INSTITUTO POLITÉCNICO DE BRAGANÇA
Escola Superior de Tecnologia e Gestão

Predictive Modeling of Media Audience Based on Time Series

Bruno Filipe Lopes da Silva - a44657

Thesis presented to the School of Technology and Management in the scope of the Master in Informatics.

Supervisors:

Prof. Paulo Alexandre Vara Alves

Prof. José Eduardo Fernandes

Bragança

October 2025

Predictive Modeling of Media Audience Based on Time Series

Bruno Filipe Lopes da Silva - a44657

Thesis presented to the School of Technology and Management in the scope of the Master in Informatics.

Supervisors:

Prof. Paulo Alexandre Vara Alves

Prof. José Eduardo Fernandes

Bragança

October 2025

Dedication

The path to our desired destination is not always easy. Obstacles, unexpected challenges, and moments that, whether we like it or not, leave a lasting mark on us often arise. It is precisely in those moments that a push, a word, or a gesture can make all the difference. That is why I want to dedicate this achievement to all those who, in their own way, contributed to helping me reach this goal.

I dedicate this dissertation to my parents. To my father, who emigrated to give me a better future, a man who never needed words to show me the true meaning of love. To my mother, who so often gave up something so that I would never go without.

Not everything that is felt is said, but I know well everything you have done for me. Thank you for teaching me values, for showing me that nothing "falls from the sky", for teaching me to be resilient and never to lower my head.

Thank you for being the examples that you are. This achievement is not mine alone, it is yours too.

Acknowledgment

I would like to express my deepest gratitude to my supervisors, Professor Dr. Paulo Alves, Professor Dr. José Fernandes, and Professor David Dias, for their invaluable support, guidance, and availability throughout this journey. Your patience, dedication, and high standards were fundamental to the development and completion of this dissertation. Your suggestions and feedback were instrumental in my academic and personal growth.

I would also like to thank my family for their constant encouragement.

A sincere thank you to my friends and colleagues for their support, companionship, and motivating words, which made this journey lighter and more rewarding. Your support was essential in maintaining the balance between personal, professional, and academic life.

Finally, I would like to acknowledge all those who, directly or indirectly, contributed to this project.

To all of you, my most heartfelt thanks.

Abstract

The rapid evolution of media consumption habits and the increasing competition between television and digital platforms have intensified the need for accurate audience forecasting. Understanding how audiences fluctuate over time is crucial for broadcasters, advertisers, and content producers seeking to optimize programming strategies and allocate resources efficiently. This dissertation presents a comprehensive study on the prediction of television audience ratings using machine learning and statistical models. The work compares multiple modelling approaches, including Linear Regression, Ridge Regression, Random Forest, Gradient Boosting (LightGBM), Long Short-Term Memory (LSTM) networks, and the SARIMA statistical model. The analysis was conducted on four datasets derived from Portuguese television audience data, covering pre- and post-COVID-19 periods and incorporating different program types schemes. It is important to emphasize that exclusively exogenous variables were used, that is, variables external to the audience generation process itself, deliberately excluding endogenous variables, in order to evaluate the predictive capacity of the models based only on contextual and programmatic factors. A rigorous preprocessing pipeline was implemented, including data cleaning, feature encoding, temporal normalization, and seasonality analysis. Hyperparameter optimization was performed using grid and randomized search methods, and models were evaluated according to MAE, RMSE, MSE, and R^2 metrics.

The results demonstrate that ensemble-based methods, particularly Random Forest and LightGBM, consistently outperform linear and statistical baselines, achieving R^2 scores above 0.93. The LSTM network effectively captured temporal dependencies but showed sensitivity to the reduction of training data in the post-COVID subsets, while

the SARIMA model proved less suitable for capturing nonlinear audience dynamics. The study also identifies clear evidence of seasonal and behavioural patterns in television audiences, which can be leveraged to improve future forecasting models. Future research directions include the integrating of external data sources such as social media and streaming platform metrics. Such extensions could further enhance the contextual understanding of audience behaviour and support data-driven decision-making in the broadcasting industry.

Keywords: television audience forecasting, machine learning, ensemble models, LSTM, SARIMA, LightGBM, time series, media analytics

Resumo

A rápida evolução dos hábitos de consumo de media e a crescente concorrência entre a televisão e as plataformas digitais intensificaram a necessidade de previsões precisas de audiências. Compreender as variações das audiências ao longo do tempo é fundamental para emissores, anunciantes e produtores de conteúdo que procuram otimizar as suas estratégias de programação e alocar recursos de forma eficiente. Esta dissertação apresenta um estudo abrangente sobre a previsão das audiências televisivas utilizando modelos de aprendizagem automática e modelos estatísticos. O trabalho compara múltiplas abordagens de modelação, incluindo Regressão Linear, Regressão Ridge, Random Forest, Gradient Boosting (LightGBM), redes Long Short-Term Memory (LSTM) e o modelo estatístico SARIMA. A análise foi realizada sobre quatro conjuntos de dados derivados de audiências da televisão portuguesa, abrangendo períodos pré e pós-COVID-19 e incorporando diferentes esquemas de tipologia de programas.

É importante salientar que foram utilizadas exclusivamente variáveis exógenas, ou seja, variáveis externas ao próprio processo de geração de audiências, excluindo deliberadamente variáveis endógenas, com o objetivo de avaliar a capacidade preditiva dos modelos apenas com base em fatores contextuais e programáticos. Foi implementado um processo de pré-processamento, incluindo limpeza de dados, codificação de variáveis, normalização temporal e análise de sazonalidade. A otimização de hiperparâmetros foi realizada através de métodos de pesquisa em grelha (*grid search*) e pesquisa aleatória (*randomized search*), e os modelos foram avaliados segundo as métricas MAE, RMSE, MSE e R^2.

Os resultados demonstram que os métodos baseados em ensemble, particularmente o

Random Forest e o LightGBM, superam consistentemente os modelos lineares e estatísticos de referência, alcançando valores de R^2 superiores a 0,93. A rede LSTM mostrou-se eficaz na captação de dependências temporais, mas revelou sensibilidade à redução dos dados de treino nos subconjuntos pós-COVID, enquanto o modelo SARIMA se mostrou menos adequado para capturar dinâmicas não lineares das audiências. O estudo identifica ainda evidências claras de padrões sazonais e comportamentais nas audiências televisivas, que podem ser explorados para melhorar modelos de previsão futuros. As direções futuras de investigação incluem a integração de fontes de dados externas, como métricas de redes sociais e de plataformas de streaming. Estas extensões poderão reforçar a compreensão contextual do comportamento das audiências e apoiar a tomada de decisões baseada em dados na indústria televisiva.

Palavras-chave: previsão de audiências televisivas, aprendizagem automática, modelos ensemble, LSTM, SARIMA, LightGBM, séries temporais, análise dos media

Contents

List of Figures

Acronyms

ANN Artificial Neural Network.

ANNs Artificial Neural Networks.

ARIMA AutoRegressive Integrated Moving Average.

EEG Electroencephalogram.

ESTiG Escola Superior de Tecnologia e Gestão.

GBM Gradient Boosting Machine.

GBMs Gradient Boosting Machines.

IPB Instituto Politécnico de Bragança.

IQR Interquartile Range.

LightGBM Light Gradient Boosting Machine.

LSTM Long Short-Term Memory.

MAE Mean Absolute Error.

MAPE Mean Absolute Percentage Error.

ML Machine Learning.

MSE Mean Squared Error.

OLS Ordinary Least Squares.

RMSE Root Mean Squared Error.

RNN Recurrent Neural Network.

RNNs Recurrent Neural Networks.

SARIMA Seasonal AutoRegressive Integrated Moving Average.

SVM Support Vector Machine.

SVMs Support Vector Machines.

TWA Tagus World Analytics.

Chapter 1

Introduction

Television, despite the profound transformations that have taken place in the media ecosystem over recent decades, remains one of the most relevant means of communication in contemporary society. The growing fragmentation of audiovisual offerings and the competition from new digital platforms, such as streaming services and social media, have significantly changed consumption habits and audience patterns. In this context of high volatility and multiple external factors influencing audience behavior, predicting television ratings has become a strategic challenge for both broadcasters and advertisers, who rely on accurate forecasts to optimize programming and advertising investment [1].

Audience forecasting is therefore a complex, multidimensional problem with a strong temporal dependency [2]. Factors such as program type, time slot, day of the week, season, sporting events, holidays, or even unforeseen occurrences can significantly affect audience levels. With the advancement of machine learning and deep learning algorithms, it has become possible to develop more sophisticated approaches capable of capturing nonlinear relationships and complex temporal dependencies [2], [3]. This evolution opens up new possibilities for the development of more accurate and adaptable predictive models that reflect the dynamic nature of contemporary audiovisual consumption.

The art of achieving success in television is not something that happens by chance; it is the result of careful study and strategic planning. Television audiences are strongly influenced by a variety of factors, including social and demographic habits, psychological

circumstances, and even feelings of loneliness. The success of new program launches largely depends on the ability to attract and retain viewers. However, it is often observed that many television projects fail to meet expectations, resulting in disappointment and poor performance [4].

In this context, the present dissertation aims to develop and evaluate predictive models of television audiences, based on historical data from Portuguese channels and employing machine learning and deep learning techniques.

The objective is to compare the performance of traditional models, such as Linear Regression, Random Forest, Gradient Boosting, with a recurrent neural network model of the Long Short-Term Memory (LSTM) type, analyzing their generalization ability and predictive accuracy. Model performance will be assessed using standard forecasting metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE) and Root Mean Squared Error (RMSE), ensuring a consistent and quantitative basis for comparison.

To achieve this goal, an extensive dataset was constructed and explored, collected through the YUMI platform [5], containing detailed information on the programming and audience ratings of the main national television channels (RTP1, RTP2, SIC, and TVI) between 2021 and 2025.

The study includes an in-depth exploratory analysis, the creation of derived variables, the treatment of outliers, and the design of different training and testing scenarios to ensure the robustness and validity of the obtained results.

In addition to the comparison between models, this work seeks to contribute to a better understanding of the factors that influence television consumption in Portugal, identifying seasonal, temporal, and behavioral patterns that may help media organizations interpret audience fluctuations more accurately.

By integrating statistical and computational approaches within a unified experimental framework, this dissertation aims to offer both a practical and scientific perspective on the application of forecasting techniques in real-world media contexts.

1.1 Tagus World Analytics and MarkData

Tagus World Analytics (TWA) was founded in 2019 and is a software development company specialized in processing different types of media data, namely television, internet, and radio. In order for these data to acquire strategic value, they must be transformed into relevant and structured information. This is precisely where TWA plays a fundamental role, by providing software solutions capable of performing this processing and analysis efficiently.

TWA is part of the Marktest Group, founded on June 26, 1980, a world-leading company in software for media research and planning. Currently, Marktest is present in more than 30 countries and is recognized as one of the pioneers in this market, standing out for its leadership and innovation in the analysis of media communication data.

This industrial collaboration demonstrates the practical relevance of the research. Accurate audience forecasting is not only an academic challenge but also a strategic asset for companies such as Tagus World Analytics and Markdata. Anticipating audience fluctuations enables resource optimization, better planning, and more effective advertising targeting, all essential factors for competitiveness in the media sector.

1.2 Problem Statement

Television audience forecasting represents a central challenge within the field of data analysis applied to media, given the dynamic and multifactorial nature of audience behaviour. The ability to anticipate fluctuations in viewership enables the optimization of programming schedules, guides advertising strategies, and maximizes the return on advertising investments.

However, the phenomenon under study exhibits a high degree of complexity, resulting from the interaction between temporal, contextual, and content-related factors. The audience of a programme depends not only on its genre but also on the time of broadcast, the day of the week, the season of the year, and external circumstances, such as holidays,

political events, or major sporting competitions.

In this context, the present dissertation proposes the development of a predictive model capable of estimating the expected television audience for a given programme or time slot, based on exogenous variables known in advance. The proposed approach aims to create a decision-support system that can anticipate peaks or drops in viewership, contributing to a more efficient management of television resources and the optimization of programming strategies. Temporal and seasonal patterns in historical audience data will be explored.

1.2.1 Objectives

The primary objective of this research is to develop a robust and generalizable television audience forecasting model capable of predicting audience behaviour across different temporal periods with a satisfactory degree of accuracy. The model aims to capture temporal and seasonal patterns in viewership, enabling broadcasters to anticipate audience fluctuations and make data-informed programming and advertising decisions. The objectives of this study are as follows:

- Identify and select relevant exogenous variables that influence television audience levels, such as broadcast time, programme type, season, day of the week, and the occurrence of holidays or special events.
- Develop a predictive modelling framework that relies on generalizable features rather than programme-specific identifiers (example, programme title or host/pivot), ensuring applicability to new and previously unseen programmes.
- Explore temporal and seasonal patterns in historical audience data to improve the model's predictive accuracy.
- Evaluate the forecasting performance of different modelling approaches using standard metrics, ensuring reliability and robustness.

The objectives of this dissertation prioritize generalization and robustness over memorization of individual cases. By focusing on exogenous and broadly applicable variables,

the model is designed to anticipate audience behaviour for new programmes and time slots without requiring continuous retraining. This approach emphasizes practical applicability and ensures that the forecasting system can support long-term strategic decision-making in the television industry.

1.3 Structure of the Document

The remainder of this dissertation is organized into six main chapters, as follows: Chapter 1 introduces the context and motivation of the study, outlines the problem statement, and defines the main objectives pursued throughout the research.

Chapter 2 reviews previous studies and theoretical foundations relevant to television audience forecasting. It explores key concepts in time series prediction, machine learning models, and performance evaluation metrics.

Chapter 3 describes the procedures followed for data collection, preprocessing, and exploratory data analysis. It presents the dataset characteristics, feature engineering steps, treatment of outliers, and variable selection process. This chapter also discusses the analytical framework adopted to prepare the data for modeling.

Chapter 4 focuses on the implementation and optimization of the predictive models. It explains the preprocessing techniques, model configurations, cross-validation strategy, and training procedures applied to each algorithm. Furthermore, this chapter details the hyperparameter tuning process and identifies the optimal configurations for each model to ensure the best predictive performance.

Chapter 5 presents and discusses the experimental results obtained for each model, comparing their performance through quantitative metrics. The chapter also provides a critical interpretation of the findings, highlighting patterns, limitations, and insights derived from the analysis.

Chapter 6 summarizes the main findings of the dissertation, discusses their implications, and suggests potential directions for future research and practical applications in the field of television audience forecasting.

Chapter 2

Theoretical Background

This chapter provides the theoretical foundation necessary to understand the concepts, techniques, and methodologies used throughout this dissertation. It begins with a review of the current state of the art in the field, followed by an overview of the fundamental principles of Machine Learning and its most common models used for predicting television audiences. The chapter then introduces the core ideas behind Deep Learning, highlighting its relevance and advantages over traditional approaches. Finally, it presents the main evaluation metrics employed to assess model performance, establishing the basis for the experimental analysis conducted in later chapters.

2.1 State Of The Art

Historically, television audience measurement has predominantly relied on sample-based methodologies, such as Nielsen ratings, which utilize set-top boxes and survey techniques to estimate viewership. Two of the most fundamental metrics in this context are rating and share. Rating refers to the percentage of a specific demographic group watching a given television program, whereas share reflects the percentage of the audience tuned into that program compared to all viewers watching television at that moment. These two metrics have long served as essential tools for broadcasters in assessing audience size and devising both content and advertising strategies [6].

While these methods have provided valuable insights into audience behavior, they are not without significant limitations. One of the primary issues is sample bias, which can lead to inaccurate representations of the broader audience.

This limitation is echoed by Webster, who notes that audience measurement techniques are often based on samples that vary in size and can result in audience polarization, further complicating the accuracy of viewership data [7].

In addition, the lack of granularity in these traditional methods means that they often fail to account for the nuances of audience engagement. For example, neural processing reliability can predict audience preferences, indicating that more sophisticated metrics could improve understanding of viewership dynamics beyond mere numbers [8].

The evolution of digital television and the rise of social media have further complicated audience measurements. Traditional metrics struggle to keep pace with the fragmented media landscape, where viewers engage with content across multiple platforms [9].

In conclusion, while traditional sample-based approaches to TV audience measurement have laid the groundwork for understanding viewership, they are increasingly inadequate in capturing the full spectrum of audience engagement in digital landscape of today. The integration of new technologies and methodologies is essential for developing more accurate and nuanced audience measurement systems that reflect reality.

In recent years, the relationship between television, streaming, and social media has become one of the central focuses in audience studies. Currently, there exists an ecosystem in which digital platforms and social networks play a key role in content consumption. A study by Larsen et al. [10], which analyzed the talk show Skavlan using large volumes of Facebook data, showed that social engagement, particularly the interactions generated by more controversial episodes was associated with subsequent increases in television audiences. This result demonstrates that online repercussions have a significant influence, even when marked by negative reactions. Complementing this perspective, Ueoka and Ishii [11] also examined the influence of Twitter interactions on the audience of a Japanese drama series. The results indicated that the dynamics of comments throughout the season accompanied variations in audience numbers, reinforcing the hypothesis

that social engagement is intrinsically connected to viewership. More recently, data from Nielsen show that, for the first time, streaming consumption has surpassed both broadcast and cable television in the United States. This milestone highlights a structural shift in audience behavior [12].

The COVID-19 pandemic had a significant influence on television audiences, altering consumption patterns. During lockdown, there was a general increase in the time spent in front of the screen, both on traditional television and streaming platforms. Lockdown also led to a marked increase in binge watching, practices that became mechanisms of emotional regulation and a search for symbolic companionship during isolation [13] [14]. During lockdown periods, there was a sharp increase in television audiences, especially for informational content. In countries such as Italy, exposure to television news proved to be decisive in shaping cognitive, emotional, and behavioral responses related to the adoption of individual protective measures [15].

Recent advances in Machine Learning (ML) have significantly transformed the landscape of television audience prediction, using a variety of algorithms to analyze historical viewership data and forecast future trends. Traditional methods such as linear regression and decision trees have been fundamental in this domain, providing a straightforward approach to modeling relationships within the data. However, the introduction of more sophisticated models, including Support Vector Machine (SVM) and Gradient Boosting Machine (GBM), has markedly improved the accuracy of predictions by capturing complex patterns in audience behavior.

Linear regression has long been a staple in predictive analytics due to its simplicity and interpretability. It allows for modeling relationships between viewership metrics and various independent variables, such as time of day, genre, and marketing efforts [16]. Decision trees, on the other hand, offer a more nuanced approach by segmenting data into branches based on feature values, which can reveal intricate decision-making processes that influence audience behavior [17]. The interpretability of decision trees makes them particularly appealing for media managers who need to understand viewer preferences and trends [18].

In recent years, ensemble methods such as gradient boosting have gained traction for their ability to enhance predictive performance by combining multiple weak learners to create a robust model. Gradient Boosting Machines (GBMs) have been shown to outperform traditional methods in various applications, including audience prediction, by effectively minimizing prediction errors through iterative learning [19].

Support Vector Machines (SVMs) represent another advancement in the field, particularly in their capacity to handle high-dimensional data and complex relationships. SVMs work by finding the optimal hyperplane that separates different classes of data, making them suitable for audience segmentation tasks where distinguishing between viewer types is crucial [20]. Their application in audience prediction has shown improved accuracy compared to simpler models, particularly in scenarios where the data exhibit significant variance [20].

In addition, the integration of big data analytics with machine learning techniques has further enhanced the ability to predict TV audiences. For instance, studies have shown that emotional engagement, as measured through Electroencephalogram (EEG) metrics (a technique used to measure the brain's activity through electrodes placed on the scalp and in this situation is used to assess emotional and cognitive responses to the content being viewed), can significantly influence audience behavior and engagement with TV content [21]. This underscores the importance of incorporating psychological factors into predictive models to achieve a comprehensive understanding of audience dynamics.

The evolution of machine learning algorithms, from traditional linear regression and decision trees to advanced techniques such as SVMs and GBMs, has revolutionized the ability to predict TV audiences. These advancements not only enhance the accuracy of predictions, but also provide deeper insights into the factors driving viewer behavior.

Machine learning models have significantly advanced the field of television audience prediction, however, their effectiveness largely depends on the choice of algorithm. Each method presents distinct strengths and weaknesses that directly influence its suitability for specific applications. In practice, it is therefore essential to critically assess both the

advantages and limitations of these models. Different algorithms vary in terms of accuracy, interpretability, and computational efficiency, making them more or less appropriate depending on the characteristics of the available data and the specific objectives of the prediction task.

