-level labels described in Section 3.6.2, the next step was to prepare a clean and structured dataset suitable for machine learning. This required a combination of filtering, feature transformation, and encoding to ensure that heterogeneous information from both outputs and requests could be integrated into a single tabular representation. Through this pipeline, raw outputs, rule-of-thumb checks, and LLM-derived features were systematically transformed into a structured machine-learning table. This ensured that the models were trained on a dataset that was both representative of real-world output requests and sufficiently clean to support robust predictive performance.

The first step was to remove inconsistencies between institutional labels and rule-of-thumb classifications. In particular, outputs that had been marked as *Reprovado* by Statistics Portugal but which were flagged as *SAFE* by the RBOSDC system were excluded, as these cases represented likely folder-level artefacts rather than genuinely unsafe content. Outputs with missing rule-of-thumb classifications were also discarded.

Next, a set of core variables was selected, including identifiers, output type, dataset code, rule-of-thumb diagnostics, frequency table structure, model degrees of freedom, variables referenced, and filters applied. Textual noise in the variable lists was cleaned through regular expressions, removing artefacts such as boilerplate words or formatting tokens. Variables were then parsed into Python lists and transformed into a sparse binary representation using a multi-label binariser, thereby enabling their inclusion as features without imposing artificial ordering.

To harmonise the column names and avoid conflicts, all feature names were normalised to ASCII, lowercased, and stripped of punctuation, with deduplication applied where necessary. Categorical fields such as output type were expanded into dummy indicators, and interaction features were created, for example the product of row and column category counts to approximate table size. Rule-of-thumb outputs (*SAFE*/*SAFE*) and associated checks were recoded into binary form, and the institutional label (*Aprovado*/*Reprovado* for *SAFE*/*SAFE*, respectively) was likewise mapped to a numeric target variable.

Additional features were engineered from the structured information extracted during the output and output request feature extraction process. Mentions of variables, sensitive variables, filters, and aggregation terms were all transformed into binary dummy indicators, allowing their presence or absence to be explicitly represented in the feature space. For example, if an output request referenced the variable rendimento, a corresponding feature var_rendimento was set to one. In the case of sensitive variables, these dummy columns (e.g., request_sens_rendimento) were further collapsed into aggregate flags, such as whether any sensitive variable of any category was referenced.

This layered encoding strategy ensured that detailed request-level information was preserved in a machine-readable form, while the aggregate indicators provided higher-level summaries that capture the overall presence of sensitive content, filtering logic, or aggregation strategies. The combination of granular and aggregate features allowed the machine learning models to leverage both fine-grained signals and broader contextual patterns in predicting disclosure safety.

Finally, to balance the dataset and avoid domination by majority classes, stratified sampling

was applied within each combination of rule-of-thumb and institutional labels, capping the number of examples per cell at 200. This produced a balanced dataset suitable for model training and evaluation, while preserving diversity across output types and request contexts. To verify robustness, we also experimented with training on the full, imbalanced dataset. In this case, evaluation was performed on the same imbalanced test distribution, allowing direct comparison of the balanced versus imbalanced training regimes. Performance was assessed using metrics such as PR-AUC and MCC, which remain informative under imbalance and thus enabled a fair comparison between the two approaches.

Models considered

Given the tabular structure of the engineered features, we focused on gradient-boosted decision trees, which are widely recognised as the state of the art for heterogeneous tabular data. We evaluated two mature implementations, XGBoost [11] and LightGBM [31], chosen for their robustness, efficiency on sparse high-dimensional features, and ability to produce calibrated probability outputs suitable for disclosure control.

Both models were trained under a common recipe. We used stratified K-fold cross-validation with early stopping to prevent overfitting, and applied class weights to mitigate the natural imbalance between safe and unsafe cases.

To assess the marginal value of different sources of information, we performed ablation experiments based on cumulative feature bundles. The baseline set, B0, consisted of RBOSDC features such as rule flags and violation intensities. Bundle B1 extended this with structural descriptors of the output. Bundle B2 corresponded to B0 + B1, B3 to the variables and filters extracted from the output request, and Bundle B4 incorporated all the bundles (B0 + B1 + B3).

Metrics and operating points

For clarity, we adopted the convention of treating the unsafe class as the positive class. True positives (TP) are outputs predicted unsafe and truly unsafe, false negatives (FN) are unsafe outputs misclassified as safe and correspond to the *confidentiality errors*, false positives (FP) are safe outputs unnecessarily flagged as unsafe and correspond to the *inefficiency errors*, and true negatives (TN) are safe outputs correctly identified as such [19].

Evaluation was driven primarily by the need to minimise false negatives, that is, the risk of classifying a genuinely unsafe output as safe. Accordingly, recall (sensitivity) for the unsafe class was the main optimisation target, with the false negative rate (FNR) defined as $1-\text{Recall}$ monitored closely.

Secondary metrics were reported for completeness and comparison: overall accuracy, precision, precision-recall curve (PR AUC), the Matthews Correlation Coefficient (MCC), the latter two being particularly informative under class imbalance [12]. Precision and the false positive rate (FPR) were also tracked, but they were not the primary tuning objectives.

Under this convention, recall is computed as $\text{TP}/(\text{TP}+\text{FN})$, precision as $\text{TP}/(\text{TP}+\text{FP})$, and the false positive rate as $\text{FP}/(\text{FP}+\text{TN})$.

To quantify the incremental value of different sources of information, all metrics were reported across the cumulative feature bundles B0–B4. This allowed us to evaluate not only the absolute performance of the models, but also the marginal contribution of structural

descriptors, request-derived variables and filters, and metadata to the overall predictive accuracy.

3.6.4 LLM Statistical Disclosure Control (LLMSDC)

Finally, an AI-driven layer was introduced to complement the rule-of-thumb and machine learning components. This layer, implemented as an AI agent, evaluates outputs against qualitative disclosure control principles derived from official guidelines. Its role is to reconcile deterministic classifications with contextual considerations, thereby providing a final decision together with an explanatory rationale. In practice, the agent operates as a principles-based validator that can confirm, refine, or override prior classifications depending on the circumstances.

The workflow is built upon structured prompt engineering, designed to optimise the reliability and interpretability of the principles-based evaluation. Several design strategies were employed. First, the prompts incorporated a chain-of-thought reasoning structure, requiring the model to validate the rule-of-thumb classifications before proceeding to a qualitative assessment of the contextual risks. Second, context awareness was embedded into the process by allowing conditional overrides: when a violation was flagged as unsafe under deterministic rules but the surrounding context indicated reduced risk, the AI agent could justify and record a principles-based exception. Third, few-shot prompting was used to anchor the model's behaviour. Realistic examples of flagged violations accompanied by contextual justifications provided a template for consistent reasoning. Fourth, contrastive instructions were included to reinforce boundaries by specifying what the model should not produce - for instance, narrative explanations in free text or deviations from the prescribed output schema. Finally, strict format control was applied by constraining the output to structured JSON, ensuring seamless downstream integration into the automated workflow.

Our principles-based layer uses structured prompts to align outputs with qualitative rules (e.g., sensitive attributes, combination risk). This mirrors LLM-based compliance pipelines that extract norms from unstructured policy text and apply them to structured artifacts [10]. As in that literature, we privilege explainability: decisions are returned with compact rationales and referenced risk factors.

Model selection

Three variants of large language models (LLMs) were evaluated. The first was *open-mistral-7b*, a 7B parameter dense model used as a lightweight baseline. The second was *open-mixtral-8x22b*, a mixture-of-experts model with substantially higher capacity for reasoning. The third was *mistral-medium-latest*, which was tested both in its original form and after fine-tuning on disclosure-control instructions. This setup enabled a systematic comparison across general-purpose open models, a stronger mixture-of-experts architecture, and a domain-adapted model, allowing us to assess both baseline performance and the benefits of task-specific fine-tuning.

Fine-tuning Setup for Output Checking

While pre-trained LLMs demonstrate strong general capabilities, they often struggle with highly specific tasks such as statistical disclosure control (SDC). To adapt the model to this

domain, fine-tuning was applied. As discussed in Section 2.4.3, fine-tuning allows a pre-trained LLM to specialize on task-specific data, improving performance compared to a purely general-purpose model [36].

In this study, the approach followed an *instruction-tuning* paradigm: each training example combined structured and unstructured context (e.g., the researcher's description of the requested output, results of rule-of-thumb checks, principles-based guidance, and machine learning predictions) with a gold structured answer. The target format was a strict JSON schema containing a classification label (*SAFE*, *Unsafe*, or *Needs_Info*), a rationale, agreement with the machine learning model, identified risk factors, and recommended actions. This ensured that the model not only classified but also produced transparent and auditable justifications.

The training dataset consisted of approximately 200 labelled outputs from Statistics Portugal's historical requests. Each observation included the following elements: All_context_final, rb_check, rb_checks, principles_rules_text, ML_label, ml_score, and the expert-assigned Output_checking_label. By explicitly mapping these inputs to the JSON schema, the model was guided to integrate heterogeneous information into structured assessments.

The base model selected for fine-tuning was mistral-medium-latest, chosen as a balance between performance and efficiency. The fine-tuning process optimized the model over three epochs. A supervised fine-tuning (SFT) setup was adopted, where the model directly learned from input–output pairs, combined with an instruction-tuning design to ensure adaptability to heterogeneous requests. More advanced strategies such as alignment-tuning or parameter-efficient fine-tuning were considered in the literature (see Section 2.4.3), but were not required in this study given the modest dataset size and the highly structured JSON format. In this context, SFT was sufficient to adapt the model without the overhead of reinforcement learning or preference-based optimisation.

Evaluation design and metrics

The evaluation of the large language model (LLM) for output checking followed the same principles and metrics as those used in the machine learning stage (Section 3.6.3). In particular, accuracy, precision, recall, false positive rate (FPR), Matthews Correlation Coefficient (MCC), and the area under the precision–recall curve (PR AUC) were reported, with recall and FNR receiving particular emphasis given the operational need to avoid false negatives.

To disentangle the contribution of different information sources, three complementary evaluation settings were considered. In the *rule-of-thumb only* (RB-only) condition, the agent's decision was based exclusively on deterministic rule checks. In the *machine learning only* (ML-only) condition, the agent relied solely on the predictions of the supervised model. In the *combined* (RB+ML) condition, the agent integrated both sources together with its principles-based reasoning to produce the final classification.

In addition to benchmarking performance against human-assigned ground truth, we also assessed the level of agreement between the LLM's decisions and those of the rule-of-thumb and machine learning layers individually. This comparison provides insight into the extent to which the LLM reinforces, complements, or diverges from the existing approaches. Finally, the proportion of cases in which the model returned a *Needs_Info* label was tracked, as these correspond to outputs where the system abstains from making a definitive decision and instead signals the need for human review.

This evaluation design enabled a unified comparison across rule-of-thumb, ML, and LLM components, while highlighting the specific added value of the LLM in capturing principles-

based considerations that go beyond deterministic or statistical thresholds.

3.7 Prototype Development

As a proof of concept, we developed a prototype system that replicates the full output-checking pipeline on a single request. Given an output and its corresponding request document, the prototype performs the same preprocessing steps described for the training data: metadata extraction, feature engineering, and alignment of variables and filters. The processed output is then passed sequentially through the three layers of the framework:

1. **Rule-of-thumb checks (RBOSDC)** - deterministic safeguards that flag outputs violating disclosure thresholds.
2. **Machine learning model (MLSDC)** - a trained classifier that estimates the probability of unsafe disclosure and outputs both a binary label and a probability score.
3. **LLM-based assessment (PBOSDC)** - a large language model that evaluates contextual factors and justifies the final classification.

This prototype illustrates how the system can be applied at the level of individual requests, ensuring that the same logic used for large-scale training and evaluation is directly transferable to practical use cases.

Chapter 4

Implementation and Results

This chapter describes the implementation of the proposed system, detailing the practical steps taken to build the metadata layer, standardise outputs, and integrate the different output checking components (rules, machine learning, and LLMs). Each section corresponds to a key building block of the pipeline, from metadata retrieval to feature extraction, model training, and evaluation.

4.1 Metadata retrieval and standardisation

The first step in the system's implementation was to build a consistent metadata layer that would later be used to contextualize the features extracted from the outputs and output requests. Since Statistics Portugal's metadata is dispersed across heterogeneous documents and formats, we implemented a retrieval and standardisation pipeline to consolidate this information into a single structured catalog. The following subsections describe this process step by step.

4.1.1 INE catalog PDF

Our starting point was the public document *Bases de microdados anonimizados para fins de investigação científica* [27], which lists the microdata available for research. We scanned each page, captured the current dataset label shown as SUBTEMA, and extracted all the links corresponding to the detailed metadata. We also pulled an adjacent year tag when it was present in the link text (e.g., 2005/2006, 2019, …). The output is a link catalog with columns Dataset, Year, Link Text, and Link.

From the same PDF, we extracted the dataset code, short term (the acronyms/aliases found in parentheses or after dashes), and name (the SUBTEMA line), the parent theme (TEMA), the stated periodicity (PERIOCIDADE), and the descriptive notes (TIPO DE DADOS). Finally, we also derived the available years for each dataset by parsing ranges (e.g., 1990 a 1994, 2010-2013), slash forms (2005/2006), conjunctive forms (2019 and 2020), and standalone four-digit years. The years were collected and de-duplicated per dataset. Table 4.1 displays a sample of three examples extracted.

Table 4.1: Sample of retrieved metadata with available years and details.

code	name	theme	frequency	Available years	Details
1.1	Inquérito aos Orçamentos Familiares – IOF/IDEF	Condições de Vida das Famílias	Quinquenal	1980/1981, 1989/1990, 1994/1995, 2000, ...	É um inquérito realizado por entrevista direta junto...
4.7	Recibo de Renda Eletrónico (RER)	Construção	Anual	2017, 2018, 2019, 2020, 2021, 2022, ...	Bases de Dados obtidas por via administrativa...
14.3	Índice de Preços na Produção de Produtos Industriais	Conjuntura	Mensal	2010, 2014, 2015, 2016, 2017, 2018, ...	O Índice de Preços na Produção de Produtos Industriais, tem como objetivo...

4.1.2 Retrieving each dataset's metadata

Using the harvested link table, we downloaded the referenced documentation into a structured folder tree, grouping by (Dataset, Year). We then walked the folder tree and pulled variable dictionaries using format-specific strategies. Throughout, we normalize text by lowercasing and removing accents to improve header matching, and we infer the year from the file name or parent folder when necessary (supporting ranges and single years).

Because datasets frequently evolve over time, with variables added, removed, or renamed between reference periods, we explicitly preserved the year dimension in the catalog. This ensures that outputs for a given year (e.g., 2010) are aligned with the correct set of variables available at that time, rather than with later or earlier versions of the dataset.

Since the files were in different file types (.pdf, .xlsx, .docx, .html), we developed format-specific parsers to retrieve the data of each type.

For .xlsx files, we initially attempted a direct read using the sheet's header row. If columns were unclear, we ran a dynamic header detection pass. We looked for columns whose normalised names match *variable* (e.g., variavel, name, codigo) and *description* (e.g., descricao, label, designacao).

In order to handle .pdf files (structured and irregular), since the files varied widely in structure, we used three complementary extractors. For documents that delineate sections like *Dimensões* or *Variáveis extrapoladas*, we collected the bullet points under each section as variable entries, tagging the section in description. For fixed-width tables, we parsed rows with *Sigla* and *Descritivo*, building (variable, description) pairs. Finally, for files listing variables with running numbers, we delimited the relevant region using begin_after/stop_after anchors and extracted position, variable, label by regex.

Finally, for .html files, we parse tables with *BeautifulSoup*, detected the header row, and mapped columns to *variable*, *description*, and optionally *type*/*n* via normalised keyword sets. When filenames carried a report code (e.g., report_X_YYYY), we attached the suffix to the dataset_code feature.

Although the automated pipeline covered most cases, some datasets required a manual pass. Mixed-language headers, scanned or irregular .pdf files, ambiguous year notations, and inconsistent variable headers occasionally defeated the scripts we developed. For this reason, we reviewed those items by hand and corrected the extracted records. Due to time constraints, we were not able to review the information for all datasets, but focused on those

most commonly used at Statistics Portugal.

