

BABEȘ-BOLYAI UNIVERSITY CLUJ-NAPOCA
FACULTY OF MATHEMATICS AND
COMPUTER SCIENCE
SPECIALIZATION COMPUTER SCIENCE

DIPLOMA THESIS

Bidders Recommender for Public Procurement Auctions using Artificial Intelligence

Supervisor
Adriana Mihaela Coroiu

Author
Silaschi Iasmina-Oana

2023

ABSTRACT

The primary objective of this research paper is to design and implement a specifically tailored application that caters to the unique needs of the Romanian public procurement auctions domain. This application is engineered to rank bidders based on their compatibility and suitability, an assessment determined by the precision of various machine learning models. The innovative creation of this application aims to enhance the effectiveness, fairness, and transparency of the procurement process, an endeavor accomplished by leveraging artificial intelligence techniques, including data analysis and predictive modeling.

In order to conduct a thorough empirical evaluation, a real-life case study is undertaken, centered around a substantial dataset procured from numerous Romanian public procurement tenders. This dataset, graciously provided by an esteemed economist, represents a vast mass of information. More precisely, it encompasses data from a total of 289,472 businesses, 47,974 contracting authorities, and 42,474 bids. The utilization of this expansive dataset ensures a robust and comprehensive analysis, thereby permitting the extraction of meaningful insights and facilitating the drawing of reliable conclusions.

Detailed data analysis and data clustering performed on this dataset have disclosed intriguing specifics and, perhaps, certain anomalies such as unusually low bidding contract values. In the quest to rank bidders effectively, the performance of nine intelligent algorithms is evaluated and compared. The top-performing algorithms in this context appear to be Decision Trees, along with ensemble methods derived from them, namely the Random Forest Classifier and the Extra Trees Classifier.

The outcomes of this study are satisfying, outperforming the metrics of a similar study conducted in the same domain. Given the potential concerns about Romania's integrity in dealing with contracting companies for public procurement auctions, this study holds significant value in its ability to validate the decision-making process employed in previous auctions. Furthermore, the application shows great potential in providing indispensable insights for upcoming tenders, thereby addressing and potentially alleviating the challenges inherent in this domain.

By casting light on the effectiveness of past procurement decisions and offering strategic guidance for future acquisitions, this research holds substantial implications for improving transparency and streamlining the bidder selection process within the public procurement area. Therefore, with potential for future work upon it, it promises to become a valuable addition in enhancing the integrity and efficiency of public procurement in Romania.

Contents

1	Introduction	1
1.1	Motivation	2
1.2	Structure of the Paper	2
2	Theoretical Background	3
2.1	Public Procurement Auctions	4
2.1.1	Romanian Electronic Public Procurement System	4
2.1.2	CPV Codes	5
2.2	Machine Learning	6
2.2.1	Intelligent Algorithms	7
2.2.2	Data Analysis	9
2.3	Software Application Development	9
2.3.1	Frontend Development with React	9
2.3.2	Backend Development with Flask	11
3	Related Work	12
3.1	Predicting the Success of a Organization in Public Procurement	12
3.2	Other Machine Learning Approaches on Public Procurement	13
3.3	Overview upon Related Work	14
4	Bidders Recommender Application	15
4.1	Methods	16
4.2	Data set	19
4.2.1	Data Description	21
4.2.2	Data Analysis	23
4.2.3	Data Clustering	29
4.2.4	Data Preprocessing	32
4.3	Computational Experiments	37
4.3.1	Data Encoding	37
4.3.2	Machine Learning Algorithm	39
4.3.3	Parameters Fine Tuning	40

4.3.4	Ranking Predictions	41
4.3.5	Random Forest vs. Extra Trees	42
4.3.6	Specialized Literature Comparison	43
4.4	Implementation Description	44
4.4.1	Frontend	45
4.4.2	Backend	47
4.5	Application Testing	48
4.5.1	Unit Testing	48
4.5.2	Integration Testing	49
5	SWOT Analysis	50
5.1	SWOT Matrix	51
5.2	SWOT Description	52
5.2.1	Strengths	52
5.2.2	Weaknesses	52
5.2.3	Opportunities	53
5.2.4	Threats	53
6	Conclusions and Future Work Directions	54
6.1	Concluding Remarks	54
6.2	Future Work Directions	54
6.3	Acknowledgements	55
	Bibliography	56

Chapter 1

Introduction

This application will rank bidders considering the best fit for a public auction. Taking into consideration the persistent issues regarding Romania's integrity in the case of contracting companies for public procurement auctions, this application could validate if indeed the best decision has been made in auctions from the past, as well as be helpful regarding this matter in future tenders.

Even though this subject has already been researched, it has never been specifically tailored for tenders from Romania and using a data set of organizations from this country. Therefore, considering the specific legislation of this European country, it will have original results. Moreover, besides computing the ideal candidate for the contracting authority, my application will also rank the bidding companies according to their compatibility.

Artificial Intelligence can be used in various ways to assist the public procurement auctions process, such as analyzing large amounts of data to make predictions about bid success, automating the evaluation of bids, identifying patterns in previous auction results, and detecting fraud or misconduct. These functionalities can increase the efficiency, fairness, and transparency of the process, and stimulate suppliers into being more competitive.

A real case study is proposed where a data set of public procurement tenders in Romania is analyzed, including information about the contracting authorities, companies, and bids. The sample consists of 47,974 contracting authorities, 289,472 companies and 42,474 bids. With this data set, experiments can be conducted to evaluate the performance of machine learning-based ranking tools for the tenders, which can help to improve the efficiency, fairness and transparency of the procurement auction process and provide insights for future tenders.

The accuracy will be computed and compared for multiple intelligent algorithms, including Decision Trees, Random Forests and Extra Trees. Moreover, in an effort to obtain the best results with the machine learning algorithms, comparisons of results using different encoding types for the data, such as one-hot encoding or label encoding

will also be performed. With the help of grid search, optimal parameters for the best performing algorithms will be selected, therefore ensuring the best possible results. As the application also aims to rank more organizations as potential winners for a bid, the accuracy will be measured for the winner being amongst multiple predictions as well. Furthermore, a comparison of accuracy with a similar paper of interest on this topic will be also conducted.

1.1 Motivation

The main motivating factor behind the creation of this application is the fact that an esteemed economist from the Romanian economical sector expressed the need for such a functionality. Furthermore, as integrity is a crucial requirement for public procurement auctions in Romania, an implementation of artificial intelligence system that is not biased by human influences could be tremendously beneficial to the economic landscape. Its ability to increase transparency, fairness, and efficiency regarding tenders is of the utmost importance to economists.

1.2 Structure of the Paper

The paper begins by introducing the **Theoretical Background** in *chapter 2*, defining topics such as public procurement auctions, machine learning and software development technical details.

Following the theoretical background, the **Related Work** in the field is explored. The discussion from *chapter 3* emphasizes previous studies on predicting the success of a company in public procurement, other machine learning applications in public procurement, and provides an overview.

The main focus of the paper is on the **Bidders Recommender** system, which is detailed in *chapter 4*. This chapter covers various aspects of the system, including the methods used, the dataset employed, data description and analysis, data clustering, data preprocessing, computational experiments conducted, and the implementation of the application, as well as the testing conducted.

A **SWOT Analysis** is performed in *chapter 5*, summarizing the key insights into a matrix SWOT figure, that identifies the strengths, weaknesses, opportunities and threats of the bidders application. For each of these four categories, the particular aspects of the application are meticulously detailed.

Finally, the paper concludes with *chapter 6*, **Conclusions and Future Work Directions**, summarizing the key findings and conclusions drawn from the research presented. Furthermore, it indicates a potential avenue for enhancing the application through future endeavors.

Chapter 2

Theoretical Background

The *Theoretical Background* chapter provides a comprehensive overview of the fundamental concepts and technologies that form the foundation of this paper. It explores several key areas related to public procurement auctions and the development of an intelligent bidders recommender system.

The chapter begins by examining the concept of public procurement auctions, focusing on their significance and relevance in the context of government procurement processes. It also highlights the Romanian electronic public procurement system, shedding light on its features and functionalities.

Furthermore, the chapter delves into the importance of common procurement vocabulary (CPV) codes, which play a crucial role in classifying and categorizing procurement items. The discussion includes an exploration of how CPV codes are used in public procurement and their implications for the bidders recommender system.

Another aspect covered in this chapter is the application of intelligent algorithms. It explains the role of machine learning and data analysis techniques in developing an effective recommender system for bidders in public procurement auctions. The chapter provides an overview of the various intelligent algorithms tested in the research.

Additionally, the Theoretical Background chapter addresses the technical aspects of frontend development with React and backend development with Flask. It discusses the frameworks and technologies utilized to build the user interface and server-side functionalities of the bidders recommender system. This section provides insights into the development process and the considerations taken into account when designing an efficient and user-friendly system.

By exploring these key theoretical aspects, the Theoretical Background chapter establishes a strong foundation for the subsequent chapters, laying the groundwork for the implementation and evaluation of the bidders recommender system in the context of public procurement auctions.

2.1 Public Procurement Auctions

A public procurement auction, according to the standard dictionary definition [lica], is the administrative procedure for the purchase of goods, the provision of services or the execution of works by over 20,000 entities from the public sector in Romania.

Also named public purchases, their main objective is to offer and ensure the purchase of goods and services by the state. Suppliers can be individuals who carry out independent activities, for instance certified natural persons (PFAs), Small and medium-sized enterprises (SMEs) or large companies, as long as they meet the required conditions.

The process entails the contracting authorities publishing tender notices. The tender notice placed by the entity interested in contracting should have all the required specifications such that the bidders can submit an appropriate offer. Afterwards, a winner will be chosen from the received offers.

Considering that each public procurement auction must be made in compliance with the legal provisions and regulations set forth by the Romanian authorities, they have to be exclusively placed on the platform SEAP, as established by Law no. 98/2016 [adr] on public procurement. Therefore, all auctions are announced and organized on the Romanian public procurement platform: SEAP.

2.1.1 Romanian Electronic Public Procurement System

The Romanian Electronic Public Procurement System, also known as SEAP, is an electronic platform, used by all public authorities and institutions in Romania, that ensures the transparency of the public procurement process and procedures. The name "S.E.A.P." is an acronym that stands for the Romanian equivalent of the Electronic Public Procurement System. According to [licc], public authorities purchase the products and services required for carrying out the activity or works for the administrative unit in question electronically using the SEAP platform.

It is a centralized system that ensures transparency and fairness among the bidders, hence following the principle of transparency, which is one of the seven principles of the European legislative card that represents the pillars of the public procurement system throughout the European Union. Through the term transparency it is meant making all the necessary information to organize and attribute a public procurement auction to everyone that is interested, which is exactly what SEAP does.

The platform known as SEAP seeks to provide openness in both the public auction process and the procedures governed by the applicable law. A corporation must complete a few processes and have all of the necessary tools, such as a computer with internet connection, in order to register in the computerized public procurement system. The National Agency for Public Procurement is the organization that pioneered

the regulation of these economic activities since public procurement is conducted using public funds and necessitates a number of rigorous regulations, as described in [licb].

As for registering in the system, both the contracting authorities that will publish the purchase notices and the bidders who wish to participate in the auctions in order to win the contracts need to be registered in the SEAP.

The participants in the system are the contracting authorities and the bidders:

- **Contracting authorities**

The contracting authorities are entities that can organize auctions, meaning public institutions. The contracting authority is the one that decides what type of auction to organize, based on the estimated value of the purchase. There are a series of thresholds that frame public procurement.

- **Bidders**

The bidders are economic agents suppliers of products, works and services required in a public procurement contract.

The main objectives of SEAP are:

- Transparency and efficacy of the procurement process involving public funds,
- Simplifying the procedure for participating in the tender of suppliers,
- Efficient and standardized work procedures,
- Reducing public expenses by reducing purchase prices,
- Providing public information about public procurement processes,
- Auditing the public procurement process,
- Ensuring a highly secure and trusted environment for public funds management.

2.1.2 CPV Codes

The CPV (common procurement vocabulary) code serves as the common denominator for all SEAP announcements. CPV codes are utilized in nearly all platform contexts, as stated in [sim], including the steps that define an announcement, procedure, or product. In order to standardize the terms that contracting authorities and entities use to define public procurement contracts, the CPV Code creates a single categorization system for public procurement.

A primary vocabulary for identifying a contract's object and a secondary vocabulary for supplying additional high-quality information make up CPV. The primary

vocabulary is based on a tree structure made up of codes with a maximum of 9 digits (an 8-digit code plus a check digit) linked to a formula that describes the kind of suppliers, works, or services that are the subject of the contract.

- Divisions are identified by the first two digits (XX000000-Y)
- Groups are identified by the first three digits (XXX00000-Y)
- Classes are identified by the first four digits (XXXX0000-Y)
- Categories are identified by the first five digits (XXXXX000-Y)

Within each category, the last three digits each provide a higher level of precision. The ninth digit is used to validate the numbers before it. To fully describe what the procurement contract's object is, more vocabulary can be utilized. An alphanumeric code that correlates to a name in the supplementary vocabulary enables further explanations of the nature or precise location of the products being purchased.

The alphanumeric code includes:

- a first level composed of a letter corresponding to a section
- a second level composed of four digits, the first three representing a subdivision and the last being a check digit.

The use of the CPV is mandatory in the European Union starting from February 1, 2006. The CPV 2008 version is the current CPV version for:

1. To complete the participation notices
2. To search for business opportunities in Tenders Electronic Daily (TED)
3. To find public procurement contract notices from TED.

Contracting authorities should try to find the code that most accurately describes the intended acquisition. Although in certain situations the contracting authorities could select several codes, it is important to select only one code for the name of the announcement. If the level of precision of the CPV is insufficient, contracting authorities should refer to the division, group, class or category that best describes the intended acquisition - a more general code that can be recognized by having more zero digits.

2.2 Machine Learning

Machine learning (ML) is a subfield of artificial intelligence (AI) and computer science that utilizes data and algorithms to mimic how people learn, progressively improving its accuracy. Intelligent algorithms and data analysis are crucial parts for gaining reliable results using artificial intelligence, as further described in 2.2.1 and 2.2.2.