Linear Regression remains one of the most widely used models in predictive analytics due to its simplicity and ease of interpretation. Fast computation and straightforward implementation are possible due to its core assumption of linearity. It is particularly valuable when the relationship between variables is expected to have a consistent and proportional trend, like predicting viewership based on time slots or show categories. Its main weakness is its inability to capture non-linear relationships, which are frequently present in the dynamic and multifaceted patterns of audience behavior. Furthermore, it is sensitive to outliers and multicollinearity among features. The application of linear regression to mixed datasets that involve categorical and continuous variables, which is commonly used in audience analysis, underperforms, as noted by El Fayq et al. [22]. Despite this, Gupta et al. [23] emphasize that linear models can outperform more complex algorithms and be more understandable for decision makers in well-structured, small-scale datasets.

Decision Trees, on the other hand, offer a more flexible framework for modeling non-linear relationships. They are particularly appreciated for their interpretability, as the tree structure can visually demonstrate how predictions are made. Decision trees are susceptible to overfitting, particularly when not properly pruned, and their outcomes can be unpredictable. Small changes in input data can lead to completely different tree structures. While Mahimkar and Goel [24] argue that decision trees are ideal for exploratory analysis and rule-based audience segmentation, Hanif [25] notes their limitations in generalizing across diverse viewer groups, especially when the feature space becomes large and noisy.

In recent years, advances in Deep Learning have significantly transformed television audience forecasting, mainly due to the ability of these models to handle complex, large-scale sequential data. Among the most promising architectures are LSTM networks,

designed to overcome the limitations of traditional recurrent neural networks, such as the vanishing gradient problem, thus enabling the capture of long-term dependencies in time series [26].

Recent studies have shown encouraging results in this field. A study by Cammarano et al. [27] employed an LSTM-based approach, combining Twitter interaction metrics with audience data, and demonstrated that this model outperformed traditional algorithms such as Random Forest, achieving a reduced MSE of 0.058 in predicting the popularity of television programs regardless of genre. In the same study, Ridge regression was also tested, obtaining a higher MSE of 0.110.

Among the many time series models used for prediction, AutoRegressive Integrated Moving Average (ARIMA) and its seasonal counterpart, Seasonal AutoRegressive Integrated Moving Average (SARIMA) stand out for their ability to capture temporal and seasonal dynamics in viewer behavior.

ARIMA models are particularly effective when analyzing non-seasonal viewership data. They combine autoregressive and moving average elements with differencing to make time series stationary, making them well suited for short-term forecasts of general viewing trends [28].

SARIMA models extend ARIMA by incorporating seasonal autoregressive and moving average terms. This allows for better performance when forecasting TV ratings that show clear periodic patterns. For example, Shao [29] demonstrated that SARIMA provided improved accuracy over ARIMA in predicting new media viewing behavior, particularly when seasonal factors were evident in the data.

2.2 Machine Learning

Machine Learning is a subfield of Artificial Intelligence that consists of a set of methods enabling the construction of models capable of learning patterns from data and making predictions on new cases [30].

Machine Learning algorithms are generally divided into four main categories: Supervised Learning, Unsupervised Learning, Semi-Supervised Learning, and Reinforcement Learning [31], as illustrated in Figure 2.1.

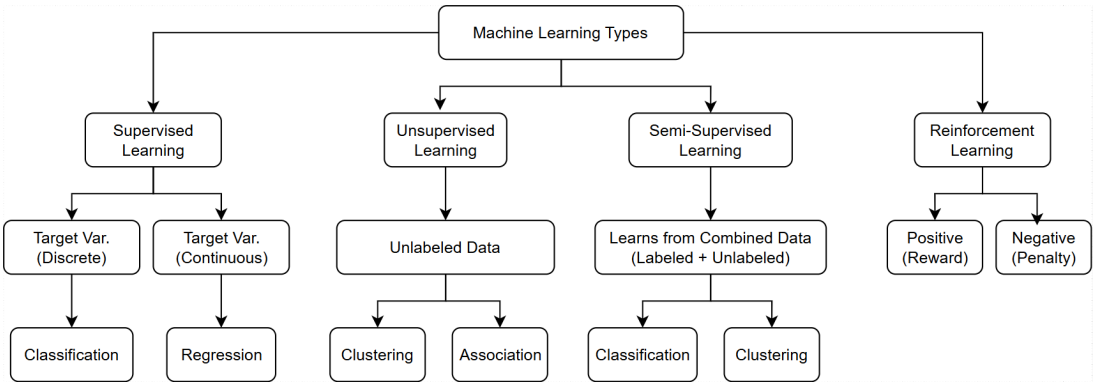


Figure 2.1: Machine Learning Types [31]

Supervised Learning, the model is trained with labeled data, meaning that both the input variables (features) and the output variables (labels) are known. The objective of the model is to learn a function that relates the input variables to the output variable, allowing it to make predictions on new data [32].

This approach is used in both regression problems, where the output variable is numerical and continuous (for example, predicting the audience of a television program or a specific time slot as proposed in this dissertation), and classification problems, where the output variable is categorical (for example, classifying programs as having "high" or "low" audience).

Supervised Learning is currently considered the most mature and widely applied approach within the field of Artificial Intelligence [33].

Unlike supervised learning, unsupervised learning works with unlabeled data, that is, only the input variables are known, and the system needs to discover hidden patterns. One of the most common tasks in this category is clustering, in which the algorithm attempts to group similar data points into clusters. Another important task is dimensionality reduction, which aims to simplify the number of variables while preserving the essential information [34].

Semi-supervised learning emerges as a hybrid approach between supervised and unsupervised learning. In this case, the model is trained with a small amount of labeled data and a large amount of unlabeled data. This technique has proven particularly useful in areas such as image recognition and natural language processing [35].

In reinforcement learning, the model learns through interaction with a dynamic environment. It performs actions and receives rewards or penalties according to the outcomes, adjusting its behavior to maximize the cumulative reward over time. This technique is widely used in fields such as robotics and gaming [36].

2.2.1 Machine Learning Models

This section presents some of the most relevant models applied to regression tasks, namely Linear Regression and Ridge Regression, which represent linear and regularized methods, and Random Forests and Gradient Boosting Machines, which correspond to tree-based and ensemble techniques.

Linear Regression

Linear regression is a foundational statistical method that establishes a relationship between a dependent variable y and one or more independent (explanatory) variables x_i [37]. In its most general form (multiple linear regression), it can be written as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \varepsilon \tag{2.1}$$

In this model:

- β_0 is the intercept (the expected value of y when all $x_i = 0$);
- β_i are the regression coefficients associated with each explanatory variable x_i;
- ε represents the residual error term, capturing the variation not explained by the model.

The goal of linear regression is to estimate the coefficients β_i that best fit the observed data. This is typically achieved through the Ordinary Least Squares (OLS) method, which minimizes the sum of squared residuals:

$$\text{Minimize } \sum_{i=1}^{m}(y_i - \hat{y}_i)^2 \tag{2.2}$$

where y_i denotes the observed value and $\hat{y}_i$ the predicted value from the model [38].

Ridge Regression

Ridge Regression is an extension of the traditional Linear Regression model that introduces a regularization mechanism to prevent overfitting and mitigate multicollinearity among predictors. While OLS estimates the regression coefficients by minimizing the sum of squared residuals, Ridge Regression adds a penalty term proportional to the square of the coefficients magnitudes (known as L2 regularization) [39], [40]. Formally, the Ridge objective function can be expressed as:

$$\min_{\beta} \left(\|y - X\beta\|^2 + \lambda \|\beta\|^2 \right) \tag{2.3}$$

where $\lambda \geq 0$ is the regularization parameter that controls the strength of the penalty, β represents the vector of model coefficients, and X and y denote the matrix of predictors and the response vector, respectively. When $\lambda = 0$, the Ridge estimator reduces to the classical OLS solution; as λ increases, the coefficients are progressively shrunk toward zero.

The analytical solution for the Ridge estimator is given by:

$$\hat{\beta}_{ridge} = (X^T X + \lambda I)^{-1} X^T y \tag{2.4}$$

where I is the identity matrix. This modification ensures numerical stability and robustness even when predictors are highly correlated [39].

By introducing this penalty, Ridge Regression achieves a better balance between bias

and variance, often improving generalization performance on unseen data.

Random Forests

The Random Forest algorithm is an ensemble learning method that combines multiple decision trees to improve predictive performance and robustness. Proposed by Breiman [41], it builds upon the idea that aggregating the predictions of many weak learners can lead to a strong and stable model. Each decision tree in the ensemble is trained with a random sample of the training data (bootstrap sampling), and at each internal split of the tree, a random subset of predictor variables is selected. This process, known as bagging (bootstrap aggregating), reduces variance and the risk of overfitting, increasing the model's generalizability [42].

Formally, given a set of B individual decision trees $T_b(x)$ trained on different subsets of the data, the Random Forest prediction is obtained by averaging (for regression) or taking a majority vote (for classification):

$$\hat{f}_{RF}(x) = \frac{1}{B} \sum_{b=1}^{B} T_b(x) \tag{2.5}$$

where $\hat{f}_{RF}(x)$ denotes the ensemble prediction for input x. The randomness introduced in both data sampling and variable selection makes the trees partially independent of each other, reducing the correlation between them and improving the overall performance of the ensemble.

Random Forests are non-parametric and can model complex, nonlinear relationships between variables without requiring prior assumptions about data distribution.

Feature importance analysis, an inherent property of Random Forests, also allows researchers to identify which variables most strongly influence audience ratings, adding interpretability to the model [41].

Gradient Boosting Machines

The Gradient Boosting Machine is an ensemble learning technique that, like Random Forests, combines multiple decision trees to achieve high predictive performance. However, unlike Random Forests, which build independent trees in parallel, Gradient Boosting constructs trees sequentially, with each new tree trained to correct the residual errors made by the previous ensemble [43].

The algorithm aims to minimize a loss function $L(y, \hat{y})$, which quantifies the error between the true values y and the model predictions $\hat{y}$. The model is built iteratively, starting from an initial approximation:

$$\hat{f}_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma) \tag{2.6}$$

Then, for each iteration $m = 1, 2, \ldots, M$, the following steps are performed:

Compute the residuals:

$$r_{im} = - \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right]_{f(x)=\hat{f}_{m-1}(x)} \tag{2.7}$$

Fit a regression tree $h_m(x)$ to predict the residuals r_{im}.

Update the model with a learning rate ν:

$$\hat{f}_m(x) = \hat{f}_{m-1}(x) + \nu \cdot \gamma_m h_m(x) \tag{2.8}$$

The process continues until the validation error stops improving or a maximum number of iterations M is reached. The learning rate ν controls how much each new tree contributes to the final model, acting as a regularization term that helps prevent overfitting [44].

Intuitively, Gradient Boosting can be understood as a form of functional gradient descent, where the algorithm incrementally fits weak learners (typically shallow decision trees) to the gradient of the loss function [40]. Although each tree individually has low

predictive power, the ensemble of many such trees can model highly complex, nonlinear relationships in the data.

Modern frameworks including XGBoost, LightGBM, and CatBoost extend the foundational algorithm introduced by Friedman, offering faster training, regularization, and parallelization capabilities that make them widely adopted in real-world forecasting tasks [45].

2.3 Deep Learning

Deep Learning is a subfield of Machine Learning that focuses on using artificial neural networks with multiple layers to model complex, high-dimensional data representations. While traditional machine learning algorithms rely on manually engineered features, Deep Learning models automatically learn hierarchical abstractions from raw data, making them particularly effective for large datasets and nonlinear relationships [3], [46].

Artificial Neural Networks (ANNs) are inspired by the structure and functioning of the human brain. The first formal model of an artificial neuron was proposed in 1943 by Warren McCulloch and Walter Pitts, who developed a logical system capable of simulating the behavior of biological neurons, establishing the foundation for modern artificial neural networks [47].

These networks have the ability to acquire, organize, and retain knowledge, in a way that resembles the human brain, albeit in a simplified form (Stanford University, 2024). Artificial neural networks are, in part, derived from the biological neural networks that constitute the human nervous system, where information is processed through the interaction of interconnected neurons [48].

By analogy to the biological structure, dendrites represent the inputs in an artificial neural network, as they receive signals from other neurons; the cell nucleus can be compared to the processing node or artificial neuron, where the weighted sum of inputs and the activation function are applied; synapses correspond to the weights, which determine the relative importance of each input; and the axon represents the output, transmitting

the resulting signal to other units [48], [49].

The analogy between biological and artificial neural networks continues to inspire ongoing research. Recent studies show that incorporating biologically inspired mechanisms, such as dendritic computation and synaptic plasticity, can enhance the efficiency and robustness of artificial neural networks [50].

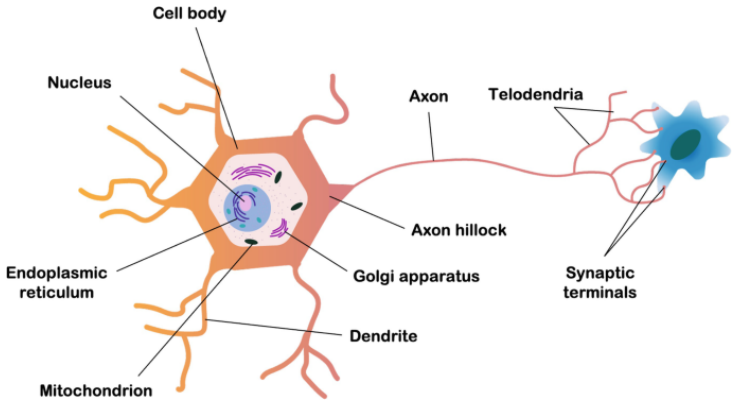


Figure 2.2: Biological Neural Network [51]

A Deep Neural Network (DNN) consists of an input layer, one or more hidden layers, and an output layer. Each layer is composed of units (or neurons) that transform their inputs through a weighted linear combination followed by a nonlinear activation function. The general form of the transformation at layer l is given by:

$$h^{(l)} = f(W^{(l)}h^{(l-1)} + b^{(l)}) \tag{2.9}$$

where $W^{(l)}$ and $b^{(l)}$ represent the weights and biases of layer l, $h^{(l-1)}$ denotes the activations from the previous layer, and $f(\cdot)$ is a nonlinear activation function [3].

An Artificial Neural Network (ANN) is typically composed of three main types of layers: input layers, hidden layers, and output layers. This structure forms the foundation of most neural architectures used in deep learning today [3] The input layer is the first layer of the network, responsible for receiving the data. The number of neurons in this layer typically corresponds to the number of features (or variables) in the dataset.

Between the input and output layers lie the hidden layers, which are responsible for detecting intermediate or higher-level features in the data. Each neuron in these hidden layers receives weighted inputs from the previous layer and applies a non-linear activation function, allowing the network to model complex relationships [52]. The number of hidden layers and neurons per layer depends on the complexity of the model and the task being solved. Increasing the number of hidden layers can improve the network ability to capture non-linear relationships, although it may also increase computational cost and the risk of overfitting [53]. Finally, the output layer produces the network final prediction. The number of neurons in this layer depends on the type of problem. For binary classification, there is typically one output neuron, which produces a probability value (between 0 and 1) through a sigmoid activation function. For multi-class classification, the output layer contains one neuron per class, often combined with a softmax activation to produce class probabilities [54]. For regression problems, the output layer usually has one neuron (for predicting a single continuous value) or multiple neurons (if predicting several continuous targets simultaneously). In this case, the output layer generally uses a linear activation function, allowing the network to output unrestricted real values [52].

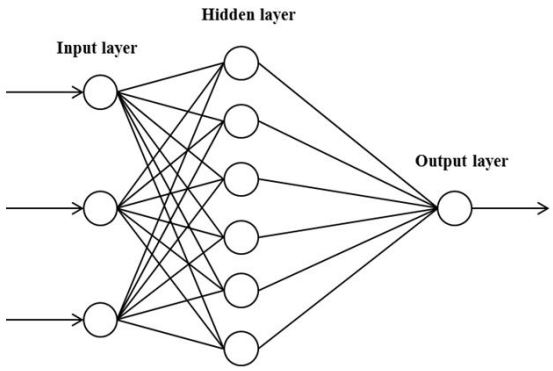


Figure 2.3: Structure of ANN with one hidden layer [55]

Activation functions play a fundamental role in artificial neural networks, as they introduce nonlinearity into the model. This property is essential for enabling the network to learn and represent complex mathematical functions that could not be modeled through purely linear transformations of the input [56].

By allowing nonlinear transformations of the data between neurons, activation functions enable the neural network to learn complex relationships and patterns present in the data [57].

Moreover, the choice of the activation function in the output layer strongly depends on the type of problem being addressed. For regression tasks, it is common to use a linear activation function in the output layer, allowing the model to produce continuous values within a real range. In contrast, classification problems often employ functions such as softmax or sigmoid, which enable the network outputs to be interpreted as probabilities associated with different classes [58].

Backpropagation is a fundamental algorithm used to train feed-forward neural networks. It operates by minimizing the cost function through the iterative adjustment of the network's weights and biases. During each epoch, the model updates these parameters in order to reduce the learning error by following the direction of the negative gradient, thus moving toward a local minimum of the cost surface.

The algorithm works in two main phases, a forward pass, in which input data propagate through the network to produce an output, and a backward pass, where the computed error is propagated backward from the output layer to the input layer. In this process, the algorithm calculates the partial derivatives of the error with respect to each parameter, which allows the network to efficiently update all weights and biases [59].

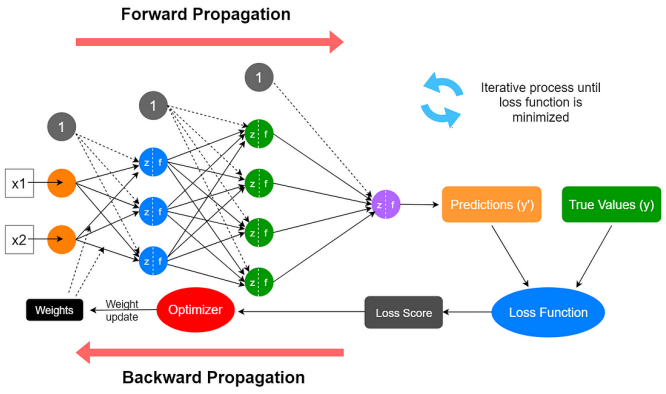


Figure 2.4: Backpropagation process [60]

Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are models specifically designed for processing sequential or temporal data. Unlike feedforward neural networks, which treat each input independently, RNNs incorporate feedback connections that enable them to maintain an internal memory, preserving information about previous states over time [61].

Formally, the hidden state of an Recurrent Neural Network (RNN) at time t is defined as:

$$h_t = f(W_h h_{t-1} + W_x x_t + b) \tag{2.10}$$

where x_t represents the input at time t, h_{t-1} is the hidden state from the previous step, W_h and W_x are weight matrices, b is a bias term, and $f(\cdot)$ is a nonlinear activation function. The corresponding output is calculated as:

$$\hat{y}_t = g(W_y h_t + c) \tag{2.11}$$

where $g(\cdot)$ is an activation function suitable for the given task (softmax for classification or a linear function for regression for example).

The fundamental characteristic of RNNs is recurrence, the presence of internal connections that cause the output at a given time step to depend not only on the current input but also on the previous hidden state. This structure creates a feedback loop where the hidden state h_{t-1} is used as part of the input for the next step h_t, allowing the model to maintain a dynamic memory of past events [62]. This memory, represented by the hidden state, serves as a summary of all temporal information accumulated up to the current step, enabling the network to identify sequential dependencies and patterns in the data.

Due to this recurrent structure, RNNs are able to capture short-term dependencies in sequential data. However, during training, these networks often face the problem of vanishing or exploding gradients, which makes it difficult to learn long-term dependencies [61], [62].

Long Short-Term Memory

LSTM networks are an advanced variant of RNNs to overcome the vanishing gradient problem inherent in traditional RNNs. The LSTM architecture introduces a memory cell and three gating mechanisms, the forget gate, the input gate, and the output gate, that regulate the flow of information through time.

The internal functioning of an LSTM cell is described by the following equations:

$$\begin{aligned} f_t &= \sigma(W_f[h_{t-1}, x_t] + b_f) \\ i_t &= \sigma(W_i[h_{t-1}, x_t] + b_i) \\ \tilde{C}_t &= \tanh(W_C[h_{t-1}, x_t] + b_C) \\ C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\ o_t &= \sigma(W_o[h_{t-1}, x_t] + b_o) \\ h_t &= o_t * \tanh(C_t) \end{aligned} \tag{2.12}$$

, where f_t is the forget gate, which decides which information to discard; i_t is the input gate, which determines what new information to store; $\tilde{C}_t$ is the candidate cell state; is the updated cell state (long-term memory); o_t is the output gate; and h_t represents the hidden state (short-term memory). In LSTM networks, the concept of memory is expanded and managed more precisely through these gating mechanisms [63]. Gates control the flow of information within the memory cell, determining what should be kept, updated, or discarded at each time step.