At the end of this processing step, we had a metadata dataframe for each dataset at Statistics Portugal, containing dataset code, dataset name, year, variables, and variable descriptions, if available. Table 4.2 provides a sample of this product.

Table 4.2: Sample of variable files by dataset and year (truncated lists).

dataset_code	year	variable_description
1.1	2010	ADP_cod, pond_adp, qtd_01111, qtd_01112, qtd_01113, …
1.1	2015	ADP_cod, pond_adp, Nuts1, Nuts2, Tipau, Trim, Aloj_tipo, Aloj_PerOcup, Aloj_RegOcup, Aloj_Div_cod, …

Finally, we concatenated all extractions into a single frame, harmonized the schema, trimmed whitespace, normalised accents, and de-duplicated records, generating two final concrete products:

1. a **dataset catalog** (df_metadata) with codes, names, themes, frequency, available years, and notes;
2. a **variable catalog** (df_complete) listing (dataset, year, variables, variables description).

These two datasets provide the context needed to align outputs with their source datasets in the following steps, supporting the extraction of information from outputs and output requests, as well as the rule-of-thumb, machine learning, and LLM-based checks.

4.2 Historical data retrieval

To evaluate the system, design prompts, and train models, we required a historical dataset that combined (i) the *output requests* submitted by researchers; (ii) the corresponding *outputs* they produced; and (iii) INE's final classification of those outputs (approved/blocked). Constructing this dataset was not straightforward due to several challenges:

- **Heterogeneous folder structures.** Requests and outputs were dispersed across researcher folders with day-level subfolders; structures were inconsistent, and some material was nested inside .zip archives.
- **Multiple file formats.** Outputs were stored as .csv, .xlsx, and .ods, but also as mixed code–output files such as .log, .do, and .txt.
- **Inconsistent output layouts.** Even within the same output type, presentation varied. For example, a frequency table might include a clearly labeled "Frequency" column, but in many cases, the same information appeared under different names, complicating automated extraction.
- **Incomplete or inconsistent labels.** The administrative classification (approved/blocked) was sometimes missing, ambiguous, or incorrect.

To address these challenges, we developed Python scripts to (i) crawl and index files; (ii) pair output requests with outputs; (iii) extract and standardise output request documents; and (iv) extract and standardise outputs. These steps are described in the following sections.

4.2.1 Collecting file paths and pairing outputs with requests

In this step we scanned the Statistics Portugal archives, distinguishing *output requests* from *outputs*, assigning identifiers that also work inside .zip archives, and pairing each output with the closest request by directory proximity.

We traversed the full tree under R:\lsb\SafeCenter\2025, inspecting every folder and each .zip archive encountered, including nested zips up to a fixed maximum depth (to prevent infinite recursion).

To distinguish output request files from outputs, we applied a simple regex that captures the forms used in practice (e.g., "lista de preenchimento", "safe center/safecenter/safe centre", case-insensitive). Files matching this pattern were classified as *output requests*; all others were considered candidate *outputs*.

To refer unambiguously to files, we generated two identifiers: output_source_id, containing the path to each output file, and request_source_id, containing the corresponding output request path. Pairing was not based on filename similarity. Instead, for each output we walked upwards through its directory ancestors and selected the first ancestor directory containing a request. This reflects how teams typically organize their work (requests and outputs stored together or one level apart) and is robust to naming variation.

Additional fields such as city, researcher, and date were derived from the directory structure when present.

This step produced a unified file index, df_ext, which contains the paths and pairings of requests and outputs. In the following sections, this index is used to locate files, drive feature extraction (variables, filters, groupings), enrich records with metadata, and support both rule-of-thumb checks and principles-based assessments.

4.2.2 Extracting output request documents

Here we described how we turned each request workbook (Excel) into tidy rows. In short, we loaded every sheet, located the header row, cleaned and standardized the columns, extracted the small metadata block at the top, attached lineage (file/sheet/path), and appended the result to a single table. This was how we built df_combined.

Since most output request documents were .xlsx files, we opened them with openpyxl and fell back to xlrd for legacy .xls. All the documents had a similar structure with the small metadata block at the top and then a table with features such as *Nº de ordem do resultado*, *Variáveis utilizadas*, *Regras de análise (filtros)*, *Tipo de resultado (tabela; gráfico; correlação; modelo de regressão;...)* and an observation for each output described.

In order to detect the headers of the table, and since the layout varied, we did not assume a fixed row. Instead, we normalised strings (lowercase, stripped accents, collapsed spaces and variants like *n.º/nº/n.o* to *no*), searched for a header cell matching *no de ordem do resultado* and if not found, we used a small numeric heuristic: looked for a *1,2* sequence in a column and took the header to be the row above. If we still could not identify a header, we logged a warning and skipped that sheet. Once we knew the header row, we took that row as the set of column names and dropped fully empty rows/columns (keeping the key column intact).

Then, to extract the information in a block of labels at the top, we read the first 14 rows and, for each label, took the value to its right: *investigator* (*Nome do investigador:*), *project title* (*Título do projeto de investigação:*), *date* (*data:*), *data source* (*Nome da(s) fonte(s) de dados:*), and *brief description* (*Breve descrição dos resultados produzidos:*). We attached these as new columns to every row from

that sheet. Furthermore, we also added the sheet_name, source_file, and request_source_id to keep a clear trail back to the original file. Table 4.4 provided one observation of one output request.

Table 4.4: Example of output request with the extracted features *dataset_name*, *sheet_name*, *dataset_code*, *Variáveis utilizadas*, *Regras de análise (filtros)*, and *Tipo de resultado*.

dataset_name	sheet_name	All_dataset_codes	Variáveis utilizadas	Regras de análise (filtros)	Tipo de resultado
Quadros de Pessoal 2013–2022	Outras BDs	Quadros de Pessoal 2013–2022. Outras BDs	ANO; NUT3_est; CAE5_est	Cálculo do número de estabelecimentos em cada combinação ANO/NUT3_est para cada uma das três CAEs seguintes: 87100, 87301 e 88101.	tabela

Finally, we concatenated the extracted content from all request files into a single dataset, df_combined. Together with df_ext (the request–output index), this dataset formed the basis for subsequent processing. Specifically, it allowed us to (i) retrieve and align the corresponding outputs; (ii) standardise and enrich the requests with variables, filters, and grouping information; and (iii) join them with official dataset metadata. These combined resources then fed into both the deterministic rule-of-thumb checks and the principles-based assessments described in the following sections.

4.2.3 Extracting outputs

In this step, we focused on the extraction of output files, transforming each path from the pairing index into a standardized format with enough lineage to trace its original source. The goal was to turn the diverse and often irregular raw outputs - spreadsheets, CSV/TSV files, and Stata .do/.log/.txt scripts - into a common, tidy structure suitable for subsequent output checking.

For spreadsheets and delimited files (.xlsx, .xls, .ods, .csv, .tsv), we first probed sheets or sampled rows to avoid expensive full reads, skipping hidden or excessively large ones. Extracting the relevant information was not always straightforward, since there was no guarantee that column names or row layouts followed a consistent pattern. In frequency tables, for example, the relevant counts could appear under columns named *Frequency*, *Freq.*, or sometimes under vague numeric headers, while percentage or share columns had to be ignored. We therefore relied on controlled dictionaries of expected terms and light heuristics to map columns and rows into a stable schema.

A substantial proportion of outputs were not spreadsheets but Stata logs or scripts, which usually contain several outputs in a single file. Initially, we avoided these due to their complexity, but later we developed a parser that could be generalized across files, since most logs follow a regular structure. We parsed the files as text streams and segmented them into blocks that resembled tables, triggered by common commands such as sum, tab/tab1, table, correlate, ttest, or reg. For one-way tabulations, we extracted frequencies, percentages, and cumulative percentages, while two-way tables were reconstructed into long format with category_a, category_b, and Freq.. For sum outputs we mapped the summary statistics (Obs, Mean, Std. Dev., Min, Max) into structured rows, and for sum, detail we additionally recovered percentiles and the *Smallest*/*Largest* values. Regression diagnostics were parsed into

stable keys (e.g., underid_f, weakid_sw_chi_sq), and we also tracked contextual information such as dataset usage commands (use, append), filtering operations (keep if, drop if), and blocks bracketed by preserve/restore. Each record was assigned a output_id, corresponding to the order in which they appear in the file, and an output_type drawn from a controlled vocabulary (e.g., *Frequency tables*, *Percentiles*, *Regression diagnostics*) and enriched with counts when available.

Because both spreadsheets and logs often presented results in a wide form, an important step was to normalize them into a long format, where each row represents a single observation with explicit variable and category labels. This reshaping ensures consistency across outputs and facilitates downstream feature extraction. An example of this transformation is shown in Figure 4.1.

Magnitude table (wide): Mean income

	Gender		
Region	Female	Male	Other
North	21,300	22,700	20,900
South	18,900	19,400	18,100

⇓ melt to long

Magnitude table (long)

variable_A	category_A	variable_B	category_B	target_variable	value
Region	North	Gender	Female	income_mean	21300
Region	North	Gender	Male	income_mean	22700
Region	North	Gender	Other	income_mean	20900
Region	South	Gender	Female	income_mean	18900
Region	South	Gender	Male	income_mean	19400
Region	South	Gender	Other	income_mean	18100

Figure 4.1: Transformation of a magnitude table from wide to long format.

Outputs stored inside .zip archives were opened in memory (or via temporary files for logs) and parsed in the same way, with lineage extended to capture both the outer archive and inner member path. After parsing, all rows were merged back with their index entries from df_ext, ensuring that every output carried a unique identifier and full provenance. Even when a file did not produce a table, we still produced a placeholder record labeled *No data*, so that the downstream pipeline could process all outputs uniformly without silent omissions.

In the end, this step returned a table with output IDs, the normalised outputs, and complete lineage across file types. While the parser successfully covered the most common table-like structures, highly irregular layouts remain a limitation, and further refinements should be done. Nevertheless, the resulting dataset provides a reliable foundation for the rule-of-thumb and principles-based disclosure checks described in the following sections.

Summary of the extracted outputs

The groups of outputs we extracted was overwhelmingly composed of outputs that were cleared for release by Statistics Portugal. Of the items for which the institutional final label could be recovered, 48,273 were labelled *Aprovado* or *SAFE* and 1,646 *Reprovado* or *UNSAFE*, implying a rejection rate of roughly 3.3% and an approval rate near 96.7%. This strong predominance of approved outputs is consistent with day-to-day practice in output checking, where most requests fall into routine patterns that can be released once basic safeguards are verified. It also underscores the intrinsic class imbalance that any predictive component must confront, with direct implications for the choice of loss functions, evaluation metrics, and sampling strategies used later in the machine-learning experiments.

With respect to output form, three classes dominate the archive: *Summary and test statistics* (20,904 outputs), *Frequency tables* (12,460), and *Means, indices, ratios, indicators* (10,820). More specialised categories such as *Magnitude tables* (668) and *Linear regression coefficients* (257) appear far less frequently, while maxima/minima/percentiles (18) and a small number of *Unknown* cases (10) are rare. In practical terms, this composition suggests that simple tabular and summary outputs remain the modal unit of disclosure review, and it motivates the emphasis placed in the rule-of-thumb layer on count and concentration checks for tables, as well as degrees-of-freedom safeguards for model-like outputs.

4.3 Feature Extraction/Engineering

The extraction step (Section 4.2.3) yielded a unified table of outputs in long format, with stable identifiers and lineage. However, many disclosure checks cannot be applied directly to the raw extracted values. Instead, they require secondary feature computation, either from the structure of the output itself (e.g., row/column proportions), from reported model statistics (e.g., degrees of freedom), or by linking to external sources such as metadata and output requests. This subsection describes how these features were computed and standardized, organized by information source and mapped to the rules in Section 3.3.

4.3.1 Output Features

Once outputs are normalised into long format, several rule-of-thumb features can be derived. Some of these are directly available in the outputs themselves, while others require additional computation.

The most immediate feature is the **cell count**, which corresponds to the unweighted size of each cell in a frequency table or, in the case of summary or model outputs, the reported number of observations. Another directly extractable feature is the presence of **perfect or null correlations**, which can be flagged whenever a correlation coefficient takes the values -1, 0, or $+1$. In regression outputs, it is also possible to detect whether an **intercept or other coefficient is present**, thereby verifying the rule that at least one coefficient must be withheld. For multivariate methods, the **variable count** is obtained by simply recording the number of distinct variables included in a factor or correspondence analysis.

Other features must be computed rather than read directly. For example, **row and column proportions** are obtained by calculating, within each partition, the share of each cell in the corresponding total and flagging cases where that share exceeds 90%. Similarly, when regression models report the number of observations and estimated parameters, the **degrees**

of freedom can be inferred as $df = n_{\text{obs}} - n_{\text{params}} -$ constraints. whenever the statistic is not explicitly provided. Finally, in the case of magnitude tables (means or totals), group totals can be computed directly, and, if the relevant values are printed, largest-contributor shares can also be derived; otherwise, these must be obtained from the underlying dataset, as described later.

4.3.2 Output Request Features

Extracting the relevant features from output requests is one step that can highly enrich the output for disclosure control by capturing the scope and logic of the analysis. However, as explained in Section 3.5.2, most output requests have missing information, organized in an unstructured way or have misspellings. It is, therefore, a complex process. In this section, we explain how to transform unstructured researcher-provided metadata into clean, structured features suitable for statistical disclosure control (SDC), a multi-stage AI-assisted pipeline was implemented.

As a first step, and because researchers often entered information into the wrong fields, we generated a composite feature named All_context. This field aggregates the content of several request columns *project_title*, *brief_description*, *variables used*, *analysis rules (filters)*, and *result type (table; chart; correlation; regression model; …)*. By consolidating all available textual context into a single input, we ensured that no relevant detail was discarded at the outset of processing.

The pipeline relies extensively on large language models to interpret this aggregated context. We used the *open-mixtral-8x22b* model for most extraction and validation tasks, typically running at a moderate temperature (0.7) for feature generation to allow controlled variability, and at zero temperature for validation steps to ensure determinism. To optimise performance and reliability, results of repeated calls were cached so that identical inputs did not require multiple API requests. In addition, we implemented a backoff mechanism to handle API rate limits and transient failures, together with a minimal-shape guard that guaranteed even failed rows returned a structured placeholder JSON object marked with row_failed. This design ensured robustness, prevented silent data loss, and allowed all downstream modules to operate uniformly on a consistent schema.

Output Type Detection

The first feature extracted from each output is its intended type. Identifying the correct output type is crucial because it determines which disclosure control rules and principles apply in later stages. For this task, we implemented an AI agent based on the *open-mixtral-8x22b* model, configured with a temperature of 0.7 and a maximum token limit of 200. The choice of a temperature slightly below 1 was deliberate: it introduces controlled variability at this initial stage, which helps the model accommodate the diverse ways in which researchers describe outputs, while still ensuring stable mapping to a fixed taxonomy.

The taxonomy itself was predefined and fixed, consisting of the following categories: *Frequency tables*, *Magnitude tables*, *maxima, minima, percentiles*, *Modes*, *Means, indices, ratios, indicators*, *Concentration ratios*, *Higher moments of distributions*, *Graphs (descriptive statistics or fitted values)*, *Linear regression coefficients*, *Non-linear regression coefficients*, *Estimation residuals*, *Summary and test statistics*, *Factor analysis*, *Correlation coefficients*, and *Correspondence analysis*. By constraining the system to this label set, we ensured consistency and avoided wrong or overly specific categories that would complicate downstream processing.

Prompt design played a key role in achieving reliable classification. The model was guided through a single-turn structured format that integrated several design choices. First, the task description explicitly listed the allowed label set and the mapping rules that had to be respected. Second, multilingual support was embedded directly in the instructions, enabling the model to handle both Portuguese and English examples, as these languages often appear interchangeably in output requests. Third, a few-shot prompt provided real examples drawn from real-world requests, paired with their correct labels, thus anchoring the model's behaviour. In addition, contextual prompting embedded domain-specific cues, such as aggregation levels, filters, and statistical measures, which increased robustness in ambiguous cases. Finally, negative prompting was used to explicitly instruct the model to ignore repetitive or irrelevant text and to return only valid labels.