2.2.1 Intelligent Algorithms

Artificial intelligence (AI) refers to an intelligent system that simulates a certain form of human reasoning, knowledge, and expertise for solving one (or several) given problem(s), as argued by [LZ15]. Therefore, any algorithm based on this concept can be referred as artificial intelligent algorithm. Supervised learning is a prominent area of research in machine learning.

The key characteristic of supervised learning, as described in [CCD08], is the availability of labeled training data. This approach involves a "supervisor" providing instructions to the learning system regarding the correct labels to associate with the training examples. Typically, these labels represent class categories in classification problems. By utilizing the annotated training data, supervised learning algorithms generate models that can be used to classify unlabeled data.

1. Logistic Regression

The logistic regression model is based on the probabilities of a two-level outcome of interest, which are the odds of the event happening divided by the odds of the event not happening, according to [LaV08].

2. Decision Trees

Decision Trees are classifiers that predict class labels for data items based on their features. They use a hierarchical tree structure and make decisions at each node to assign class labels to the data items. As emphasised in [KS08], decision trees are simple, interpretable, and widely used in various scientific fields.

3. Random Forests

Random Forests are an ensemble of tree predictors that utilize random vectors to make predictions. They exhibit low error rates, are robust to noise, and provide insights into variable importance, as argued by [Bre01]. They are applicable to both classification and regression problems.

4. Extra Trees

Extra Trees, short for Extremely Randomized Trees, as described in [Die00], is an ensemble learning algorithm that combines multiple decision trees to make predictions. It randomly selects subsets of features and splits nodes using random thresholds, leading to a more diverse set of trees compared to other algorithms. The final prediction is obtained by averaging or voting among the individual tree predictions.

5. Ada Boosting

Ada Boosting, or Adaptive Boosting, is a boosting algorithm that iteratively trains weak classifiers on different subsets of the training data. As [SF12] stated, it assigns higher weights to misclassified samples in each iteration, forcing subsequent weak classifiers to focus more on these samples. The final prediction is obtained by combining the weighted predictions of all the weak classifiers.

6. K-Neighbors Classifier

K-Neighbors Classifier is a non-parametric algorithm used for classification. According to [Bis06], it assigns a new sample to the class most common among its K nearest neighbors in the training data, where K is a user-defined parameter. The distance metric used to determine proximity can vary, commonly using Euclidean distance.

7. Light GBM (Gradient Boosting Machine)

Light GBM, short for Light Gradient Boosting Machine, is a gradient boosting framework that aims to provide high performance and efficiency. According to [lig], it uses a tree-based learning algorithm similar to other boosting methods, but employs a more efficient strategy for finding the best split points during the tree construction process. Light GBM also supports various advanced features like handling large-scale datasets and parallel computing.

8. Support Vector Machines (SVM)

Additionally helpful for classification issues, a support vector machine (SVM) is a computer algorithm that learns from examples to assign labels to objects. In the words of [Nob06], it can be trained to recognize patterns or make predictions by analyzing a dataset of labeled examples.

9. Gaussian NB (Naive Bayes)

The naive Bayes classifier is a popular classification technique employed in data mining and machine learning. While it is recognized for its efficiency, it relies on the assumption of attribute conditional independence, which can be a drawback, as explained in [JT17]. To address this limitation and enhance its performance, researchers have proposed algorithms that integrate discriminative approaches and ensemble methods. These modifications aim to overcome the inherent challenges of the naive Bayes classifier and improve its effectiveness in diverse applications.

2.2.2 Data Analysis

Data Analysis in Machine Learning is the act of analyzing, cleaning, converting, and modeling data with the purpose of identifying relevant information through informing conclusions and assisting decision making, as stated in [Pri22]. Vital libraries in Python to conduct data analysis are: SciPy, Numpy, Matplotlib and Pandas. Explanatory Data Analysis is a method of studying a dataset in order to summarize its essential properties. The data must be analyzed in order to identify the distribution of datasets, select the best machine learning algorithm, obtain the correct features and examine the machine learning algorithm and sharing relevant findings.

Data Encoding

A thorough comprehension of data is required for successful analysis. Prior to doing the actual analysis, the data is encoded to help algorithms and increase efficiency, according to [PPP17].

One Hot Coding is the coding scheme that finds the most extensive application. It involves converting a single variable with n observations and d unique values into d binary variables, with each observation representing the presence (1) or absence (0) of a particular binary variable, as specified in [Lan13]. Label Encoding is a straightforward approach that assigns a unique numerical value to each distinct value in a column, as explained in [Yad21].

2.3 Software Application Development

Web Software Application Development entails creating software applications that operate over a network. It necessitates both frontend and backend development: the frontend manages user interaction, while the backend handles server-side logic and database interactions.

The combination of well esteemed frameworks such as React and Flask in order to create an application based on machine learning is a quite popular choice amongst software developers. The theoretical specific traits of these frameworks will be explained in greater detail in 2.3.1 and 2.3.2.

2.3.1 Frontend Development with React

React is a well-known JavaScript framework used for frontend development. It enables you to create user interfaces using individual pieces called components, as advertised on [Rea]. Afterwards, it easily lets you to combine them into entire screens, pages and apps. It is an innovative concept that follows a declarative approach, hence allowing

for a reusable and maintainable coding style. As it employs a declarative model, it boosts the efficiency and flexibility of applications. It generates uncomplicated views for each state in your application and competently updates and renders the correct component when your data alters. The declarative view simplifies the predictability of your code and eases debugging.

In essence, React that was built with the main purpose of displaying data in a user interface, according to [Gac15]. It is a component-centric frontend library that handles only the view portion of an MVC (Model View Controller) architecture. React promotes the creation of modular user interfaces and advocates for the development of recyclable UI (user interface) components that exhibit dynamic data. Each component in a React application takes responsibility for producing a separate, reusable chunk of HTML code. The capacity to encapsulate components within other components enables the construction of intricate applications from straightforward building elements. A component may maintain its internal state, for instance, a `TabList` component may keep a record of the open tab in its memory.

Material-UI (MUI) [mui], a prominent UI library for React, embodies the principles of Material Design. Offering an extensive selection of pre-structured inputs, including text fields, checkboxes, radio buttons, and select dropdowns, MUI facilitates easy integration into React applications. In the context of MUI, a broad array of UI tools exists, designed to streamline the introduction of new features. Users can utilize Material UI, an all-inclusive component library, as a foundation. Alternatively, it accommodates the integration of bespoke design systems with components that are primed for production use.

Axios [axi] is a prevalent library in the React ecosystem, primarily designed for dispatching asynchronous HTTP requests to REST endpoints. It is highly effective for performing CRUD (Create, Read, Update, Delete) operations within React applications, serving as a bridge for communication with the backend. Embedded within the framework, Axios supports the Promise API, promoting an elegant, readable style for handling asynchronous operations in React. With Axios, API requests are dispatched from the React application, and the ensuing data can then be leveraged across various components within the application. The popularity of Axios among the React developer community is considerable, as evident from the over 78,000 stars on its GitHub repository. Axios acts as a robust, efficient instrument in React for managing asynchronous data exchange between frontend and backend architectures.

In summary, the integration of React's input handling with Material-UI inputs simplifies form-focused web development. Axios further enhances this process by enabling efficient API calls, ensuring seamless data exchange between frontend applications and the backend.

2.3.2 Backend Development with Flask

Flask is a Python-based micro web framework. It is characterized as a microframework since it does not necessitate the usage of any specific tools or libraries. It is lightweight and flexible, widely recognized for its simplicity and ease of use, according to [Gri18]. It allows developers to quickly build APIs for web applications using Python's familiar syntax. Thus, this framework offers routing and request handling in a simplified and intuitive manner.

Furthermore, Flask has a vibrant and supportive community [dev] that has contributed numerous extensions, libraries, and resources. These extensions provide additional functionality for tasks like authentication, database integration, and form handling, enabling developers to leverage existing solutions and reduce development time.

Contemporary web frameworks, such as Flask, employ the routing mechanism, a valuable tool aiding users in recalling application URLs. This technique facilitates direct access to the desired page, bypassing the need to navigate from the homepage. Within the Flask framework, the `route()` decorator functions as the binding element that connects a URL to a specific function, such as `@app.route()`, illustrating the theoretical intersection of user interface design and efficient web navigation.

Flask is especially useful in the context of applications based on artificial intelligence as it seamlessly enables for the use of pre-trained machine learning models. The Flask backend can read and run a machine learning method, which is often saved in a file that contains either the trained model or the necessary parameters to compute predictions. This machine learning integration in a Flask application serves as the foundation for intelligent and data-driven functionality.

Aside from algorithm integration, Flask provides robust capabilities for data preparation, which is an important step in preparing data for machine learning algorithms. Within the Flask context, cleaning data, dealing with missing values, scaling features, and encoding categorical variables are all easy to incorporate operations.

The usage of JSON (JavaScript Object Notation) format for data return is paramount for seamless interaction between backend and frontend applications, a concept supported by [som19]. JSON offers a universally accepted, lightweight, and human-readable framework for data representation. Thus, the adoption of JSON paves the way for straightforward data consumption, enhances compatibility, and ensures efficient amalgamation of frontend and backend technologies.

To encapsulate, Flask stands as a lightweight and adaptable web framework for Python that distinguishes itself with its simple and user-friendly nature. Its approach paired with an intuitive API makes it a prime candidate for the construction of web applications and APIs. Flask enables developers to initiate their projects swiftly and customize as per their needs.

Chapter 3

Related Work

Because of its economic importance and risk of corruption, the topic of public procurement auctions has received a lot of attention in recent years. However, the use of Machine Learning in public procurement is largely unexplored. Despite minimal study in this arena, promising results indicate that machine learning could revolutionize the world of public procurement auctions.

The most relevant articles found based on this field of study are "Bidders recommender for public procurement auctions using machine learning" by Rodríguez et al. (2020) [GRMBP⁺22], "Collusion detection in public procurement auctions with machine learning algorithms" by Rodríguez et al. (2022) [GRRMOFVB20], "Application of neuro-fuzzy system for predicting the success of a company in public procurement" by Pamučar et al. (2022) [PBPM22], and "A rank-and-compare algorithm to detect abnormally low bids in procurement auctions" by Conti et al. (2012) [CDN12].

3.1 Predicting the Success of a Organization in Public Procurement

In [GRRMOFVB20] it is presented a machine learning-based approach to recommend bidders for public procurement auctions, with the aim of increasing transparency and competition in the bidding process. They use an algorithm to forecast which bidders will have the greatest chance of winning the auction after analyzing data from public auctions in Spain.

In this study, the intelligent algorithm used is the random forest algorithm. The outcomes of the case study proved to be positive: the victorious bidding company falls within the suggested group of companies, ranging from 24% to 38% of the tenders, contingent on varying trial conditions and scenarios.

However, one of the limitations of this study is the fact that there is no ranking of recommended companies. Thus, the organizations do not have an associated

probability for winning.

Furthermore, in [PBPM22] a neuro-fuzzy system for appraising and forecasting a construction firm's success in public tenders is discussed. This model allows businesses to maintain sustainability by examining their own standing in the market.

The model is derived from data collected over a seven-year research period. Data from the first six years were utilized to refine the model, while the data from the final year were employed for testing and verification. The neuro-fuzzy model was adjusted using the Artificial Bee Colony algorithm. Out of a cumulative 239 tests, in 89% of the scenarios, the anticipated ranking from the neuro-fuzzy model perfectly matched the actual rank achieved in the public tender.

3.2 Other Machine Learning Approaches on Public Procurement

Identifying unusually low proposals in procurement auctions is a known issue, as their approval may result in the winning bidder failing to deliver the service or task conferred by the auction, posing a significant risk for the auction manager. A rank-and-compare algorithm, as detailed in [CDN12], is utilized to detect such outlier bids and assist auction managers in forming an effective denial decision.

Formulaic interpretations and trial results are offered for the probability of detection and the likelihood of false alarms. The proposed span of usage for the detection algorithm excludes situations with numerous bidders (exceeding 20) and highly varied bids (coefficient of variation exceeding 0.15). An upsurge in the count of bidders results in opposing outcomes, as it reduces both the false alarm probability and the detection probability. If the bids are dispersed widely, it results in a doubly detrimental impact, with increased false alarms and reduced detections. The existence of multiple outlier bids also deteriorates the performance of the algorithm. Nevertheless, the approach is fairly resilient to the existence of courtesy bids.

Artificial intelligence algorithms have found applications in various fields, including the analysis of economic data, particularly in the context of public procurement auctions. One notable area of focus is the detection of collusion, a prevalent issue observed globally in public sector procurement. Collusion entails the illicit collaboration among competing companies, wherein they conspire to submit predetermined bids for upcoming procurement opportunities.

In [GRMBP⁺22], there are tested eleven Machine Learning algorithms in order to detect collusion in public procurement auctions. Although the utilization of machine learning (ML) in public procurement has yet to be extensively explored, its potential for identifying collusion is highly promising. ML algorithms, while requiring a

significant amount of historical auction data for calibration, exhibit remarkable flexibility and can yield meaningful detection rates even with limited information. Their information-intensive nature positions ML algorithms as valuable tools in the detection and prevention of collusion in public procurement processes.

Summarizing the graphical results from [GRMBP⁺22], we can conclude that the top three best performing intelligent algorithms were

1. Extra Trees Classifier,
2. Random Forest Classifier,
3. Ada Boost Classifier.

Hence, based on the aforementioned studies, it is derived that these algorithms are undoubtedly worth considering for future artificially intelligent enhanced economical endeavours.

3.3 Overview upon Related Work

Taking the aforementioned accuracy percentages into consideration, as conducted by [GRRMOFVB20] and [GRMBP⁺22], we can conclude that the Random Forest algorithm is shown to be a reliable intelligent algorithm given economical data, specifically public procurement auctions. Moreover, the Extra Trees Classifier also shows great promise, being the best ranked algorithm by [GRMBP⁺22]. The results from [PBPM22] are also impressive, proving the neuro-fuzzy model to be very effective as well.