This allows the LSTM to distinguish between short-term memory (h_t) and long-term memory (C_t), enabling it to preserve relevant information across longer periods and remove what is no longer useful. Thanks to these mechanisms, LSTMs can preserve and update relevant information across long sequences, allowing the model to learn complex temporal dependencies [63], [64].

2.3.1 Statistical models

This section addresses two of the most widely used models for this type of analysis, ARIMA and SARIMA, an extension of the former that incorporates seasonal components. Both models allow for modeling and forecasting future values based on past observations, providing an interpretable and statistically grounded approach to time series forecasting.

ARIMA

ARIMA model is a classical statistical method for time series forecasting. Proposed by Box and Jenkins, ARIMA combines three components, autoregression (AR), differencing (I), and moving average (MA), to model both trend and temporal dependencies in a stationary time series [65].

The general ARIMA(p, d, q) model can be expressed as:

$$\Phi_p(B)(1 - B)^d y_t = \Theta_q(B)\varepsilon_t \tag{2.13}$$

, where y_t is the observed value at time t, B is the backshift operator ($By_t = y_{t-1}$), $\Phi_p(B)$ and $\Theta_q(B)$ are polynomials of order p and q, corresponding to the autoregressive and moving average parts, d denotes the degree of differencing to achieve stationarity, and ε_t is a white noise error term [66]

The autoregressive component captures the dependence of the current value on past observations, the integration component removes trends through differencing, and the moving average component models the dependence on past forecast errors.

SARIMA

SARIMA model extends ARIMA by incorporating additional seasonal autoregressive and moving average components.

SARIMA model can be expressed as:

$$\Phi_p(B)\Phi_P(B^s)(1 - B)^d(1 - B^s)^D y_t = \Theta_q(B)\Theta_Q(B^s)\varepsilon_t \tag{2.14}$$

, where (p, d, q) are the non-seasonal ARIMA parameters, (P, D, Q) are the seasonal ARIMA parameters, s is the seasonal period (example, $s = 7$ for weekly seasonality in daily data), B is the backshift operator ($By_t = y_{t-1}$), and ε_t is the error term [67].

2.4 Model evaluation metrics

Evaluating model performance is a fundamental step in the forecasting process, as it quantifies how accurately a model predicts future values of a time series. Regardless of the type of model, it is essential to measure how close the predicted values are to the actual observations [68].

In the context of television audience forecasting, the most widely adopted performance metrics are described below.

MAE

The MAE measures the Mean Absolute Error between predictions and actual values. It is defined as:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.15}$$

A lower MAE value indicates more accurate predictions. MAE is intuitive and easy to interpret, representing the average deviation between actual and predicted audience levels.

Mean Squared Error (MSE)

The MSE computes the average of squared differences between predictions and actual values:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.16}$$

MSE penalizes larger errors more severely due to squaring. It is particularly useful when significant deviations, such as sudden spikes in audience ratings, should be emphasized during model evaluation.

RMSE

RMSE is the square root of MSE:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.17}$$

Lower RMSE values indicate better model performance, especially when penalizing large errors is important.

Mean Absolute Percentage Error (MAPE)

MAPE expresses the average relative error in percentage terms:

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{2.18}$$

However, care must be taken when actual values (y_i) approach zero, as the metric can become unstable.

Coefficient of Determination (R^2)

R^2 measures how much of the variability in the observed data is explained by the model:

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} \tag{2.19}$$

An R^2 value close to 1 indicates that the model explains most of the variance in the data, whereas values near 0 suggest poor explanatory power. Although R^2 is widely used in regression analysis, it may not be reliable for strongly seasonal or autocorrelated time series [68], [69].

Chapter 3

Methodological Approach and Data Analysis

This chapter outlines the methodological approach adopted in this research. It begins by presenting the research questions that guided the study, followed by the formulation of hypotheses derived from both theoretical and empirical considerations. Subsequently, the procedures related to data collection, preprocessing, and exploratory analysis are detailed, forming the basis for the subsequent modelling and forecasting stages.

3.1 Research Questions

- How can historical audience data and external contextual factors be combined to improve the accuracy of television audience forecasts?
- Which modelling approaches (example, statistical, machine learning, or hybrid) provide the best predictive performance for audience estimation?
- What are the most influential variables affecting audience fluctuations?

3.2 Hypotheses

The hypotheses presented in this section were formulated based on the theoretical framework and the state of the art, reflecting the main factors identified as relevant to television audience forecasting. These hypotheses aim to assess the impact of incorporating exogenous variables, derived temporal representations, and nonlinear models on the predictive performance of the forecasting models.

The hypotheses tested in this study are as follows:

- Incorporating exogenous variables (example, holidays, special events, season, day of the week) significantly improves the accuracy of television audience forecasts compared to models based solely on historical data.
- The inclusion of derived temporal variables, such as seasonal components of hour and month represented using trigonometric functions (sine and cosine), enhances model performance by capturing nonlinear cyclic patterns in audience behaviour.
- Nonlinear models, particularly recurrent neural networks, demonstrate a superior ability to capture complex temporal dependencies and seasonal patterns compared to traditional linear models, resulting in more robust forecasts.

3.3 Data Collection and Preprocessing

The quality and reliability of predictive models strongly depend on the data used in their development. For this reason, data collection, exploration, and preparation are fundamental steps in any forecasting project. This section describes the process of obtaining and analyzing the dataset, as well as the main techniques applied to data cleaning, transformation, and the selection of relevant variables. Topics such as exploratory data analysis (EDA), feature engineering, the detection and treatment of outliers, and the evaluation of correlations between variables are discussed. These steps ensure data integrity and consistency.

3.3.1 Dataset

The present investigation relied on data from the YUMI platform, developed by the company TWA, whose purpose is to provide structured television information. Within the scope of the methodological definition, it was established that data collection would cover the period starting in 2021, encompassing the four main Portuguese generalist television channels, namely RTP1, RTP2, SIC, and TVI.

However, given the large scale and complexity of the available information, it proved unfeasible to extract and store the entire dataset for the period 2021 to 2025 in a single dataset within the YUMI platform. To overcome this technical limitation, the data collection was divided into four independent datasets, each corresponding to a specific annual interval: 2021, 2022, 2023, and finally the biennium 2024–2025 (up to March 2025). Subsequently, the four datasets were concatenated, thus creating a single consolidated dataset, which served as the analytical basis for the present study. With regard to its structure, the final dataset consisted of a comprehensive set of variables, which are organized in the following tables, 3.1 and 3.2.

Table 3.1: Structural variables of the dataset

Variable	**Description**	**Type**
Channel	Identification of the TV channel	Categorical
Date	Transmission date (year, month, day)	Temporal
Program	Title of the program	Categorical
Type 1	Main category (e.g., fiction, sports, news, etc.)	Categorical
Type 2	Subcategory derived from Type 1 (e.g., series, movie, etc.)	Categorical
Start Time	Program start time (02:30:00–26:29:59)	Temporal
End Time	Program end time (02:30:00–26:29:59)	Temporal
Duration	Total duration of the program	Numerical
Day of Week	Day of the week of the broadcast	Categorical
Host/Pivot	Name of the host/pivot (if available)	Categorical

Table 3.1 presents the structural variables of the dataset, which describe the main

characteristics of each television program, such as transmission date, duration, and categorical attributes. Complementarily, Table 3.2 lists the audience-related variables, which quantify viewer engagement through different segmentation dimensions, including gender, age group, and geographic region.

Table 3.2: Audience variables of the dataset

Variable	**Description**	**Segmentation**
Universe ratings (%)	Percentage of viewers in the Portuguese population	Global
Universe shares (%)	Percentage of viewers among those watching TV at the same moment	Global
Masculine ratings/shares (%)	Male audience distribution	Gender
Feminine ratings/shares (%)	Female audience distribution	Gender
Age group ratings/shares (%)	Audience distribution by age groups: 4–14, 15–24, 25–34, 35–44, 45–54, 55–64, 65–75+	Age
Regional ratings/shares (%)	Audience distribution by region (North, Center, Lisbon, Alentejo, Algarve)	Geographic
Sales Responsible ratings/shares (%)	Audience metrics for individuals responsible for sales	Global
Non–responsible for sales ratings/shares (%)	Audience metrics for individuals not responsible for sales	Global

3.3.2 Exploratory Data Analysis

The dataset employed in this study initially contained 45 columns (considering "channel" as the header) and 696,142 rows, covering the chronological period from January 1, 2021, to March 31, 2025. Exploratory data analysis was conducted to ensure data integrity and consistency, as well as to identify relevant patterns for subsequent television audience studies. It is important to note that the temporal scope of the dataset encompasses a period strongly influenced by the COVID-19 pandemic, particularly during the

year 2021. During the COVID-19 pandemic, television consumption increased sharply, including among segments of the population that traditionally did not watch TV frequently, such as young people [70]. This context is analytically relevant, as the lockdown measures and mobility restrictions implemented during that time had a direct impact on media consumption patterns. Specifically, the increase in time spent at home likely led to higher television viewership and shifts in viewing habits, both in terms of scheduling and preferred content types. Therefore, the inclusion of data from this period should be considered when interpreting the results, given its potential influence on the observed audience behavior trends.

Data Preparation

The "Date" column was converted to the datetime format to enable consistent temporal operations. Missing values were primarily identified in the Host/Pivot column and were replaced with "Unknown". The "Duration" column has also been converted to seconds. To ensure the correct temporal sequence, the "Date" column was sorted chronologically.

Feature Engineering

Preliminary analysis revealed opportunities to generate new variables from existing information to enrich modeling and capture seasonality or temporal patterns showed in Table 3.3.

The "Start Time" and "End Time" columns exhibit cyclical characteristics: for instance, 24:00 and 01:00 are temporally close but numerically distant, which could mislead predictive models into interpreting them as highly different values. To address this issue, these columns were transformed into trigonometric representations (sine and cosine), preserving temporal proximity between adjacent hours. Similarly, the "Month" column was converted into trigonometric values, allowing models to correctly interpret annual seasonality. After the creation of derived variables and necessary transformations, the dataset comprised to 50 columns.

Table 3.3: Feature Engineering

Variable	**Description**	**Source**
Season	Season of the year (Summer, Autumn, Winter, Spring)	Created from the month and day in the Date column
Month	Month of the broadcast	Extracted directly from the Date column
Holiday	Binary indicator (yes/no) for whether the day is a holiday	Determined from the Date
StartTimeSin	Trigonometric representation of program start time (sine)	Transformed from Start Time column
StartTimeCos	Trigonometric representation of program start time (cosine)	Transformed from Start Time column
EndTimeSin	Trigonometric representation of program end time (sine)	Transformed from End Time column
EndTimeCos	Trigonometric representation of program end time (cosine)	Transformed from End Time column
MonthSin	Trigonometric representation of month (sine)	Transformed from Month column to capture annual seasonality
MonthCos	Trigonometric representation of month (cosine)	Transformed from Month column to capture annual seasonality

In order to enable the use of categorical information in predictive modeling, several categorical variables were transformed into dummy variables. The variables converted included: "Program", "Day", "Type1", "Type2", "Host/Pivot", "Holiday", and "Season". The conversion to dummy variables is necessary because most machine learning algorithms require numerical input. By creating a binary representation for each category, the model can interpret the presence or absence of a specific category in each observation. This transformation preserves the categorical information while making it compatible with algorithms such as regression and decision trees.

During this process, the parameter ($drop\_first = True$) was applied. This choice is justified by the issue of multicollinearity, which occurs when one variable can be perfectly predicted by a combination of other variables. In the context of dummy variables, if all categories are included, the sum of the dummy columns would always equal one, introducing perfect linear dependence. Dropping the first category prevents this problem while retaining all the necessary information, as the omitted category is implicitly represented

by the combination of the remaining dummies.

Outliers

The detection and treatment of outliers is a fundamental step in data preparation, as these values can affect both exploratory analysis and the performance of predictive models. However, their interpretation must be contextualized, distinguishing between recording errors and real, relevant events.

Outliers were identified using the Interquartile Range (IQR) method, which is widely applied in statistical data analysis. The IQR corresponds to the difference between the third quartile (Q3) and the first quartile (Q1), representing the central dispersion of 50% of the data. According to the Tukey rule, outliers are defined as values that satisfy the following conditions:

$$\text{IQR} = Q_3 - Q_1$$

$$T_{\min} = Q_1 - c \cdot \text{IQR}, \quad T_{\max} = Q_3 + c \cdot \text{IQR}$$

$$x \text{ is an outlier if } x < T_{\min} \text{ or } x > T_{\max}$$

Where c is decided by the user, but usually set to 1.5 [71].

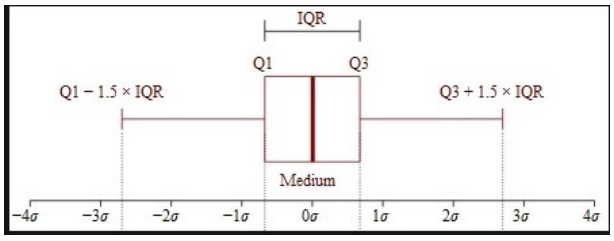


Figure 3.1: IQR [72]

In addition, boxplots were used to visually represent outliers, allowing a clear identification of data points that lie outside the conventional limits of the distribution, as illustrated in Figures 3.2 - 3.4.

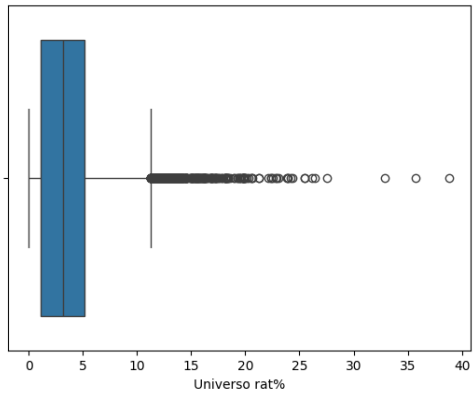


Figure 3.2: Outliers TVI

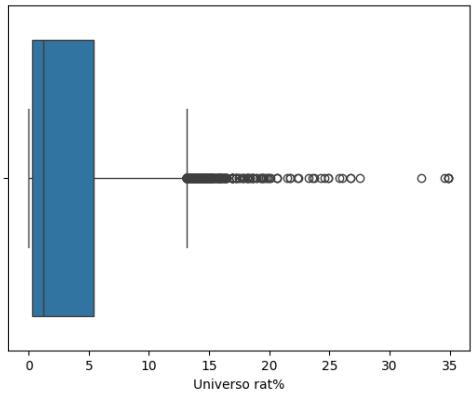


Figure 3.3: Outliers SIC

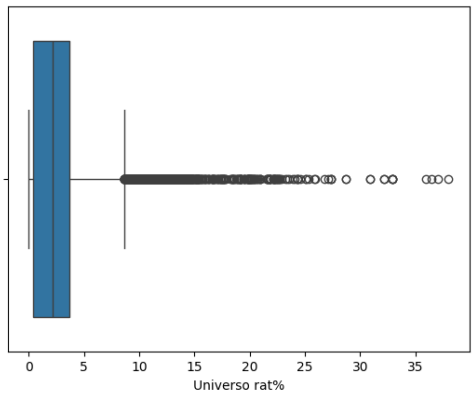


Figure 3.4: Outliers RTP1

The analysis revealed that each TV channel presents a set of values classified as outliers, according to the thresholds defined by the IQR:

- TVI: values above 11% of rat% were classified as outliers. A total of 623 outliers were identified in 224,885 observations. Most of these cases correspond to sports broadcasts, particularly football matches, explaining the audience peaks observed. The maximum value recorded was 38.9%, associated with a sports event.

- SIC: values above 13% of rat% were classified as outliers. A total of 435 outliers were identified in 194,889 observations. Again, the largest peaks correspond to sports broadcasts, with the maximum value reaching 34.6%.

- RTP1: values above 8.5% of rat% were classified as outliers. A total of 2,841 outliers were identified in 140,716 observations, a relatively higher number compared to the other channels. Sports events once again stand out (with a maximum peak of 38%), as well as the entertainment show "O Preço Certo", which reached audience values ranging between 8.7% and 12.2%, thus exceeding the IQR-defined threshold.

Following the identification of outliers, it was decided not to remove them from the dataset, since these represent real phenomena of high relevance in television consumption, such as sports broadcasts or popular entertainment shows. Excluding these values would result in the loss of critical information for the purpose of this research, which aims precisely to forecast television audience patterns, including unexpected peaks.
The possibility of creating a dummy variable to flag observations classified as outliers was also considered. However, this approach was deemed unnecessary, as the dataset already includes explicit variables that capture the factors responsible for such peaks. Specifically:

- The program type variable distinguishes categories such as sports, drama, or entertainment, enabling the model to learn that certain types (especially sports) are associated with higher audience peaks.

- The start and end time in sine and cosine variables further strengthen this differentiation, as the highest audience events tend to consistently occur in specific time slots (for example, prime-time sports broadcasts). Consequently, even programs of the same type broadcast at different times can be distinguished by the model in terms of their audience impact.

Exploratory Data Analysis

This aims to visually characterize audience behaviour and identify patterns that will inform the subsequent modelling stage. We analyse the temporal evolution of the target variable, Universe Rating (%), over the study period, stratified by the four channels (RTP1, RTP2, SIC, and TVI). The approach focuses on describing short and medium term dynamics, highlighting potential seasonal patterns (weekly and annual), and deriving direct implications for feature selection in predictive models.

To smooth out daily volatility and highlight underlying patterns, each daily series was accompanied by two smoothing curves:

- 7-day rolling mean: captures short-run (weekly) dynamics, typically reflecting differences between weekdays and weekends/prime time.
- 30-day rolling mean: highlights medium-run trends and seasonality, associated with monthly variation or persistent schedule effects.

The joint reading of these curves helps separate transient noise from persistent movements, supporting the later inclusion of temporal features such as day of the week, month, and season in the predictive models, as illustrated in Figures 3.5 - 3.8.

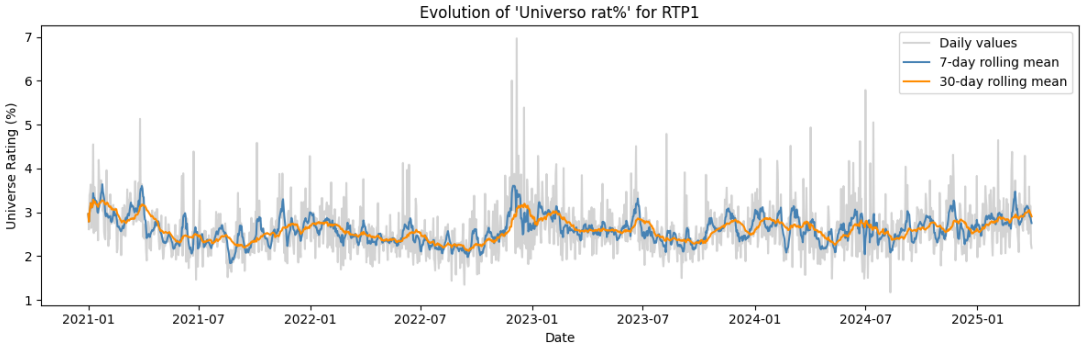


Figure 3.5: RTP1: stable medium-run trend; weekly oscillations and episodic peaks.

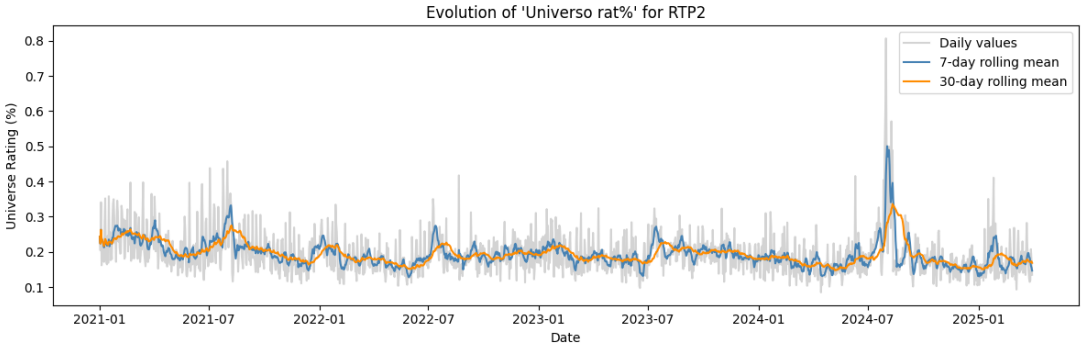


Figure 3.6: RTP2: low baseline; isolated mid-2024 surge and rapid mean reversion.