The raw model predictions were subsequently processed by a two-stage validation pipeline. The first stage consisted of deterministic, rule-of-thumb cleaning and canonicalisation. Here, pre-cleaning removed prefixes and harmonized whitespace, while accent-insensitive normalization ensured that both Portuguese and English references could be matched. A bilingual dictionary then mapped common surface forms directly to canonical categories; for instance, *max/min/quartis* were mapped to *maxima, minima, percentiles*, and *indices/racios/indicadores* were mapped to *Means, indices, ratios, indicators*. Pattern rules captured broader method families across languages, for example, mapping any regress* form to *Linear regression coefficients*, poisson|logit|probit|glm to *Non-linear regression coefficients*, and correla* to *Correlation coefficients*. In parallel, verbatim detection identified any full canonical labels present in the text, while fuzzy matching with RapidFuzz (token_sort_ratio, thresholds $\tau_\ell = 90$ for labels and $\tau_k = 88$ for keys) allowed recovery of labels even when phrasing diverged. Candidates were then resolved through a priority hierarchy (exact match $\rightarrow$ dictionary mapping $\rightarrow$ fuzzy match $\rightarrow$ relaxed match), before being de-duplicated and filtered to retain only valid categories.

The optional second stage introduced LLM-based validation. In this step, the cleaned candidates, together with the raw text, were re-evaluated using a deterministic run of *open-mixtral-8x22b* (temperature = 0). This validator, constrained to the fixed label set, enforced three requirements: (i) preserve only allowed labels (including mapped synonyms); (ii) flag out-of-set or ambiguous items; and (iii) return a structured JSON with the keys valid, labels, and errors. Table 4.5 illustrates five examples of extracted output types, shown alongside the original features *Variáveis utilizadas*, *Regras de análise (filtros)*, and *Tipo de resultado*.

Table 4.5: Example of extracted output metadata with variables, analysis rules, result type, and validator classification.

Variáveis utilizadas	**Regras de análise (filtros)**	**Tipo de resultado**	**Extracted output type**
numberofALbyquarter; numberofALbyquarter-after4	Regressão	log out file	Means, indices, ratios, indicators
postcode, quarter, numberoftransactions, newbuilt...	Código Stata	Stata do file	Frequency tables; Means, indices, ratios, indicators
IMI & IMT: coddestino, ano	keep if coddestino==7	Gráfico de linhas	Graphs (descriptive statistics or fitted values)

By combining structured prompting, deterministic cleaning, and optional LLM-based validation, the system achieved a robust mechanism for classifying outputs into a controlled taxonomy. This hybrid approach proved particularly effective in accommodating the noisy and inconsistent phrasing found in real requests, while maintaining the reliability required for an operational disclosure control workflow.

Metadata

To continue the analysis of the information contained in output requests, it was necessary to link these documents to the corresponding metadata. This step was essential because it provided additional context to the outputs, as well as the basis for validating the researchers' descriptions against the official dataset documentation, allowing the system to identify variables used, filters applied, and types of aggregation performed with greater precision.

The task was complicated by the fact that researchers did not always reference datasets in a consistent way. Some requests referred to datasets using their full names, either in Portuguese or English, others relied on short terms or formal dataset codes (e.g., 1.8, 4.4), and still others included informal abbreviations. Furhtermore, some documents reported more than one used dataset. To accommodate this variety, we developed a multi-stage process that combined direct extraction, text cleaning, and fuzzy matching, ensuring that the relevant datasets could be identified with high recall.

The first stage involved the direct extraction of explicit identifiers. Regular expressions were used to capture dataset codes in the n.n format, while a dictionary of known short terms (compiled from the metadata catalog) was used to detect references in free text. When such a match was found, the associated dataset codes were retrieved immediately.

Once these explicit identifiers were removed, the remaining text was cleaned to reduce noise in subsequent steps. This involved stripping out date-like expressions (e.g., 09-2005, 3-2019), generic placeholders such as *Safe Center* or *Outras bases de dados*, and various formatting inconsistencies such as underscores, duplicated phrases, or extraneous punctuation. The result was a simplified representation of the dataset name, better suited for comparison with the official registry.

The cleaned text was then compared against the metadata catalog using fuzzy matching. Both the request text and the catalog entries were normalised by lowercasing and removing diacritics before being split into smaller components at common delimiters (commas, semicolons, periods). Each component was compared to the metadata dataset names using RapidFuzz and the token_sort_ratio scorer. Matches above a similarity threshold of 70 were retained as candidate dataset codes.

Finally, the codes obtained from regex extraction, dictionary look-ups, and fuzzy matching were combined into a unified list of unique dataset references for each request. This allowed the system to recover the correct dataset(s) even when references were incomplete, abbreviated, or expressed in multiple languages.

Once the data set(s) were identified, the corresponding metadata were retrieved. This metadata included not only variable names and descriptions but also information on dataset size and structure, sample design, available geographic granularity, temporal coverage, and any sensitivity labels. Together, these contextual details provided the foundation for subsequent feature extraction and were indispensable for both the rule-of-thumb and principles-based disclosure checks.

Variable Extraction and Metadata Matching

A central task in the pipeline is the extraction and standardisation of variables referenced in output requests. This process begins with a *pre-metadata stage*, where candidate variables are identified directly from the researcher's free-text descriptions, without reference to the official metadata repository. The rationale for this design choice is to maximise recall: researchers often employ their own terminology, abbreviations, or bilingual expressions that may not appear in the structured metadata. By capturing these surface forms early, the system preserves valuable semantic cues that can later be reconciled to canonical variable names.

The pre-metadata stage combines deterministic and LLM-based techniques in four steps. First, deterministic candidate mining applies regular expressions and lexical heuristics to identify tokens or n-grams that resemble variable names, while filtering out false positives such as method names or years. Second, the cleaned text is passed to a large language model, which is instructed to output only a list of variable candidates, thereby capturing implied variables that do not follow strict naming conventions. This LLM is configured with a moderately high temperature to encourage broader coverage and capture a wider spectrum of variable candidates. Third, these LLM results are refined deterministically by restoring formatting (e.g., underscores, singular/plural forms) and applying fuzzy matching to stabilise naming. Finally, the union of all candidates is validated by a second LLM prompt, with a lower temperature to minimise randomness, constrained to return structured JSON. This validator enforces de-duplication, corrects near-duplicates, and flags errors, yielding both a high-recall union view and a higher-precision intersection view (or majority-vote compromise) for downstream use.

Once a candidate list is available, the system proceeds to cross-reference it against Statistics Portugal's metadata repository. This step enriches each variable with type information (categorical, continuous, binary, date), coding schemes and measurement levels (e.g., NUTS 2 versus NUTS 3, quarterly versus annual), sensitivity classifications (for example, income marked as high risk), and provenance notes such as whether a variable is derived, imputed, or drawn from administrative records. Incorporating this metadata can highly enrich the whole process because it directly informs disclosure control decisions: sensitive variables trigger stricter thresholds, binary variables require special handling in regressions and correlations, and fine-grained geographic or temporal references increase re-identification risk.

In cases where structured metadata is incomplete or unavailable, a fallback strategy is applied. Here, a lightweight LLM prompt is used to infer likely variable type (quantitative, categorical, binary) and potential sensitivity based on a curated sensitive-variable list. The prompt is strictly structured to guarantee parseable output and supported by few-shot examples to improve consistency. For instance, given the input Rendimento global, Coleta líquida from the IRS dataset, the system returns structured classifications identifying both variables as quantitative. Although inherently probabilistic and less reliable than structured metadata, this fallback ensures broader automation coverage and enables preliminary disclosure checks even for under-documented datasets.

Table 4.6 displays a sample of four examples of the extracted variables following this procedure along with the original features *Variáveis utilizadas* and *Regras de análise (filtros)*.

Together, this layered approach - pre-metadata extraction, metadata cross-referencing, and LLM-based fallback - provides both breadth and depth. It allows the system to capture researcher-specific terminology while aligning it with official metadata, enrich variables with sensitivity and structural attributes, and maintain resilience when documentation gaps exist.

Table 4.6: Examples of variable extraction: original variables, applied filters, and LLM-normalised candidates.

Variáveis utilizadas	Regras de análise (filtros)	Extracted variables
VAB e Volume de negócios. Proveniente do SCIE e Ficheiros de Empresa	–	[vab, volume_de_negocios]
Pessoal, volume de negócio, Subsidios à exploração	Periodo temporal: 2004–2010	[pessoal, volume_de_negocio, subsidios_a_exploracao]
Número de trabalhadores, rbase, prest_reg, hnormais, idade, género	Cálculo de apuramentos pela célula definida por ANO, género e idade	[ano, rbase, prest_reg, hnormais, idade, genero]
ANO, NL_RENDIMENTO_GLOBAL, NL_COLETA_LIQUIDA	Período temporal: 2021; Exceto observações com valores nulos	[ano, nl_rendimento_global, nl_coleta_liquida]

These steps ensure that the subsequent disclosure control models operate on a consistent and context-aware representation of variables.

To reduce noise and improve interpretability, extracted variables were grouped by thematic similarity or shared root terms. For example, all variants of rendimento (rendimento_bruto, rendimento_liquido, rendimento_das_familias, etc.) were collapsed into a single category rendimento. A similar approach was applied to housing-related variables (e.g., proprietario, residencia, casa_aluguel) and employment terms (tipo_de_emprego, independente). This grouping ensured that conceptually equivalent or highly related variables were not treated as independent signals.

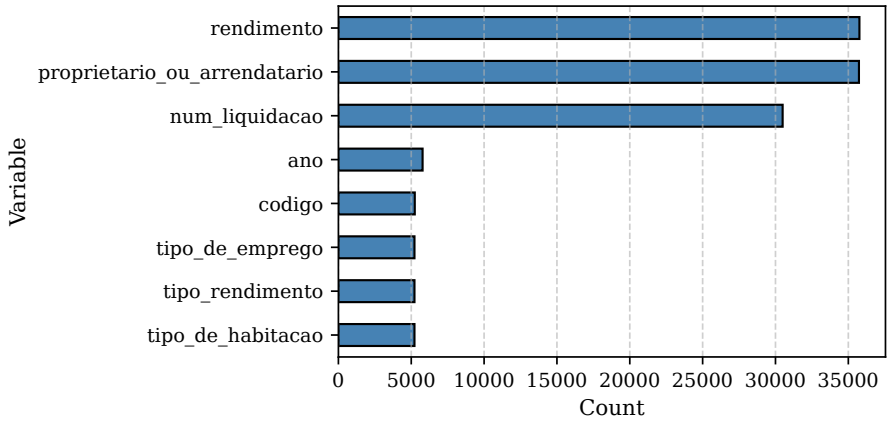

Figure 4.2: Top 8 most frequent variables extracted from researcher requests after normalisation and grouping. Related expressions (e.g., *rendimento_bruto*, *rendimento_liquido*, *rendimento_das_familias*) were collapsed into the generic category *rendimento*.

Figure 4.2 displays the eight most frequent variables after grouping. The prominence of rendimento and housing status (proprietario_ou_arrendatario) underscores their centrality in disclosure risk evaluation, while other categories such as ano, num_liquidacao, and tipo_de_emprego also emerge as recurrently referenced variables. Before grouping, these variables were fragmented across dozens of surface forms, but the normalisation and consolidation process increased clarity and robustness for downstream modelling.

After extracting variables, a similar pipeline was applied to retrieve both aggregation variables and sensitive variables from the output request documents. As with variable extraction, this process combined deterministic rules with LLM-based interpretation and validation. The extracted items were subsequently grouped by thematic similarity, resulting in a more structured and interpretable representation.

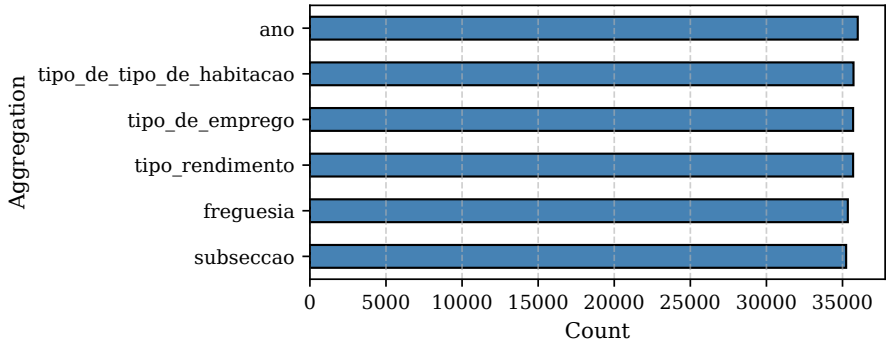

Figure 4.3: Top 6 aggregation variables extracted from output requests.

Figure 4.3 shows the most frequent aggregation variables identified in the requests. The dominance of structural variables such as *ano*, *freguesia*, and *tipo_de_habitacao* highlights the strong role of temporal, geographic, and housing-related dimensions in shaping output requests.

In turn, Figure 4.4 presents the most common sensitive variables detected by the system. Although fewer in number compared to aggregation variables, these are can be relevant for disclosure safety assessment. The prominence of *rendimento* and related income variables underscores their central role in the statistical outputs most prone to confidentiality concerns.

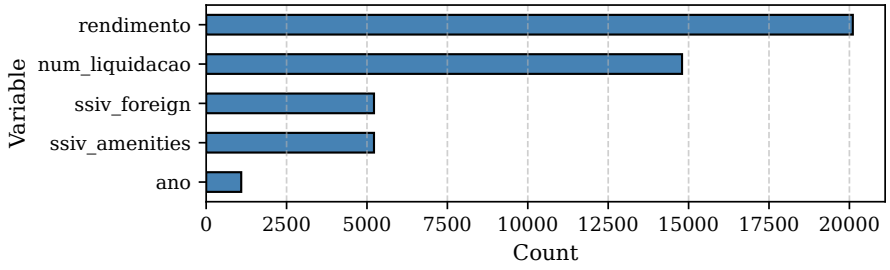

Figure 4.4: Top 5 sensitive variables extracted from output requests.

Filter Extraction

The extraction of analytical filters relied on the free-text column *Regras de análise (filtros)*. In practice, this field frequently encodes implicit or explicit restrictions such as inclusion or exclusion conditions, temporal limits, geographic scopes, population qualifiers, binary flags, or even reported sample sizes. To capture this information in a structured and reproducible way, we implemented a hybrid extraction and validation pipeline.

The first stage of the pipeline applies deterministic parsing through a function that normalises the text by removing accents and harmonising case, after which a series of regular

expressions are applied to detect common filter structures. The output is a structured dictionary with explicit keys for include, exclude, time, geography, population, n_observations, and notes. This deterministic layer provides high reproducibility, but may miss more complex or less standardised phrasing.

To complement this, the same text is also processed by an LLM-based extractor, which queries the *open-mixtral-8x22b* model with temperature fixed at 0 to ensure determinism. The prompt is strictly constrained to produce JSON conforming to a predefined schema. This stage enables the capture of richer or less conventional filter descriptions, including those expressed in natural language or embedded within compound phrases, that are not easily detected by rule-of-thumb methods.

The outputs of both extraction strategies are then reconciled using a function that produces a unified filter structure that combines deterministic and LLM-derived elements. Duplicates are removed, and a record of *support* is retained for each rule, indicating whether it was identified by the deterministic parser, the LLM, or both. For scalar fields such as time bounds or sample sizes, deterministic values are given precedence where available, while list-based fields such as geography or population are merged across both extractors.

Finally, the merged structure is passed through a validation and pruning function. At this stage, the original free-text description is reintroduced alongside the merged filters, and the LLM flags items that do not correspond to genuine dataset restrictions. For example, statistical concepts such as *quantile* or *median* which sometimes appear in the result-type field, are identified as non-filters. The validator also detects logical inconsistencies, such as the same variable appearing in both inclusion and exclusion lists. Items flagged as spurious or contradictory are annotated accordingly, and a pruned version of the filter set is produced for downstream processing. Table 4.7 provides a sample of the extracted filters.

Table 4.7: Examples of extracted variables, filters, and validated filter text.