The limitations of [GRRMOFVB20] are the lack of ranking, as well as the fact that only one approach of intelligent algorithms was experimented with in the paper. Furthermore, the only data sets used were from Spain tenders, the regulations regarding public procurement auctions being slightly different among countries. Hence, the application is specifically tailored to tenders from Spain, as proven by the computational experiments, and it is unknown what tweaks should be made to accommodate different economical characteristics and how it would perform on others datasets.

Therefore, the related work provides a strong basis for further exploring of the application of artificial intelligence in the economical realm of public procurement auctions. It lays a strong foundation for us to continue looking into how machine learning can be used in the economic field of public procurement auctions. These studies show us that machine learning and mixed models like the neuro-fuzzy system can really help improve prediction and decision-making in this complex area. Hence, a good starting point for more research and development in this area is provided.

Chapter 4

Bidders Recommender Application

Identifying suitable bids for procurement projects in today's dynamic and competitive business market can be a hard and time-consuming task. The importance of efficient and accurate bidder recommendation systems in facilitating informed decision-making has grown. In response to this demand, the Bidders Recommender Application has been developed, combining advanced machine learning algorithms with intuitive user interfaces to assist economists in selecting the most suitable companies for bidding on a given project.

This chapter focuses on the methods employed in the development of the Bidders Recommender Application. The process begins with a thorough examination and analysis of the given data, followed by preprocessing methods to increase the data's relevance. Afterwards, the data is encoded such that the multitude of machine learning algorithms tried can process it. Building upon the chosen machine learning model, the development process extends to the creation of a lightweight backend using Flask. The backend is responsible for integrating the pretrained model, enabling real-time prediction capabilities, and handling requests from the frontend. Simultaneously, a user-friendly frontend is developed using React, providing economists with a seamless interface to interact with the algorithm and input the required characteristics for bid prediction.

The Bidders Recommender Application's integration of advanced machine learning algorithms with a user-friendly interface seeks to streamline the process of selecting appropriate bidders for procurement projects. Economists can make better informed decisions by utilizing the power of data-driven decision-making, boosting the efficiency and efficacy of the bidding process. This chapter explains the details of each step, from the methodologies employed and the rationale behind their selection, to details regarding the data set, its analysis, preprocessing and encoding, as well as the experiments conducted, implementation details and application testing.

4.1 Methods

To achieve optimal performance, multiple machine learning techniques are studied and tested utilizing preprocessed and encoded data. This iterative process involves training and testing multiple models, mainly considering their predictive accuracy. Through rigorous experimentation, the best-performing algorithm is identified, which forms the backbone of the Bidders Recommender Application.

The methods tried to solve this problem are Decision Tree Classifier, Random Forest Classifier, Gaussian NB (Naive Bayes), Logistic Regression, Support Vector Machines (SVM), Ada Boost, K-Neighbors Classifier, Light GMB (Gradient Boosting Machine), and Extra Trees Classifier.

The template used to compute the experiments in the effort to find the best performing algorithm for this task outlines a common workflow for building and evaluating a machine learning model, as can be seen in the code below 1:

```
1  # 1. Import necessary libraries and modules
2  < . . . >
3
4  # 2. Load and preprocess data
5  data = pd.read_csv(<...file...>)
6  X = data.drop('organizationId', axis=1)
7  y = data['organizationId']
8
9  # 3. Split the data into training and testing sets
10 X_train, X_test, y_train, y_test =
11 train_test_split(X, y, test_size=0.2, random_state=42)
12
13 # 4. Define the model
14 clf = <...Classifier...>(<...parameters...>)
15
16 # 5. Train the model on the training data
17 clf.fit(X_train, y_train)
18
19 # 6. Evaluate the model on the testing data
20 < . . . >
21
22 # 7. Print the performance metrics
23 < . . . >
```

Listing 1: Code representing the template to use a ML algorithm on the data.

All of the necessary libraries and modules that are used throughout the code are included in the beginning, for instance the *pandas* library, as it is always used irrespective of the ML algorithm, as can be seen on line 5. Afterwards, the already preprocessed and encoded data is loaded from the CSV file, and the input features (X) and the target variable (y) are defined. The *organizationId* column is dropped from the input features, as it is the target variable. Furthermore, the data is divided once again into X_{train} , X_{test} , y_{train} , and y_{test} , considering 80% of the data for training and 20% of the data for testing. From this point on, the algorithm gets personalized with the specific model that is tested, with its specific set of parameters. Then, it is trained using the fit method, which takes the input features (X_{train}) and the corresponding target variable (y_{train}) as arguments. This step allows the model to learn from the training data and adjust its internal parameters. Lastly, but not least, some performance metrics are performed and then printed in order to compare the algorithms amongst themselves, the accuracy being the most relevant method in this context.

The supervised learning algorithms that have been used in order to find the best fit for the problematic of the bidders recommender system are, as previously defined in 2.2:

1. Logistic Regression

As stated in 1., logistic regression is a preferred algorithm for issues involving binary classification. In the context of the bidders recommender application, the event that should happen can be defined by the organization winning the bid.

$$\text{Odds} = \frac{\text{Probability of organization winning the bid}}{\text{Probability of organization not winning the bid}}$$

The probability that a bidder will obtain a tender can be predicted using the logistic regression algorithm based on a variety of independent variables, which in this case could rely on the numeric CPV code and the previous winning organizations.

2. Decision Trees

In the context of predicting organizations for bids based on bid characteristics, decision trees can be used effectively. By utilizing the features of a bid, such as its specifications, most importantly the CPV code, decision trees can classify and predict the suitable organization or category for that particular bid. The decision tree algorithm analyzes the bid's characteristics at each node and makes decisions based on specific criteria, ultimately assigning the appropriate organization or category label to the bid. Decision trees are particularly valuable in this context due to their simplicity, interpretability, and ability to handle categorical variables,

as described in 2., making them well-suited for predicting organizations based on bid characteristics.

3. **Random Forests**

Random Forests can be applied to the bidders app to predict suitable organizations for bids based on bid characteristics. By utilizing an ensemble of decision trees, Random Forests combine multiple predictions to provide accurate results. They are particularly useful in handling noisy data and can effectively handle both classification and regression problems, as previously mentioned in 3.. Random Forests can be used in the bidders app to identify and recommend organizations that are most likely to be a good fit for specific bids, considering the bid characteristics as input, especially the CPV codes.

4. **Extra Trees**

In the context of the bidders recommender app, Extra Trees can be employed to predict suitable organizations for bids based on bid characteristics, the most relevant one being the CPV code. By leveraging the ensemble of Extra Trees described in 4., the app can benefit from improved accuracy and robustness in recommending organizations. The algorithm's randomization process ensures a diverse set of predictions, enhancing the app's ability to handle different bid scenarios effectively.

5. **Ada Boosting**

AdaBoosting, also known as Adaptive Boosting as it was already mentioned in 5., can be highly relevant in the context of the bidders recommender app. This boosting algorithm can improve the accuracy and performance of the recommendation system by iteratively training weak classifiers on different subsets of the training data. In the context of the bidders recommender app, AdaBoosting can enhance the recommendation system by iteratively learning from the bid data. By assigning higher weights to misclassified bids or challenging cases, AdaBoosting can focus more on improving the accuracy of recommendations for those specific bids. This adaptive approach helps to identify and address potential challenges or complexities in the bidding process.

6. **K-Neighbors Classifier**

The K-Neighbors Classifier is a relevant algorithm for the bidders recommender app, as it can assist in classifying bids based on their similarity to known training data. This non-parametric algorithm is specifically designed for classification tasks. In the context of the bidders recommender app, the K-Neighbors Classifier

can be used to recommend organizations for bids by comparing the bid characteristics to the known training data. The algorithm finds the K most similar bids and assigns the most prevalent class label (organization) among those neighbors to the new bid, process generically defined in 6..

7. **Light GMB (Gradient Boosting Machine)**

The performance and efficiency of Light GBM make it a great choice for the bidders recommender app. By leveraging its tree-based learning algorithm and efficient split point selection strategy, as described in 7., Light GBM can effectively handle the bid data and hopefully make accurate predictions. Its support for large-scale datasets and parallel computing further enhances its capabilities, enabling faster processing and better scalability. Hence, it is a promising option for the problematic at hand.

8. **Support Vector Machines (SVM)**

A Support Vector Machine (SVM) is a great choice for the bidders app's recommender system for matching organizations to bids based on CPV codes. According to 8., it can effectively handle non-linear relationships, learn from labeled examples, and find optimal decision boundaries. SVMs provide accurate predictions, robustness to noise, and can handle high-dimensional data, making them well-suited for recommending organizations based on CPV codes in the bidders app.

9. **Gaussian NB (Naive Bayes)**

The naive Bayes classifier is a great choice for the bidders app's recommender system of organizations to bids based on CPV codes. It offers computational efficiency, handles categorical features effectively, and can handle high-dimensional data. Despite its assumption of attribute conditional independence, a drawback mentioned in 9., it is easy to implement and interpret, making it an option that is worth trying out for the scope of predicting winning companies for given bids.

4.2 Data set

The dataset utilized for this study is made up of three tables, each containing useful information about the public procurement procedures in Romania. These tables provide information about various traits of the procurement process, such as bidding procedure features, winning organizations mapping, and contracting authorities mapping.

- **1st Table:** Public Procurement Auction Announcements

Firstly, we have data represented in CSV format regarding Public Procurement Auction Announcements, which can be visually represented in the form of a table:

1	noticeNo
2	contractValue
3	title
4	currencyCode
5	publicationDate
6	countyCode
7	contractingAuthorityName
8	cpvCodeName
9	cpvCode
10	cpvCodeType
11	cNoticeNo
12	cNoticeEstimatedContractValue
13	cNoticePublicationDate
14	cNoticeTitle
15	organizationName
16	organizationId

Table 4.1: Public Procurement Auction Announcements

The first table 4.1 consists of 42,475 public procurement procedures and is defined by 16 distinct characteristics. This table is the starting point for assessing and comprehending the procurement problematic.

- **2nd Table:** Contracting Authorities

Secondly, we have data represented in CSV format regarding Contracting Authorities, which can be visually represented in the form of a table:

1	contractingAuthorityId
2	name
3	CUI

Table 4.2: Contracting Authorities

The second table 4.2 contains a one-to-one mapping of the existing winners' names across the entire production dataset to the internal ID of Spend.ro. With

289,473 entries, this mapping facilitates the identification of winning bidders in the dataset. The internal ID of Spend.ro can be found in the organizationID column. Additionally, where available, the company's tax identification number (CUI) is provided, with or without the "RO" prefix, depending on the company's VAT status.

- **3rd Table:** Bidding Companies

Thirdly, we have data represented in CSV format regarding Bidding Companies, which can be visually represented in the form of a table:

1	organizationId
2	name
3	CUI

Table 4.3: Bidding Companies

The third table 4.3 provides a one-to-one mapping of the existing contracting authorities' names in the production dataset to the internal ID of Spend.ro. This mapping encompasses 47,945 entries and enables the identification of contracting authorities involved in the procurement procedures. The internal ID of Spend.ro can be found in the contractingAuthorityId column. Furthermore, if an official identifier of the authority (CUI) exists, it is also included, with or without the "RO" prefix, depending on the entity's status, including some entities with commercial status.

4.2.1 Data Description

- **1st Table:** Public Procurement Auction Announcements

The data presented in 4.1 contains the following fields:

- a) **noticeNo** - the number of the award announcement related to the purchase in SEAP (helps to identify easier procedures for various analysis flows)
- b) **contractValue** - the contractual value introduced by the contracting authority in the award notice
- c) **title** - the title given by the contracting authority of the procedure in the award announcement (text description of the main object of the procedure)
- d) **currencyCode** - procedure currency (RON, EUR, USD)
- e) **publicationDate** - Date of publication publication
- f) **countyCode** - the county of the contracting authority

- g) **contractingAuthorityName** - the name of the contracting authority, as it appears in SEAP
 - h) **cpvCodeName** - the name of the COMMON CODE PROCUREMENT VOCABULUY (CPV) used in the procedure (e.g. afforestation services)
 - i) **cpvCode** - Numeric Code of CPV (cf. European nomenclature, e.g. 77231600-4)
 - j) **cpvCodeType** - only works procedures (code 2) and services (code 3), not product supply, were selected, the supply is less relevant to the context of the project
 - k) **cNoticeNo** - the number of the announcement of participation related to the award announcement above (helps to identify easier procedures)
 - l) **cNoticeEstimatedContractValue** - the estimated value of the contract in the participation announcement (the award results resulting from the competition between the bidders)
 - m) **cNoticePublicationDate** - the date of publication of the participation announcement related to the award announce
 - n) **cNoticeTitle** - the title given by the contracting authority of the procedure in the participation announcement (may be more object / descriptive than that put in the award announcement) 0
 - o) **organizationName** - the winner or winners of the procedure (can be consorti)
 - p) **organization** - the Code or the internal numeric codes of Spend.ro for winners (the same winner can have several codes)
- **2nd Table: Contracting Authorities**

The data presented in 4.2 contains the following fields:

- a) **contractingAuthorityId** - the internal numeric code of Spend.ro for contracting authorities
- b) **name** - the name of the contracting authority, as it appears in SEAP
- c) **CUI** - official identifier of the authority ('CUI'), with or without RO in front, depending on the status of the entity - some may have a commercial regime

- **3rd Table: Bidding Companies**

The data presented in 4.3 contains the following fields:

- a) **organizationId** - the internal numeric code of Spend.ro for organizations

- b) **name** - the name of the organization, as it appears in SEAP
- c) **CUI** - official identifier of the organization ('CUI'), with or without RO in front, depending on the status of the entity - some may have a commercial regime

4.2.2 Data Analysis

Data analysis is a crucial step in machine learning as it plays a fundamental role in understanding, preprocessing, and extracting meaningful insights from the available data. Before training a machine learning model, it is essential to thoroughly analyze the dataset to gain insights into its characteristics, identify patterns, and assess its quality. Data analysis helps in understanding the distribution of features, detecting outliers, handling missing values, and performing data transformations if necessary.