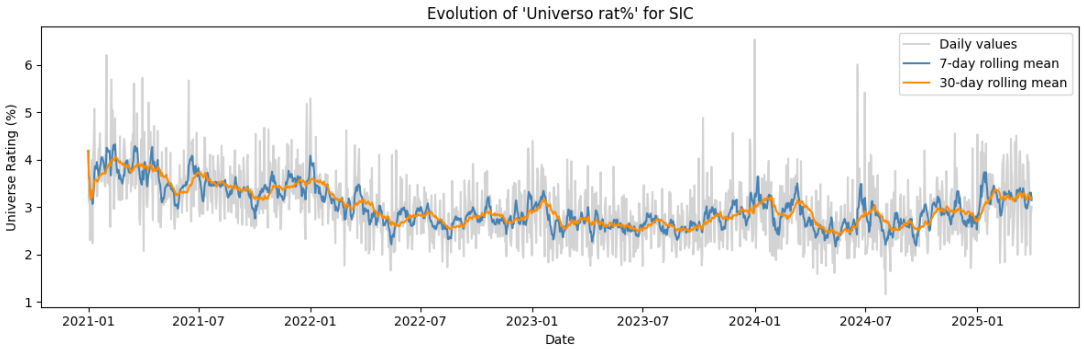


Figure 3.7: SIC: short-run volatility; decline in 2021–2022 and recovery in 2023–2025.

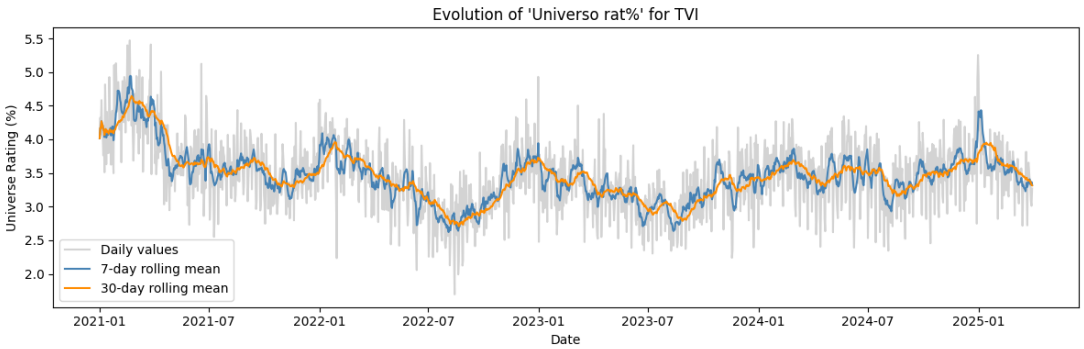


Figure 3.8: TVI: high baseline; pronounced weekly cycles; medium-run upward drift.

Across the four broadcasters, TVI and SIC exhibit the highest baselines and the greatest short-run variability, with frequent peaks. In TVI, these bursts coincide with a medium-run upward drift in the 30-day mean during 2023–2024, while SIC follows a

U-shaped medium-run path, softening through 2021-2022 and recovering from 2023 onward. RTP1 maintains a stable baseline with moderate, recurrent spikes: the 7-day curve oscillates around the 30-day curve, indicating a clear weekly rhythm but only weak-to-moderate medium-run seasonality. RTP2 records the lowest average levels and the tightest dispersion, interrupted by an isolated surge in mid-of-2024 that briefly lifts both rolling means before swift reversion, most likely associated with the 2024 Olympic Games, suggesting a steady audience punctuated by occasional specials. Overall, the contrast between the 7-day and 30-day curves makes weekly periodicity salient, especially in TVI and SIC, while medium-run movements are channel-specific, supporting the inclusion of day-of-week, month, season, and genre as exogenous features and the retention of event-driven peaks as informative signals.

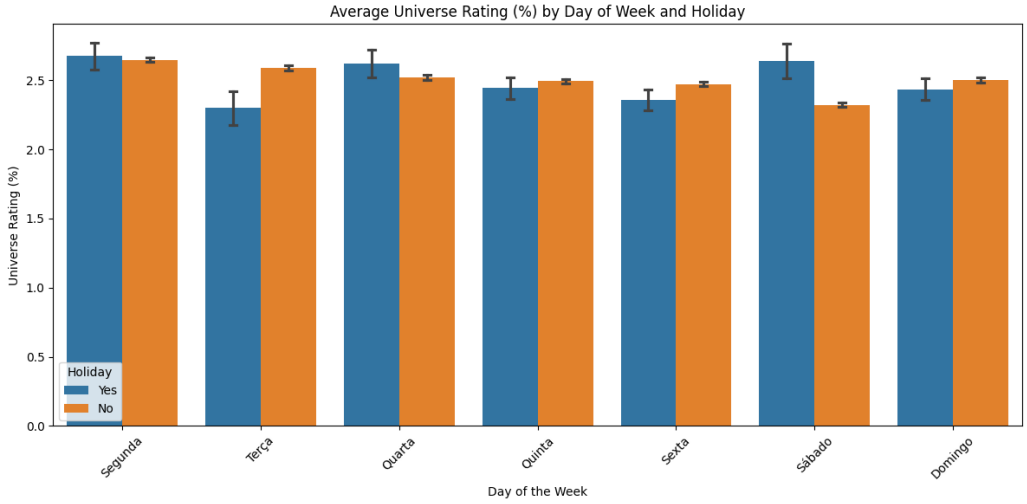


Figure 3.9: Average rating by day of week and holiday

The Figure 3.9 presented illustrates the average television audience throughout the week, distinguishing between regular days and holidays. The analysis shows that both the day of the week and the holiday condition significantly influence viewing patterns. It is observed that Monday consistently concentrates the highest audience levels, regardless of whether it is a holiday. This result may be associated with the fact that Monday marks the beginning of the workweek. Another relevant aspect is the behavior observed on Saturdays during holidays, television audiences increase significantly. In contrast, during

holidays that occur on Tuesdays or Fridays, there is a decline in audience levels.

The analysis of the 95% confidence intervals (CI 95%), represented by the error bars, adds an important layer to the interpretation. On regular days, the bars are shorter, suggesting greater consistency and predictability of audience behavior. On the other hand, on holidays, the bars are often longer, reflecting greater variability in viewing patterns. This variability may be related to the diversity of ways in which families and individuals spend holidays, some choosing leisure activities outside the home, while others staying at home, increasing their television exposure.

In summary, the analysis demonstrates that television consumption is strongly conditioned by temporal variables, not only by the day of the week but also by the occurrence of holidays. These findings highlight the importance of incorporating temporal dimensions into audience studies.

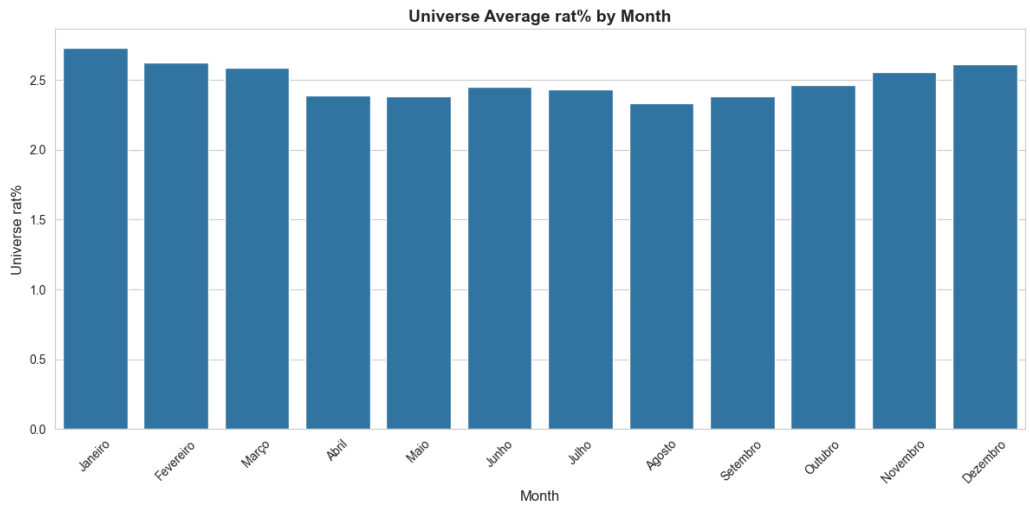


Figure 3.10: Average rating by Month (All Channels)

Figure 3.10 presents the average audience aggregated by month of the year for the complete dataset, encompassing all four channels, allowing the observation of seasonal behaviour throughout the analysed period. A relatively stable annual pattern can be observed, with higher values during the winter months, particularly in January, February, and December, and gradual reductions during the spring and summer months (April to August). This behaviour is consistent with the annual cycle of television consumption,

characterised by greater time spent at home during colder months and lower viewing activity during summer. These results confirm the presence of annual seasonality, albeit moderate, and justify the inclusion of temporal variables such as month and season in the predictive models, as they capture structural variations in audience behaviour across the year.

To gain a more detailed understanding of the seasonal differences across channels, the same monthly analysis was carried out individually for each of the four broadcasters. Figures 3.11 - 3.14 presents the evolution of the average monthly audience for RTP1, RTP2, TVI, and SIC, respectively. The analysis reveals distinct patterns among the channels: TVI shows a clear peak during the first months of the year (January and February), followed by a gradual decline until August and a subsequent recovery towards December. SIC exhibits a similar trend, marked by a decrease during the spring and summer months and a recovery at the end of the year. RTP1 follows a comparable seasonal trajectory, while RTP2 displays an isolated peak in July. Overall, the results indicate a clear presence of seasonality in television audience behaviour.

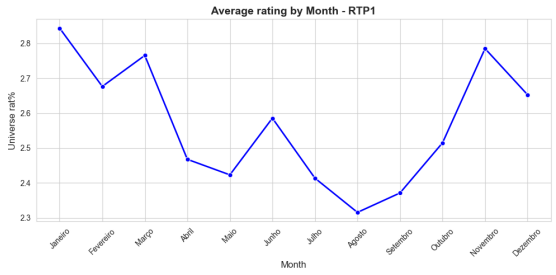


Figure 3.11: Avg. rating by Month (RTP1)

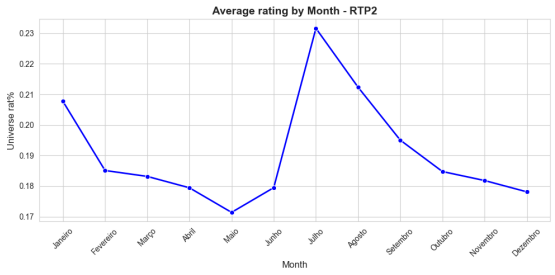


Figure 3.12: Avg. rating by Month (RTP2)

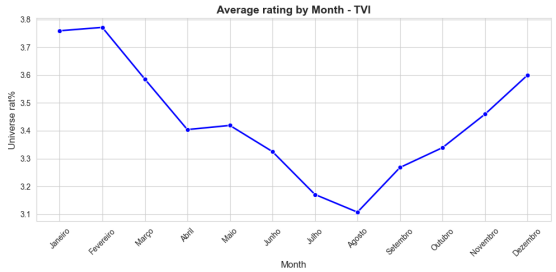


Figure 3.13: Avg. rating by Month (TVI)

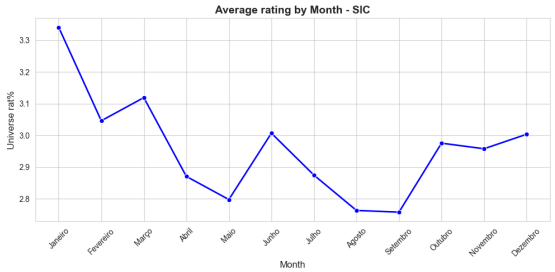


Figure 3.14: Avg. rating by Month (SIC)

Figures 3.15 - 3.18 presents the average audience aggregated by season for each channel. These results consolidate the evidence of annual seasonality identified in the monthly analyses and reinforce the relevance of including season and month variables in the predictive audience models, as they capture structural variations in viewer behaviour throughout the year.

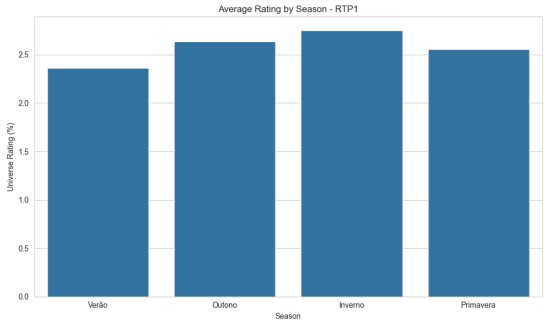


Figure 3.15: Avg. rating Season RTP1

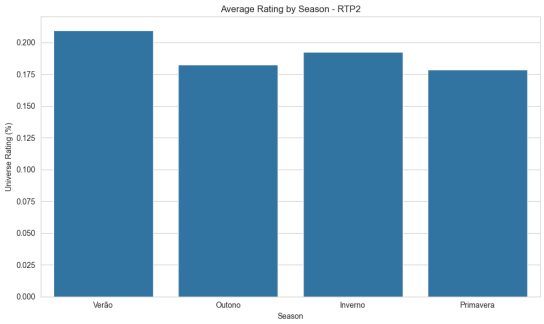


Figure 3.16: Avg. rating Season RTP2

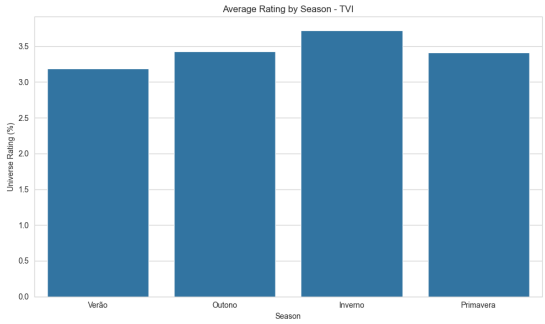


Figure 3.17: Avg. rating Season TVI

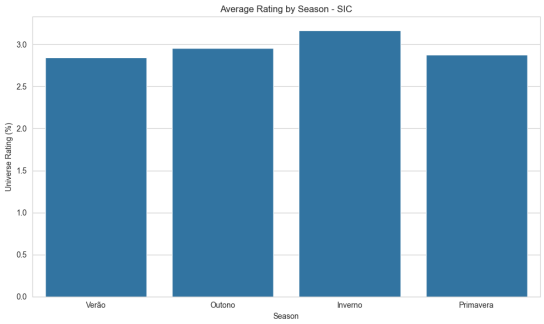


Figure 3.18: Avg. rating Season SIC

The analysis of average audience by hour required specific preprocessing of the hourly data, given that the considered television schedule extended from 02:30 to 26:29. For visualization purposes, hours beyond 23h were converted, that is, hour 24 was represented as 0, hour 25 as 1, and hour 26 as 2. This transformation was applied solely to enable the construction of the graphs, ensuring coherent interpretation within the interval.

The results showed in Figures 3.19 and 3.20 confirmed the existence of strong hourly seasonality in television consumption. There is low viewership during the early morning hours, a gradual increase from late morning, an intermediate peak in the early afternoon, and above all, a pronounced maximum during prime time, followed by a progressive decline as the night advances.

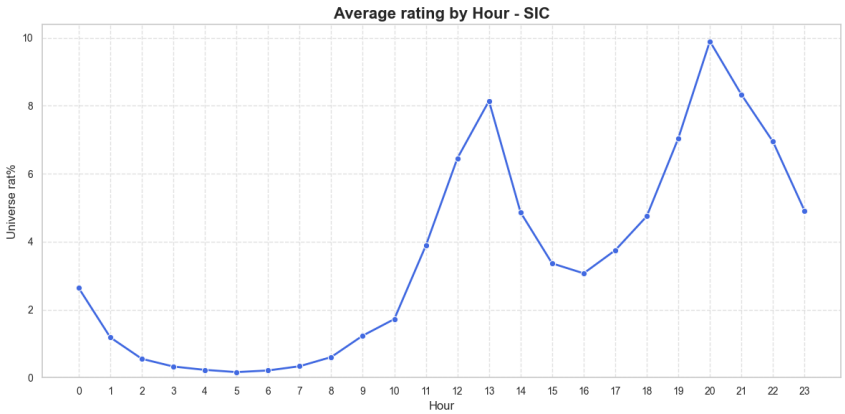


Figure 3.19: Average rating by Hour (SIC)

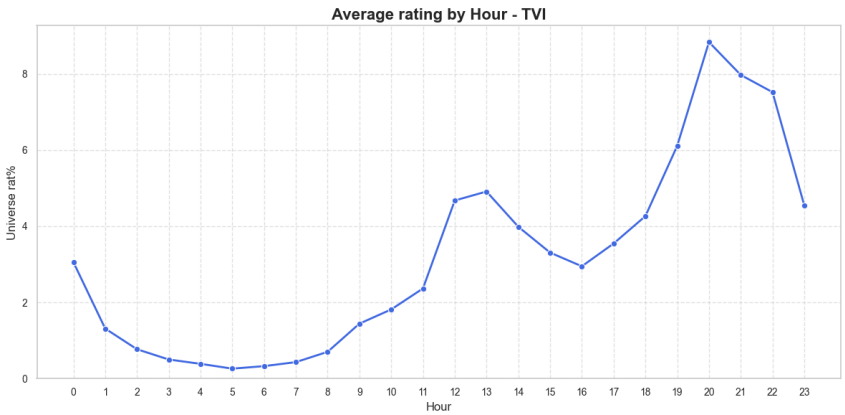


Figure 3.20: Average rating by Hour (TVI)

Figure 3.21 shows the evolution of ratings over time for the main generalist TV channels in Portugal, highlighting variations between channels and the occurrence of peaks associated with specific events or programs.

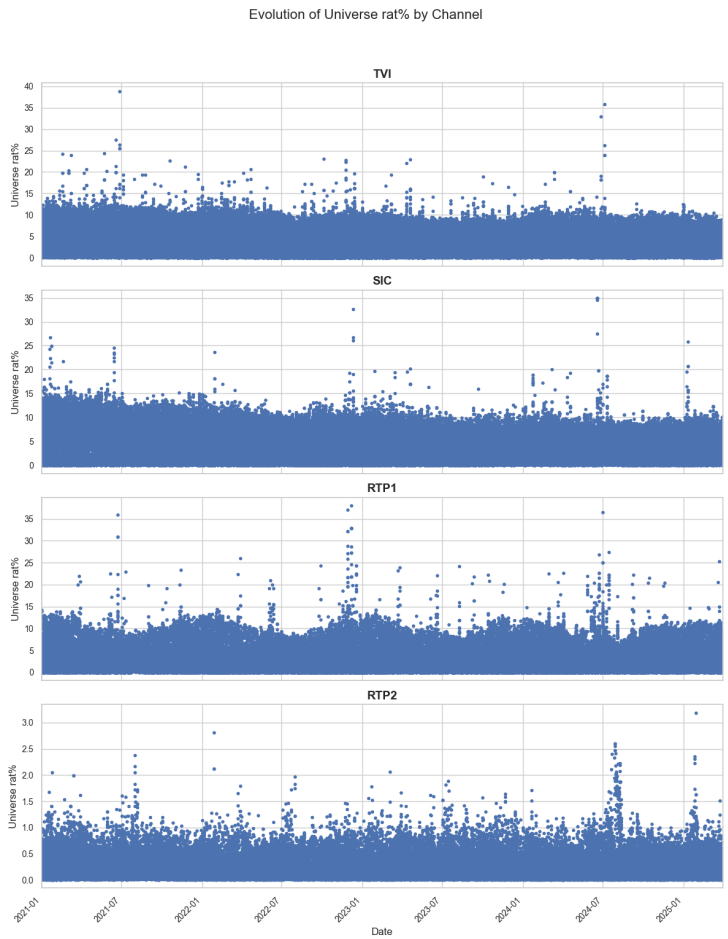


Figure 3.21: Evolution of Universe rat% by channel

I considered it relevant to extend the investigation to the endogenous variables present in the dataset. These variables are only known after the broadcast and therefore cannot be directly used in predictive models. However, their graphical and descriptive exploration proves to be important, as it allows for a broader understanding of audience behaviour and provides a complementary perspective to analyses based on exogenous factors.

The influence of age group and gender on television audiences is a relevant phenomenon that deserves careful analysis. Studies show that both age and gender play significant roles in programming preferences and media consumption habits.

First, age group is a critical determinant in television consumption. Research indicates that audience patterns vary considerably across different age groups, with younger people

tending to consume more digital content and interact on social media platforms, which impacts their preferences for traditional television programs [73].

Regarding gender, data reveal divergences in program preferences between men and women. Men usually prefer sports and action content, while women tend to be more interested in dramas and romantic comedies [74].