Variáveis utilizadas	**Regras de análise (filtros)**	**filters_llm_text_validated**
Todas	análise do rendimento das famílias nas freguesias...	rendimento_familiar > 18000, reformado = true
Ano, Concelho Fiscal, Freguesia Fiscal, NIF-A	Análise da morada dos declarantes na freguesia da Misericórdia (Lisboa) antes de 2019...	freguesia = “misericordia (lisboa)”, ano < 2019, freguesia = “nao classificado”
Todas	Estatísticas descritivas	no filters

By combining deterministic extraction with LLM interpretation and validation, this pipeline balances reproducibility with flexibility. The deterministic stage ensures that standard patterns are consistently captured, the LLM broadens coverage to encompass more varied phrasings, and the validation layer reduces false positives and inconsistencies. Finally, a process similar to the one explained in Section 4.3.2 was applied by grouping the filters by thematic similarity. The result is a comprehensive yet cleaner representation of the analytical restrictions associated with each output request, with explicit traceability back to both extraction and validation sources.

Final Consistency Validation

The last stage of the output request feature pipeline introduces a global consistency validator. While earlier modules extract specific components such as variables, filters, and group terms independently, this step reconciles them holistically against the original context of the request. The goal is to ensure that the final structured representation is both internally coherent and fully supported by the researcher's description.

The validator is implemented as a hybrid LLM-deterministic procedure. A structured prompt is built from the aggregated All_context field together with the intermediate feature lists: variables, sensitive variables (categorical and numerical), filters, and grouping terms. The prompt enforces strict anti-hallucination rules, instructing the model not to introduce new content, to exclude unsupported items, and to return a machine-parseable JSON object wrapped in <json> tags. To guarantee stability, the model (*open-mixtral-8x22b*) is run at zero temperature with a maximum token limit of 600.

The JSON schema requires the validator to return three blocks: (i) the final section, which contains the cleaned and normalised lists of variables, sensitive variables, filters, and groups; (ii) an issues list, recording mismatches, unsupported items, or structural problems detected; and (iii) a notes field, where the validator provides brief reasoning or flags ambiguity. All tokens are normalised to lowercase, accents are removed where possible, and multi-word phrases are collapsed into short surface forms directly evidenced in the context.

By performing this final consistency validation, the system produces a reconciled feature set with both high recall and high precision. It ensures that each variable, filter, and grouping term included in the structured representation is supported by the researcher's request, that contradictory or spurious items are excluded, and that all outputs conform to a strict schema suitable for automated disclosure risk assessment.

Split outputs

At the end of this process, in some cases, a single record refers to multiple output types, implying that it actually represents more than one distinct output. To ensure that each output can be independently processed during the output-checking workflow, the record is split into multiple entries, each corresponding to a single output type specified in the original request.

This separation is necessary because certain features required for disclosure safety classification depend directly on the output type. By isolating outputs in this way, we ensure that each record can be associated with the appropriate set of disclosure rules and principles. In practice, this step serves as the final harmonisation stage in the processing of output requests, enabling consistent and accurate feature extraction in the subsequent phases of the system.

4.4 Rule-of-thumb output checking

The first automated layer of disclosure control is based on a set of deterministic *rule-of-thumb* checks. This component operates on the normalised outputs described in Section 4.2.3, and produces an initial classification of each output into *SAFE*, *SAFE*, or *NO_DATA*. The approach is intentionally conservative, favouring the identification of potential risks, and is designed to provide a transparent baseline that can later be refined by principles-based assessments.

The procedure can be divided into three stages. In the pre-computation and harmonisation stage, raw values are standardised to create a consistent basis for rule application. Common count fields such as Obs, Freq., and N are coerced into a unified N_Observations column, ensuring comparability across outputs. For frequency and magnitude tables, cell shares within rows and columns are computed, with the maximum share stored as an indicator of dominance. For regression-like outputs, lightweight model metadata are extracted, including the number of observations and regressors, residual degrees of freedom (when reported), and whether at least one coefficient appears withheld. Additional heuristics capture whether the coefficients are entirely categorical and provide a rough count of variables represented in the output block.

The second stage involves applying the disclosure checks themselves, tailored to the detected output type. For tabular statistics, a cell-count rule is applied, classifying outputs as *SAFE* when cell counts fall between one and nine. A dominance check flags cross-tabulations in which a single cell exceeds 90% of its row or column. For model-based outputs, rules are applied to residual degrees of freedom, requiring a minimum threshold of ten, and to the presence of withheld coefficients, which must be observable for the output to be considered safe. Additional checks include warnings for outputs composed entirely of categorical variables, a minimum-variable requirement for multivariate methods such as factor or correspondence analysis, and diagnostic tagging of maxima and minima.

These checks are combined per output type, such that, for example, frequency tables are subject to both the cell-count and dominance rules, while regression coefficients are subject to a broader set of model-based checks. After each output is checked according to all the rules as described in Section 3.3, if all the checks returned safe, the output is labeled as safe. Otherwise, it is labeled as unsafe.

Finally, in the aggregation and reporting stage, row-level results are collapsed to produce a single rule-of-thumb label for each output_id. A conservative ordering ensures that any unsafe finding dominates: *SAFE* takes precedence over *SAFE*, which in turn takes precedence over *NO_DATA*. Alongside the label, a compact dictionary is generated that records the percentage of rows violating each rule, together with contextual diagnostics such as the smallest positive cell count or the distribution of maxima/minima tags. A canonical output type is also assigned to each output, based on the most frequently detected type within the block.

Edge cases are handled explicitly. When no parsable data are available, a placeholder record is produced with Rule_based = NO_DATA. In cases where multiple inferred sub-outputs overlap, the most conservative label is retained. Some checks, such as dominance in magnitude tables, are currently implemented as placeholders and return Not evaluated until parsing is extended. The overall output of the RBOSDC module is therefore a structured and explainable table with one row per output_id, containing the assigned rule-of-thumb label and the summary of violated rules. This table serves both as a baseline assessment and as structured input to subsequent principles-based and machine learning evaluations.

Applying thesis rule-of-thumb check to the extracted outputs gave us the distribution shown in Table 4.8. A large proportion of the material consisted of *Summary and test statistics*, which were almost always labelled as *SAFE*. This outcome is consistent with the fact that such outputs systematically satisfied the two key rules: a sufficient number of observations and positive degrees of freedom. Since these checks are deterministically enforced, this category did not provide meaningful variation for further modelling.

Table 4.8: Distribution of outputs by type and rule-of-thumb classification.

output_type	No data	SAFE	UNSAFE
Frequency tables	24	9062	3374
Linear regression coefficients	250	7	0
Magnitude tables	668	0	0
Means, indices, ratios, indicators	59	10694	67
Summary and test statistics	13976	6928	0
Unknown	10	0	0
Maxima, minima, percentiles	12	6	0

Outputs classified as *Means, indices, ratios, and indicators* presented a different challenge. While this category contained both *SAFE* and *SAFE* cases, it was highly imbalanced: fewer than 1% of outputs were unsafe. This imbalance limited the potential for robust machine learning experiments, as the models tended to converge on trivial classifiers dominated by the majority class.

By contrast, *Frequency tables* displayed substantial variation across all three categories (*SAFE*, *SAFE*, and No data), making them the most informative setting in which to test rule-of-thumb and predictive approaches. They also represent one of the most common and disclosure-sensitive types of outputs in official statistics. For this reason, subsequent analysis concentrates primarily on frequency tables, using them as the reference case for exploring both rule-of-thumb and principles-based output checking.

4.5 Output Classification labelling

To train and evaluate the ML- and LLM-based approaches, we required a per-output target indicating whether an output is safe for release. At Statistics Portugal, however, approval is issued at the file/folder level: a submission is either approved or rejected as a whole. Importantly, it is *not* possible for a finally-unsafe output to appear in an approved (safe) folder: if at least one output is finally unsafe, the entire folder is labelled *SAFE*. Conversely, some individually safe outputs can appear inside a folder labelled *SAFE*, because the institutional decision is made for the whole submission.

This folder-level labelling creates ambiguity when we wish to assign labels to *individual* outputs. To mitigate this, we combined the institutional folder label with the *rule-of-thumb* (RBOSDC) check from Section 4.4 and filtered cases where the folder-level label is a poor proxy for the output-level truth. Concretely, we applied the following inclusion/exclusion rules seen in 4.9.

The rationale is as follows. If a folder is *SAFE*, all contained outputs are finally safe. We therefore keep both the "easy" cases (also safe by rules) and the "override" cases (flagged unsafe by rules but deemed safe after principles-based review). If a folder is *SAFE*, at least one output is finally unsafe, but not necessarily all of them. To avoid falsely labelling safe outputs as unsafe, we *drop* outputs that are safe by the rule-of-thumb yet inherit the folder's unsafe label. By contrast, outputs that are unsafe by the rule-of-thumb are retained and labelled *SAFE*, representing genuinely disclosive (or borderline) content under deterministic

Table 4.9: Inclusion and exclusion rules for reconciling folder-level labels (INE) with rule-of-thumb (RBOSDC) checks.

Folder label (INE)	Rule-of-thumb	Action	Notes
SAFE	SAFE	Keep	Safe under both checks
SAFE	UNSAFE	Keep	Principles-based override scenario
UNSAFE	UNSAFE	Keep	Deterministically unsafe
UNSAFE	SAFE	*Drop*	Likely safe output inside an unsafe folder

rules.

This curation has three main effects. First, it reduces noise introduced by folder-level aggregation, thereby preventing the models from learning spurious associations. Second, it ensures retention of all genuinely safe outputs, including those that were safe under both rule-of-thumb and principles-based checks. Third, it preserves the two signal-rich regimes we care about for modelling: (i) *principles-based overrides* (rule-of-thumb unsafe but finally safe), which the ML/LLM models should learn to recognise; and (ii) *deterministically unsafe* cases (rule-of-thumb unsafe, folder unsafe), which anchor the unsafe class. The resulting target is therefore aligned with institutional practice while remaining faithful to per-output safety.

4.6 Machine Learning Output Checking (MLSDC)

This section reports the implementation and evaluation of the machine learning layer designed to assess the disclosure safety of each output. The MLSDC module complements the rule-of-thumb checks by learning patterns from historical outputs and request-side features. Its predictions are not used in isolation, but are subsequently combined with the final LLM-based validation step described in Section 4.6.6.

For each output, the ML stage produces two additional signals that feed directly into this later stage:

- **ML check**: a binary SAFE/UNSAFE classification (encoded as 0 for SAFE and 1 for UNSAFE);
- **ML score**: a continuous probability (ml_score) representing the estimated likelihood of an unsafe disclosure.

This design ensures that deterministic, statistical, and probabilistic signals are all available to the final LLM decision-making step.

4.6.1 Training Data Characteristics

Having established how output-level labels were derived, we next examine the characteristics of the training data constructed from these labels, focusing on frequency tables as the reference case.

After filtering out observations where the rule-of-thumb result was No data and outputs not corresponding to frequency tables, the resulting dataset contained 12,428 outputs. Of these, 12,364 were labelled *SAFE* and only 64 were labelled *SAFE* by Statistics Portugal.

This raw distribution is highly imbalanced, with the vast majority of outputs classified as *SAFE*. To mitigate the risk of machine learning models overfitting to the majority class, we applied stratified sampling over the joint distribution of the institutional label (output_checking_label) and the rule-of-thumb label (rule_based). Specifically, up to 200 samples were drawn per cell of the contingency table, yielding a more balanced training dataset. This approach retains diversity across categories while ensuring sufficient representation of unsafe cases for learning. Table 4.10 shows the resulting distribution.

Table 4.10: Balanced sampling of outputs by rule-of-thumb and institutional labels.

	SAFE (0)	**UNSAFE (1)**
Rule-of-thumb = 0	200	0
Rule-of-thumb = 1	200	64

The features implemented in the machine learning pipeline were drawn from three main families. First, we included signals derived directly from the rule-of-thumb disclosure control (RBOSDC) checks. These features capture core disclosure heuristics such as the number of cells with fewer than ten observations, the share of such small cells, indicators for whether any row or column shows dominance above 90%, the proportion of concentrated cells, the presence of maxima or minima, and the degrees of freedom where reported.

Second, structural characteristics of each output were encoded. These features represent the declared output type, expressed as one-hot indicators, and basic measures of output shape such as the number of rows, columns, and reported statistics.

Third, we incorporated request-side signals wherever available. Free-text descriptions of variables used and filters applied were cleaned, lowercased, accent-stripped, and then vectorised through hashing to avoid high-cardinality encodings. In addition, a binary flag was included to indicate whether any term from the internal glossary of sensitive variables appeared in the request text.

All features were stored in numeric form. Textual elements were hashed into sparse numeric vectors, which integrate well with tree-based models without requiring scaling. Metadata-derived features such as sample design, survey vintage, or microdata-derived dominance measures were described in the methodology but are not yet included in this implementation, as they would require access to the original datasets.

Correlation Analysis

Before turning to the results, it is important to note that structural features derived directly from the outputs - such as the proportion of small cells or the dominance of individual contributors - exhibited strong correlations with the unsafe label. This behaviour is expected, as such features are themselves deterministic components of the RBOSDC checks. To avoid redundancy, the correlation analysis presented here focuses instead on request-side variables, which provide complementary and less trivial signals of disclosure risk.

To this end, we computed correlations between individual variables extracted from requests and the final output-checking labels. The results are shown in Figures 4.5 and 4.6.

The analysis shows that only a handful of variables display non-negligible associations with the unsafe label. The strongest positive correlation is observed for num_liquidacao (0.12),

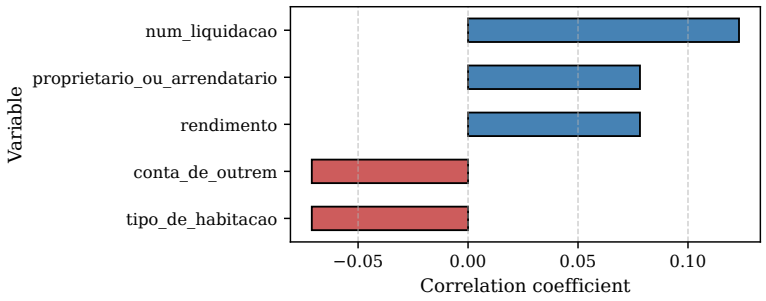

Figure 4.5: Correlation of extracted variables with the unsafe label.

followed by proprietario_ou_arrendatario and rendimento (both ≈ 0.08). Several other variables, including conta_de_outrem, tipo_de_habitacao, and tipo_de_emprego, exhibit small negative correlations (around -0.07).

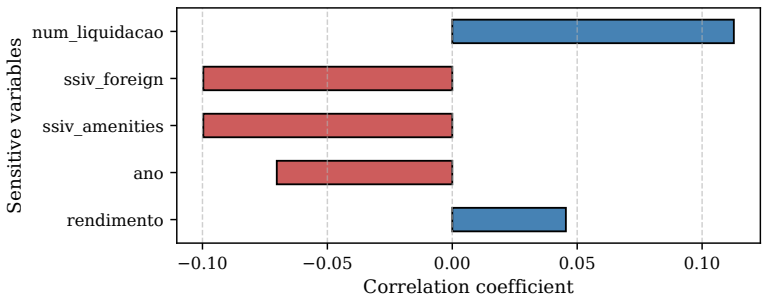

Figure 4.6: Correlation of sensitive-variable group with the unsafe label.

Among the sensitive-variable group, correlations remain weak, with rendimento and ano showing coefficients around 0.05–0.06, while others approach zero. Importantly, the analysis of aggregation variables yielded correlations all below an absolute magnitude of 0.025, and was therefore omitted as not informative.

Overall, these results confirm that no single feature strongly determines disclosure risk. Instead, unsafe classifications are better understood as the outcome of interactions among multiple variables and contextual conditions, reinforcing the case for using machine learning to capture such complex patterns.

4.6.2 Models and training setup

The models implemented were XGBoost [11] and LightGBM [31], trained with class weights to address imbalance. Training followed stratified five-fold cross-validation at the output level. Grouped CV by request, probability calibration, and the allow/review/block policy described in Section 3.6.3 are planned for the next iteration. Predictions were binarised at a global threshold of 0.5.