Average Bids per Contracting Authority	Average Wins per Company
17.58	2.25

Table 4.4: Average Bids and Wins in Public Procurement Auctions

On average, as shown in 4.4 each contracting authority creates 17.58 bids, and each company wins 2.25 of the public procurement auctions.

CUI, a Romanian economical terminology, stands for "Unique Registration Code". It serves as a unique identifier for tax and administrative purposes. VAT stands for Value Added Tax, and only the authorities with their CUI starting with the prefix "RO" are considered tax payers.

Category	Number	Percentage
Authorities	47,975	100%
CUI Authorities	30,576	63.74%
VAT Authorities	2,485	5.18%

Table 4.5: Authorities Statistics

Table 4.5 highlights the distribution of authorities within different categories and provides insights into the relative proportions of CUI authorities and VAT authorities compared to the total number of authorities.

In respect to validating the CUI, the algorithm from [Con20] has been used.

Limited Liability Companies (LLC), a type of business entities that are permitted under state laws [LLC23], are distinguished in Romanian by the keyword "SRL" or "S.R.L." included in their names, according to [Con23].

Category	Number	Percentage
Organizations	289,473	100%
CUI Organizations	238,167	82.34%
Valid CUI Organizations	230,770	79.72%
SRL Organizations	168,860	58.33%
VAT Organizations	110,799	38.27%

Table 4.6: Organizations Statistics

The findings in table 4.6 shed light on how organizations are distributed across various categories and offer insights into the relative representation of CUI organizations, valid CUI organizations, SRL organizations, and VAT organizations in relation to the total number of organizations.

The two most important numerical features of the bids data are the contract value and the estimated contract value. A comparison of their description in terms of various statistics (mean, median, percentiles, standard deviation, minimum and maximum) is made in table 4.7. Unfortunately, almost half of the data does not have a value for the estimated contract value, more precisely 41.89% of the bids. Even if for the estimated contract value only the relevant 58.11% of the bids are considered, for contract value there are considered all of the bids. All the represented values are in the RON currency.

Statistic	Contract Value	Estimated Contract Value
Mean	316,491,010.93	5,945,641.58
Median	289,964.26	524,193.54
25th Percentile	55,000.00	61,220.00
50th Percentile	289,964.26	524,193.54
75th Percentile	1,392,072.00	2,800,000.00
Standard Deviation	63,784,933,478.50	77,540,599.89
Maximum	13,145,876,360,789	4,401,990,443
Minimum	0.01	0.01

Table 4.7: Contract Value and Estimated Contract Value Statistics

Contract Value

- The mean contract value is 316,491,010 RON, indicating the average of contracts.
- The median contract value is 289,964 RON, which represents the middle value in the sorted bid contract values.
- The 25th percentile contract value is 55,000 RON, meaning that 25% of the bid contract values are below this amount.

- The 75th percentile contract value is 1,392,072 RON, indicating that 75% of the bid contract values are below this amount.
- The standard deviation of the bid contract values is 63,784,933,478 RON, indicating a high degree of variability in the data.
- The maximum contract value is 13,145,876,360,789 RON, which represents the highest value observed.
- The minimum bid contract value is 0.01 RON, indicating that even small contract values are present.

Estimated Contract Value

- The mean estimated contract value is 5,945,641 RON, representing the average estimation of bid contract values.
- The median in this case is 524,193 RON, indicating the middle estimation value.
- The 25th percentile estimated contract value is 61,220 RON, meaning that 25% of the estimations are below this amount.
- The 75th percentile estimated contract value is 2,800,000 RON, indicating that 75% of the estimations are below this amount.
- The standard deviation of the estimated contract values is 77,540,599 RON, indicating a significant variation in the estimations.
- The maximum estimated contract value is 4,401,990,443 RON, which represents the highest estimated value for a bid.
- The minimum value is 0.01 RON, alike the minimum contract value.

The useful insights drawn from 4.7 can be used to understand the distribution and variability of the data.

Currency Code

In the case of 'currencyCode', a data transformation is necessary in order to normalize the values. Therefore, a currency has to be chosen and all other bid values that are not in the respective currency have to be exchanged. In order to choose the ideal currency, we firstly needed to find what currencies we have among the dataset, and the findings concluded there are only three options:

- RON (the currency code for the Romanian Leu)

- USD (the currency code for the United States Dollar)
- EUR (the currency code for the Euro).

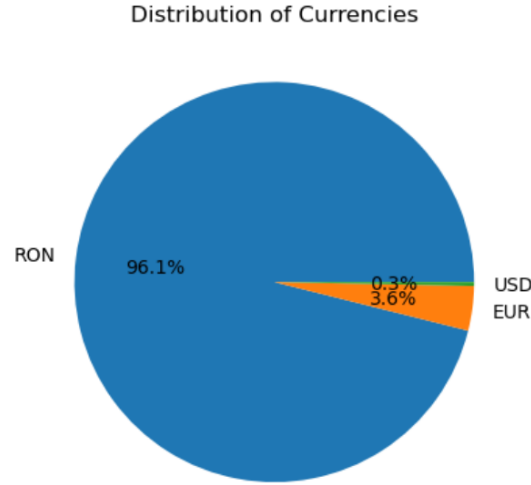


Figure 4.1: Graph representing the distribution of currencies among the bids.

Considering the graphical representation represented in 4.1, of the distribution of these three currency codes, it makes most sense to convert all the bids to RON, as it has the largest distribution and it is the national currency of Romania, country for which this study is most relevant.

As the exchange rate is different nowadays compared to when the bids were published and won, we need to take the date when the bids were placed into consideration.

In order to standardize the currency codes used in the dataset, a currency conversion is performed using the `forex.python` library. This library provides functionality to retrieve exchange rates between different currencies.

The conversion process begins by fetching the exchange rate for a specific date between the original currency code specified in each bid and RON. This exchange rate is obtained using the `CurrencyRates` class from the `forex.python.converter` module.

County Codes

Geographical positioning is evidently an important aspect in choosing a winner for a bid, especially because many companies only apply to bids that are in their region. Therefore, first of all it was relevant to check if there are counties with no bids posted at all, but this was not the case. There are 42 different instances of county codes in the dataset, representing all the 41 counties from Romania and the municipality of Bucharest.

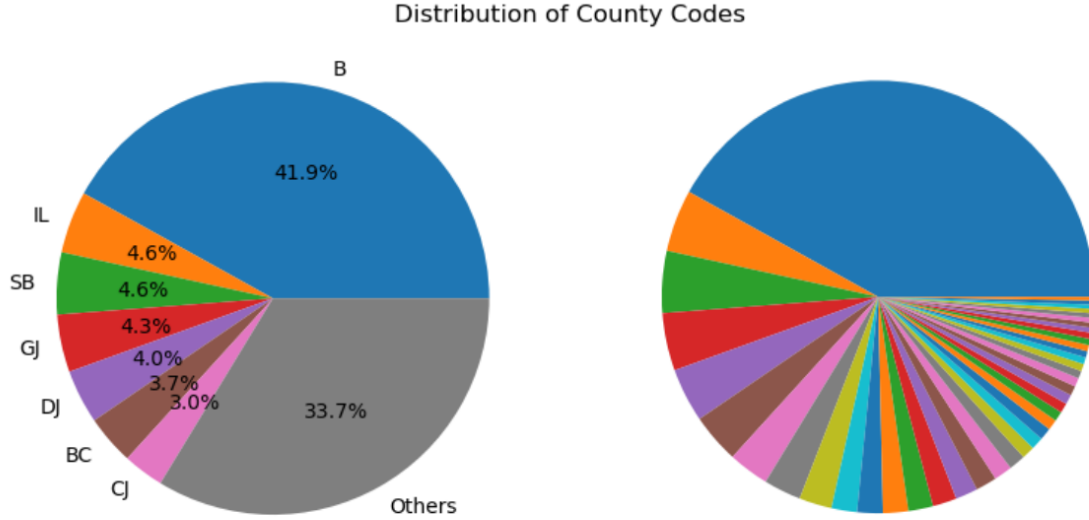


Figure 4.2: Graph representing the distribution of counties among the bids.

The distribution of the data in terms of county codes can be seen in graph 4.2. As expected, the county code "B" which stands for the municipality of Bucharest had the most published public procurement auctions, making up for almost half of all the bids. Besides the capital, the rest of the counties share from 0.26% to 4.6% of the bids, with the county code "GR", standing for Giurgiu, being the one with the least bids, only 0.26%. Thus, the data is diverse enough to take into account all the counties from Romania.

Feature Importance

According to the client, a specialist in the field of economics, the most important feature is the CPV code. Moreover, another contextual information that was stated to be relevant by him are the initially estimated value of the contract, the actual price of the winning company, the county of the contracting authority and the CPV code type.

Also, it is worth mentioning that some contracting authorities choose profound CPV codes, going into great detail regarding the specific nature of the bid, whilst others choose more general CPV codes, perhaps even for the same activity. Therefore, I have split the CPV codes into divisions, groups, classes and categories, as they were previously explained in the chapter regarding CPV codes 2.1.2.

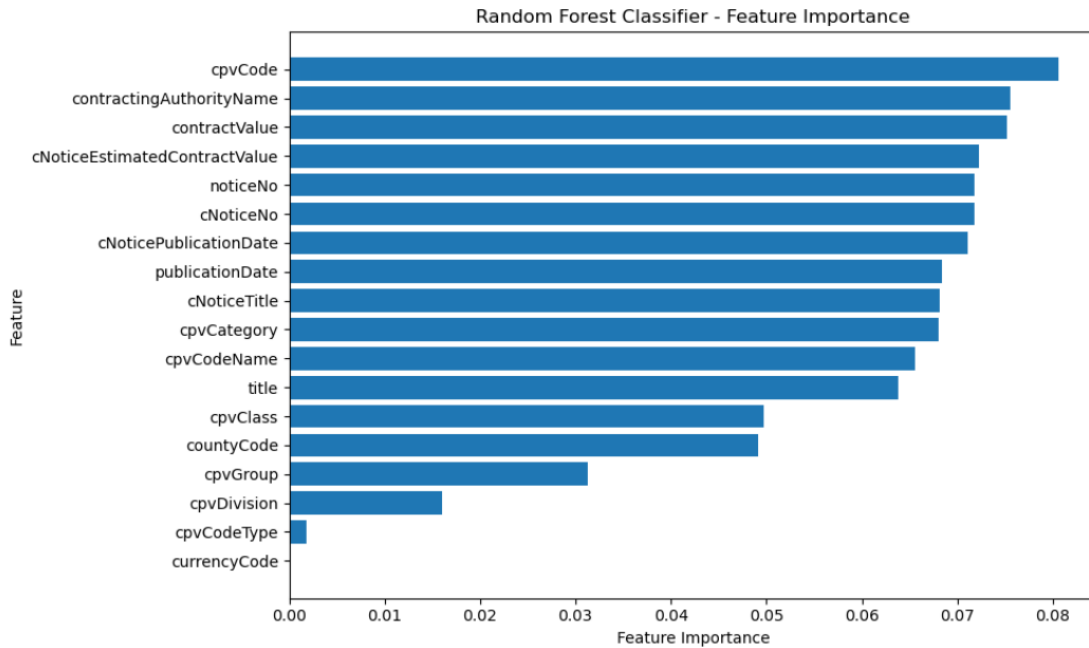


Figure 4.3: Graph representing the importance of the features used in the ML algorithm.

After conducting feature importance analysis, as seen in 4.3, using a random forest classifier to identify the most significant features, the premise that the CPV code is the most important feature was validated. However, the idea that the CPV code type would provide useful insight for this machine learning model was invalidated, being ranked almost the last one, only followed by the currency code column. This last column was obviously not going to be relevant anymore, as all the data has been normalized and converted in the RON currency, so it is logical that it gives the model nothing to learn from.

Out of the newly created columns, divisions, groups, classes and categories, the CPV categories proved themselves to be the most relevant ones, thus they were kept as features in the model. The others, divisions, groups, classes, were too generic and did not provide any new information, and could potentially lead to over-fitting, hence they were dropped.

Even though ranked rather badly by the feature importance analysis, the county code was kept at the advice of an economist due to its importance regarding the concept of where certain companies usually operate.

4.2.3 Data Clustering

There are two CPV code types in this analysis: services and works. Thus, it would make sense to try data clustering with two clusters. However, as it can be seen in the graph 4.4, the elbow method was still performed in order to validate this theory and assure the optional number of clusters to be used.

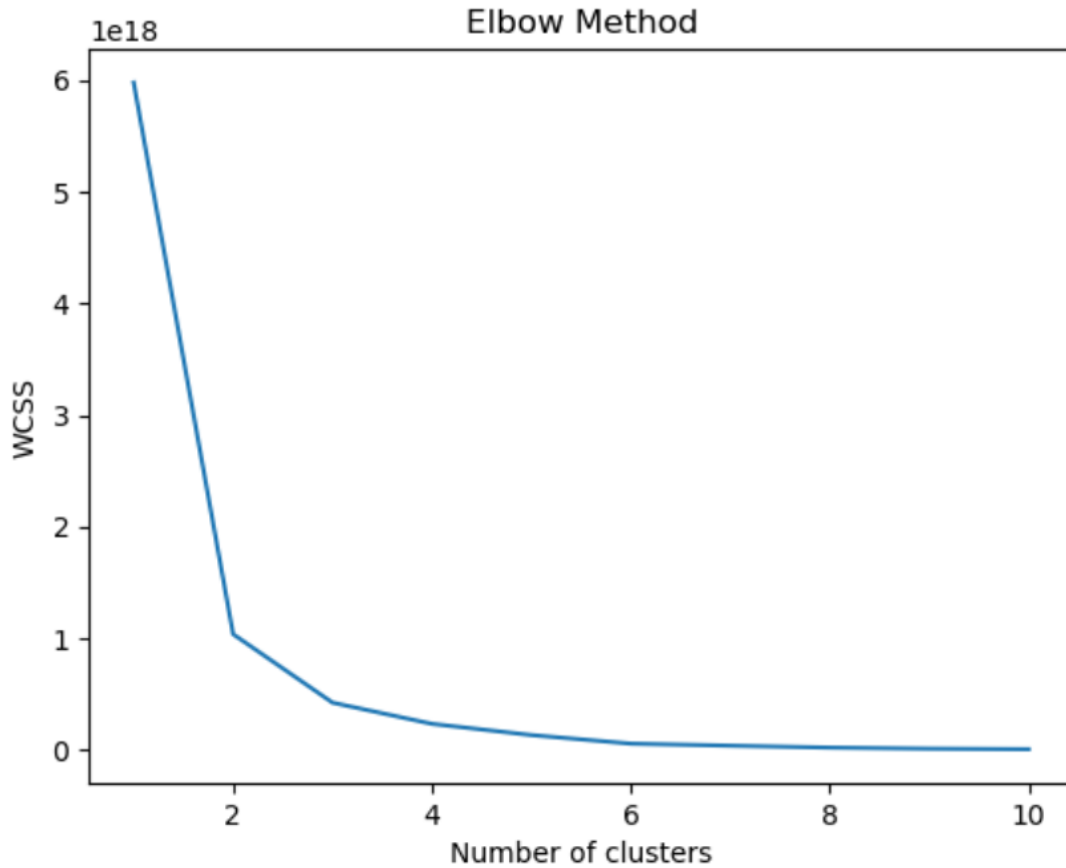


Figure 4.4: Graph representing the elbow method to find the ideal number of clusters.