The analysis of television consumption in Portugal shows that television has the greatest affinity among women, older individuals, and residents of the northern region of the country [75].

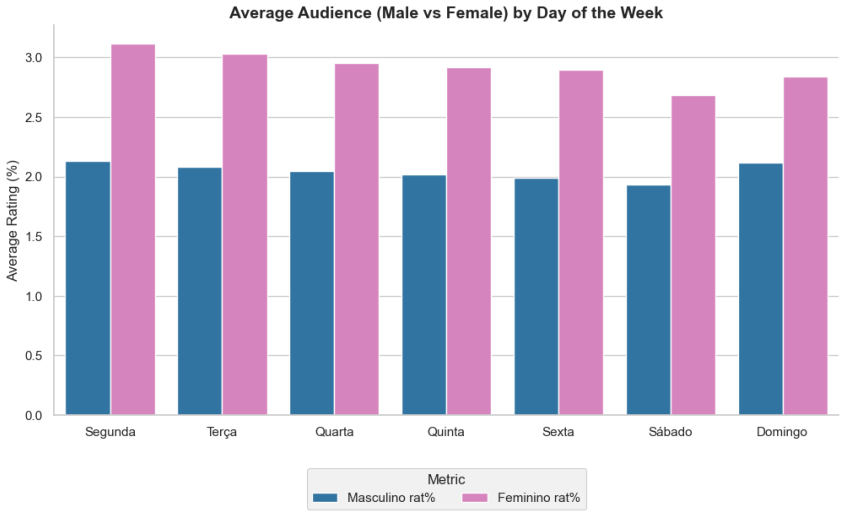


Figure 3.22: Average Audience (Male vs Female) by day of the Week

Figure 3.22 illustrates the variation in average audience by gender throughout the week. It is possible to observe that female viewers consistently show higher average ratings than male viewers across all days, with a particularly noticeable gap at the beginning of the week.

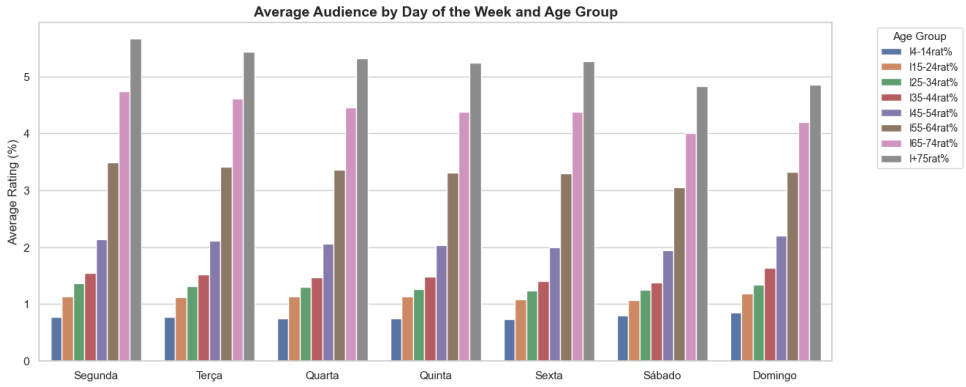


Figure 3.23: Average Audience by day of the Week and Age Group

Figure 3.23 expands this analysis by examining audience behaviour across different age groups. The results show that older age groups tend to have higher average ratings, while younger viewers exhibit lower levels of traditional television consumption throughout the week. Figures 3.22 and 3.23 confirm the demographic patterns discussed above.

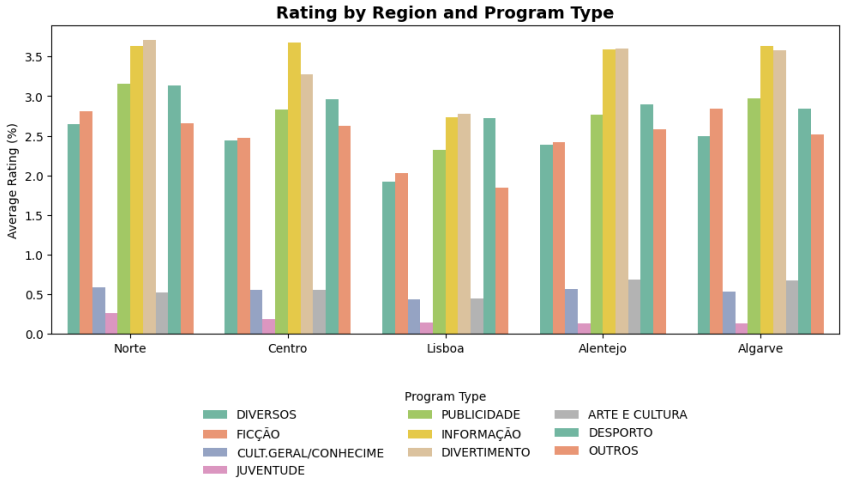


Figure 3.24: Average Audience by Region and Program Type

The analysis of Figure 3.24 reveals that the regions of the North, Center, Alentejo, and Algarve consistently present higher ratings compared to Lisbon, the Portuguese capital. This difference indicates that, proportionally, a smaller percentage of the Lisbon population is watching television.

It is important to recall that the television rating is a relative measure; in this case,

it represents the percentage of the population in a given region that was watching the program in question. Thus, more populated regions, such as Lisbon, do not necessarily present higher ratings, since this indicator is calculated within the region's own base.

$$\text{Rating (By Region) (\%)} = \frac{\text{Number of viewers in the region}}{\text{Total population of the region}} \times 100$$

In this context, the lower ratings observed in Lisbon can be interpreted as a reflection of sociocultural differences between regions. According to recent studies, audiovisual consumption in Portugal has been fragmenting, with more urban and younger areas, such as Lisbon and the Metropolitan Area, being more prone to digital and on-demand consumption [76]. This trend is further supported by [77], which highlights a greater penetration of streaming platforms in urban regions and among younger age groups.

Additionally, data from Marktest in 2024 shows that, despite television still maintaining high national reach (84.7%), there are signs of a progressive migration of consumption toward digital platforms. [75]

Figure 3.24 illustrates that Portuguese television behavior is not homogeneous.

Feature Correlation

The analysis of correlations among variables constitutes a fundamental step in understanding the dataset and selecting the most appropriate features for predictive modeling. This process involves differentiating between endogenous and exogenous variables, interpreting the limitations of linear correlation measures, and considering multicollinearity among candidate features.

Endogenous variables are those whose values are only known after the occurrence of the observed event. Examples in this dataset include ratings and shares segmented by gender or by geographic region. These variables are highly relevant for descriptive analyses, as they provide detailed insights into audience behavior. However, they are not suitable as predictive features because their values are unavailable at the time a forecast must be made. Using these variables in predictive models would introduce data leakage,

compromising the validity of the results.

Exogenous variables, in contrast, are known a priori (before the event occurs). In this dataset, they include attributes such as broadcast time (Start Time and End Time), program type (Type1 and Type2), season, day of the week, and holiday indicators. These variables are therefore suitable candidates for model features, as they allow the model to anticipate audience patterns without relying on information from the future.

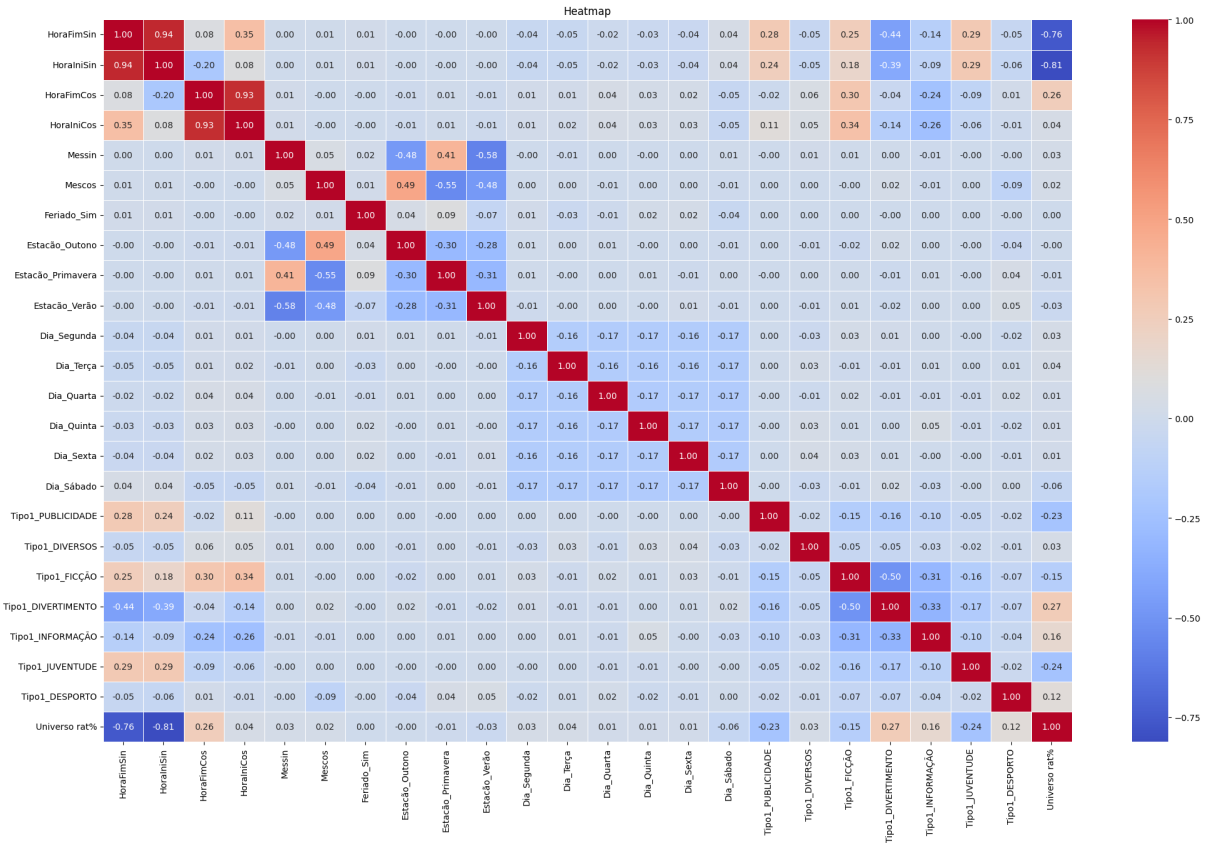


Figure 3.25: Heatmap for Type1

The correlation analysis using a heatmap of exogenous variables revealed that some features, which were expected to be highly relevant, such as Holiday, exhibited a low correlation with the target variable, as illustrated in Figure 3.25 and Figure 3.26.

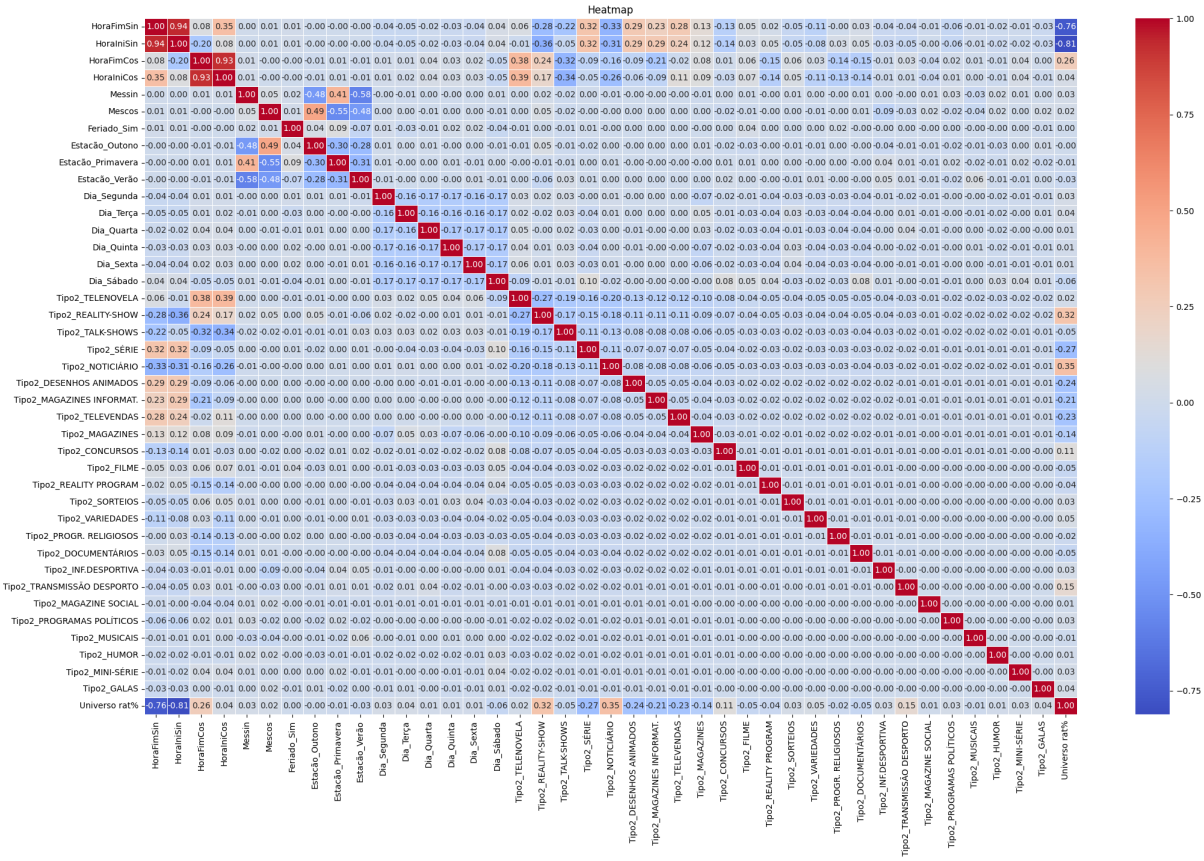


Figure 3.26: Heatmap for Type2

This can be explained by the fact that Pearson correlation, which is the default method in Pandas, captures only linear relationships between variables. Many relationships in this dataset are non-linear. For Example:

- The Type1 variable does not exhibit a linear relationship with ratings, since certain types, such as sports, tend to be broadcast only at specific times (example, late afternoon or prime time) or on specific days (weekends).

- Similarly, Holiday or Season influence audience behavior in a conditional manner: a holiday with a major sports event may produce a high audience peak, while a holiday with regular programming may not.

Thus, the apparent lack of linear correlation does not imply that these variables are irrelevant. Sophisticated models, are capable of capturing non-linear relationships and

complex interactions between features.

Variables such as Program and Host/Pivot were considered but ultimately excluded for two main reasons:

- Limited generalizability: During prediction, the model may encounter programs or presenters that were not seen during training. In such cases, unseen values would be treated as noise, reducing predictive robustness.
- For Host/Pivot, most records were labeled as "Unknown," significantly limiting the utility of this feature.

Excluding these variables ensures that the model relies on generalizable, consistently available features, improving both robustness and interpretability.

In this dataset, Type1 (main program type) and Type2 (subtype) naturally exhibit a high correlation, as subtypes are directly dependent on the main type. To reduce redundancy and avoid predictive instability, only one of these types was selected. The choice and its justification will be presented and explained in the following chapters.

Although non-linear models are less sensitive to multicollinearity, this practice helps reduce the risk of overfitting and enhances model interpretability.

Chapter 4

Data Preparation and Model Development

This chapter describes the development and implementation of the predictive models used in this research. Following the approach and exploratory analysis presented in the previous chapter, the focus now shifts to the practical application of these concepts. The process includes the stages of data preprocessing, variable selection, model training, and optimization. Thus, the aim of this chapter is to systematically and transparently describe the procedures that led to the construction of the television audience forecasting framework. Also highlighting difficulties encountered and how they were overcome.

4.1 Data Preprocessing

To ensure chronological consistency, the dataset was first ordered according to the temporal variable. Subsequently, an analysis was conducted to identify the programs that appeared most frequently in the dataset. It was observed that the programs titled “Station Marker”, “Advertisement Marker”, and “Time Signal” had a high number of occurrences, showing a significant predominance within the dataset. These markers correspond to short transition elements between different broadcast segments, for example, between a program and a commercial break, or between two different programs, and typically have

a very short duration, usually between 5 and 10 seconds. After closer examination, it was concluded that such segments do not convey meaningful information for audience prediction and, in fact, could introduce noise into the model.

Although the model was not directly trained on program titles, each program is associated with a categorical feature, Type1, which classifies the broadcast into broader categories. The issue arises because the Publicity category (Type1 = "Publicity") included not only these short transition segments, but also commercial breaks and sponsorships. To preserve the integrity of other relevant records within this category, the removal was performed based on the program variable, selectively excluding only the short separators mentioned above.

Different experiments were conducted to assess the impact of this cleaning step. Models were trained with and without these markers records, and the comparative results are presented in the following chapter. Additionally, further tests were carried out by excluding other types of segments, such as "Commercial Breaks", "Program Promotions", and "Sponsorships".

As mentioned in the previous chapter, the dataset included broadcasts covering the COVID-19 pandemic period, during which television viewership was abnormally high due to confinement measures. To assess the robustness of the models, separate experiments were performed using:

- Full dataset (including both COVID and post-COVID periods);
- Subset containing only post-pandemic data;

Thus allowing evaluation of the model stability under distinct viewing conditions.

Regarding the Type2 variable (which represents subcategories of Type1), an imbalance was identified, some subcategories appeared only once or twice across the entire four-year period. To mitigate sparsity and improve model generalization, infrequent categories with extremely low representation were removed in tests involving Type2-based models.

The implementation of categorical variable encoding was carried out using the get_dummies() function from the Pandas library. This function performs one-hot encoding, as described

in the previous section, allowing the categorical variables to be transformed into numerical format for model training.

To prevent multicollinearity, the parameter `drop_first=True` was applied. In one-hot encoding, all dummy variables are linearly dependent since their sum equals 1 for each observation. By dropping the first category of each encoded variable, one dummy column is removed, thereby eliminating perfect linear dependency and ensuring numerical stability, especially in linear models. Table 4.1 illustrates how the variable Season was transformed into dummy variables.

Table 4.1: Example of one-hot encoding applied to the variable Season using `get_dummies()` with `drop_first=True`.

Season (original)	**Spring**	**Summer**	**Autumn**
Winter	0	0	0
Spring	1	0	0
Summer	0	1	0
Autumn	0	0	1

When using `drop_first=True`, the first category (Winter in this case) is not explicitly represented in the dataset. Rows corresponding to this reference category take the value 0 in all dummy columns, which implicitly identifies them as belonging to the base category. This strategy simplifies the model matrix while maintaining complete information about all categories.

4.2 Cross-validation Strategy and Train/Test Split

The target variable (y) corresponds to the Universe Rating (%), representing the proportion of the Portuguese population watching television at a given time. All other relevant variables were used as predictors (X), excluding identifiers and variables known only after broadcast time.

Before model training, the predictor variables were normalized using the StandardScaler function from the scikit-learn library [78]. Standardization transforms each feature to have zero mean and unit variance according to the following expression:

$$x' = \frac{x - \mu}{\sigma}$$

where μ and σ denote the mean and standard deviation of the variable, respectively. This procedure ensures that all predictors contribute equally to the model training process, preventing features with larger numeric ranges from dominating the learning algorithm. Such normalization is particularly important for models based on gradient descent, such as linear or ridge regression, as well as for tree-based ensemble methods when hyperparameters depend on variance scaling [78].

After normalization, the dataset was divided into training and testing subsets using the train_test_split() function from scikit-learn, adopting a 70/30 ratio and a fixed random seed (42) to ensure reproducibility. The training subset was used to fit the models, while the test subset served for out-of-sample evaluation.

This procedure was applied to machine learning models (Linear Regression, Ridge Regression, Random Forest, and Gradient Boosting Machines).

For the LSTM, a different preprocessing strategy was adopted due to the sequential nature of the model. Unlike traditional regression models, which assume that observations are independent, LSTMs require input data to be structured as temporal sequences that preserve chronological dependencies [63]. Because neural networks are sensitive to feature magnitude, both predictors and the target variable were normalized to the range $[0, 1]$ using the MinMaxScaler from the scikit-learn library. This scaling improves convergence stability and prevents exploding or vanishing gradients during training. The normalization was applied independently to the predictor set (X) and the target variable (y)

To model temporal dependencies, input sequences were created using a sliding window approach. A window size of 100 observations was defined, corresponding approximately to one full day of continuous television data. To determine this value, the average number

of observations per day in the dataset was first calculated, in order to estimate how many rows represented a typical daily period. This average was then used as a reference to define the temporal window size employed in the LSTM model, ensuring that each input sequence represented a complete daily cycle. For each position i in the series, the input (X_i) consists of the 100 previous observations ($i - 100$ to $i - 1$), and the target (y_i) is the value at position i. This transformation converts the input into a three-dimensional tensor of shape (samples, timesteps, features), which is the expected input format for LSTM networks.