Table 4.11: Model performance using the balanced dataset by feature bundle (5-fold CV). B0 refers to the group of features extracted from the RBOSDC check, B1 to the features related to the structure of the output, B2 is B0+B1, B3 refers to the features extracted from the output request document and B4 is B2+B3

Bundle	model	PR AUC	MCC	Precision	Recall	Accuracy	FNR
B0	LGBM	0.4508	0.6210	0.4835	0.9800	0.8475	0.0200
B1	LGBM	0.4465	0.5437	0.4027	0.9800	0.7931	0.0200
B2	LGBM	0.4396	0.6260	0.4894	0.9800	0.8504	0.0200
B3	LGBM	0.8187	0.8972	0.8540	0.9800	0.9711	0.0200
B4	LGBM	0.8544	0.8935	0.8457	0.9800	0.9712	0.0200

4.6.3 Evaluation protocol and metrics

Evaluation focused on fold-averaged PR AUC, Matthews Correlation Coefficient (MCC), precision, recall, and the false positive rate (FPR) at the 0.5 threshold. Extended metrics such as calibrated operating bands and reliability plots will be added in future runs.

4.6.4 Predict principle-based label

In contrast to the rule-of-thumb prediction task, the second set of experiments aimed at predicting the *final* output-checking label assigned by Statistics Portugal. As described in Section 4.5, this label incorporates both the rule-of-thumb checks and the possibility of principles-based overrides, and therefore represents the ground-truth decision actually used in practice.

This task is more challenging than replicating RBOSDC, since the model must capture not only deterministic rule violations but also the contextual exceptions that human checkers allowed under PBOSDC. In effect, the target variable reflects the institutional integration of strict thresholds with discretionary principles-based reasoning.

To evaluate model performance under this setting, we followed the same experimental design as in the rule-of-thumb task, reporting results for both balanced and unbalanced datasets. The balanced version was created through stratified sampling to ensure equal representation of SAFE and UNSAFE cases, while the unbalanced version preserved the natural prevalence of labels. Comparing performance across these two regimes allowed us to test both robustness and sensitivity to class imbalance.

Table 4.11 compares the results accross the different feature bundles in the balanced setting. Metrics are averaged over 5 stratified folds; the decision threshold is 0.5.

The baseline RBOSDC bundle (B0) achieves strong recall (98%) and reasonable balanced performance (MCC ≈ 0.62). However, precision remains modest (0.48), meaning that while the model captures nearly all unsafe cases, it still produces a substantial number of false positives, wrongly labelling a lot of outputs as *SAFE*. Adding only structural descriptors (B1) lowers overall performance, with both precision and MCC dropping, suggesting that structural features alone provide insufficient discriminatory power and mainly push the classifier toward over-predicting the unsafe class. Combining RBOSDC and structural features (B2) restores performance to levels comparable with B0, confirming that structural signals add

little when rule-of-thumb features are already present.

A notable improvement appears when the features extracted from the output requests are introduced (B3). Precision rises sharply to 0.85 and MCC to 0.90, while recall remains at 98%. This indicates that textual features such as referenced variables and filters provide crucial context for identifying when rule violations are overridden under PBOSDC. The gains highlight the value of request-side information in capturing patterns not encoded in deterministic rules.

The strongest results arise from the full bundle (B4), which integrates RBOSDC features, structural descriptors, and request-derived signals. Performance remains very high, with precision and MCC both close to 0.85–0.89, recall at 98%, and overall accuracy above 97%. Importantly, the false negative rate is consistently low (2%), which is critical given that missed unsafe outputs represent the most serious disclosure risk. This demonstrates that combining deterministic, structural, and request-derived features yields the most reliable approximation of institutional decision-making.

Ablation results (B0–B4) confirm that request-derived features improve precision at fixed recall, aligning with reports that LLM-derived textual features complement structured predictors [21, 29].

Taken together, the results confirm that while RBOSDC-derived features are sufficient for detecting deterministically unsafe cases, richer representations are required to approximate the final institutional labels. In particular, request-text features emerge as the most valuable complement, enabling the models to capture the contextual nuances underlying principles-based overrides.

Besides aggregate metrics, model performance was further examined through Receiver Operating Characteristic (ROC) and Precision-Recall (PR) curves (Figure 4.7). The ROC curve illustrates the trade-off between true positive and false positive rates across thresholds, yielding an AUC of 0.98, which indicates excellent overall discrimination. The PR curve, more informative under class imbalance, shows an area under the curve of 0.81, confirming that the model maintains reasonably high precision while retaining strong recall. Together, these curves reinforce the robustness of the trained model and highlight its ability to identify unsafe outputs without overfitting to the majority safe class.

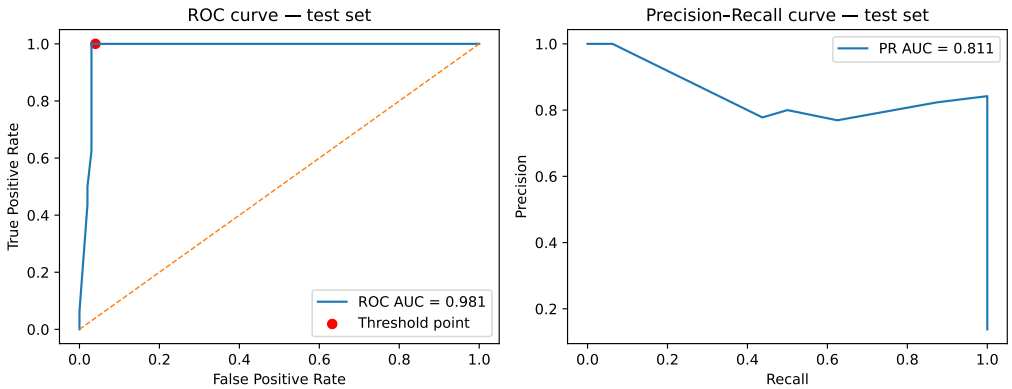

Figure 4.7: ROC and Precision–Recall curves on the test set.

Table 4.12: Model performance using the unbalanced dataset by feature bundle (5-fold CV). B0 refers to the group of features extracted from the RBOSDC check, B1 to the features related to the structure of the output, B2 is B0+B1, B3 refers to the features extracted from the output request document and B4 is B2+B3

Bundle	model	PR AUC	MCC	Precision	Recall	Accuracy	FNR
B0	LGBM	0.0508	0.2016	0.0492	1.0000	0.8287	0.0000
B1	LGBM	0.0450	0.1815	0.0415	1.0000	0.7953	0.0000
B2	LGBM	0.0620	0.2019	0.0493	1.0000	0.8291	0.0000
B3	LGBM	0.2011	0.3715	0.1458	1.0000	0.9480	0.0000
B4	LGBM	0.3288	0.4827	0.2402	1.0000	0.9717	0.0000

Comparison with non-balanced results.

The results obtained on the non-balanced dataset (Table 4.12) differ markedly from those on the balanced dataset. Because the natural distribution of outputs is highly skewed towards *SAFE*, recall for the unsafe class remains perfect (FNR = 0) across all bundles, but precision drops dramatically, especially for the RBOSDC-only bundles (B0–B2), where fewer than 5% of predicted unsafe cases are actually unsafe. This behaviour reflects the model's tendency to over-predict the majority class, achieving high apparent accuracy (≈83–97%) simply by labelling most outputs as *SAFE*.

As richer features are introduced, performance improves. Text-derived signals (B3) roughly quadruple precision compared to B0–B2 and nearly double MCC, indicating that request-level information provides critical context absent in rule-of-thumb and structural features alone. The full feature set (B4) yields the strongest results, with precision rising to 24% and MCC approaching 0.48, alongside very high overall accuracy (≈97%).

When contrasted with the balanced dataset, the trade-off becomes clear. Balancing exposes the models to more unsafe examples, which improves their ability to discriminate between genuinely safe and unsafe outputs, as reflected in substantially higher PR AUC and MCC. By contrast, in the imbalanced regime, the models achieve superficially strong accuracy through majority-class dominance but remain much weaker at identifying unsafe cases with confidence. For this reason, both perspectives are informative: the balanced dataset highlights the model's discriminative capacity, while the non-balanced dataset reflects performance under realistic deployment conditions.

4.6.5 Predicting rule-of-thumb labels

We also did experiments trying to predict the rule-of-thumb label directly from the engineered features. For this particular case, given that the dataset was balanced in terms of the target variable, the training dataset included all the 12,428 observations mentioned in Section 4.6.1.

Results comparing the results from feature bundles (see 4.6.2)ranging from RBOSDC-only signals to the full combination of RBOSDC, structural descriptors, and request-derived features are summarised in Table G.3. Both XGBoost and LightGBM were tested, but as results were nearly identical across all feature bundles, we report only LightGBM in the main

Table 4.13: Model performance (rule-check prediction) by feature bundle (5-fold CV). B0 refers to the group of features extracted from the RBOSDC check, B1 to the features related to the structure of the output, B2 is B0+B1, B3 refers to the features extracted from the output request document and B4 is B2+B3.

Bundle	Model	PR AUC	MCC	Precision	Recall	Accuracy
B0	LGBM	1.0000	1.0000	1.0000	1.0000	1.0000
B1	LGBM	0.9424	0.8329	0.7698	0.9665	0.9469
B2	LGBM	1.0000	1.0000	1.0000	1.0000	1.0000
B3	LGBM	0.3630	0.4023	0.4174	0.6539	0.7933
B4	LGBM	1.0000	1.0000	1.0000	1.0000	1.0000

text for clarity. Full comparison tables including LightGBM and XGBoost are provided in Appendix G.1.

The results highlight several important dynamics in the prediction of rule-of-thumb labels. First, the RBOSDC-only features (B0) already yield perfect performance across all metrics. This is unsurprising, since the target labels are themselves defined by RBOSDC violations, and therefore the same signals that drive the label are directly encoded in the predictors.

When structural descriptors are used in isolation (B1), performance drops considerably (PR AUC $\approx$ 0.94, MCC $\approx$ 0.83). This shows that while table-level characteristics such as size, declared output type, and number of reported statistics contain information correlated with disclosure risk, they are not sufficient on their own to fully reproduce the rule-of-thumb decisions.

Combining RBOSDC features with structural descriptors (B2) restores perfect performance (all metrics = 1.0), confirming that structural context adds only marginal refinements once rule-derived features are available.

Request-text features in isolation (B3) remain weak (PR AUC $\approx$ 0.36, MCC $\approx$ 0.40), indicating that free-text mentions of variables and filters are too noisy and indirect to reliably capture the deterministic rule logic. Nevertheless, these features are not without value, as they later contribute in the prediction of principles-based overrides where contextual interpretation matters.

Finally, when all feature families are combined (B4), the models again achieve perfect classification. This is consistent with the dominance of RBOSDC-derived signals, with additional feature sets providing redundancy rather than incremental gains in this specific task.

Overall, the experiments confirm that RBOSDC signals are the decisive drivers of performance in replicating rule-of-thumb labels, while structural descriptors offer some limited standalone predictive value, and request-text features serve as complementary signals more relevant for the principles-based prediction task.

Beyond their direct application to frequency tables, these experiments also provide indirect value for the treatment of magnitude tables. Specifically, RBOSDC rules 1 and 3 -which concern minimum cell counts and distribution over rows and columns - can be approximated by predicting disclosure safety on magnitude tables using the same feature logic. In principle, researchers are expected to submit the corresponding frequency tables alongside magnitude outputs, thereby allowing deterministic checks to be applied directly. In practice, however, such frequency tables are often omitted, as mentioned in Section 3.3. In these cases, the

rule-of-thumb prediction framework offers a pragmatic substitute by leveraging structural and request-derived features to approximate the missing checks.

4.6.6 LLM Checking (LLMSDC)

The implementation of the Large Language Model to Statistical Disclosure Control system builds on a layered approach that integrates deterministic preprocessing, machine learning, and a prompt-driven evaluation stage powered by large language models.

In the preprocessing stage, output metadata and diagnostics derived from rule-of-thumb (RB) checks are standardized into model-ready features. This process begins with the normalization of output types, where the variable output_type is cleaned and mapped to a canonical set such as frequency tables, magnitude tables, or linear regression coefficients. To capture the structural dependencies between outputs and applicable rules, a deterministic mapping defines which diagnostics are relevant to each output type; for example, frequency tables are systematically associated with the cell count and row/column distribution checks. Finally, these diagnostics are summarized into human-readable strings that transform raw metrics into concise interpretative statements. Each rule is expressed in natural language, mapping binary status values to *SAFE* or *SAFE* and including key indicators such as the number and percentage of violating cells. An illustrative summary might state: *"Cell counts rule (cells < 10) SAFE - Violating cells: 0; Percent violating: 0.0%; Evaluated cells: 434."*

Following preprocessing, the system constructs prompts for the LLM evaluation stage (see Fig. 4.8). Each prompt integrates four elements in a deterministic sequence: the researcher's original context and request description (All_context_final), the compact summaries of the relevant rule-of-thumb checks (rb_checks), the principles-based guidance drawn from authoritative disclosure control frameworks (principles_rules_text), and the output of the machine learning classifier, including its predicted label and associated probability score (ML_label, ml_score). The system prompt instructs the model to return only valid JSON conforming to a predefined schema, thereby enabling full automation of the subsequent processing pipeline.

To ensure robustness and operational reliability, the serving layer incorporates multiple safeguards. All requests are fingerprinted and cached to minimize redundant computation. API calls are managed with rate-limiting mechanisms and exponential backoff strategies to handle service overloads or temporary failures (HTTP 429/5xx). Additionally, strict parsing routines enforce JSON integrity by extracting only the first valid object from the LLM response. When a valid object cannot be identified, the system defaults to a controlled safe error state, thereby maintaining consistency and avoiding accidental disclosure of unsafe outputs.

Through this design, the LLMSDC framework combines deterministic standardisation, machine learning prediction, and principles-guided LLM reasoning in a structured and auditable workflow, with safeguards that ensure both reliability and interpretability.

This structured process ensures that principles-based reasoning is applied consistently, while deterministic RB checks and machine learning context provide grounding for the LLM assessment.

Evaluate the disclosure risk of a **SINGLE output** from data analysis.

Researcher description: Cross-tabulation of income by age group in Lisbon, 2021.

Rule-of-thumb (RB) precheck: Cell counts rule (cells <10): **SAFE** (0 viol.; 0.0%; 434 eval.) | Distribution rule (≥90% row/col): **UNSAFE** (14 viol.; 3.2%; 434 eval.; 420 safe).

Principles-based guidance: Disclosure control must ensure that no individual dominates a cell; suppress results with unsafe small counts; apply contextual thresholds.

ML model summary: Label: **SAFE** | Score: 0.21

Instructions: Decide the overall label (allowed: *SAFE*, *SAFE*, *Needs_Info*). Consider the ML label as input; if you disagree, state why. Be concise and cite specific risks. If choosing *Needs_Info*, list up to 3 missing items. Return **STRICT JSON ONLY** (no markdown, no commentary).

Figure 4.8: Example structured prompt for the LLM evaluation stage.

Implementation of LLM Fine-Tuning

As introduced in Section 3.6.4, we experimented with three open-source baselines: *open-mistral-7b*, a lightweight single-expert model, and *open-mixtral-8x22b*, a larger mixture-of-experts architecture. In addition to these pre-trained models, we implemented a fine-tuning procedure using the Mistral API. The workflow comprised three main stages: (i) preparing the training dataset; (ii) uploading the data and configuring the fine-tuning job; and (iii) monitoring and deploying the resulting fine-tuned model.

The fine-tuning dataset was derived from the same data used in the machine learning stage, but restricted to the training split (train_df). This ensured that the held-out test set (test_df) remained unseen during fine-tuning and was later used for evaluation. The data were transformed into a JSONL file using the *messages-only* format required by the Mistral API. Each record contained the structured prompt - composed of the researcher description, rule-of-thumb checks, principles-based guidance, and ML outputs - and the corresponding reference label.

This setup enforced supervised fine-tuning (SFT) with explicit instruction-tuning: the model was repeatedly shown how to map heterogeneous context into a fixed schema.

The training file was uploaded to the API with the declared purpose *fine-tune*. The fine-tuning job was created using the base model mistral-medium-latest, chosen as a compromise between inference cost and representational power. The hyperparameters included three training epochs, consistent with common practice for small domain-specific datasets to avoid overfitting. Duplicate-job handling was also implemented to prevent repeated submissions.

Upon success, the fine-tuned model was assigned a unique identifier, which can be queried in the job metadata. This identifier allows the model to be used in subsequent inference calls by replacing the base model name with the fine-tuned ID.