Based on the result of the graph 4.4, the elbow method suggests that either two or three clusters could be considered as potential choices for the ideal number of clusters. As stated above based on the economical meaning of the CPV code types, two is the preferred choice. Furthermore, three clusters could also make sense, representing the two CPV code types, as well as perhaps a category that represents a mixture of organizations from the first two clusters, that could perform well in both sectors, services and works.

Data Clustering with Two Clusters

Metric	Value
Gini Index	0.4164420475061019
Purity	0.010186248564438009
Silhouette Score	0.3581224757953581

Table 4.8: Clustering Metrics for 2 Clusters

Gini Index: The Gini index measures the inequality or impurity within the data-set. With two clusters, the Gini index seen in 4.8, of 0.4164 suggests a moderate level of impurity or inequality in the distribution of the winning companies across the clusters. This means that the two clusters may not be equally balanced in terms of the number of companies or their characteristics.

Purity: Purity, in this context, indicates the proportion of instances (winning companies) belonging to the most dominant cluster. With a purity value of 0.0101, noted in 4.8, it suggests that the majority of instances are not concentrated in a single cluster. This could mean that the two clusters have a relatively even distribution of winning companies or that there are no clear dominant clusters.

Silhouette Score: The silhouette score measures how well-defined the clusters are, considering both the cohesion within clusters and the separation between clusters. With a silhouette score of 0.3581, as seen in 4.8, it indicates that the clusters have a reasonable separation, suggesting that the winning companies within each cluster are more similar to each other than to those in other clusters. However, it's worth noting that the score is not extremely high, indicating that there may still be some degree of overlap or ambiguity in the clustering results.

Overall, based on these metrics seen in 4.8, it seems that the clustering analysis has resulted in moderately impure clusters, with a relatively even distribution of winning companies and reasonable separation between the clusters.

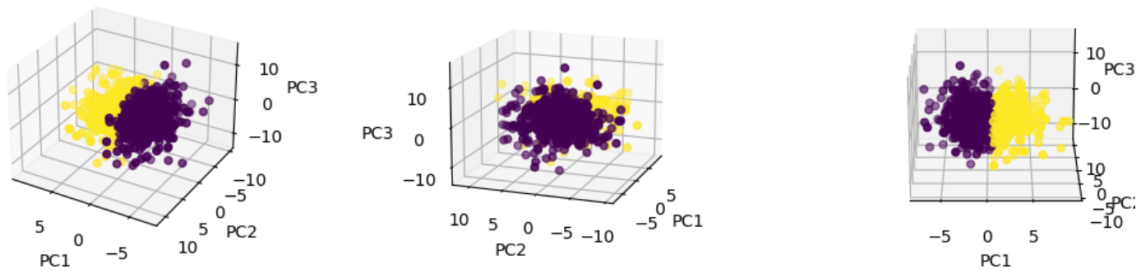


Figure 4.5: Graph representing data clustering with 2 clusters.

Data Clustering with Three Clusters

Metric	Value
Gini Index	0.6333356395190954
Purity	0.012882608478553952
Silhouette Score	0.2282416400777638

Table 4.9: Clustering Metrics with 3 clusters

Gini Index: The Gini index measures the inequality or impurity within the dataset. With a Gini index of 0.6333 as seen in 4.9, it suggests a relatively high level of impurity or inequality in the distribution of the winning companies across the clusters. This indicates that the three clusters may not be well-separated or distinct from each other, and there is a significant degree of mixing or overlap between the clusters.

Purity: Purity measures the proportion of instances (winning companies) belonging to the most dominant cluster. With a purity value of 0.0128 from 4.9, it indicates that the majority of instances are not concentrated in a single cluster. This further supports the notion that the three clusters have a relatively even distribution of winning companies, and there may not be a clear dominant cluster.

Silhouette Score: The silhouette score measures how well-defined the clusters are, considering both the cohesion within clusters and the separation between clusters. A silhouette score of 0.2282, as seen in 4.9, suggests a relatively low separation between the clusters. It indicates that the clusters may have a considerable overlap or ambiguity, and the instances within each cluster might not be distinctly different from those in other clusters.

Based on these metrics from 4.9, it seems that the clustering analysis with 3 clusters might not have produced well-separated or highly distinct clusters. The third cluster could be a mix or overlap of the first and second clusters, representing the instances that are more questionable or difficult to assign to a specific cluster.

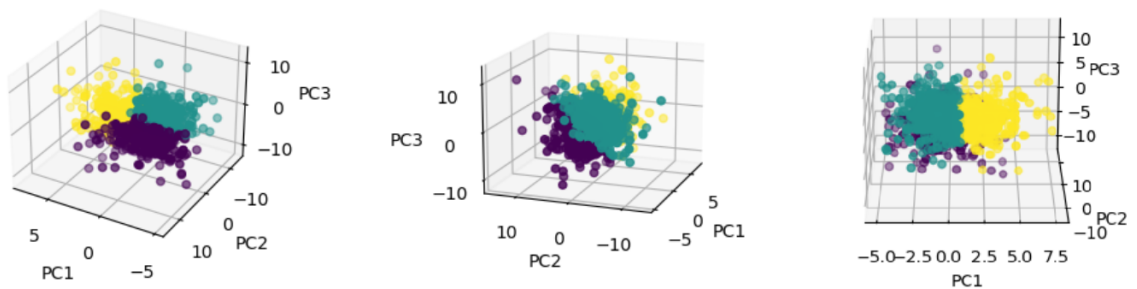


Figure 4.6: Graph representing data clustering with 3 clusters.

4.2.4 Data Preprocessing

Any data analysis or machine learning project must include data preprocessing because it helps to ensure the accuracy, reliability, and suitability of the data for further analysis. Data preprocessing is carried out in this case to address various data issues and improve the dataset's usability.

The preprocessing handled situations where multiple values were combined within a single row by splitting and restructuring rows, enabling readability for the machine learning algorithm. Currency conversion removes differences brought on by different currency formats, allowing for fair comparisons. Rows with zero contract values were removed to prevent bias or distortion in later analyses. Furthermore, actions like eliminating pointless columns and encoding categorical variables increased data efficiency and the compatibility with machine learning algorithms.

Data Filtering

In the data filtering section, the focus is on why certain rows have been filtered out of the data set due to inconsistencies. We start with an initial number of 42,475 rows and 16 columns in the csv data file, which we convert into a pandas data frame.

Firstly, an abnormality is having a row with missing or NaN (Not a Number) value, where one should be. Therefore, we want to drop all such rows from our data frame. This will be performed as depicted in listing 2.

```
1 bids.dropna(inplace=True)
```

Listing 2: Code representing Data Filtering for missing / NaN values.

The change from code snippet 2 let to a significant decrease in the number of total rows in the data set, from the initial 42,475 to almost half, 24,679 rows.

Having the value "0" for the contract value or for the estimated contract value is considered an abnormality, and thus such rows should be filtered out of the data set, as presented in listing 3. There are 1431 rows with 0 as the contract value and 3863 rows with the estimated contract value equal to 0. These rows have been dropped from the data set. However, considering the overlap between these two categories, there are a total of 4594 nonconforming rows concerning this aspect, thus 20,085 rows left.

```
1 data = data.drop(data[(data['contractValue'] == 0) |  
2                   (data['cNoticeEstimatedContractValue'] == 0)].index)
```

Listing 3: Code representing Data Filtering for values of 0.

Data Restructuring

As a result of 4.2.4, we start with a selection of 20,085 rows in our data set. Considering the fact that in our winner column we may have multiple winners comma separated, this poses an issue for the understanding of machine learning models. Thus, we will split them one by one, duplicating all the other columns for each of the newly split value of the id of the winner. We will add these newly created rows in the data set, and delete the old ones with the initial multiple winners, as depicted in 4.

```
1 new_bids = data.append(new_rows, ignore_index=True)
2 bids_with_only_one_winner_per_row = new_bids.drop(rows_to_delete)
```

Listing 4: Code representing Data Restructuring.

This change led to the addition of 11,678 new rows in the data set, meaning we now have 31,763 rows in our data frame.

However, duplicate rows in a dataset can potentially lead to overfitting in machine learning models. Hence, it was checked if there are any duplicated rows, which there were and have been deleted from our data, as shown in the code snippet 5.

```
1 bids.drop_duplicates(inplace=True)
```

Listing 5: Code representing Data Filtering for duplicate values.

As a final result of data filtering and restructuring, we are finally left with the number of 20,027 rows in our data frame.

Data Normalization

In the initial data set, there were three types of currencies: RON, EUR and USD. Considering the fact that the RON currency was prominent, leading with 96.1% as graphically represented in 4.1, it was best to exchange all the values for the bids with EUR or USD currencies to RON, in order to normalize the values.

As time passed, currency exchange rates clearly changed, and therefore a solution was needed in order to make the changes as accurately and fairly as possible. Thus, a simple API for currency conversion and exchange rate retrieval was used, as depicted in code snippet 6.

```
1 from forex_python.converter import CurrencyRates
2 c = CurrencyRates()
3 . . .
4 exchange_rate = c.get_rate(currency_code, 'RON', date)
```

Listing 6: Code representing Data Normalization of currency codes.

Overall, this code 6 allows retrieval of the exchange rate between a given currency, in this case only EUR or USD, and the Romanian New Leu for a specific date, which was taken from the details of each particular bid, using the `forex-python` library. The data is normalized, having comparable values now, as they are all in RON.

Data Trimming

All of the features that were not of interest to the scope of the bidder recommender have been dropped from the data frame. Considering that `'currencyCode'` was now only `'RON'`, no insights at all could be learned from it, and it was redundant and hence also dropped, as highlighted in 7.

```
1 data = data.drop(['currencyCode', ...], axis=1)
```

Listing 7: Code representing Data trimming.

As a result of 7, the number of columns of the data set have been trimmed from 16 to only 6.

Data Splitting

Taking into consideration the spitting in smaller parts of CPV codes as explained in 2.1.2, we will refactor CPV codes into: `cpv`, `check digit`, `division`, `group`, `class` and `category`.

```
1 data['cpv'] = data['cpvCode'].str.split('-', expand=True)[0]
2 data['cpvCheckDigit'] = data['cpvCode'].str.split('-', expand=True)[1]
3 data['cpvDivision'] = data['cpv'].str.slice(stop=2)
4 data['cpvGroup'] = data['cpv'].str.slice(stop=3)
5 data['cpvClass'] = data['cpv'].str.slice(stop=4)
6 data['cpvCategory'] = data['cpv'].str.slice(stop=5)
```

Listing 8: Code representing CPV splitting.

As seen in the feature importance graphic 4.3, not all of these are relevant to the machine learning algorithm, so not all were considered. Only the original CPV code and 'cpvCategory' were used finally, the data frame having 7 columns now.

Data Encoding

In terms of data encoding, there were tried three approaches: One-Hot Encoding, Label Encoding and a combination of these two encodings, considering different data frame columns for each. The best encoding method for this specific data set and algorithms used is a combination of One-Hot Encoding for the CPV codes, the most important data feature, and Label Encoding for all the other relevant columns. This was chosen according to a comparison of accuracy metrics, as it will be detailed in chapter 4.3.

```
1 import pandas as pd
2
3 # one-hot encode the column 'cpvCode'
4 data_encoded = pd.get_dummies(data, columns=['cpvCode'])
```

Listing 9: Code representing One-Hot Encoding.

From the code snippet 9, on line 4, the `pd.get_dummies()` function from the pandas library is used to perform the one-hot encoding. It creates new columns for each unique value in the 'cpvCode' column, and assigns a value of 1 or 0 to each row based on whether the original 'cpvCode' value matches the corresponding column.

By passing the `columns=['cpvCode']` argument to `pd.get_dummies()`, it specifies that only the 'cpvCode' column should be one-hot encoded. The already numerical columns in the data will not be changed, whilst even though the other categorical columns in the data frame will remain unchanged for the moment, they will be label encoded in listing 10.

Label encoding converts categorical variables into numerical representations, allowing machine learning models to process them effectively. A representation of how it has been used in the scope of this data encoding part can be seen in listing 10. The remaining categorical columns to label encode, which will be in the variable `columns_to_encode`, are:

- 'noticeNo'
- 'title'
- 'publicationDate'

- 'contractingAuthorityName'
- 'cpvCodeName'
- 'cNoticeNo'
- 'cNoticePublicationDate'
- 'cNoticeTitle'
- 'cpvCategory'
- 'countyCode'.

```
1 from sklearn.preprocessing import LabelEncoder
2
3 # add label encoding to the other categorical columns
4 label_encoder = LabelEncoder()
5 for col in columns_to_encode:
6     data[col] = label_encoder.fit_transform(data[col])
```

Listing 10: Code representing Label Encoding.