Since time series data cannot be randomly shuffled without destroying temporal relationships, the dataset was divided into training, validation, and testing subsets sequentially (without shuffling). 70% of the data were used for training, 15% for validation, and the remaining 15% for testing.

This chronological partitioning ensures that the model is always trained on past data and evaluated on unseen future data, thereby avoiding data leakage and better simulating real-world forecasting conditions. In summary, the LSTM data preprocessing follows:

1. Feature scaling using MinMaxScaler for both predictors and target variables;
2. Construction of sequential input-output pairs through a sliding window of 100 observations.
3. Chronological data splitting into training (70%), validation (15%), and testing (15%) subsets without shuffling.

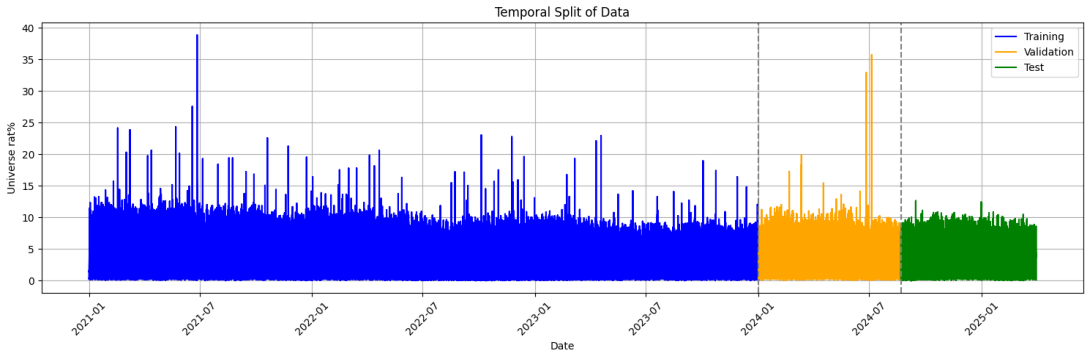


Figure 4.1: LSTM temporal split for full dataset

Figure 4.2 illustrates the temporal split applied to the complete dataset, covering the full analysis period. To further investigate potential structural changes in audience behaviour after the COVID-19 pandemic, a separate dataset was created containing only post-pandemic data. The temporal division adopted for this subset, shown in Figure 4.2, follows the same proportions for training, validation, and testing, ensuring methodological consistency across both experiments.

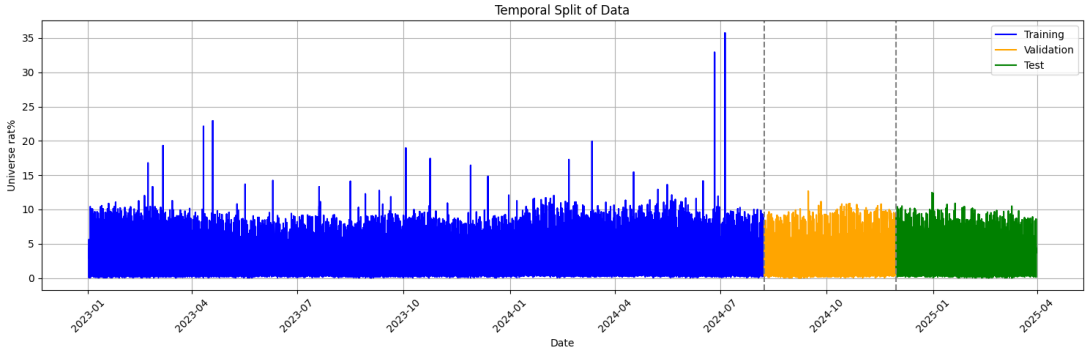


Figure 4.2: LSTM temporal split for post-pandemic dataset

For the SARIMA model, the dataset was chronologically ordered using the "Data" column as index and divided into a 70/30 train-test split, ensuring temporal consistency, no shuffling was applied.

4.3 Model Training

This section describes the process of training the predictive models developed for television audience forecasting. The training phase involves defining the initial configuration of each model and subsequently optimizing its parameters to achieve the best possible predictive performance. The models were trained using the datasets prepared in the previous stages, following consistent procedures to ensure comparability across approaches.

4.3.1 Model setup and configuration

This section presents the initial configuration and implementation details of the predictive models used in this study. For each algorithm, the main parameters, data handling procedures, and evaluation methods are described to ensure methodological transparency.

Linear Regression

The first predictive model implemented was the Linear Regression model, using the LinearRegression() class from the scikit-learn library. To enhance the robustness of the evaluation and mitigate the dependency on a single data split, a 5-fold cross-validation procedure was subsequently applied to the training set. This approach allows the model to be trained and validated across five distinct partitions of the training data, providing a more stable and generalizable estimate of its performance. Table 4.2 summarizes the configuration used for the Linear Regression model.

Table 4.2: Model configuration for Linear Regression.

Parameter	**Description**
Algorithm	Linear Regression (`LinearRegression()`)
Regularization	None
Data normalization	`StandardScaler()`
Validation method	5-Fold Cross-Validation
Evaluation metrics	MAE, RMSE, MSE, R^2
Implementation library	scikit-learn

After training the model, the regression coefficients were analyzed to interpret the influence of each explanatory variable on the target variable. In linear models, each coefficient represents the expected variation in the dependent variable. Positive coefficients indicate that the variable contributes to an increase in audience, while negative coefficients indicate a reduction associated with that characteristic.

The analysis reveals some relevant patterns. Variables related to the broadcast schedule, particularly the trigonometric transformations of the start times, show the largest coefficients in magnitude, confirming that the broadcast time plays a decisive role in audience variation. Among the categorical variables, program types such as "News" and "Sports Broadcasts" show positive coefficients, while "Talk Shows" and "Magazines" present negative ones. Seasonal variables such as Summer and Spring are associated with audience reductions, whereas Autumn and Winter have a positive impact.

Ridge Regression

The Ridge Regression model was trained using the Ridge() class from the scikit-learn library. The implementation used standardized predictors to ensure that the penalty term was applied uniformly across features with different scales.

The hyperparameter α was initially set to 1.0. Additional tests with alternative α values were performed during hyperparameter tuning.

Ridge Regression was chosen as a natural extension of the Linear model, providing a controlled form of regularization that improves model robustness to correlated predictors while maintaining interpretability. Table 4.3 summarizes the configuration used for the Ridge Regression model.

Table 4.3: Model configuration for Ridge Regression.

Parameter	**Description**
Algorithm	Ridge Regression (Ridge())
Regularization type	L2 penalty
Regularization strength (α)	1.0
Data normalization	`StandardScaler()`
Validation method	5-Fold Cross-Validation
Evaluation metrics	MAE, RMSE, MSE, R^2
Implementation library	scikit-learn

Random Forest

The model was implemented using the RandomForestRegressor() class from the scikit-learn library.

The model used 100 decision trees (n_estimators = 100). The maximum depth of the trees was left unconstrained (max_depth = None) to allow the model to learn complex feature interactions, while the random_state = 42 ensured reproducibility. Parallel computation (n_jobs = -1) was used to accelerate model training.

The analysis of the feature importances obtained from the Random Forest Regressor model made it possible to identify the variables that most contributed to the model predictive performance. This indicator represents the relative importance of each explanatory variable, measuring how much each one contributes to reducing impurity within the decision trees.

The results showed that, in addition to the variables related to the start time and end time, factors such as “Saturday”, as well as program types (reality shows, soap operas, and sports broadcasts, also stood out.

Furthermore, the season of the year exhibited a relevant degree of importance, indicating the presence of seasonal patterns in the data. The month variables, which had been previously transformed into sine and cosine components to capture the cyclical nature of time, also contributed to a better temporal representation within the model, reinforcing their overall importance.

Table 4.4 summarizes the configuration used for the Random Forest model.

Table 4.4: Model configuration for Random Forest.

Parameter	**Description**
Algorithm	Random Forest Regressor (RandomForestRegressor())
Number of trees (n_estimators)	100
Maximum tree depth (max_depth)	None
Bootstrap sampling	Enabled
Feature selection per split	sqrt
Data normalization	Not required
Validation method	5-Fold Cross-Validation
Evaluation metrics	MAE, RMSE, MSE, R^2
Implementation library	scikit-learn

The feature selection per split follows the default setting of the Random Forest algorithm, where $\sqrt{p}$ features (with p being the total number of features) are randomly selected at each split. This approach reduces correlation between trees and improves model generalization.

Gradient Boosting Machine

The Gradient Boosting model was implemented using the LGBMRegressor() class from the lightgbm library.

The model was configured with 100 boosting iterations (n_estimators = 100) and a learning rate of 0.1, providing a balance between convergence speed and overfitting control. Each decision tree was allowed to grow without an explicit depth limitation (max_depth = -1), enabling the model to capture non-linear relationships within the data. (num_leaves = 31) that controls the maximum complexity of individual trees, and (min_child_samples = 20), which enforces a minimum number of samples per leaf. LightGBM includes built-in regularization mechanisms such as L1/L2 penalties.

Table 4.5: Model configuration for Light Gradient Boosting Machine.

Parameter	**Description**
Algorithm	LGBMRegressor()
Number of boosting iterations (n_estimators)	100
Learning rate	0.1
Maximum tree depth (max_depth)	No limit (-1)
Number of leaves (num_leaves)	31
Regularization	L1/L2 built-in
Data normalization	StandardScaler()
Validation method	3-Fold Cross-Validation
Evaluation metrics	MAE, RMSE, MSE, R^2
Implementation library	LightGBM

Long Short-Term Memory

To capture the temporal dependencies inherent in television audience data, a LSTM was implemented.

An initial baseline LSTM network was implemented using the Sequential() API from the TensorFlow/Keras framework. The network architecture consisted of a single LSTM layer with 64 units, followed by a dense output layer with one neuron for the audience rating prediction. The `tanh` activation function was used in the recurrent layer to maintain stable gradients during training, and the `Adam` optimizer was selected due to its proven efficiency in adaptive learning rate adjustment. The model minimized the Mean Absolute Error (MAE) loss function. Table 4.6 summarizes the configuration of the baseline LSTM model.

Table 4.6: Model configuration for the LSTM network.

Parameter	**Description**
Architecture	Sequential LSTM Neural Network
LSTM units	64
Activation function	`tanh`
Output layer	Dense(1)
Optimizer	Adam
Loss function	MAE (Mean Absolute Error)
Batch size	16
Epochs	50
Data normalization	MinMaxScaler (features and target)
Input shape	(window length, number of features)
Validation method	Temporal split (70% train, 15% validation, 15% test)
Evaluation metrics	MAE, RMSE, MSE
Implementation library	TensorFlow / Keras

This preliminary version of the network was intended as an exploratory model to assess the effectiveness of recurrent architectures for audience forecasting. Subsequent experiments (presented in the next chapter) include an extensive hyperparameter optimization process. The baseline LSTM was initially trained for 50 epochs, with the training and validation losses monitored throughout the process.

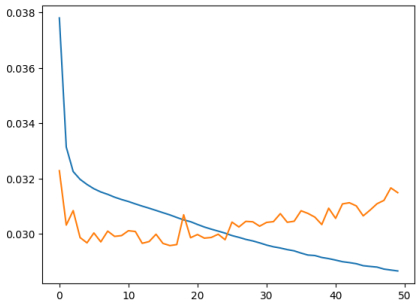


Figure 4.3: LSTM overfitting in first attempt

Figure 4.3 shows the evolution of the losses during the model training process. The blue line represents the training loss, while the orange line corresponds to the validation loss. An analysis of the curves reveals that, after a certain number of epochs, the validation loss stopped decreasing, whereas the training loss continued to decline. This behaviour indicates that the model began to fit too closely to the training data, thereby compromising its ability to generalize, a phenomenon known as overfitting. This occurs when the model learns noise or patterns specific to the training set that do not generalize to unseen data, leading to reduced performance on new examples.

To address the overfitting issue, several training strategies were implemented to improve the model generalization and stability. Specifically, EarlyStopping was used to monitor the validation loss and halt the training process when no further improvement was observed after ten consecutive epochs (patience = 10), while restoring the model weights corresponding to the best validation performance. In addition, ModelCheckpoint was applied to automatically save the best-performing model during training, ensuring that the most optimal version was retained. Furthermore, the ReduceLROnPlateau callback was employed to dynamically adjust the learning rate by reducing it by 50% (factor = 0.5) when the validation loss stagnated for seven epochs (patience = 7), with a minimum learning rate of 0.0001.

ARIMA/SARIMA

To explore classical time series forecasting approaches, a Seasonal Autoregressive Integrated Moving Average with Exogenous Regressors (SARIMAX) model was implemented using the statsmodels library. The model was fitted to the logarithmic transformation of the target variable in order to stabilize variance.

The non-seasonal order (p, d, q) = (1, 0, 1) corresponds to a first-order autoregressive and moving average structure, capable of capturing short-term temporal dependencies in the series. In this case, the first-order autoregressive term (AR(1)) indicates that the audience of a given program tends to depend on the audience of the immediately preceding program, reflecting potential continuity in viewer behavior between consecutive

broadcasts. The moving average term (MA(1)), in turn, models the forecast error of the previous program.

It is also important to note that in the dataset, the same program is often divided into multiple consecutive entries with identical titles, in other words, the same content may appear repeated two or three times in sequence. This characteristic further reinforces the relevance of the first-order autoregressive component, since the audience levels of repeated programs tend to remain similar across adjacent broadcasts, strengthening the short-term correlation captured by the AR(1) term.

Initially, a non-seasonal configuration (seasonal_order = (0, 0, 0, 0)) was tested. However, to account for cyclic patterns, a seasonal component was introduced with parameters (P, D, Q, s) = (0, 0, 1, 100). Since each observation in the dataset represents a program rather than a day, the SARIMA model cannot interpret calendar time directly, it relies solely on the index position of observations. For this reason, the periodicity parameter (s) was estimated empirically as the average number of programs broadcast per day, approximately 100.

The parameter values can be interpreted as follows:

- $P = 0$: no seasonal autoregressive component;
- $D = 0$: no seasonal differencing was applied;
- $Q = 1$: inclusion of one seasonal moving average term;
- $s = 100$: represents one complete daily cycle in terms of program count.

The non-seasonal component (p, d, q) captures short-term dependencies between consecutive programs, whereas the seasonal component (P, D, Q, s) represents cyclic patterns, such as daily audience fluctuations.

Table 4.7: Model configuration for the SARIMA model.

Parameter	**Description**
Algorithm	SARIMAX
Data transformation	Log-transform
Non-seasonal order (p, d, q)	(1, 0, 1)
Seasonal order (P, D, Q, s)	(0, 0, 1, 100)
Remarks	Periodicity s=100: corresponds to one daily cycle
Stationarity enforcement	False
Invertibility enforcement	False
Validation metric	MAE, RMSE, MSE
Implementation library	statsmodels

4.3.2 Hyperparameter tuning

After training the baseline models, a systematic hyperparameter tuning phase was conducted with the aim of maximizing the performance and generalization capability of each algorithm.

Hyperparameters are variables that control the learning process itself, such as the regularization strength, tree depth, learning rate, or the number of hidden units, and are defined prior to training.

The objective of this phase was to identify the optimal configuration for each model by testing multiple parameter combinations and evaluating their impact on predictive performance, minimizing the MAE, RMSE, and MSE errors.

For all models, tuning was mainly performed using a Grid Search strategy combined with Cross-Validation.

In the case of the LSTM, the process was complemented with additional regularization mechanisms, early stopping and model checkpointing, to mitigate the risk of overfitting. These mechanisms allowed the training to stop when the validation loss ceased to improve

and saved the weights corresponding to the best performance.

In the following subsections, the tuning methods and the results obtained for each model are described. A summary table is also presented, showing the final optimized configuration and comparing it with the baseline parameters defined in the previous chapter.

Hyperparameter Tuning for Ridge Regression

Ridge Regression contains a single but crucial hyperparameter, the regularization strength, denoted by α.

The tuning process for the Ridge model was carried out using a Grid Search combined with 5-Fold Cross-Validation, implemented through the GridSearchCV class from the scikit-learn library. The grid search systematically evaluates a predefined set of α values, training the model for each configuration and estimating its average performance across the validation folds.

The regularization parameter α was tuned over a logarithmically spaced grid between 10^{-3} and 10^{3}, comprising 20 values in total. This range was chosen to cover both very weak and very strong regularization regimes.

Table 4.8 summarizes the final hyperparameter configuration obtained for the Ridge Regression model after the tuning process.

Table 4.8: Optimized hyperparameters for Ridge Regression.

Parameter	**Description**
Algorithm	Ridge Regression (Ridge())
Regularization type	L2 (Ridge)
Regularization strength (α)	1.44
Cross-validation folds	5
Evaluation metric	MAE (Mean Absolute Error)
Data normalization	StandardScaler
Implementation library	scikit-learn

The tuned Ridge model demonstrated improved validation performance compared to the baseline configuration, this optimized version was subsequently used for the final evaluation on the test dataset.

Hyperparameter Tuning for Random Forest

The Random Forest model contains several hyperparameters that directly influence its predictive performance, and generalization capability.

The most relevant ones include the number of trees in the ensemble (n_estimators), the maximum depth of each tree (max_depth), the minimum number of samples required to be at a leaf node (min_samples_leaf), the minimum number of samples required to split an internal node (min_samples_split), and the number of features considered at each split (max_features).

The optimization of hyperparameters for the Random Forest model was conducted using a Grid Search strategy combined with 5-Fold Cross-Validation, implemented through the GridSearchCV class from the scikit-learn library. This approach exhaustively evaluates every combination of parameters in the search grid, training and validating the model across multiple data splits to ensure statistical robustness.

The tested grid of hyperparameters was defined in Table 4.9:

Parameter	**Values**
n_estimators	[50, 100, 200]
max_depth	[None, 10, 20, 30]
min_samples_split	[2, 5, 10]
min_samples_leaf	[1, 2, 4]
max_features	[None, 'sqrt']

Table 4.9: Parameter grid used for Random Forest tuning.

The grid search results indicated that the model achieved the lowest validation error with the following configuration showed in Table 4.10:

Table 4.10: Optimized hyperparameters for the Random Forest model.

Parameter	**Description**
Algorithm	RandomForestRegressor()
n_estimators	100
max_depth	20
min_samples_split	2
min_samples_leaf	2
max_features	sqrt
Cross-validation folds	5
Data normalization	StandardScaler
Implementation library	scikit-learn

Hyperparameter Tuning for Gradient Boosting Machine

The Light Gradient Boosting Machine (LightGBM) includes a large set of hyperparameters that control different aspects of the boosting process, such as tree structure, learning rate, regularization, and sampling. Given the high dimensionality of this hyperparameter space, performing an exhaustive Grid Search would be extremely computationally expensive. Therefore, a Randomized Search strategy was adopted instead, which samples a fixed number of parameter combinations from predefined distributions, providing a computationally efficient yet statistically robust alternative.

The tuning process was implemented using the RandomizedSearchCV class from the scikit-learn library. This approach randomly selects hyperparameter combinations to evaluate, allowing the exploration of a large search space while significantly reducing training time compared to the exhaustive grid search. The performance of each sampled configuration was estimated using 5-Fold Cross-Validation, and the model was evaluated according to the MAE. The tested grid of hyperparameters was defined in Table 4.11:

Parameter	Values
n_estimators	[100, 300, 500, 1000]
learning_rat	[0.01, 0.05, 0.1, 0.2]
max_depth	[3, 5, 7, 10, -1]
num_leaves	[15, 31, 50, 100]
min_child_samples	[5, 10, 20, 30]
subsample	[0.6, 0.8, 1]
reg_alpha	[0, 0.1, 0.5, 1]
reg_lambda	[0, 0.1, 0.5, 1]
min_split_gain	[0, 0.1, 0.3, 0.5]
colsample_bytree	[0.6, 0.8, 1]

Table 4.11: Parameter grid used for LightGBM tuning.

This procedure tested 100 randomly selected configurations out of all possible combinations, which ensured a comprehensive exploration of the parameter space. Table 4.12 summarizes the optimized configuration of the LightGBM model after the tuning process.

Table 4.12: Optimized hyperparameters for the LightGBM model.