Evaluation of Classification and Exception Justification

The fine-tuned and baseline LLMs were applied to the held-out test set of 116 historical outputs from Statistics Portugal. For each observation, the models produced a final approval label (*SAFE*, *UNSAFE*, or *Needs_Info*), which we compared against the human ground truth.

As described in Section 3.6.4, three evaluation settings were considered: RB-only, ML-only, and RB+ML. The assessment was carried out using the full suite of classification metrics, and we also tracked the proportion of *Needs_Info* predictions as an indicator of inconclusive automated assessments.

Table 4.14 summarises the results across the three models: two open pre-trained LLMs (*open-mistral-7b (M7b)*, *open-mixtral-8x22b (Mix22b)*, *mistral-medium-latest (Medium)*) and the domain-specific fine-tuned model (FT). In addition to standard metrics, the table reports the level of agreement with the RB and ML labels, providing insight into how closely each model aligned with these components of the hybrid pipeline.

Table 4.14: LLM output-checking results with agreement against RB and ML labels. (Acc.=Accuracy, Prec.=Precision, Rec.=Recall, MCC=Matthews Correlation Coefficient, PR AUC=Precision–Recall Area Under Curve, FNR=False Negative Rate, Agr.=Agreement, Nfo=Needs Info).

Model	Setup	Acc.	Prec.	Rec.	MCC	PR AUC	FNR	Agr. RoT	Agr. ML	Nfo
M7b	RoT-only	62.10	26.70	100.00	0.39	0.27	0.00	100.00	72.41	0
Mix22b	RoT-only	62.10	26.70	100.00	0.39	0.27	0.00	100.00	72.41	0
Medium	RoT-only	64.60	31.40	100.00	0.43	0.31	0.00	85.34	65.52	17
FT	RoT-only	62.10	26.70	100.00	0.39	0.27	0.00	100.00	72.41	0
M7b	ML-only	86.20	0.00	0.00	0.00	0.14	100.00	48.28	75.86	0
Mix22b	ML-only	84.80	57.10	100.00	0.68	0.57	0.00	54.31	68.10	37
Medium	ML-only	55.60	55.60	100.00	0.00	0.56	0.00	23.48	23.48	88
FT	ML-only	89.70	57.10	100.00	0.71	0.57	0.00	72.41	100.00	0
M7b	ML+RoT	67.60	31.40	100.00	0.44	0.31	0.00	92.24	73.28	8
Mix22b	ML+RoT	63.80	27.60	100.00	0.40	0.28	0.00	98.28	74.14	0
Medium	ML+RoT	68.90	32.70	100.00	0.46	0.33	0.00	91.30	73.91	9
FT	ML+RoT	83.60	45.70	100.00	0.61	0.46	0.00	78.45	93.97	0

The results in Table 4.14 reveal several important dynamics across model size, setup, and fine-tuning.

First, across the RoT-only setup, all models achieved perfect recall but relatively low precision, as expected given that the rule-of-thumb checks are deterministic and highly conservative, leading to more observations being labelled as *SAFE*. The larger mixtral-8x22b and the lighter open-mistral-7b achieved identical results, both aligning fully with the rule-of-thumb outputs (100% agreement) with the rule-of-thumb outputs (100% agreement), while the medium-latest variant produced slightly lower agreement (85%) and flagged seventeen cases as *Needs_Info*, meaning that the LLM considered that there was not enough information to make a decision.

Second, in the ML-only setup, the differences between base models became more pronounced. The smaller 7b model failed to generalise, predicting all outputs as *SAFE* and thus achieving zero precision and recall. By contrast, the larger mixtral-8x22b and the medium-latest model demonstrated substantially stronger predictive capacity, with MCC values of 0.68 and 0.56, respectively. Both, however, frequently concluded that more information was required before making a decision, with the medium-latest model marking as many as 88 cases as

Needs_Info. While an automated system ideally produces a complete set of labelled outputs, it is generally preferable for a model to flag uncertain cases rather than issue potentially incorrect classifications. Nevertheless, the behaviour of the medium-latest model illustrates the limits of this trade-off when the proportion of flagged cases becomes excessive. In contrast, the fine-tuned model (FT) achieved the best balance: it combined high accuracy (90%), solid precision (57%), and perfect recall, while entirely eliminating *Needs_Info* responses.

Third, under the combined ML+RoT setup, agreement with both RB and ML signals was in general higher. The larger mixtral-8x22b achieved the highest agreement with RB checks (98%), while the 7b and medium-latest hovered around 91–92%. Once again, the fine-tuned model stood out, delivering higher precision (46%) and strong overall balance (MCC = 0.61) compared to its untuned counterparts. The fine-tuned model also achieved the highest accuracy in this configuration.

In general terms, it is also relevant to highlight the behaviour of the fine-tuned model across the ML-only and ML+RoT setups. In the ML-only setting, the fine-tuned LLM achieved nearly 90% accuracy and agreed with the ML label in 100% of cases. When the rule-of-thumb (RoT) labels were also introduced, overall accuracy and agreement with the ML label decreased slightly, while agreement with the RoT checks increased. This behaviour is expected: because the ML label was explicitly designed to approximate the institutional ground truth, introducing RoT signals (which often diverge from the final decision) naturally reduces alignment with the true target.

What is important, however, is that the fine-tuned model remained relatively stable under this shift. By contrast, the non–fine-tuned models (M7b and Mix22b) experienced much sharper changes: accuracy dropped by about 20 percentage points, and their agreement with RoT checks increased by roughly 50 percentage points. This indicates that while the base models tended to overfit to whichever supervision signal was most prominent, the fine-tuned model was better able to reconcile conflicting sources of information, maintaining more consistent performance and closer alignment with the true target. This highlights the robustness of the fine-tuned model to conflicting supervision signals, whereas the base models showed instability and overfitting.

Taken together, these results underline three points: (i) model size matters, with the larger mixtral-8x22b showing the most stable alignment with rule-of-thumb checks; (ii) smaller models like 7b are prone to collapse without RB guidance, classifying nearly everything as *SAFE*; and (iii) fine-tuning clearly improves robustness, eliminating *Needs_Info* responses and shifting agreement from strict rule-of-thumb mimicry towards closer alignment with ML and ground-truth labels. Together, these findings underscore the importance of model scale and targeted fine-tuning in building reliable LLM-based disclosure control systems.

4.6.7 Prototype Implementation

A working prototype was implemented to validate the feasibility of the proposed system. The prototype takes as input one output and its corresponding request file, applies the same preprocessing and feature-extraction pipeline as used for model training, and then executes the three checking layers (RBOSDC, MLSDC, and PBOSDC) in sequence. The system returns a consolidated decision containing the rule-of-thumb results for each check, the machine learning classification and probability score, and the LLM-based rationale and final label.

Prototype output check (Frequency table)

Researcher description: project_title: Economia do Cuidado em Portugal. Used Variables: ANO, nut3_est, CAE5_est. Filters applied: cálculo do número de estabelecimentos em cada ANO/nut3_est para cada uma das três CAEs seguintes: 87100, 87301 e 88101. Type of output: tabela.

Dataset: Quadros de Pessoal **Dataset_code:** 6.3

Rule-of-thumb (RB) results: Distribution rule ($\geq$90% row/col): **SAFE** (0 viol.; 0.0%). | Cell counts rule (cells <10): **UNSAFE** (164 viol.; 20.6%; 795 eval.).

ML model summary: Predicted label: **SAFE** (0) | Probability score: 0.42

LLM rationale: The output is considered **UNSAFE**. The cell counts rule flags 164 cells with fewer than 10 observations, posing a high risk of disclosure. The LLM highlights that the presence of small counts increases the likelihood of individual identification.

Recommended actions: Increase cell counts by aggregation; suppress or coarsen sensitive categories.

Final consolidated decision: *SAFE*

Figure 4.9: Example prototype run showing integrated RB, ML, and LLM evaluation for a frequency table flagged as unsafe.

This setup mirrors the intended operational deployment, where individual researcher submissions can be processed dynamically. In practice, the prototype demonstrates that the system can deliver integrated output-checking results in real time, combining deterministic, statistical, and contextual signals into a single interpretable decision.

In addition, a fallback mechanism was introduced for cases where no request document is available. In this situation, the system processes the output directly, applying the rule-of-thumb and machine learning checks, and then invokes the LLM assessment with limited contextual information. This ensures that output checking can still proceed even in the absence of a full request document, albeit with reduced contextual sensitivity.

To illustrate the behaviour of the prototype in practice, we provide a worked example of a single frequency table evaluated by the system in Fig. 4.9. The example shows how information from the output request is combined with the results of the rule-of-thumb checks, the machine learning classifier, and the LLM rationale to produce a final consolidated decision. This demonstrates the end-to-end integration of deterministic, statistical, and contextual signals into a single interpretable output.

Chapter 5

Conclusions & Discussion

This dissertation contributes a novel hybrid architecture that integrates deterministic rules, ML classification, and LLM reasoning for disclosure control. It is, to our knowledge, the first to demonstrate such an approach using real historical outputs from Statistics Portugal. Motivated by the pressing need for scalable disclosure control, the work developed a hybrid system that integrates metadata retrieval, feature engineering, rule-of-thumb checks, machine learning classification, and LLM-based reasoning into a unified pipeline.

The results presented in Chapter 4 demonstrate that such a system is both feasible and effective. The rule-of-thumb layer provided a transparent and conservative baseline, the machine learning models captured complex patterns and contextual exceptions, and the LLM component enabled flexible, principles-based reasoning. Together, these elements were shown to reproduce institutional decisions with high recall while offering interpretable justifications and actionable recommendations. Importantly, the prototype implementation illustrated how the approach can operate in real time, producing consolidated decisions for individual outputs in a way that balances automation with transparency and auditability of decisions.

In this final chapter, we discuss the implications of these findings for the practice of statistical disclosure control, reflect on the limitations of the proposed system, and outline promising directions for future research.

5.1 System Performance

One of the central aims of this thesis was to explore the role of large language models (LLMs) in the disclosure safety assessment process. Although LLMs were indeed crucial in extracting meaningful information from output requests, the evaluation conducted suggests that, when it comes to the actual classification of outputs into *Safe* or *Unsafe*, deterministic rule-of-thumb systems and machine learning (ML) models such as XGBoost or LightGBM often provide a more reliable basis for decision-making. This reflection highlights the relative strengths and limitations of each approach, discussing considerations that statistical offices should take into account when deciding which methods to implement. Table 5.1 summarizes the comparison between these methods.

Rule-of-thumb approaches proved highly effective in our work for ensuring compliance with established disclosure rules, as they are fully deterministic and transparent. However, their application required outputs to be structured in a particular format, leaving little flexibility for irregular or unexpected cases. This limitation is well recognised in tools such as

ACRO and SACRO [47], which face similar challenges of coverage and adoption across different types of outputs. Our results therefore confirm both the strengths and constraints of deterministic checks: reliable when the format is controlled, but brittle when outputs deviate.

Machine learning models, such as XGBoost and LightGBM, showed strong performance in our experiments, particularly in capturing cases where exceptions to strict rules were justified under principles-based reasoning. Prior work, notably *Towards Machine Learning-Assisted Output Checking for Statistical Disclosure Control* [17], demonstrated that ML techniques can generalise rules not explicitly present in training data. Our results extend this by showing that ML can complement rule-of-thumb checks when trained on historical outputs, especially in identifying *grey areas* where deterministic rules are too strict or permissive. Nevertheless, ML depends on consistent, structured inputs, and predictions degrade with irregular or poorly formatted outputs.

By contrast, *COACH* [46] and *COACH+* [45] explored ML with Human-in-the-Loop feedback at the cell level, classifying individual table entries as safe or unsafe before aggregating results. This fine-grained approach provides valuable signals but does not exploit request-side descriptions or broader structural features, and it amplifies class imbalance since most cells are safe. Our system complements these efforts by operating at the output level, integrating summaries of rule checks with engineered structural features and metadata from researcher requests, which better reflects how institutional decisions are taken. Similarly, compared with CASD [41], which emphasises broad file-level descriptors to achieve wide coverage, our approach focuses on disclosure-specific features tied to statistical content. Taken together, these approaches are complementary: cell-based models excel at detailed flagging, file-centric ones ensure coverage, and output-level frameworks bring contextual and metadata-driven reasoning into the decision process.

LLMs, in contrast, bring complementary strengths to the process. While they may not outperform rule-of-thumb or ML methods in terms of pure classification accuracy, they excel in providing explanations and contextual justifications for their decisions, as well as in assisting in the data cleaning and extraction process. Furthermore, they are capable of leveraging contextual understanding to assess disclosure risk even when outputs are inconsistently or poorly formatted. This ability to articulate why a particular output might be at risk - for example, by explicitly referencing small cell counts or dominance in a distribution - directly enhances transparency. It allows both researchers and output checkers to understand the reasoning behind automated decisions better, reducing the *black box* problem that often arises with ML. This makes them a valuable tool for improving communication between the statistical office and the researcher, especially in cases where the outcome is contested or requires clarification.

Fine-tuning an LLM further enhanced its effectiveness in this setting. Unlike the base models, which often became unstable when exposed to conflicting rule-of-thumb and machine learning signals, the fine-tuned model demonstrated robustness and maintained stable performance across setups. Crucially, it eliminated inconclusive *Needs_Info* responses and consistently aligned more closely with the institutional ground truth, underscoring the value of targeted domain-specific adaptation for disclosure control.

Taken together, these reflections suggest that the three approaches should not be viewed as competitors but rather as complementary tools. Rule-of-thumb checks guarantee compliance and provide a non-negotiable baseline. ML methods add adaptability and capture exceptions in cases where sufficient structured historical data exists. LLMs, in turn, fill a distinct role: they handle unstructured data, improve transparency, and provide user-facing

Approach	Strengths	Limitations	Best suited when...
Rule-of-thumb	Deterministic, transparent, guaranteed compliance with formal rules.	Rigid: requires outputs in specific, standardised formats; brittle with irregular cases.	Outputs are well structured and standardisation is enforced.
Machine Learning (e.g., XGBoost, LightGBM)	Captures exceptions to strict rules; generalises to "grey areas"; high predictive accuracy with structured data.	Dependent on consistent, labelled historical data; performance drops with irregular or novel outputs.	Sufficient historical labelled data are available, and outputs follow a consistent structure.
Large Language Models (LLMs)	Provide explanations, justifications, and adaptive feedback; handle unstructured or irregular outputs; enhance transparency and usability.	Lower classification accuracy than ML/rules; outputs may vary in consistency.	Transparency and user interaction are priorities, or when outputs are not fully standardised.

Table 5.1: Comparison of rule-of-thumb, machine learning, and LLM approaches for output checking.

explanations that enhance trust. For statistical offices, the choice of which methods to implement should therefore depend on two key considerations: (i) whether there is access to sufficiently structured historical data to train robust ML models; and (ii) whether outputs can be standardized in ways that make deterministic rules effective. In contexts where neither condition is fully met, LLMs may serve as a bridge by offering interpretable reasoning and adaptive feedback, ensuring that disclosure control processes remain both practical and understandable.

5.1.1 Comparison with Manual Output Checking

Manual output checking is a time-intensive process that requires skilled experts to review each research output against Statistical Disclosure Control (SDC) rules. This task often involves detailed case-by-case judgement, making it both costly and prone to delays when large volumes of outputs must be reviewed.

The system developed in this dissertation addresses these challenges by standardizing output formats and automating rule-of-thumb (RoT) assessments. By providing structured outputs and preliminary automated evaluations, it substantially reduces the burden on human checkers, who can then focus their attention on genuinely borderline cases rather than repetitive routine checks. In practice, reviewers benefit from having cleaner inputs and clearer contextual information, which accelerates decision-making and improves consistency.

Equally importantly, automation makes it possible to embed dynamic checking mechanisms such as *SWAR*, whereby researchers receive real-time feedback on whether their outputs are likely to be approved or flagged for disclosure risks. Allowing researchers to adjust ana-

lyses proactively reduces rejection rates and minimizes the need for multiple review iterations. While human oversight remains essential for ensuring confidentiality and addressing complex cases, the proposed system significantly decreases the time and effort required from expert reviewers, enhances transparency, and streamlines the overall output checking process.