Code snippet 10 shows how label encoding is performed on the remaining categorical columns of the data using the `LabelEncoder` class from the `sklearn.preprocessing` module. The method `fit_transform()` from line 6 is called for each data column and fits the label encoder to the column's unique categories. It then transforms the original categorical columns with their corresponding label encoded representations, meaning encoded integer values, as it can be seen in 4.7.

As a result of data encoding, the number of columns in the data frame skyrocketed from 7 to 1121. The final format of the data frame is: 20027 rows x 1121 columns.

	no°	contrac	tit	pul	c	cont	cpv	cNo°	cNotice	cNo°	...	c	c	c		c	c	c		
	0	18935	2424229.00	1712	14825	3	807	893	19303	17263180.36	15155	...	0	0	0	0	0	0	0	0
	1	20052	1252530.00	15988	14825	3	1240	741	20032	3608558.57	15860	...	0	0	0	0	0	0	0	0
	2	20065	133657.00	13401	14825	11	659	620	20026	153780.00	15847	...	0	0	0	0	0	0	0	0
	3	20068	151470.00	12016	14825	32	1432	85	19840	152340.00	15654	...	0	0	0	0	0	0	0	0
	4	18263	28211330.00	1656	14825	28	1203	825	17782	20915712.00	13621	...	0	0	0	0	0	0	0	0
...
24674	0	500055.00	13437	2	24	1240	621	229	702780.00	223	...	0	0	0	0	0	0	0	0	0
24675	4	25499990.00	10041	2	11	493	551	348	36748811.00	321	...	0	0	0	0	0	0	0	0	0
24676	7	3469477.32	16928	2	3	231	986	312	8901279.27	290	...	0	0	0	0	0	0	0	0	0
24677	18	12400.00	12950	1	3	1511	598	770	32258.00	615	...	0	0	0	0	0	0	0	0	0
24678	17	9520.00	12951	0	3	1511	598	790	20161.00	627	...	0	0	0	0	0	0	0	0	0

Figure 4.7: Data frame after encoding: only numerical values.

4.3 Computational Experiments

The computational experiments were computed gradually with the following goals: finding the best version of data encoding, deducting the most appropriate machine learning model for the respective data, fine tuning its parameters in hopes to increase its accuracy, and computing the accuracy in various scenarios, such as the case of more predictions suggested, comparative studies between well performing algorithms from this study and with specialized literature.

Experiments were conducted using two different computing environments. Firstly, the local machine was utilized, specifically the DataSpell IDE. This allowed for convenient and efficient testing and development within a familiar setup. Additionally, remote machines with ample RAM capacity, specifically 25.5 GB of run-time memory powered by Google Colab Pro were utilized, as depicted in 4.8. These high-memory machines provided the better computational resources to handle the experiments and ensure smooth execution.

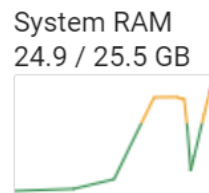


Figure 4.8: System RAM usage.

```
Traceback...
MemoryError: could not allocate 487981056 bytes
```

Figure 4.9: Local system (DataSpell IDE) Memory Error.

Your session crashed after using all available RAM. As a Colab Pro user, you have higher

Figure 4.10: Google Colab Pro Memory Error.

The experimenting environments crashed very often due to the high RAM memory usage, as can be seen in 4.9 and 4.10 which led to results data loss, especially concerning Grid Searches.

4.3.1 Data Encoding

In the first attempt of conducting computational experiments, the emphasis was on finding the best encoding variants for the data. There were tried variations of either

dividing the CPV code in smaller sub-codes, as explained in 2.1.2, or simply using it as it is. Apart from this alteration, there were compared encoding methods, more precisely One-Hot Encoding, Label Encoding, as well as a combination of these two.

To exclude potential bias of the ML algorithm chosen, initially the options were tried with the method of Decision Tree Classifier. Afterwards, there was an attempt proceed in the same way, trying these encoding options using the Gaussian NB method, but the accuracy was so disappointing that the efforts were stopped after the first iteration, which was computed with the use of One-Hot Encoding.

The following experiments with this scope were performed with a better suited algorithm for this problematic, the Random Forest Classifier, trying to delimit between the best encoding method according to the previous experiments, and a combination of it and the other option. The numeric metrics for the cases mentioned can be observed in the table 4.10.

Method	Setup	Metrics			
		Accuracy	Precision	Recall	F1 Score
Decision Tree Classifier	One-Hot Encoder	0.10	0.51	0.39	0.04
Decision Tree Classifier	Label Encoded	0.10	0.49	0.39	0.04
Decision Tree Classifier	OHE, divided CPV	0.11	0.50	0.39	0.05
Gaussian NB	One-Hot Encoder	0.01	0.93	0.05	0.0002
Random Forest Classifier	OHE, divided CPV; max depth = 37	0.12	0.52	0.38	0.05
Random Forest Classifier	OHE, divided CPV; max depth = 12, all data columns Label Encoded	0.14	0.74	0.21	0.05

Table 4.10: Model Performance Metrics - Different Data Encoding

Considering the results from 4.10, it can be concluded that from the first tree attempts, the One-Hot Encoding method incorporating the division of the CPV code was the best setup method for the data so far. The experiment with Gaussian NB can be overlooked due to its poor performance, which made further analysis into it redundant. Overall, as it is deduced from the experiments with the Random Forest

Classifier method, the best results were obtained using One-Hot Encoding for the most important features, the sub-codes of the CPV code, and Label Encoding for the other columns. Thus, this will be the configuration that will be used later on in the following experiments in regards to data encoding.

4.3.2 Machine Learning Algorithm

Moving forward and taking into account the data encoding configuration deducted from 4.10, the best ML algorithm is next up to be chosen. Therefore, for an almost standard setup for each of them, the intelligent algorithms defined in 2.2 and studied in 4.1 are put to the test and recorded in 4.11, being primarily compared and sorted by their accuracy, but also appreciated in terms of precision, recall and F1 score.

Method	Setup	Metrics			
		Accuracy	Precision	Recall	F1 Score
Logistic Regression	random_state=42	0.00	1.00	0.00	0.00
Support Vector Machines	random_state=42	0.01	1.00	0.00	0.00
Ada Boost	random_state=42	0.01	1.00	0.00	0.00
Gaussian NB	-	0.01	0.94	0.06	0.00
K-Neighbors Classifier	-	0.01	0.62	0.23	0.00
Decision Tree Classifier	random_state=42	0.13	0.51	0.41	0.05
Light GMB	random_state=42	0.03	0.83	0.07	0.01
Random Forest Classifier	n_estimators=100, random_state=42, max_depth=23	0.19	0.63	0.32	0.07
Extra Trees Classifier	random_state=42, max_depth=30	0.21	0.58	0.39	0.09

Table 4.11: Model Performance Metrics - Different ML Algorithms

As it can be easily observed from 4.11, the methods Random Forest Classifier and Extra Trees Classifier outperform greatly all of the other ML algorithms in terms of accuracy. In light of their very similar and equally promising results, these are the main two algorithms we will further study in this paper, in hopes to find the optimal one for our Bidders Application.

4.3.3 Parameters Fine Tuning

For both of these two algorithms, Random Forest Classifier and Extra Trees Classifier, the fine-tuning methods Random Grid Search and Grid Search were performed in order to choose the optimal values for their parameters and hopefully increase their accuracy.

First up is the Random Forest Classifier method, for which two more different parameter setups have been tried and added in 4.12 alongside the previous ones from the previous step, to gradually visualize the increase in accuracy.

Method	Setup	Metrics			
		Accuracy	Precision	Recall	F1 Score
Random Forest Classifier	OHE, divided CPV; max depth = 37	0.12	0.52	0.38	0.05
Random Forest Classifier	OHE, divided CPV; max depth = 12, all data columns Label Encoded	0.14	0.74	0.21	0.05
Random Forest Classifier	n_estimators=100, random_state=42, max_depth=23	0.19	0.63	0.32	0.07
Random Forest Classifier	n_estimators=100, random_state=42, max_depth=30	0.21	0.59	0.37	0.08

Table 4.12: Random Forest Performance Metrics - Different Setups

As it can be clearly observed from 4.12, the accuracy increased in each step, but it increased significantly more after performing the fine-tuning of the parameters. Specifically, it increased with approximately 35.71% only due to the tweaking of parameters, and once again with 10.53% just from the increasing of the max_depth parameter.

Secondly, plenty different configurations have been further on tried for the Extra Trees Classifier method, listed in 4.13. These configurations have been deducted in multiple attempts to run Random Grid Search and Grid Search, both with just smaller samples of the data and the whole data frame. Unfortunately, numerous times no conclusion was reached because of RAM memory crashes during the running of the algorithms. The maximization of the max_depth parameter yielded better results, but was also the main consumer of RAM, thus the principal reason the process crashed.

Method	Setup	Metrics			
		Accuracy	Precision	Recall	F1 Score
Extra Trees Classifier	max_depth=10	0.13	0.69	0.25	0.05
Extra Trees Classifier	max_depth=50, max_features=6, min_samples_leaf=3, n_estimators=500	0.13	0.86	0.12	0.05
Extra Trees Classifier	n_estimators=100, random_state=42, max_depth=50	0.15	0.56	0.45	0.07
Extra Trees Classifier	random_state=42, max_depth=20, min_samples_split=5, n_estimators=200	0.19	0.74	0.25	0.08
Extra Trees Classifier	random_state=42, max_depth=30	0.21	0.58	0.39	0.09
Extra Trees Classifier	random_state=42, max_depth=30, min_samples_split=5, n_estimators=200	0.23	0.63	0.36	0.10

Table 4.13: Extra Trees Performance Metrics - Different Setups

From analyzing the computational results from 4.13, it is obvious that the accuracy has increased significantly, with a 0.1 difference from the first setup compared to the last and best performing setup from this study, 0.23. The significant 76.92% percentage increase in accuracy from the initial setup was entirely due to the fine tuning of the Extra Trees Classifier's parameters: max_depth, min_samples_split and n_estimators.

4.3.4 Ranking Predictions

Considering the best performing algorithms, Random Forest and Extra Trees, predicting multiple organizations for a bid will be implemented, ranking them by probability. The accuracy will be computed as 1 if the desired winner is amongst those n companies, and 0 otherwise.

Considering different values for n, the number of forecast organizations, for instance 3 and 5, the accuracy of the actual winner being amongst them clearly increases, as it can be observed in 4.14.

Classifier	n	Accuracy
RandomForestClassifier	1	0.21
RandomForestClassifier	3	0.33
RandomForestClassifier	5	0.38

Table 4.14: Random Forest Performance Metrics - Different Number of Predictions

Deducting from 4.14, for the case in which $n=3$, the accuracy rises with 57.14% compared to $n=1$. Moreover, for the case in which $n=5$, the accuracy increases with approximately 80.95% from the initial base case of $n=1$.

The performance of the Extra Trees Classifier grows significantly, in terms of the winning company being in the predicted organizations, as the number of forecast organizations (n) is increased.

Classifier	n	Accuracy
ExtraTreesClassifier	1	0.23
ExtraTreesClassifier	3	0.32
ExtraTreesClassifier	5	0.36
ExtraTreesClassifier	10	0.41

Table 4.15: Extra Trees Performance Metrics - Different Number of Predictions

As shown in 4.15, simply for the increase of n , from 1 to 3, the accuracy rises with 39.13%, from 0.23 to 0.32. Moreover, from 0.23 to 0.36 there is an 56.52% increase in accuracy, only increasing the value of n from 1 to 5, and an 78.26% increase considering the case in which n becomes 10. In the aforementioned cases, all values are manageable in terms of manually looking into the status of the companies to choose the best firm, proving a useful smaller sample of firms of interest.

4.3.5 Random Forest vs. Extra Trees

Considering the results for predicting three or five potential winning companies for a bid, the two best performing algorithms for this problem, Extra Trees Classifier and Random Forest Classifier are compared in 4.16. They are both ensemble machine learning algorithms that belong to the family of decision tree-based methods.

n	Accuracy (Extra Trees)	Accuracy (Random Forest)
1	0.23	0.21
3	0.32	0.33
5	0.36	0.38

Table 4.16: Models Performance Comparison - Comparison between Random Forest and Extra Trees for Different Number of Predictions

The results shown above in 4.16 imply that the Random Forest Classifier performs slightly better than the Extra Trees Classifier for a bigger value of n , even though for $n=1$ the Extra Trees Classifier was the leader in terms of accuracy, with a maximum accuracy of 23%, as initially seen in 4.3.3.

Thus, considering the accuracy metrics from 4.16, we can conclude that the best accuracy values identified are as summarized in 4.17. Sadly, as all of these best performing algorithms were experimentally computed in Google Colab, in an 25.5 GB high RAM memory environment, they were not eligible to be chosen in the application.

Classifier	n	Accuracy
RandomForestClassifier	1	0.21
ExtraTreesClassifier	1	0.23
ExtraTreesClassifier	3	0.32
RandomForestClassifier	3	0.33
ExtraTreesClassifier	5	0.36
RandomForestClassifier	5	0.38
ExtraTreesClassifier	10	0.41

Table 4.17: Models Performance Comparison - The Best Accuracy Scores

4.3.6 Specialized Literature Comparison

Taking into consideration the computational results of [GRRMOFVB20] using a Random Forest Classifier, the following parallel with the experiments conducted in this paper can be made:

Classifier	n	Specialized Literature	My Accuracy
RandomForestClassifier	1	0.17	0.21
RandomForestClassifier	5	0.31	0.38

Table 4.18: Models Performance - Comparison with Specialized Literature

Thus, considering the experimental data results as seen in 4.3.6, it can be deducted that the Random Forest Classifier model's performance, as measured by accuracy, is slightly higher in both cases compared to the specialized literature's reported values. For $n=1$, the specialized literature reported an accuracy of 0.17, while the achieved accuracy was 0.21. Similarly, for $n=5$, the specialized literature reported an accuracy of 0.31, whereas the achieved accuracy was 0.38. These results suggest that the applied model outperformed the one presented in the specialized literature, indicating the potential effectiveness of the implemented approach.