Parameter	Description
Algorithm	LGBMRegressor()
n_estimators	300
learning_rate	0.05
max_depth	10
num_leaves	50
min_child_samples	10
subsample	0.8
colsample_bytree	0.8
L1 regularization (reg_alpha)	0.5
L2 regularization (reg_lambda)	0.1
min_split_gain	0
Cross-validation folds	5
Evaluation metric	MAE
Implementation library	LightGBM

Hyperparameter Tuning for LSTM

The tuning of LSTM network was carried out through a systematic grid search, aimed at identifying the combination of architectural and training hyperparameters that maximized model performance while preventing overfitting. Given the computational complexity of neural networks, the search focused on a restricted yet representative set of parameters related to the number of hidden units, activation functions, optimizers, dropout rate, and batch size.

The grid search was implemented using the GridSearchCV class from the scikit-learn library, integrated with the Keras deep learning framework via the KerasRegressor() wrapper. The base model was constructed through a custom function, allowing each set of hyperparameters to be passed dynamically during training. To preserve the temporal nature of the data, Time Series Cross-Validation was adopted using the TimeSeriesSplit class, which divides the data sequentially so that each validation fold corresponds to a future segment relative to the training set. This approach prevents data leakage and ensures that the model is always validated on unseen future data, maintaining the integrity of the temporal forecasting process. The grid search iteratively trained the model under all parameter combinations, evaluating the performance on each temporal split. The evaluation criterion was the MAE, which quantifies the average magnitude of forecasting errors. Each model was trained for 50 epochs, using early stopping and checkpointing mechanisms to avoid overfitting by monitoring validation loss improvements. Table 4.13 presents the hyperparameter combinations tested during the grid search. Each configuration was evaluated across three temporal folds.

Parameter	**Values**
batch_size	[16, 32, 64]
model_units	[32, 64, 128]
model_activation	['tanh', 'relu']
model_optimizer	['adam', 'rmsprop']
model_dropout_rate	[0.0, 0.2, 0.5]

Table 4.13: Grid of hyperparameters tested during LSTM tuning.

Table 4.14 summarizes the optimal hyperparameters obtained for the LSTM network after the tuning phase.

Table 4.14: Optimized hyperparameters for the LSTM model.

Parameter	**Optimized Value / Description**
Architecture	Sequential LSTM Neural Network
Hidden units	32
Activation function	`tanh`
Optimizer	`adam`
Dropout rate	0.2
Batch size	16
Epochs	50 (with Early Stopping)
Cross-validation method	Time Series Split (n_splits = 3)
Evaluation metric	MAE
Regularization techniques	Early stopping, model checkpointing
Implementation libraries	TensorFlow / Keras, scikit-learn

The optimized configuration demonstrated improved validation performance and reduced error variance relative to the baseline LSTM model. The use of TimeSeriesSplit ensured that the temporal dependencies were properly respected.

This chapter presented the complete process of data preparation and model development carried out for television audience forecasting. It began with data preprocessing, which involved cleaning, transforming, and organizing the dataset to ensure its consistency and suitability for modeling. The subsequent section described the cross-validation strategy and the temporal train/test split adopted to guarantee robust model evaluation. Finally, the model training phase detailed the configuration and implementation of various predictive algorithms, as well as the procedures applied for hyperparameter tuning. Together, these steps established a solid methodological foundation for the performance analysis and comparative evaluation of the models, discussed in the following chapter.

Chapter 5

Discussion of Results

This chapter presents a detailed discussion of the results obtained from the predictive models developed in the previous chapter. The aim is to interpret and compare the performance of the different approaches, identifying the main factors that influenced their accuracy and generalization capabilities. Beyond quantitative evaluation, this analysis seeks to derive insights into audience behaviour and assess the practical implications of the findings within the context of television viewership forecasting.

5.1 Overview of Model Performance

This section presents a comprehensive comparison of the predictive performance of all models across the different dataset configurations evaluated. Four experimental setups were tested to assess the robustness and generalization of each algorithm under varying data conditions:

- **Dataset A:** Full dataset with Type1;
- **Dataset B:** Full dataset with Type2;
- **Dataset C:** Post-COVID subset with Type1;
- **Dataset D:** Post-COVID subset with Type2;

It is important to highlight that for each of the four dataset configurations (A–D), multiple preprocessing variations were tested in the experiments. These included combinations where markers, commercial breaks, and auto-promotional breaks were either removed or selectively retained depending on their potential informational contribution.

For instance, within the configuration (Full dataset with Type1), experiments were performed both with and without commercial content, and with the exclusion of markers segments. Similarly, the Post-COVID datasets were tested under equivalent conditions.

However, to ensure clarity and conciseness, only the best-performing model results for each dataset configuration are reported in the tables below. The presented values therefore correspond to the optimal outcomes obtained among the various tested preprocessing strategies for each model–dataset pair. Each configuration was evaluated using the optimized hyperparameters obtained in the previous section. The evaluation metrics considered were the MAE, RMSE, and MSE. It is also relevant to note that, for the Linear Regression, Ridge Regression, Random Forest, and Gradient Boosting Machine models, the (R^2) was additionally computed.

However, the R^2 score was not applied to the LSTM and SARIMA models. In the case of the LSTM network, the model is trained focuses on capturing temporal dependencies and dynamic transitions between observations, which do not necessarily align with the assumptions of variance explanation underlying the R^2 formulation.

Similarly, for the SARIMA model, the residuals are autocorrelated by construction, violating the independence assumption required for the meaningful interpretation of R^2.

It is important to note that the results of the LSTM network are expressed on a normalized scale. The tables 5.1 and 5.2 presented below summarize the results obtained for each experimental setup, highlighting the best-performing model per dataset.

Table 5.1: Model performance across full datasets

Model	**Dataset A**				**Dataset B**			
	MAE	RMSE	MSE	R^2	MAE	RMSE	MSE	R^2
Linear Regression	1.218	1.701	2.710	0.714	1.147	1.570	2.403	0.746
Ridge Regression	1.227	1.700	2.893	0.702	1.147	1.570	2.465	0.746
Random Forest	0.392	0.769	0.591	0.931	0.390	0.764	0.584	0.939
LightGBM	0.398	0.787	0.620	0.934	0.387	0.777	0.604	0.937
LSTM	0.031	0.047	0.002	–	0.031	0.046	0.002	–
SARIMA	1.658	2.800	5.841	–	1.348	2.193	4.813	–

Table 5.2: Model performance on post-COVID datasets

Model	**Dataset C**				**Dataset D**			
	MAE	RMSE	MSE	R^2	MAE	RMSE	MSE	R^2
Linear Regression	1.122	1.481	2.272	0.727	1.014	1.354	1.821	0.774
Ridge Regression	1.099	1.440	2.076	0.742	1.009	1.335	1.811	0.788
Random Forest	0.348	0.595	0.354	0.951	0.346	0.589	0.347	0.956
LightGBM	0.346	0.592	0.349	0.954	0.344	0.590	0.348	0.956
LSTM	0.052	0.072	0.005	–	0.051	0.070	0.004	–
SARIMA	1.237	1.820	3.315	–	1.261	1.869	3.493	–

Tables 5.1 and 5.2 summarize the evaluation metrics obtained for all models across the four datasets considered in this study. The first table presents results for the full datasets (A and B), which include both pre- and post-COVID observations, while the second table corresponds to the post-COVID subsets (C and D). This structure allows for a clear comparison of the models predictive performance under different temporal and categorical configurations.

Across all experiments, the ensemble-based models, particularly Random Forest and LightGBM, consistently achieved the best results in terms of predictive accuracy. Both models recorded the lowest error values (MAE, RMSE, and MSE) and the highest R^2 scores, exceeding 0.93 in the full datasets and reaching up to 0.956 in the post-COVID subsets. This strong performance highlights the ability of these methods to model non-linear relationships and capture complex feature interactions that influence television audience ratings.

The linear models (Linear Regression and Ridge Regression) demonstrated moderate predictive capability, with R^2 values ranging between 0.70 and 0.78. Although they capture the general structure of audience variation, their limited flexibility restricts their performance relative to tree-based approaches. Nevertheless, the Ridge model slightly outperformed the simple linear model, confirming that regularization helped mitigate overfitting without sacrificing interpretability.

The LSTM network results indicated that the temporal structure was captured reasonably well in the full datasets, while performance deteriorated in the reduced post-COVID subset, reflecting sensitivity to data volume and sequence length.

Finally, the SARIMA model produced comparatively weaker results, with higher error values.

When comparing datasets, a general improvement is observed in the post-COVID subsets (C and D). This trend suggests that audience behaviour became more stable and predictable after the pandemic, allowing models to achieve better generalization. Moreover, datasets using the Type2 categorization (B and D) consistently performed slightly better than their Type1 counterparts, indicating that the finer granularity of program classification provided marginal but consistent gains in predictive performance.

5.2 Visual analysis of predictions vs actual data

While quantitative metrics such provide a concise assessment of model accuracy, they do not fully convey how well each model captures the temporal dynamics and fluctuations

of audience behaviour over time. For this reason, a complementary visual analysis was conducted by comparing the predicted versus actual audience values across the test set of each model.

This graphical evaluation enables the observation of how accurately each algorithm follows the real variations of television ratings, identifying periods of underestimation or overestimation, and assessing the ability of each model to capture audience peaks and troughs. By inspecting the prediction trends over time, one can also better understand the stability and generalization capacity of each approach.

In addition to evaluating the predictions on the holdout test sets, an independent external dataset was also used to assess the generalization of the trained models. Specifically, a dataset corresponding to the TVI channel programming for August 2025 was downloaded from the YUMI platform and used as an unseen dataset for out-of-sample forecasting. This additional experiment serves as a robustness check, allowing for the comparison of predicted and actual ratings for an entire month of programming that was not included in the training process. In the following subsections, the results are presented for each model individually. For every algorithm, two graphical analyses are provided:

1. The Predicted vs Actual plot for the test subset of the main dataset.

2. The Predicted vs Actual plot for the external August 2025 dataset (TVI).

5.2.1 Dataset A

Linear Regression

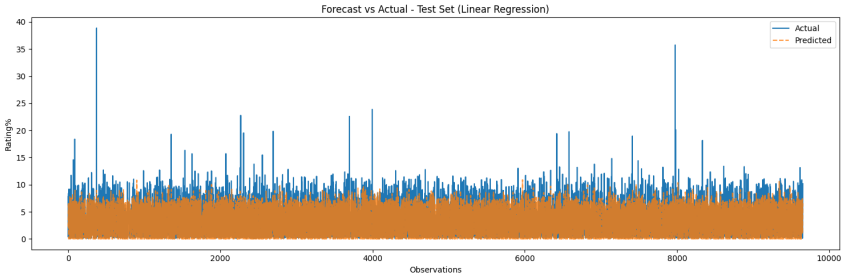


Figure 5.1: Predicted vs Actual plot for the test subset - Linear Regression (Dataset A)

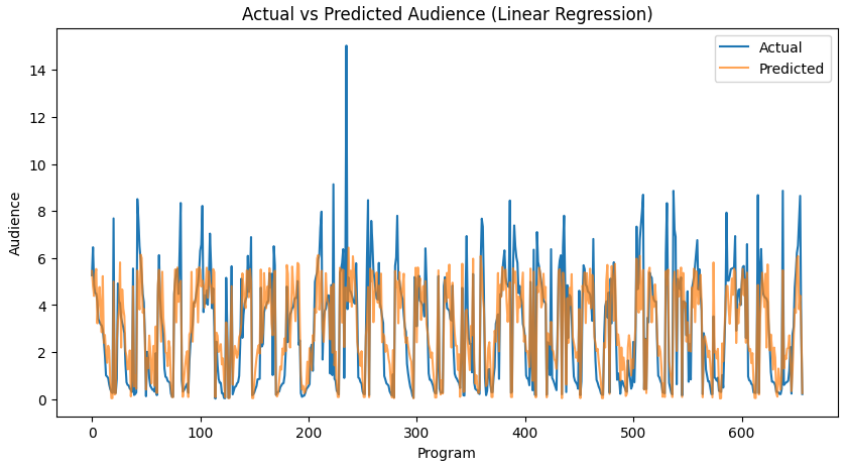


Figure 5.2: Predicted vs Actual plot for external dataset - Linear Regression (Dataset A)

The visual analysis of Figures 5.1, 5.2 shows that Linear Regression is able to follow the general trend of the audience data, but it does not adequately reproduce the amplitude of the variations. The predictions remain close to the average, smoothing out the peaks—something expected from this type of model, which is limited in capturing nonlinear relationships or abrupt fluctuations typical of events.

Ridge Regression

The results obtained were very similar to those of the linear regression, both in terms of performance metrics and in the relationship between observed and predicted values. Since the differences are not visually perceptible, it was decided not to include the corresponding graphs in the main text, but to present them only in the appendix, in Figure A.1.

Random Forest

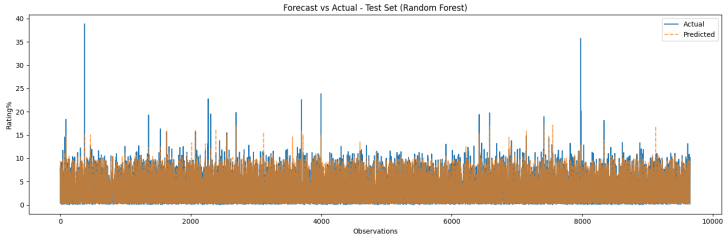


Figure 5.3: Predicted vs Actual plot for the test subset - Random Forest (Dataset A)

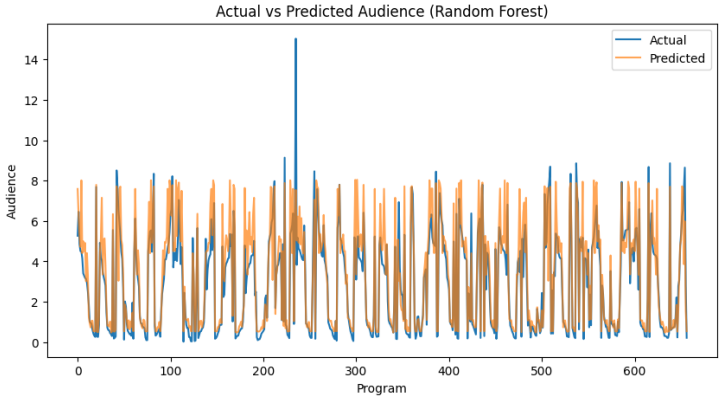


Figure 5.4: Predicted vs Actual plot for external dataset - Random Forest (Dataset A)

Figures 5.3, 5.4 shows the predicted versus actual audience values for the Random Forest model. The model achieved significantly better performance compared to the linear regression models. A visual inspection reveals that, unlike the linear regression models, which tend to smooth out variability, the Random Forest captures a much wider range of fluctuations present in the real data. Although the extreme audience peaks remain slightly underestimated, the overall prediction pattern closely follows the actual values, demonstrating strong responsiveness. Regarding the model performance on the external dataset, the same trend is observed. Overall, this model outperformed both linear regression and ridge regression across all evaluation metrics.

Gradient Boosting Machine

The GBM model showed results that were quite consistent with those of the Random Forest model, achieving very similar performance metrics. However, GBM model exhibited smoother behavior, reducing the tendency to overfit local variations in the data.

The results can be found in the figures 5.5 and 5.6

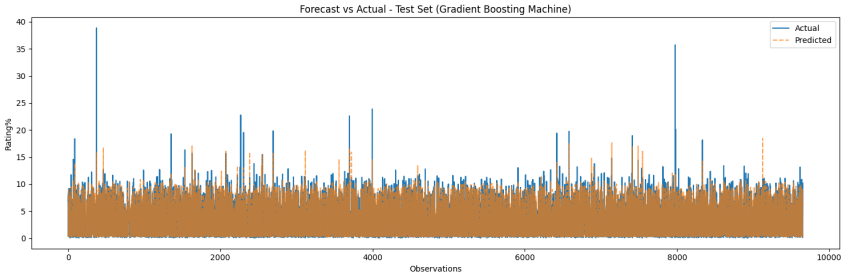


Figure 5.5: Predicted vs Actual plot for the test subset - GBM (Dataset A)

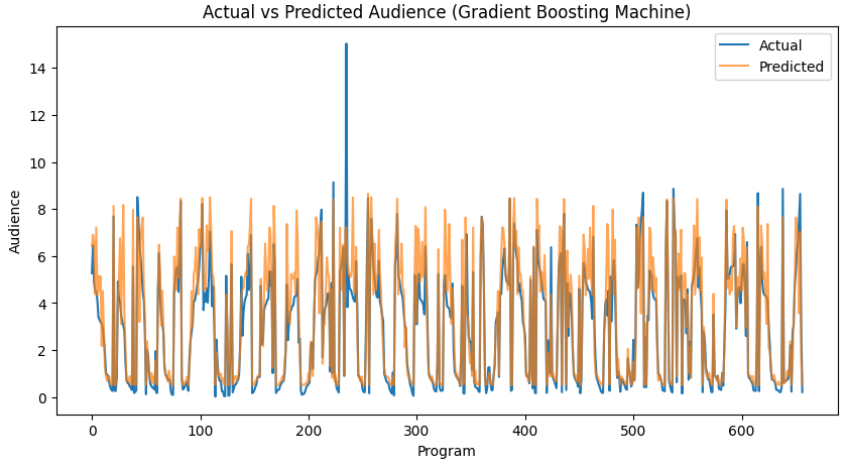


Figure 5.6: Predicted vs Actual plot for external dataset - GBM (Dataset A)

LSTM

As shown in Figures 5.7 and 5.8, the LSTM reproduces the cyclical structure of audience behaviour remarkably well, particularly the daily rise-and-fall pattern associated with television programming schedules. While some deviations occur in extreme peaks, the model successfully captures most of the medium and low-amplitude fluctuations. Compared to the tree-based models, the LSTM exhibits slightly smoother predictions, which reflect its temporal smoothing capability derived from memory cells. Overall, the LSTM achieved competitive performance relative to the other models

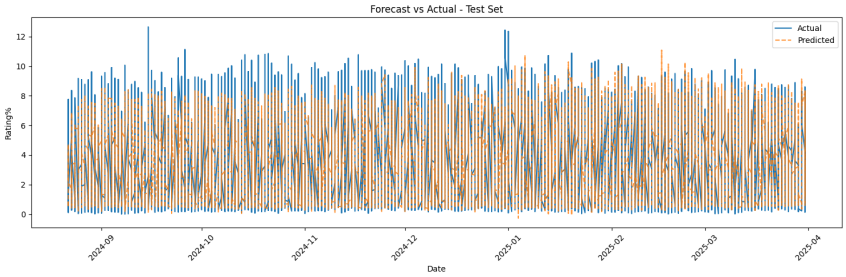


Figure 5.7: Predicted vs Actual plot for the test subset - LSTM (Dataset A)

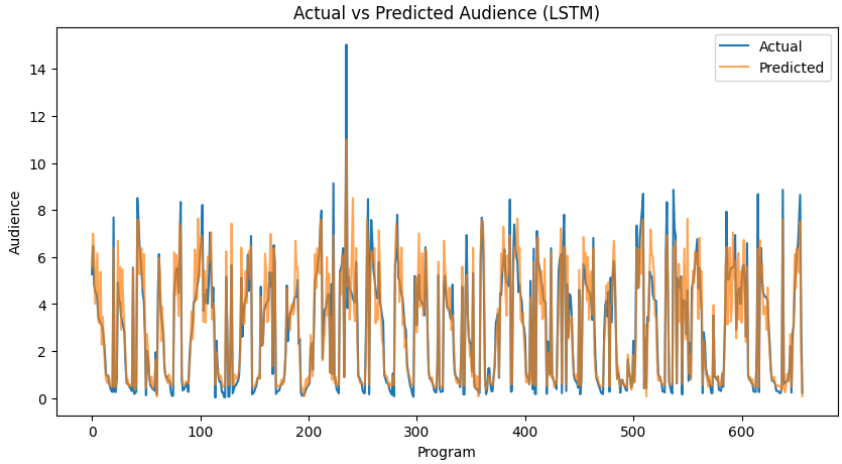


Figure 5.8: Predicted vs Actual plot for external dataset - LSTM (Dataset A)

Figure 5.9 shows the evolution of the training loss and validation loss over the epochs for the trained model. A sharp reduction in loss is observed during the initial epochs, followed by a gradual stabilization. The validation loss closely follows the training loss in the early stages, indicating good generalization ability at the beginning of training.

From around the tenth epoch onward, slight oscillations can be observed, although the phenomenon remains moderate.

Overall, the model demonstrates good performance and stability, with low final loss values. The use of checkpoints ensured that the final model corresponds to the best validation epoch, mitigating potential overfitting effects and guaranteeing the selection of the model with the best generalization capability.