5.2 Challenges and Limitations

Despite its advantages, the proposed system presents several challenges that must be addressed for full-scale adoption.

A first set of challenges concerns data retrieval and preprocessing. Outputs were dispersed across heterogeneous folder structures, stored in multiple file types (.xlsx, .csv, .log, etc.), and presented in diverse formats depending on the output type. This heterogeneity not only complicated parsing and standardisation, but also limited our ability to construct a representative historical dataset. In practice, many outputs could not be reliably extracted or aligned with their corresponding request documents due to time constraints, despite significant engineering efforts. As a result, the usable dataset was skewed towards the most structured and consistent output type: frequency tables. While this subset provided a sufficiently large and coherent basis for testing, it does not fully capture the diversity of outputs encountered in statistical practice, thereby constraining the scope of our study.

Another limitation relates to the coverage of disclosure risks. While the system effectively detects primary risks such as small cell counts or dominant contributions within tables, secondary disclosure risks - where confidential information could be inferred by combining multiple outputs - remain more complex and may still require human oversight. A promising extension would be the creation of a stored dataset of past outputs from each researcher, enabling new requests to be cross-referenced against historical submissions in order to identify cases where cumulative disclosure may arise.

Computational costs also present a potential barrier. Although AI-driven disclosure control systems are resource-intensive, the current design mitigates this challenge by leveraging lightweight models for local processing and considering cloud-based alternatives for non-confidential tasks. Importantly, since outputs are encoded before being processed, reliance on cloud-based models does not entail a risk of data leakage, thereby preserving confidentiality.

Finally, although the system design envisaged the computation of features directly from the underlying microdata (e.g., unweighted counts, dominance metrics, largest-contributor shares), this component was not implemented during the internship due to time constraints. Incorporating such features, which were available at Statistics Portugal, would further improve the robustness and accuracy of disclosure control.

In parallel, the use of AI introduces its own risks, including hallucinations, incorrect classifications, and over-reliance on automated decisions. To mitigate these issues, the system applies several safeguards: providing LLMs only with non-confidential information, enforcing strict prompt design to reduce hallucinations, constraining outputs to deterministic formats (e.g., JSON) for consistency, and requiring mandatory human validation in borderline cases or principles-based overrides. These measures align with the recommendations of the EU AI Act and help ensure that the system remains trustworthy, transparent, and accountable.

5.3 Future Work

Building on the results of this study, several directions for future development can be identified. First, while the system was primarily evaluated on frequency tables, future work should extend the pipeline to other types of outputs such as magnitude tables, regression results, and percentiles. Broadening the scope of supported outputs is essential to ensure that disclosure control mechanisms are robust across the full diversity of statistical products encountered in practice. A key step in this direction is the implementation of direct feature computation from the underlying microdata. Although the current study relied mainly on outputs, requests, and metadata, the design already anticipated the inclusion of features such as unweighted counts and largest-contributor shares. Incorporating this module would significantly strengthen accuracy and bring the system closer to the operational procedures followed in practice.

Another priority concerns request-side information. While the current variable and filter extraction pipeline is effective, it can still produce errors or inconsistencies. Fine-tuning domain-specific LLMs on annotated request data offers a promising way to reduce these errors and improve coverage, enabling more reliable use of researcher-provided context. Leveraging this rapidly evolving technology could substantially enhance the extraction of meaningful information from outputs and requests, thereby strengthening the ability of both rule-of-thumb and machine learning approaches to make accurate predictions based on structured data. In addition, future systems should include a stored dataset of past outputs, allowing new requests to be cross-referenced against historical ones. This would make it possible to capture secondary disclosure risks, where confidentiality breaches arise not from a single output but from the combination of multiple results.

Human validation will remain necessary, particularly during the early adoption phase. Evidence from *COACH* and *COACH+* [46, 45] clearly shows that Human-in-the-Loop (HITL) feedback can dramatically reduce false negatives (i.e., cases labelled as *SAFE* that are actually *UNSAFE* - referred to as false positives in their work) and improve the reliability of ML-assisted output checking. To support this in the present system, a structured feedback mechanism should be developed that allows reviewers to confirm, reject, or comment on AI assessments. These decisions can be stored and re-used to reinforce future evaluations, enabling a reinforcement-learning pipeline where the model adapts to institutional practice over time. Such an approach would allow the principles-based component to improve through human feedback, gradually increasing automation accuracy and trust.

Finally, deployment considerations merit attention. An important question is whether to rely on local or cloud-based implementations. Local models provide stronger confidentiality guarantees, while cloud solutions offer scalability and computational efficiency. Since outputs can be encoded before processing, cloud-based models could be used safely for non-confidential tasks, while sensitive material would remain restricted to local deployments. Future work should also align with emerging governance standards (e.g., the EU AI Act) to ensure responsible deployment of AI-driven disclosure control in official statistics.

Taken together, these directions provide a roadmap for evolving the current prototype into an operational, institution-grade disclosure control system that balances automation, human oversight, and transparency.

5.4 Final Conclusion

This dissertation has demonstrated that hybrid approaches integrating deterministic rules, machine learning, and large language models can meaningfully improve the efficiency, transparency, and scalability of output checking in official statistics. By combining the strengths of each component - rules for guaranteed compliance, machine learning for capturing contextual exceptions, and LLMs for explanation and adaptability - the proposed system reproduces institutional practice while reducing the burden on human reviewers.

Although challenges remain in extending coverage, incorporating secondary risk detection, and aligning with operational governance frameworks, the results confirm that automation can substantially streamline disclosure control without compromising confidentiality. Looking ahead, the adoption of such hybrid systems has the potential to transform output checking from a resource-intensive bottleneck into a collaborative, transparent, and responsive process that better serves both statistical offices and researchers.

References

[1] Yutaka Abe and Kazuhiro Minami. A case study of output checking in japan. Technical report, United Nations Economic Commission for Europe (UNECE), Conference of European Statisticians: Expert Meeting on Statistical Data Confidentiality (SDC 2023), Wiesbaden, 2023. Accessed: 2025-08-31.

[2] Kyle Alves and Felix Ritchie. Runners, repeaters, strangers, and aliens: Operationalising efficient output disclosure control. *Statistical Journal of the IAOS*, 36:1281–1293, 2020.

[3] Fabian Biester, Mohamed Abdelaal, and Daniel Del Gaudio. Llmclean: Context-aware tabular data cleaning via llm-generated ofds. In *Proceedings of Conference acronym 'XX*. ACM, 2018.

[4] Steve Bond, Maurice Brandt, and Peter-Paul de Wolf. Guidelines for output checking. Technical report, European Commission, Eurostat, 2020.

[5] Dario Buono, Marius Felecan, and Cristiano Tessitore. An introduction to large language models and their relevance for statistical offices. Statistical working paper, Eurostat, Publications Office of the European Union, 2024.

[6] Ricardo Carvalho. Ai-driven output checking for official statistics: Leveraging llms and workflow automation. In *Proceedings of the 20th Iberian Conference on Information Systems and Technologies (CISTI 2025)*, Porto, Portugal, 2025. Forthcoming.

[7] Statistics Netherlands (CBS). Catalogue of services - microdata services 2025. Technical report, Statistics Netherlands, 2024.

[8] Statistics Netherlands (CBS). Microdata: Conducting your own research, 2025. Accessed: 2025-02-03.

[9] Keyu Chen and Shiliang Sun. Cp-rec: Contextual prompting for conversational recommender systems. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI-23)*. Association for the Advancement of Artificial Intelligence (AAAI), 2023.

[10] Nanjiang Chen, Xuhui Lin, Hai Jiang, and Yi An. Automated building information modeling compliance check through a large language model combined with deep learning and ontology. *Buildings*, 14(1983), 2024.

[11] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM, 2016.

[12] Davide Chicco and Giuseppe Jurman. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC Genomics*, 21(1):6, 2020.

[13] Francisco de Arriba-Pérez and Silvia García-Méndez. Leveraging large language models through natural language processing to provide interpretable machine learning predictions of mental deterioration in real time. *Arabian Journal for Science and Engineering*, 2024.

[14] Francisco de Arriba-Pérez, Silvia García-Méndez, Javier Otero-Mosquera, and Francisco J. González-Castaño. Explainable cognitive decline detection in free dialogues with a machine learning approach based on pre-trained large language models. *Applied Intelligence*, 54:12613–12628, 2024.

[15] Instituto Nacional de Estatística (INE). Output checking no ine: Prática atual e desafios futuros. Technical report, Instituto Nacional de Estatística, 2025. Accessed: 2025-02-03.

[16] Ben Derrick, Elizabeth Green, Felix Ritchie, and Paul White. Towards a comprehensive theory and practice of output sdc. Technical report, United Nations Economic Commission for Europe (UNECE), Conference of European Statisticians: Expert Meeting on Statistical Data Confidentiality (SDC 2023), Wiesbaden, 2023. Accessed: 2025-08-31.

[17] Josep Domingo-Ferrer and Alberto Blanco-Justicia. Towards machine learning-assisted output checking for statistical disclosure control. Technical report, Universitat Rovira i Virgili, CYBERCAT-Center for Cybersecurity Research of Catalonia, UNESCO Chair in Data Privacy, 2021. Working Paper.

[18] Research Data Centre (FDZ). Access to microdata, 2025. Accessed: 2025-02-03.

[19] Elizabeth Green, Felix Ritchie, and Jim Smith. Understanding output checking: Deliverable 1 of the eurostat specific contract: Access to european microdata in eurostat safe centre - automatic checking of the output. Technical report, European Commission, Eurostat, May 2020. GOPA Worldwide Consultants in joint venture with GOPA Luxembourg.

[20] Elizabeth Green, Felix Ritchie, and Paul White. The statbarn: A new model for output statistical disclosure control. In Josep Domingo-Ferrer and Melek Önen, editors, *Privacy in Statistical Databases (PSD 2024)*, pages 284–293, Cham, 2024. Springer Nature Switzerland.

[21] Artur Grigorev, Khaled Saleh, Yuming Ou, and Adriana-Simona Mihăită. Enhancing traffic incident management with large language models: A hybrid machine learning approach for severity classification. *arXiv preprint arXiv:2403.13547*, 2024.

[22] Radiah Haque, Hui-Ngo Goh, Choo-Yee Ting, Albert Quek, and M.D. Rakibul Hasan. Leveraging llms for optimised feature selection and embedding in structured data: A case study on graduate employment classification. *Computers and Education: Artificial Intelligence*, 8:100356, 2025.

[23] Noah Hollmann, Samuel Müller, and Frank Hutter. Large language models for automated data science: Introducing caafe for context-aware automated feature engineering. In *Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS 2023)*. NeurIPS, 2023.

[24] Xiaobao Huang, Mihir Surve, Yuhan Liu, Tengfei Luo, Olaf Wiest, Xiangliang Zhang, and Nitesh V. Chawla. Application of large language models in chemistry reaction data extraction and cleaning. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24)*. ACM, 2024.

[25] Zezhou Huang and Eugene Wu. Relationalizing tables with large language models: The promise and challenges. *arXiv preprint arXiv:2402.12345*, 2024.

[26] Anco Hundepool, Josep Domingo□ Ferrer, Luisa Franconi, Sarah Giessing, Rainer Lenz, Jane Naylor, Eric Schulte Nordholt, Giovanni Seri, Peter-Paul De Wolf, Reinhard Tent, Andrzej Młodak, Johannes Gussenbauer, and Kamil Wilak. Handbook on statistical disclosure control. Technical report, Centre of Excellence on Statistical Disclosure Control, August 2025. Second edition.

[27] Instituto Nacional de Estatística (INE). Bases de microdados anonimizados para fins de investigação científica. Technical report, Instituto Nacional de Estatística (INE), September 2024. Última atualização: 18-09-2024. Dados disponíveis da 1ª (1994) à 8ª vaga (2001).

[28] Instituto Nacional de Estatística (INE). Lista de microdados para fins de investigação científica. Technical report, Instituto Nacional de Estatística, Portugal, September 2024. Última atualização em 18-09-2024.

[29] Areeba Ishtiaq, Kashif Munir, Ali Raza, Nagwan Abdelsamee, Mona M. Jamjoom, and Zahid Ullah. Product helpfulness detection with novel transformer based bert embedding and class probability features. *IEEE Access*, 12:1–15, 2024.

[30] Daniel P. Jeong, Zachary C. Lipton, and Pradeep Ravikumar. Llm-select: Feature selection with large language models. *arXiv preprint arXiv:2407.02694*, 2024.

[31] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*, pages 3146–3154, 2017.

[32] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *arXiv preprint arXiv:2205.05638*, 2022.

[33] Luyi Ma, Nikhil Thakurdesai, Jiao Chen, Jianpeng Xu, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Llms with user-defined prompts as generic data operators for reliable data processing. *arXiv preprint arXiv:2312.16351*, 2023.

[34] Anu Mohan, K Ashok, Muskan Rizwan Shaikh, Y P Dhanush, and Yajat Vishwakarma. Data integration and transformation using large language models. *Grenze International Journal of Engineering and Technology*, 9(2):1645–1649, 2023.

[35] Zan Ahmad Naeem, Mohammad Shahmeer Ahmad, Mohamed Eltabakh, Mourad Ouzzani, and Nan Tang. Retclean: Retrieval-based data cleaning using llms and data lakes. *Proceedings of the VLDB Endowment*, 17(12):4421–4424, 2024.

[36] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*, 2024.

[37] Australian Bureau of Statistics (ABS). Tablebuilder: Microdata and tablebuilder statistics, 2025. Accessed: 2025-02-03.

[38] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*,

2022.

[39] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[40] Mohaimenul Azam Khan Raiaan, Md. Saddam Hossain Mukta, Kaniz Fatema, Nur Mohammad Fahad, Sadman Sakib, Most Marufatul Jannat Mim, Jubaer Ahmad, Mohammed Eunus Ali, and Sami Azam. A review on large language models: Architectures, applications, taxonomies, open issues and challenges. *IEEE Access*, 12:26839–26855, 2024.

[41] Titouan Rigaud, Remy Marquier, Eric Debonnel, and Pengfei Liu. Checking data outputs from research works: A mixed method with ai and human control. Technical report, United Nations Economic Commission for Europe (UNECE), Conference of European Statisticians: Expert Meeting on Statistical Data Confidentiality (SDC 2023), Wiesbaden, 2023. Accessed: 2025-08-31.

[42] Felix Ritchie. Disclosure detection in research environments in practice. Technical report, Office for National Statistics, UK, January 2008. Joint UNECE/Eurostat work session on statistical data confidentiality, Manchester, UK, 17-19 December 2007.

[43] Felix Ritchie, Elizabeth Green, and James Smith. Automatic checking of research outputs (acro): A tool for dynamic disclosure checks. Technical Report KS-TC-21-005-EN-N, European Commission, Eurostat, September 2021. Statistical Working Paper.

[44] Ravid Shwartz-Ziv and Amitai Armon. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90, 2022.

[45] Manel Slokom, Jel Vankan, and Peter-Paul de Wolf. From coach to coach+: Automating output checking with human-in-the-loop. *Statistical Journal of the IAOS*, 2024. Published online: 2024-11-01; Accessed: 2025-08-31.

[46] Manel Slokom, Jel Vankan, Peter-Paul de Wolf, and Martha Larson. Coach: Computer-assisted output checking with human-in-the-loop. Technical report, United Nations Economic Commission for Europe (UNECE), Conference of European Statisticians: Expert Meeting on Statistical Data Confidentiality (SDC 2023), Wiesbaden, 2023. Accessed: 2025-08-31.

[47] Jim Smith, Richard Preen, Maha Albashir, Felix Ritchie, Elizabeth Green, Simon Davy, Pete Stokes, and Sebastian Bacon. Sacro: Semi-automated checking of research outputs. Technical report, United Nations Economic Commission for Europe (UNECE), Conference of European Statisticians, Expert Meeting on Statistical Data Confidentiality, 2023. Accessed: 2025-08-31.

[48] Hamed Taherkhani, Melika Sepindband, Hung Viet Pham, Song Wang, and Hadi Hemmati. Epic: Cost-effective search-based prompt engineering of llms for code generation. *arXiv preprint arXiv:2408.11198*, 2024.