4.4 Implementation Description

From user input to frontend-backend communication, machine learning model utilization, and training and model selection, this app allows economists to possess the power of artificial intelligence to make informed decisions about what companies are suitable for the public procurement auctions of interest.

The user, most likely an economist, effortlessly inputs bid data through an intuitive form built with React and MUI components. This data is seamlessly processed by the backend, powered by Flask, to facilitate further analysis. Leveraging a pre-trained Extra Trees Classifier, only with an accuracy of 0.11, due to memory limitations, but with a precision of 0.86, the backend generates predictions that aid decision-making.

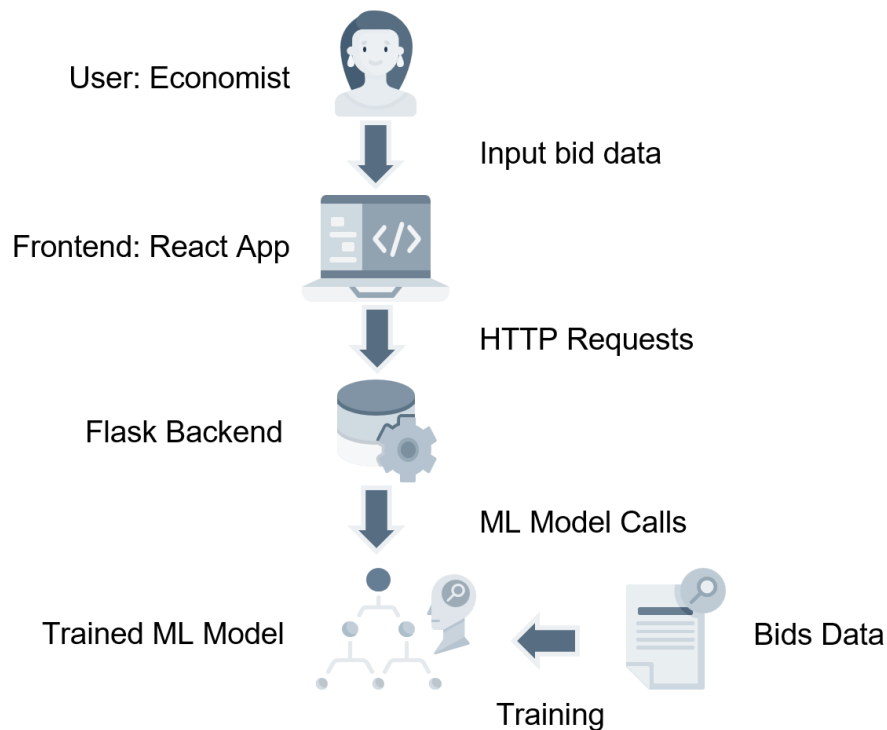


Figure 4.11: Graph representing the workflow of the application.

The workflow of the application is, as also depicted in graph 4.11:

1. The user, an economist, can input the bid data into the application using the designated form, which was created in React using MUI components.
2. The frontend then communicates with the API using HTTP requests. The backend is implemented in Flask, a lightweight Python framework.
3. The backend of the application loads a machine learning model from a file and utilizes it to generate predictions.

4. The ML algorithm was previously trained using a data set described in the following Data Set section, and was chosen due to the experiments explained in the upcoming Experiments section 4.3. The machine learning model used is a random forest classifier, however not with the best accuracy from the experiments, due to hardware limitations.

4.4.1 Frontend

The frontend of the application is written in React. With all the necessary characteristics in order to employ the machine learning model considered, and with appreciation for simplicity and readability in mind when it comes to user interface design, the form was created using MUI components.

By integrating Axios into the application, as previously defined in 2.3, the frontend can communicate with the backend server, retrieve data, send user input, and handle responses effectively. This allows for seamless data exchange between the frontend and backend, enabling a dynamic and interactive usage of the application.

```
1  const handleSubmit = async (e: FormEvent) => {
2    e.preventDefault();
3    // Sending the data to the backend
4    try {
5      // Make the API request
6      const response = await axios.post(
7        'http://localhost:5000/predict',
8        inputData);
9
10     // Extract the predictions from the API response
11     const predictions = response.data.predictions[0];
12
13     // Set the predictions as a state variable to
14     // trigger a re-render
15     setPredictions(predictions);
16     . . .
```

Listing 11: Code representing the request to the backend with Axios in React.

In the code snippet 11, it is presented how Axios is utilized to send requests to the backend API 4.4.2. In summary, the code sends form data to a backend API endpoint, awaits the response, extracts the predictions from the response, and updates the UI with the received predictions.

Bid Data

Enter the data of the bid you want winner recommendations for:



Notice Number 0	C Notice Publication Date dd/mm/yyyy 
Title	C Notice Title
Publication Date dd/mm/yyyy 	CPV Code
Contracting Authority Name	County Code
CPV Code Name	Contract Value 0
C Notice Number 0	C Notice Estimated Contract Value 0
Number of Predictions 3	SUBMIT

Figure 4.12: Screenshot representing the frontend of the application: bid input data.

The inputs represented in 4.12 are the characteristics of the bid which are needed in order to make predictions. There are three types of data: numbers, dates and strings. Notice number and C Notice Number are integers, while Contract value and C Notice Estimated Contract Value are floats, and all of them have 0 as the default set value in the form. Publication Date and C Notice Publication Date are of type date, with the possibility to nicely choose the date from the calendar or input it in the hinted format: dd/mm/yyyy. The number of predictions, 'n' defines how many organizations will be forecast for the given bid, and can take values from 1 up to 10, with a default value of 3. The rest of the fields are strings, with no default values.

Predictions:

Rank	Organization Name
1	SC ACAZIA IMPEX SRL
2	CARMENSIMI GRUP SRL
3	S.C. INTRETINERE SI REPARATII LOCOMOTIVE SI UTILAJE-CFR IRLU S.A.

Figure 4.13: Screenshot representing the frontend of the application: example output.

Regarding the output of the application, it was designed in a simple table, as can be visualized in figure 4.13. The simple design was brought to life with the use of the table component from Material-UI.

In the first column, the predictions are ranked in accordance to their respective probabilities of success in the given bid. The number of forecast organizations is determined by the user's input value, 'n'. The second column depicts the names of the companies, as provided by the backend.

Moreover, in the particular screenshot 4.13, the information used as bid details corresponds to the CPV code 50222000-7. This common procurement vocabulary code stands for "repair and maintenance services for rolling stock", so it can be observed that the predictions are relevant to the requested domain of interest.

4.4.2 Backend

The backend of the application is written in Flask, which is a Python web framework, as defined in 2.3.2. It uses various libraries such as joblib, numpy, pandas, and sklearn for data processing and machine learning. When the Flask application is run, it loads a trained machine learning model from a joblib file and a CSV file that maps IDs to names for the organizations.

Overall, the backend code provides an API endpoint that accepts input data, pre-processes it, makes predictions using a trained model, and returns the top predictions with their corresponding names as the API response, as represented in listing 12.

```
1 @app.route('/predict', methods=['POST'])
2 def predict():
3     # Get the input data from the request
4     data = request.json
5
6     # Preprocess the input data
7     data_frame = preprocess_data(data)
8
9     # Make predictions using the loaded model
10    probabilities = classifier.predict_proba(data_frame)
11
12    . . .
13
14    # Serialize response to JSON
15    return json.dumps(response)
```

Listing 12: Code representing the route in Flask.

In the code snippet 12, a POST route has been implemented using the `methods=['POST']` parameter. This route is used to receive input data from the React frontend 4.4.1. The `request.json` method is used to retrieve the JSON data sent from the web application, and the received data is preprocessed as needed by the ML algorithm. Afterwards, the predictions are made using the loaded model. They are sorted in descending order by their probabilities, and then a response with the corresponding ids and names of the forecast organizations are returned to the client, in JSON format.

4.5 Application Testing

Testing is a crucial practice in software development and various other domains where quality assurance is relevant. It involves systematically verifying and validating the system to identify errors or inconsistencies. Testing was applied during the development lifecycle to detect issues early on and ensure that the desired functionality is achieved.

4.5.1 Unit Testing

For unit testing, a main feature to consider is the machine learning algorithm's prediction accuracy. This involved generating a set of test, of 20% of the data with known outcomes and comparing the predicted results with the expected results. I have done this type of testing multiple times, considering different algorithms, various parameters

for them and different characteristics of the data, as well as different encodings.

4.5.2 Integration Testing

For integration testing, the modules that needed to be tested when integrated include the input data module, the ML algorithm module, the API module, and the output module. Testing helped me find the issues with the API. It revealed that the provided data was not suitable for the ML algorithm, causing errors.

Additionally, manually testing from the web page helped uncover a problem with the API's output format. Initially, the output was not being sent as a JSON, has caused compatibility issues. Through testing different API requests and examining the response, it became apparent that the output needed to be properly formatted as JSON to ensure seamless integration with the other components. Postman [Pos23], a popular API testing tool, was used to validate the API's functionality and ensure the proper handling of requests and responses, as depicted in 4.14.

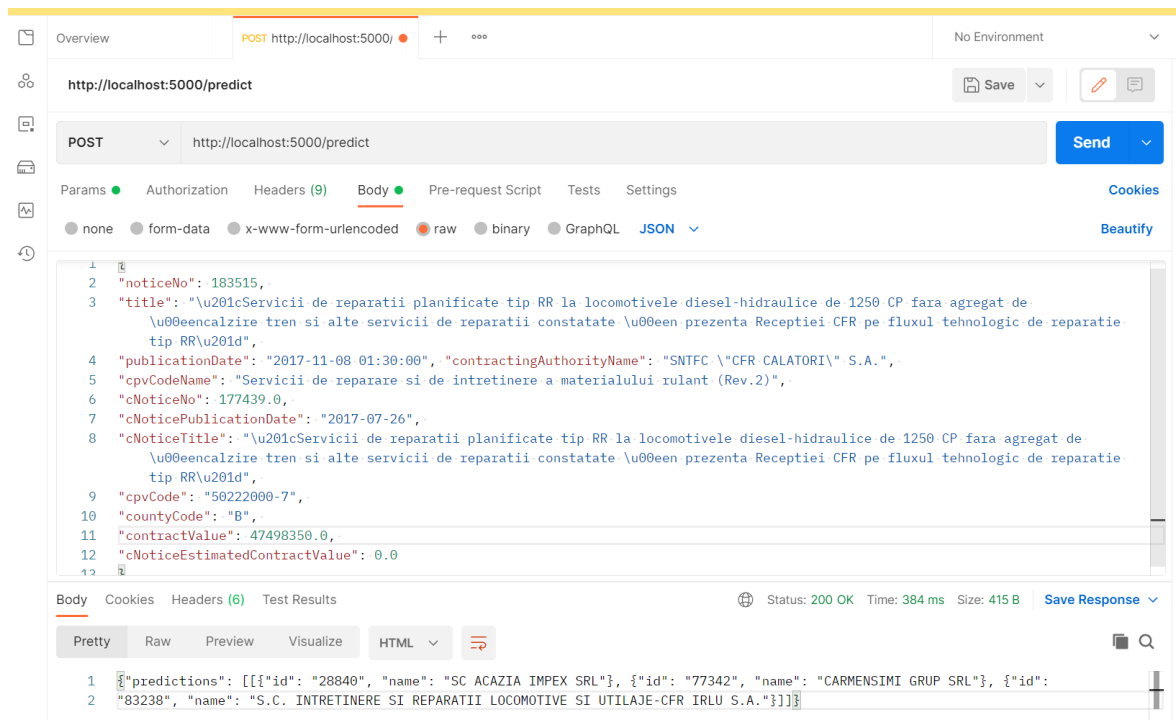


Figure 4.14: Screenshot representing the Postman request and response.

By emphasizing testing during the development process, these issues were identified and solved early on. Validating data quality and output format helped in refining the application's functionality, resulting in a more reliable and user-friendly solution. Overall, testing was instrumental in improving the application's performance, ensuring accurate predictions, and resolving compatibility issues with the API's output format.

Chapter 5

SWOT Analysis

A SWOT analysis [SWO23] is a strategy for examining four areas of the Bidders Recommender Application.

Those four areas, meaning what S.W.O.T. stands for, are:

- S: Strengths
- W: Weaknesses
- O: Opportunities
- T: Threats

These four areas can also be split into more categories:

- internal: Strengths, Weaknesses
- external: Opportunities, Threats
- positive: Strengths, Opportunities
- negative: Weaknesses, Threats

Internal factors, including strengths and weaknesses, are within the application's control.

External factors, composed of opportunities and threats, are outside the application's control.

Positive aspects, such as strengths and opportunities, provide favorable conditions. Negative aspects, such as weaknesses and threats, pose challenges or risks.