[49] Tao Tu, Eric Loreaux, Emma Chesley, Adam D. Lelkes, Paul Gamble, Mathias Bellaiche, Martin Seneviratne, and Ming-Jun Chen. Automated loinc standardization using pre-trained large language models. In *Proceedings of Machine Learning for Health (ML4H) 2022*, pages 343–355. PMLR, 2022.

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[51] Dukyong Yoon, Changho Han, Dong Won Kim, Songsoo Kim, SungA Bae, Jee An Ryu, and Yujin Choi. Redefining health care data interoperability: Empirical exploration of large language models in information exchange. *Journal of Medical Internet Research*, 26:e56614, 2024.

[52] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. In *International Conference on Learning Representations (ICLR)*, 2023.

Appendix A

Output classification

This appendix presents the mapping between Eurostat output types and the corresponding *statbarn* categories .

Table A.1: Crosswalk between Eurostat "Type of Output General" and closest *statbarn*.

Output type (image)	Image class	Closest statbarn (paper)	Paper safe?	Status (paper)
Frequency tables	Unsafe	**Frequencies**	Unsafe	Very well understood
Magnitude tables	Unsafe	**Linear aggregations** (e.g., sums/totals)	Unsafe	Very well understood
Maxima, minima, percentiles (incl. median)	Unsafe	**Position** (min/median/quantiles)	Unsafe	Provisional
Mode	Safe	**Mode**	Safe	Confirmed
Means, indices, ratios, indicators	Unsafe	**Linear aggregations** / **Calculated ratios**	Unsafe	Very well understood / Provisional
Concentration ratios	Safe	**Concentration ratios** (e.g., Herfindahl)	Safe	Provisional
Higher moments (variance, covariance, kurtosis, skewness)	Safe	**Shape** (higher moments)	Safe	Provisional
Graphs: pictorial representations of actual data	Unsafe	*No dedicated statbarn* (treat via underlying statistic; survival → **Implicit tables**)	Unsafe (when treated as implicit tables)	Provisional
Linear regression coefficients	Safe	**Correlation/regression coefficients**	Safe	Confirmed
Non-linear regression coefficients	Safe	**Correlation/regression coefficients**	Safe	Confirmed

Continued on next page

Table A.1 (continued)

Output type (image)	**Image class**	**Closest statbarn (paper)**	**Paper safe?**	**Status (paper)**
Estimation residuals	Unsafe	*No dedicated statbarn* (diagnostics/predictions)	-	-
Summary & test statistics from estimates (R^2, χ^2, etc.)	Safe	**Statistical hypothesis tests**	Safe	Provisional
Correlation coefficients	Safe	**Correlation/regression coefficients**	Safe	Confirmed
Factor analysis	Safe	*Not explicit*; closest family: **Clusters** (latent structure)	?	No knowledge
Correspondence analysis	Safe	*Not explicit*; dimensional-association method (treat via context)	-	-

Appendix B

RBOSDC summary

Table B.1: Rule-of-thumb disclosure risks and required features by output type [4].

Output type	Rule-of-thumb risk logic	Key features to extract
Frequency tables	Cells must contain at least 10 units; no cell should dominate its row/column ($> 90\%$); organisational dominance must be avoided.	• Cell count (unweighted) • Row/column group proportions • Organisational source breakdown
Magnitude tables (means, totals)	Cells must be based on at least 10 units; largest contributor must not exceed 50% of the cell value.	• Cell count • Largest-contributor proportion • Total value per cell
Maxima / minima / percentiles	Not allowed if extreme individuals could be identified; tail percentiles (e.g., 1st or 99th) are sensitive.	• Percentile position • Sample size • Value range
Modes	Mode must represent at least 10 observations.	• Frequency of modal value • Total sample size
Means, ratios, indicators	Avoid outputs dominated by few individuals; ensure contributor balance.	• Number of contributors • Contributor dominance check
Concentration ratios	The ratio must not indicate dominance of a few contributors.	• Top contributor shares • Contributor rank list

Continued on next page

Output type	Rule-of-thumb risk logic	Key features to extract
Higher moments (e.g., variance)	Must be computed with at least 10 degrees of freedom.	• Sample size • Degrees of freedom
Graphs / visuals	Graphs should not be released; only pre-cleared data may be visualised.	• Graph presence flag • Underlying data structure
Regression coefficients (linear / nonlinear)	At least one coefficient must be withheld; output must be based on at least 10 residual d.f.	• Number of coefficients • Residual degrees of freedom
Residuals	Not permitted for disclosure due to re-identification risk.	• Residual presence flag • Residual values (if provided)
Summary and test statistics	Permitted only with ≥ 10 residual degrees of freedom.	• Residual degrees of freedom • Statistic type
Factor and correspondence analysis	Require at least 2 variables and enough observations for safe output.	• Number of variables • Sample size
Correlations	Must not be exactly -1, 0, or 1.	• Correlation coefficient value

Appendix C

PBOSDC summary

Table C.1: Principles-based risk considerations and required features by output type [4].

Output type	Risk considerations	Key features to extract
Frequency tables	Small cell counts; confidential data sources; identification via combinations or rank order.	• Minimum cell count • Data source (survey/admin/derived) • Dataset confidentiality flag • Rank-order information (if disclosed) • Geographic breakdown • Data age • Sample design
Magnitude tables (means, totals)	Disclosure by dominance or extremes; sensitive variables.	• Sensitive-variable presence (e.g., income) • Dominance metrics (e.g., top-contributor %) • Cell-level aggregation • Number of contributors • Outlier detection
Regression outputs	Disclosure via influential variables, overfitting, or model explainability.	• Number of observations • Number of predictors • R^2, residual patterns • Suppressed coefficients • Subgroup sample sizes

Continued on next page

Output type	Risk considerations	Key features to extract
Percentiles / max / min / quantiles	Direct identification of extreme individuals.	• Presence of extreme values • Percentile position (e.g., top 1%) • Geography / subdomain • Sample size • Variable sensitivity
Graphs / maps / visuals	Visual identification of sparse points/outliers; residual patterns.	• Data granularity in visuals • Plot type (scatter, map, histogram) • Sample size per point • Colour coding / legend info • Regression line visibility
Residuals or diagnostics	Disclosure through uniquely large residuals or influence diagnostics.	• Residual magnitudes • Number of units contributing • Influence diagnostics (Cook's distance, leverage) • Data sensitivity
Time series / trends	Risk of deducing events or individuals from temporal anomalies.	• Temporal granularity (daily, monthly, …) • Sample size per time unit • Change points / spikes • Subgroup breakdowns

Appendix D

INE baseline thresholds

Table D.1: INE baseline thresholds and safeguards for common output classes (illustrative) [15].

Area	INE baseline (rules-of-thumb)
Small cells / class disclosure	Minimum unweighted count per cell (e.g., $n \geq 10$); row/column concentration capped (e.g., $\leq 90\%$).
Contributor dominance	Largest contributor limited (e.g., $\leq 50\%$ of cell total); apply secondary suppression where needed.
Model outputs (coefficients)	Release full coefficients only if the model is non-saturated and residual d.f. meet a minimum (e.g., ≥ 10). Otherwise treat as implicit tables.
Residuals / diagnostics	Do not release residuals or case-level diagnostics; aggregated summaries only.
Graphs / survival	Assess via the underlying statistic (e.g., survival $\rightarrow$ implicit/linked tables); avoid exact read-offs (no data labels; bin/jitter).
Percentiles / extremes	Treat minima/maxima/percentiles as high risk; allow only with strong contextual justification and sufficient group size/coarsening.

Appendix E

Features for Disclosure Risk Assessment: Definitions and Data Sources

Table E.1: Summary of features for disclosure risk assessment

Feature / Indicator	Description	Where to get it
Cell count	Ensures each output value (cell, mean, percentile) is based on at least a minimum number of units (typically 10).	Output (frequency tables); dataset or metadata (others).
Number of observations used	Used to estimate degrees of freedom and validate cell size.	Output or metadata.
Filters applied	Defines the subpopulation analysed, which can reduce sample size and increase risk.	Output request or code.
Temporal reference	Specifies the time period covered; recent or fine-grained references may increase risk.	Output request or metadata.
Geographic granularity	Level of territorial aggregation (e.g., national, region, municipality); finer levels increase disclosure risk.	Metadata or output request.
Age of the data	Older data may pose lower re-identification risk.	Metadata.
Dataset source	Confidential data sources require stricter checks.	Metadata.
Sample design	If the sample itself is confidential, even small counts are risky.	Metadata.
Group proportion (row/-column)	Ensures no cell holds more than 90% of a row or column total.	Output or dataset.
Organisational dominance	Checks if $\geq$90% of units come from a single organisation.	Dataset or metadata.
Largest contributor proportion	Ensures no single value dominates cell totals (>50%).	Dataset.
Degrees of freedom	Model-based outputs must have $\geq$10 degrees of freedom.	Dataset or researcher-provided.
Variable count in multivariate outputs	Factor/correspondence analysis must use $\geq$2 variables.	Output or dataset.
Number of coefficients reported	At least one (e.g., intercept) should be withheld.	Output.

Continued on next page

Feature / Indicator	Description	Where to get it
Variable type	Binary-only regressions resemble table means; coefficients cannot be based solely on categorical variables.	Output.
Type of summary statistic	Used to estimate degrees of freedom.	Output.
Graph presence flag	Indicates whether graphical output is submitted.	Output.
Graph type (scatter, residual, etc.)	Identifies type of visual and risk associated.	Output.
Outlier check / residual analysis	Checks for visual disclosure risk in graphs.	Output.
Granularity of axes or values	High granularity increases risk of exact identification.	Output.
Rank order of contributors	Risk increases if contributors can be externally ranked.	Output or metadata.
Usefulness / reliability of output	Low-reliability estimates may be unnecessary and risky.	Subjective - LLM or human.
Variance around percentile	Low variance implies group disclosure; high variance $\rightarrow$ uniqueness.	Output (if available).
Residual presence flag	Residuals should be excluded unless justified.	Output.
Residual value distribution	Outliers in residuals may imply individual data.	Output.

Appendix F

Prompt Examples for Output Request Feature Extraction

F.1 Extracting Variables

Prompt: Input: $json['Variáveis utilizadas']

Given the following text, extract all variable names that appear before the expression "Proveniente de" or similar. Return the variables as a comma-separated list.

Example 1 Input: Rendimento global e Coleta líquida. Proveniente das notas de liquidação de IRS Output: Rendimento global, Coleta líquida

Example 2 Input: VAB, imposto, subsídios, volume de negócios do Dataset SCIE Output: VAB, imposto, subsídios, volume de negócios

A.2 Extracting Filter Conditions

Prompt: Input: $json['Descrição']

Extract all filter conditions present in the input, such as population restrictions (e.g., age, sex, region). Return as a comma-separated list.

Example 1 Input: Frequência de mulheres com 65 anos ou mais na região Norte Output: mulheres, idade $\geq$ 65, região = Norte

Example 2 Input: Número de trabalhadores agrícolas entre 20 e 40 anos Output: trabalhadores agrícolas, idade entre 20 e 40

A.3 Extracting Intended Output Type

Prompt: Input: $json['Descrição']

From the description, identify the type of statistical output intended (e.g., frequency table, regression, mean, correlation). Return one of the following labels: Frequency table, Regression, Mean, Percentile, Correlation, Graph, Residuals, etc.

Example 1 Input: Tabela de frequências por idade e sexo Output: Frequency table

Example 2 Input: Coeficientes de regressão sobre o rendimento em função da escolaridade Output: Regression

A.4 Extracting Aggregation Levels

Prompt: Input: $json['Descrição']

Identify any aggregation levels (e.g., by sex, region, education). Return the grouping variables as a comma-separated list.

Example 1 Input: Médias salariais por setor de atividade e sexo Output: setor de atividade, sexo

Example 2 Input: Número de habitantes por NUTS 2 Output: NUTS 2

A.5 Extracting Temporal References

Prompt: Input: $json['Descrição']

Identify any time references (e.g., year, range of years, phrases like "últimos 5 anos"). Return the time period as-is.

Example 1 Input: Taxa de desemprego entre 2018 e 2022 Output: 2018–2022

Example 2 Input: Dados de escolaridade dos últimos 10 anos Output: últimos 10 anos

Appendix G

Supplementary Results

G.1 Comparison between LightGBM and XGBoost

This appendix presents the full set of results for both LightGBM and XGBoost across the three tasks:

1. Predicting the final output-checking label with a balanced dataset.
2. Predicting the final output-checking label with the original, imbalanced dataset.
3. Predicting the rule-check label (RBOSDC).

Table G.1: Model performance (balanced dataset, final label prediction) with balanced dataset by feature bundle (5-fold CV). B0 refers to the group of features extracted from the RBOSDC check, B1 to the features related to the structure of the output, B2 is B0+B1, B3 refers to the features extracted from the output request document and B4 is B2+B3

Bundle	model	PR AUC	MCC	Precision	Recall	Accuracy	FNR
B0	XGB	0.4513	0.6241	0.4859	0.9800	0.8504	0.0200
B0	LGBM	0.4508	0.6210	0.4835	0.9800	0.8475	0.0200
B1	XGB	0.4546	0.5435	0.4023	0.9800	0.7931	0.0200
B1	LGBM	0.4465	0.5437	0.4027	0.9800	0.7931	0.0200
B2	XGB	0.4473	0.6241	0.4859	0.9800	0.8504	0.0200
B2	LGBM	0.4396	0.6260	0.4894	0.9800	0.8504	0.0200
B3	XGB	0.8223	0.8705	0.8125	0.9800	0.9626	0.0200
B3	LGBM	0.8187	0.8972	0.8540	0.9800	0.9711	0.0200
B4	XGB	0.8820	0.9153	0.8821	0.9800	0.9769	0.0200
B4	LGBM	0.8544	0.8935	0.8457	0.9800	0.9712	0.0200

Table G.2: Model performance (imbalanced dataset, final label prediction) by feature bundle (5-fold CV). B0 refers to the group of features extracted from the RBOSDC check, B1 to the features related to the structure of the output, B2 is B0+B1, B3 refers to the features extracted from the output request document and B4 is B2+B3

Bundle	model	PR AUC	MCC	Precision	Recall	Accuracy	FNR
B0	XGB	0.0508	0.2019	0.0493	1.0000	0.8291	0.0000
B0	LGBM	0.0508	0.2016	0.0492	1.0000	0.8287	0.0000
B1	XGB	0.0450	0.1826	0.0419	1.0000	0.7974	0.0000
B1	LGBM	0.0450	0.1815	0.0415	1.0000	0.7953	0.0000
B2	XGB	0.0620	0.2024	0.0495	1.0000	0.8298	0.0000
B2	LGBM	0.0620	0.2019	0.0493	1.0000	0.8291	0.0000
B3	XGB	0.2011	0.3715	0.1458	1.0000	0.9480	0.0000
B3	LGBM	0.2011	0.3715	0.1458	1.0000	0.9480	0.0000
B4	LGBM	0.3288	0.4827	0.2402	1.0000	0.9717	0.0000
B4	XGB	0.3171	0.4827	0.2402	1.0000	0.9717	0.0000

Table G.3: Model performance (rule-check prediction) by feature bundle (5-fold CV). B0 refers to the group of features extracted from the RBOSDC check, B1 to the features related to the structure of the output, B2 is B0+B1, B3 refers to the features extracted from the output request document and B4 is B2+B3

model	PR AUC	MCC	Precision	Recall	Accuracy	Bundle
XGB	1.0000	1.0000	1.0000	1.0000	1.0000	B0
LGBM	1.0000	1.0000	1.0000	1.0000	1.0000	B0
LGBM	0.9424	0.8329	0.7698	0.9665	0.9469	B1
XGB	0.9390	0.8345	0.7844	0.9509	0.9491	B1
XGB	1.0000	1.0000	1.0000	1.0000	1.0000	B2
LGBM	1.0000	1.0000	1.0000	1.0000	1.0000	B2
LGBM	0.3630	0.4023	0.4174	0.6539	0.7933	B3
XGB	0.3624	0.4023	0.4174	0.6539	0.7933	B3
XGB	1.0000	1.0000	1.0000	1.0000	1.0000	B4
LGBM	1.0000	1.0000	1.0000	1.0000	1.0000	B4