5.1 SWOT Matrix

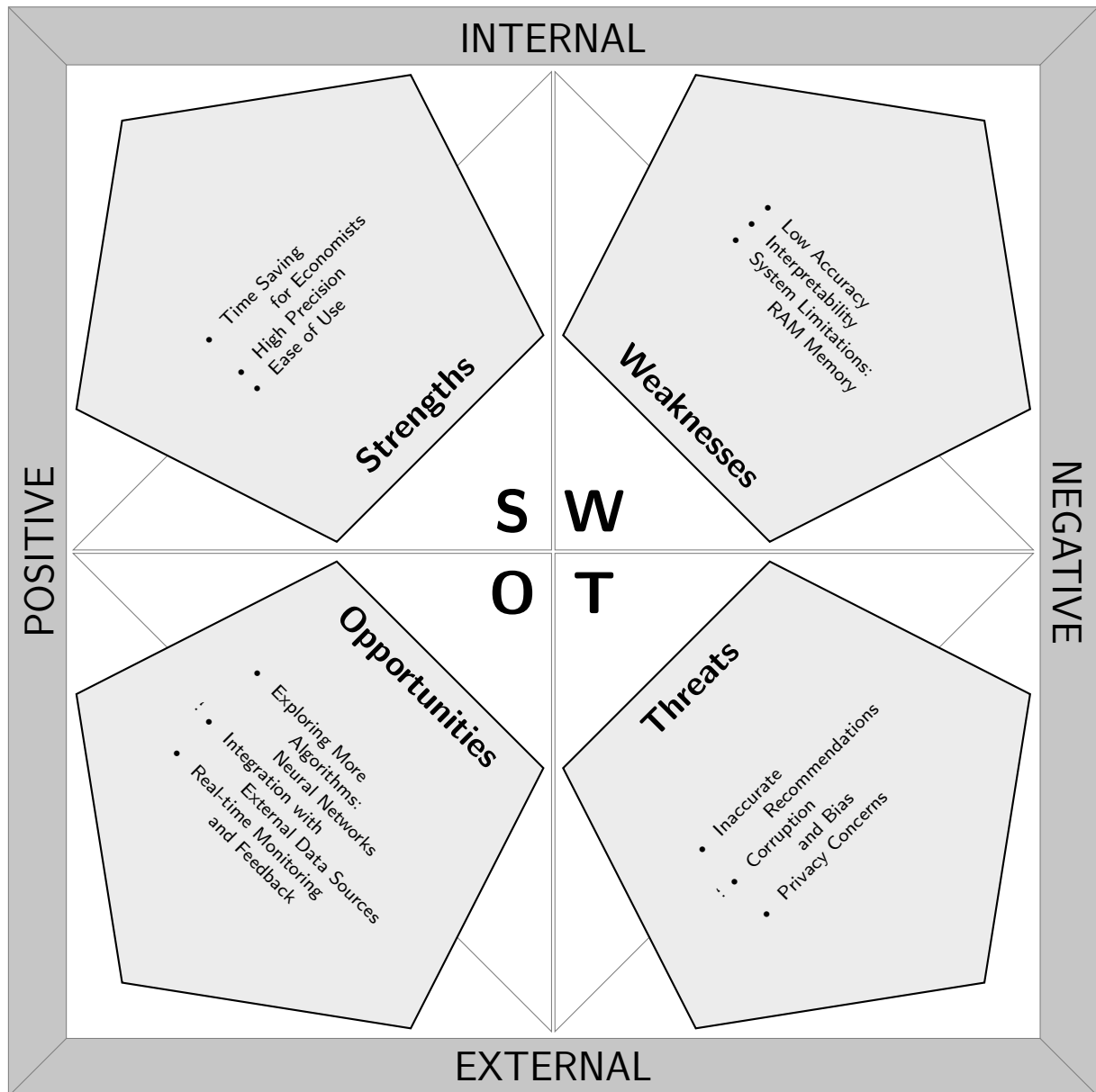


Figure 5.1: SWOT Analysis Matrix [Fer13]

5.2 SWOT Description

As pointed out in 5.1, all the above mentioned traits of the Bidders Recommender System will be described in detail as follows:

5.2.1 Strengths

1. **Time Saving for Economists:** Nowadays, time is the most crucial resource we have. Therefore, saving some extra time with the use of the Bidders Recommender System in the search for an ideal candidate can be considered a huge strength of this application for economists. Instead of going through all the potential candidates, reducing the sample of organizations to look into can be very beneficial in temporal terms.
2. **High Precision:** Precision measures the proportion of relevant and correctly recommended bidders out of the total recommended bidders. So, high precision indicates that a large percentage of the recommended bidders are relevant to the procurement auctions.
3. **Ease of use:** The application is highly intuitive, with an easy to use frontend that is pleasant to the eye. Moreover, the predictions are computed quickly, making the app's usage an effortless and quick process.

5.2.2 Weaknesses

1. **Low Accuracy:** Due to various potential factors such as insufficient or biased training data, corruption, unaccounted variables, or limitations in the algorithm implementation, the accuracy of the Bidders Recommender System is relatively low. However, it is comparable to the accuracy from another similar study, and it could still provide useful options in some cases.
2. **System Limitations (RAM memory):** Taking into consideration the platform I have used, Google Colab Pro [Goo23], I had a limitation of 25.5 GB of RAM memory. Better results could have been obtained for a maximum depth of the algorithms, however more RAM memory is needed. Moreover, also considering the limitations of my machine to a maximum of 16 GB, the best version of the algorithm could not be used in the actual Bidders Recommender Application.
3. **Interpretability:** Machine learning algorithms, especially complex ones like Random Forest or Extra Trees, can be less interpretable compared to simpler models. This lack of interpretability might make it challenging to explain the recommendations or identify the specific factors contributing to the results.

5.2.3 Opportunities

1. **Exploring Other Algorithms:** I believe further reasearch into other ML Algorithms could be proven beneficial to increase the accuracy of the current application. For instance, Neural Networks could be a great follow up algorithm to be used to improve the accuracy of the Bidders Recommender System.
2. **Integration with External Data Sources:** Perhaps more relevant information could be useful to draw more similarities between the companies with alike features. For example, maybe keywords from descriptions of the companies or the public procurement auctions could be proven useful.
3. **Real-time Monitoring and Feedback:** A very good way for the Bidders Application to improve would be to craft a system of feedback given by the users, the economists. In case it gives good predictions, they could rate the experience positively, and negatively otherwise, and the system should learn from these insights.

5.2.4 Threats

1. **Inaccurate Recommendations:** Considering the weakness of low accuracy, this could lead to suboptimal bidder selections and potential negative outcomes for the procurement process.
2. **Data Quality and Bias:** Biased or incomplete data can lead to biased recommendations or reinforce existing biases in the public procurement process, potentially compromising fairness and transparency.
3. **Privacy and Security Concerns:** Handling sensitive procurement data poses privacy and security risks. Data protection measures, including encryption, access controls, and compliance with relevant data protection regulations should be implemented.

To conclude with, the SWOT analysis highlights the system's strengths in high precision, the time saved for economists and its ease of use. However, it also identifies weaknesses in low accuracy, techincal limitations, and interpretability challenges. Opportunities include exploring other ML algorithms, integrating external data sources, and implementing real-time monitoring and feedback. Threats amass inaccurate recommendations, data quality and bias issues, and privacy concerns. By leveraging strengths, solving weaknesses, capitalizing on opportunities, and mitigating threats, the system can enhance the efficiency and fairness of public procurement auctions in Romania.

Chapter 6

Conclusions and Future Work Directions

6.1 Concluding Remarks

In conclusion, this study proposes a customised application for public procurement auctions in Romania, with the goal of ranking organizations based on their suitability to win a bid. The program employed artificial intelligence approaches to improve procurement efficiency and fairness. The empirical study, which was carried by utilizing a large dataset of public procurement auctions from Romania, yielded results comparable to specialized literature, even surpassing the accuracy of a relatively similar paper. These findings highlight the potential such an application has upon revolutionizing the field of public procurement auctions, in terms of saving time and enhancing transparency.

6.2 Future Work Directions

To improve the accuracy of the current application, further research into other ML algorithms should be conducted. Neural Networks, in particular, could be a promising choice to enhance the accuracy of the Bidders Recommender System. Additionally, incorporating more relevant information, such as keywords from company descriptions or public procurement auctions, could help draw meaningful similarities between companies with similar features.

Considering the limitation of 25.5 GB of RAM memory in the Google Colab Pro platform [Goo23], addressing memory constraints is crucial for obtaining better results. Allocating additional RAM could potentially improve the accuracy of the application. Moreover, even using the best algorithm from the experiments section could improve the application's quality of results, considering that my personal laptop's maximum RAM memory is 16 GB, and therefore a less accurate version of ML algorithm has

been used in the Bidders Recommender Application. Therefore, future work should focus on optimizing the algorithm and finding ways to overcome machine limitations to ensure the application can operate at its full potential.

Furthermore, another great method to improve the Bidders Application would be to design a mechanism that asks the users for feedback. If it makes relevant predictions, economists may assess the experience favorably, otherwise, they may rate it negatively, and the system should learn from these insights. Taking into consideration this respective user-given feedback, privacy measures should also be set in place to secure the application.

By pursuing these future work directions, the Bidders Recommender Application can be significantly improved in terms of accuracy, user experience and privacy. This will contribute to more effective and efficient public procurement auctions, benefiting both economists and the procurement process as a whole.

6.3 Acknowledgements

I would like to express my sincere appreciation to the economist Ion Marcu, for providing the data, as well as key insights into the field of economy pertaining to the scope of this paper. Specifically, I am extremely grateful for all resources provided by Mr. Marcu, especially the tree crucial CSV files which have represented the starting point of this application. The provided data played the most important role in this paper, facilitating the analysis, main functionality and conclusions presented in this paper.

Bibliography

- [adr] ADR - sistemul electronic de achiziții publice. <https://www.adr.gov.ro/seap/>.
- [axi] Axios. <https://axios-http.com/docs/intro>.
- [Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.
- [Bre01] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.
- [CCD08] Pádraig Cunningham, Matthieu Cord, and Sarah Jane Delany. Supervised learning. *Machine learning techniques for multimedia: case studies on organization and retrieval*, pages 21–49, 2008.
- [CDN12] Pier Luigi Conti, Livia De Giovanni, and Maurizio Naldi. A rank-and-compare algorithm to detect abnormally low bids in procurement auctions. *Electronic Commerce Research and Applications*, 11(2):192–203, 2012. The Role of Business Analytics in E-Commerce.
- [Con20] Contributors to Wikimedia projects. Cod de identificare fiscală - Wikipedia, September 2020. [Online; accessed 14. Jun. 2023].
- [Con23] Contributors to Wikimedia projects. Societate cu răspundere limitată - Wikipedia, May 2023. [Online; accessed 14. Jun. 2023].
- [dev] Flask. <https://dev.to/t/flask>.
- [Die00] Thomas G. Dietterich. *Ensemble Methods in Machine Learning*. Springer, 2000.
- [Fer13] Ricardo García Fernández. SWOT Matrix Snippet. GitHub repository, 2013.
- [Gac15] Cory Gackenhaimer. *Introduction to React*. Apress, 2015.

- [Goo23] Google. Google colaboratory. <https://research.google.com/colaboratory/faq.html>, Accessed: 2023.
- [Gri18] Miguel Grinberg. *Flask web development: developing web applications with python.* " O'Reilly Media, Inc.", 2018.
- [GRMBP⁺22] Manuel J. García Rodríguez, Vicente Rodríguez-Montequín, Pablo Ballesteros-Pérez, Peter E.D. Love, and Regis Signor. Collusion detection in public procurement auctions with machine learning algorithms. *Automation in Construction*, 133:104047, 2022.
- [GRRMOFVB20] Manuel J García Rodríguez, Vicente Rodríguez Montequín, Francisco Ortega Fernández, and Joaquín M Villanueva Balsera. Bidders recommender for public procurement auctions using machine learning: Data analysis, algorithm, and case study with tenders from spain. *Complexity*, 2020:8858258, November 2020.
- [JT17] Ali Haghpanah Jahromi and Mohammad Taheri. A non-parametric mixture of gaussian naive bayes classifiers based on local independent features. In *2017 Artificial intelligence and signal processing conference (AISP)*, pages 209–212. IEEE, 2017.
- [KS08] Carl Kingsford and Steven L Salzberg. What are decision trees? *Nature biotechnology*, 26(9):1011–1013, 2008.
- [Lan13] Brett Lantz. *Machine Learning with R*. Packt Publishing Limited, 2013.
- [LaV08] Michael P LaValley. Logistic regression. *Circulation*, 117(18):2395–2399, 2008.
- [lica] Ce este o licitație publică și cum funcționează. <https://www.licitatie-publica.ro/blog/ce-este-o-licitatie-publica-si-cum-functioneaza>.
- [licb] Ce este SEAP? informații generale de bază pentru începători. <https://www.licitatie-publica.ro/blog/ce-este-seap-informatii-generale-de-baza-pentru-incepatori>.
- [licc] Licitatii SEAP. <https://licitatiiseap.ro/seap/>.
- [lig] Lightgbm github repository. <https://github.com/microsoft/LightGBM>.

- [LLC23] Limited Liability Company (LLC) | Internal Revenue Service, June 2023. [Online; accessed 14. Jun. 2023].
- [LZ15] Shi Li and Yahong Zheng. A memetic algorithm for the Multi-Depot vehicle routing problem with limited stocks. In *Handbook of Research on Artificial Intelligence Techniques and Algorithms*, pages 411–445. IGI Global, 2015.
- [mui] Material-UI. <https://mui.com/>.
- [Nob06] William S Noble. What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567, 2006.
- [PBPM22] Dragan Pamučar, Darko Bozanic, Adis Puška, and Dragan Marinković. Application of neuro-fuzzy system for predicting the success of a company in public procurement. *Decision Making: Applications in Management and Engineering*, 5(1):135–153, Apr. 2022.
- [Pos23] Postman API Platform | Sign Up for Free, June 2023. [Online; accessed 13. Jun. 2023].
- [PPP17] Kedar Potdar, Taher Pardawala, and Chinmay Pai. A comparative study of categorical variable encoding techniques for neural network classifiers. *International Journal of Computer Applications*, 175:7–9, 10 2017.
- [Pri22] Priya_dharshini_... Must-Know Statistical Data Analysis Techniques in Machine Learning! *Analytics Vidhya*, September 2022.
- [Rea] React - a javascript library for building user interfaces. <https://reactjs.org/>. Accessed on May 21, 2023.
- [SF12] Robert E. Schapire and Yoav Freund. *Boosting: Foundations and Algorithms*. The MIT Press, 2012.
- [sim] Codul cpv. <https://simap.ted.europa.eu/ro/cpv#:~:text=Codul%20CPV%20stabile%C5%9Fte%20un%20singur,obiectul%20contractelor%20de%20achizi%C5%A3ii%20publice>.
- [som19] The basics of json and why it matters. <https://somedudesays.com/2019/11/the-basics-of-json-and-why-it-matters/>, 2019.
- [SWO23] SWOT Analysis, May 2023. [Online; accessed 13. Jun. 2023].
- [Yad21] Dinesh Yadav. Categorical encoding using Label-Encoding and One-Hot-Encoder. *Medium*, December 2021.