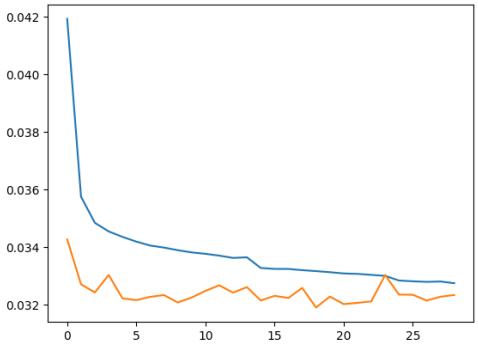


Figure 5.9: Evolution training loss and validation loss - LSTM (Dataset A)

Sarima

Figures 5.10 and 5.11 show what was already speculated based on the results obtained in the metrics; a slight overestimation is observed. This model proved to be the one with the worst results.

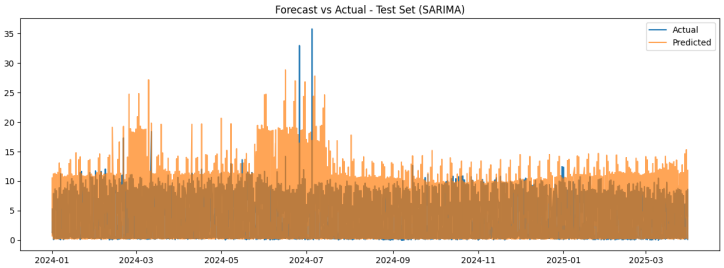


Figure 5.10: Predicted vs Actual plot for the test subset - SARIMA (Dataset A)

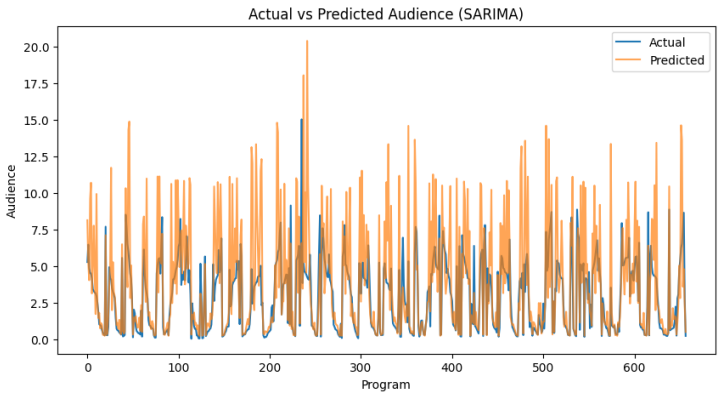


Figure 5.11: Predicted vs Actual plot for external dataset - SARIMA (Dataset A)

5.2.2 Dataset B

Following the same experimental setup described for Dataset A, the Dataset B was constructed by replacing the categorical variable Type1 with Type2.

Overall, the results obtained with Dataset B closely mirror those observed in Dataset A across all models. Performance metrics remained nearly unchanged, indicating that the substitution of Type1 by Type2 did not introduce significant additional predictive power for this dataset.

The only model that exhibited a notable deviation was the Gradient Boosting Machine (LightGBM). In this case, the finer level of categorical detail provided by Type2 allowed the model to capture slightly more specific patterns, resulting in a modest improvement and a better alignment between predicted and actual values.

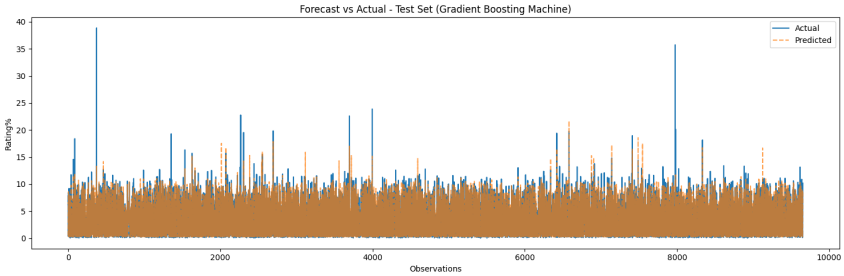


Figure 5.12: Predicted vs Actual plot for the test subset - GBM (Dataset B)

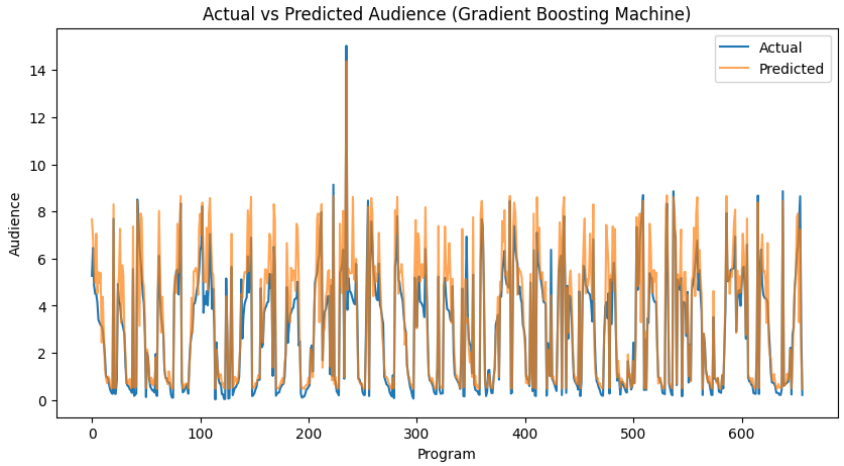


Figure 5.13: Predicted vs Actual plot for external dataset - GBM (Dataset B)

Figures 5.12 and 5.13 illustrate this behaviour, showing the comparison between predicted and real audience ratings on the test set and on the external TVI August 2025 dataset, respectively. It is possible to see in figure 5.13 that the model managed to capture the audience peaks well.

The remaining models (Linear Regression, Ridge Regression, Random Forest, LSTM, and SARIMA) produced results analogous to those described for dataset (A). For conciseness, their visual analyses are presented in the Appendix: Figure A.2 corresponds to the Linear Regression model, Figure A.3 to Ridge Regression, Figure A.4 to Random Forest, Figure A.5 to the LSTM model, and Figure A.6 to the SARIMA model.

5.2.3 Dataset C

Dataset C corresponds to the subset of data that excludes the COVID-19 period, preserving only observations from the post-pandemic period. The preprocessing, model configuration, and training procedures were identical to those applied to the previous datasets, ensuring methodological consistency across experiments.

Given the extensive visual analyses already presented, the graphical representations were placed in the Appendix: Figure A.7 corresponds to the Linear Regression model, Figure A.8 to Ridge Regression, Figure A.9 to Random Forest, Figure A.10 to Gradient Boosting Machine, Figure A.11 to the LSTM model, and Figure A.12 to the SARIMA model. Thus, the discussion in this section focuses on summarizing the general trends and preparing the transition to Dataset D, which integrates Type2 categorization and demonstrated superior predictive performance. Overall, the results, as the metrics have shown, exhibited a slight improvement, except for the LSTM model which produced poorer results mainly due to the reduced amount of available data. LSTMs require a large number of sequential examples to learn temporal dependencies, when this volume decreases, the network tends to overfit the training data and loses generalization capability.

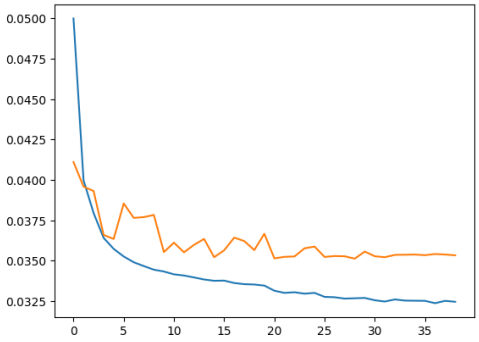


Figure 5.14: Training and validation loss curves for the LSTM model (Dataset C)

Figure 5.14 illustrates this phenomenon precisely: the validation curve stagnates and diverges from the training curve, a classic sign of mild overfitting caused by limited data and the excessive stability of the series.

5.2.4 Dataset D

Linear Regression

The Figures 5.15 and 5.16 show that the model is able to partially capture the general trend of the audience data, although it displays visible errors in the more pronounced peaks. This again indicates limitations in its ability to represent non-linear relationships between the independent variables and the audience, resulting in a certain smoothing of the extremes.

Nevertheless, it is important to highlight that, for this particular dataset, Linear Regression achieved its best performance compared to the other datasets tested.

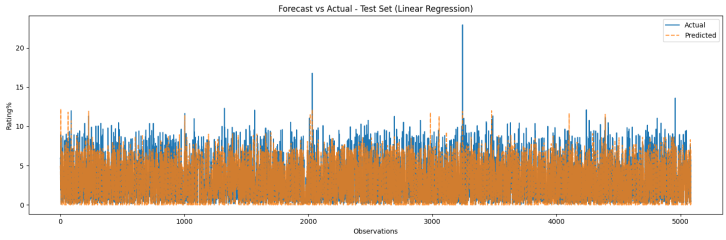


Figure 5.15: Predicted vs Actual plot for the test subset - Linear Regression (Dataset D)

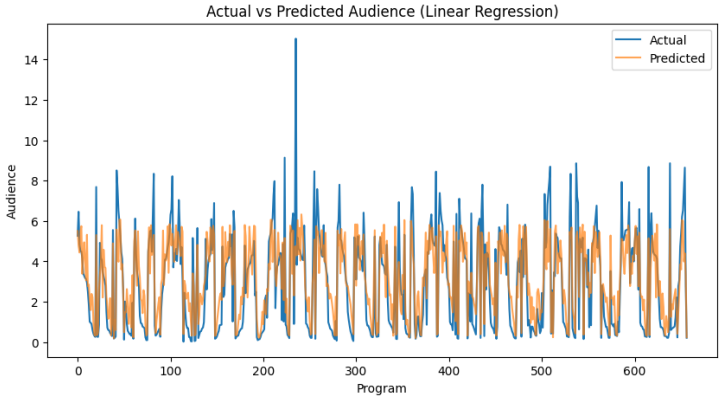


Figure 5.16: Predicted vs Actual plot external dataset - Linear Regression (Dataset D)

Ridge Regression

This model exhibits behavior very similar to that of Linear Regression, which is expected since both share the same underlying structure. However, the regularization introduced by the L2 penalty parameter slightly reduces the model's variance, making it somewhat more stable and less sensitive to outliers. Despite this, the visual improvement compared to Linear Regression is minimal, indicating that the model's linearity still limits its predictive capability. The improvement in the evaluation metrics is also minimal. The graphs related to this evaluation can be found in the appendix: Figure A.13.

Random Forest

The Random Forest model produces predictions that are much closer to the actual line. In Figures 5.17 and 5.18, it is evident that the model better adapts to local fluctuations in audience levels, accurately capturing both increases and decreases. This reflects the model's ability to handle nonlinear relationships and complex interactions among variables. However, small discrepancies can still be observed, especially in extreme values.

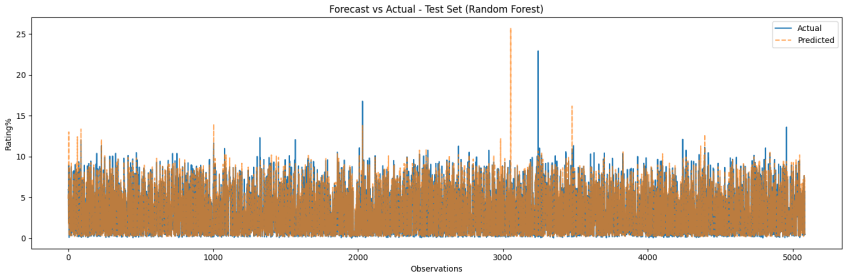


Figure 5.17: Predicted vs Actual plot for the test subset - Random Forest (Dataset D)

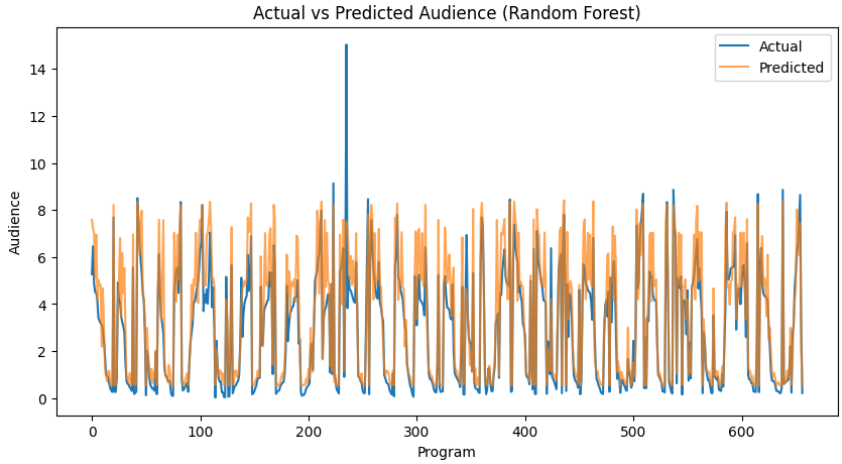


Figure 5.18: Predicted vs Actual plot external dataset - Random Forest (Dataset D)

Gradient Boosting Machine

The Gradient Boosting model shows performance quite similar to that of the Random Forest, but with an even more precise adaptation to real patterns, especially at transition points and small variations. Visually, in Figures 5.19 and 5.20, the predicted curve follows the actual one with greater accuracy, demonstrating that the model is effective in capturing the dynamics of audience behavior.

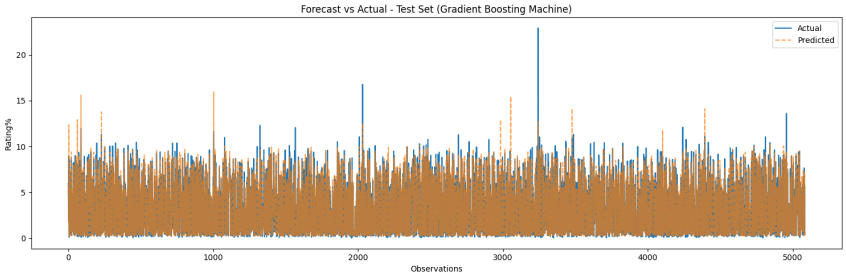


Figure 5.19: Predicted vs Actual plot for the test subset - Random Forest (Dataset D)

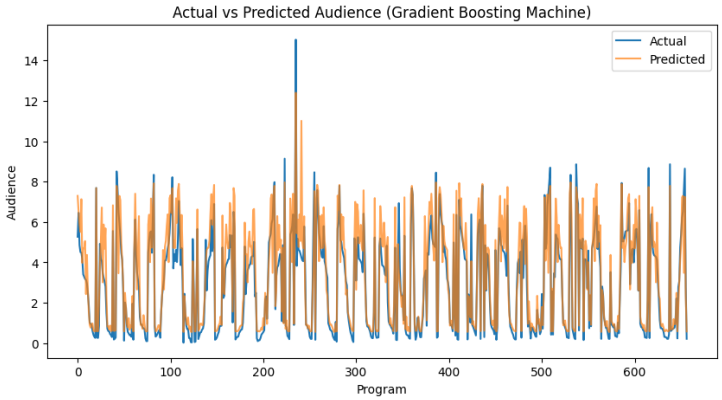


Figure 5.20: Predicted vs Actual plot external dataset - Random Forest (Dataset D)

LSTM

Figure A.14 in appendix indicate that the model still shows considerable differences between the actual and predicted values, especially during rapid fluctuations. As previously mentioned in the earlier dataset, this behavior is due to the LSTM model's limitations when dealing with a reduced amount of data, which hinders its ability to learn temporal dependencies and may lead to overfitting phenomena.

SARIMA

Figure A.15 in the appendix show a mixed performance: although the model is able to follow the seasonal behavior and some audience trends, it displays peaks and predicted values higher than the actual ones at certain points. This suggests that the model captured the seasonal component but did not fully adjust to the data's volatility.

Chapter 6

Conclusions

The present dissertation aimed to develop, evaluate, and compare different machine learning and statistical models for forecasting television audience ratings in Portugal. Through the integration of temporal, categorical, and contextual features, the study sought to assess the predictive capacity of various approaches, ranging from traditional regression algorithms to advanced ensemble methods and deep learning architectures.

The experimental results demonstrated that ensemble-based models, namely Random Forest and LightGBM, consistently outperformed linear and statistical baselines, achieving high accuracy and strong generalization across all datasets. Linear Regression and Ridge Regression provided interpretable yet less flexible predictions, while the LSTM network captured sequential dependencies but exhibited sensitivity to data volume and volatility. In contrast, the SARIMA model was limited.

An important insight derived from the experiments is the influence of external factors such as the COVID-19 period. Post-COVID datasets yielded more stable and predictable audience behaviour, resulting in improved accuracy.

From a methodological standpoint, the results validate the application of ensemble learning for audience forecasting, highlighting LightGBM as the most robust and adaptable model within this context. Furthermore, the research emphasizes the relevance of temporal features.

Despite the promising results, several limitations remain and open new directions for

further research. First, the study focused exclusively on regression-based forecasting of continuous audience values. A natural extension of this work would be to reformulate the problem as a classification task, where audience levels are categorized into discrete classes such as high, medium, and low viewership. Such an approach could be especially useful for operational decision-making, advertising placement, or scheduling optimization.

Another promising direction involves enriching the feature space with external data sources. Integrating information from social media, streaming platforms, and other online engagement metrics could provide a more comprehensive view of audience behaviour, capturing competitive dynamics between television and digital media. These additional data streams could enhance model responsiveness to external trends and improve predictive robustness.

In practical terms, the outcomes of this research have direct applicability within the operational context of the TWA company and its YUMI platform, which serves as a structured source of television audience data. The developed models could be integrated into such analytical environments to enhance forecasting capabilities, automate audience insights, and support strategic decisions related to programming, advertising, and scheduling. Future work may also explore the deployment of these models in real-time audience monitoring systems or their integration with broader audience analytics platforms. This would enable continuous model updating, adaptive learning from new data, and the provision of actionable intelligence to media stakeholders, further bridging the gap between research and industry practice.

Bibliography

[1] P. Napoli, *Audience Evolution: New Technologies and the Transformation of Media Audiences*. Columbia University Press, 2010, ISBN: 9780231520942. [Online]. Available: https://books.google.pt/books?id=twHVnyfamBUC.

[2] R. Akula, Z. Wieselthier, L. Martin, and I. Garibay, "Forecasting the success of television series using machine learning," in *2019 SoutheastCon*, 2019, pp. 1–8. DOI: 10.1109/SoutheastCon42311.2019.9020419.

[3] J. Heaton, "Ian goodfellow, yoshua bengio, and aaron courville: Deep learning: The mit press, 2016, 800 pp, isbn: 0262035618," *Genetic Programming and Evolvable Machines*, vol. 19, Oct. 2017. DOI: 10.1007/s10710-017-9314-z.

[4] P. Agnieszka, "Factors affecting viewerâ€™s television preferences: A review," *International Journal of Advances in Social Science and Humanities*, Mar. 2018. [Online]. Available: https://ijassh.com/index.php/IJASSH/article/view/56.

[5] Y. Web, *Studies*, https://info.yumianalyticsweb.com/studies/, Acedido em 1 de setembro de 2025, 2024.

[6] D. V. Singh, "Television ratings: Techniques, issues and debate," *Media Watch*, vol. 2, no. 2, pp. 70–73, 2011, Original work published 2011. DOI: 10.1177/0976091120110211.

[7] J. Webster, "Beneath the veneer of fragmentation: Television audience polarization in a multichannel world," *Journal of Communication*, vol. 55, no. 2, pp. 366–382,

2005. DOI: 10.1093/joc/55.2.366. [Online]. Available: https://doi.org/10.1093/joc/55.2.366.

[8] J. Dmochowski, M. Bezdek, B. Abelson, J. Johnson, E. Schumacher, and L. Parra, "Audience preferences are predicted by temporal reliability of neural processing," *Nature Communications*, vol. 5, no. 1, 2014. DOI: 10.1038/ncomms5567. [Online]. Available: https://doi.org/10.1038/ncomms5567.

[9] I. Jennes and J. Pierson, "Audience measurement and digitalisation," in *Proceedings of the EuroITV 2011 - 9th European Interactive TV Conference*, 2011, pp. 97–100. DOI: 10.1145/2000119.2000138. [Online]. Available: https://doi.org/10.1145/2000119.2000138.

[10] H. H. Larsen, J. M. Forsberg, S. V. Hemstad, R. R. Mukkamala, A. Hussain, and R. Vatrapu, "Tv ratings vs. social media engagement: Big social data analytics of the scandinavian tv talk show skavlan," in *2016 IEEE International Conference on Big Data (Big Data)*, 2016, pp. 3849–3858. DOI: 10.1109/BigData.2016.7841058.

[11] T. Ueoka and A. Ishii, "Consideration on tv audience rating and influence of social media," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 5460–5461. DOI: 10.1109/BigData.2018.8621881.

[12] Nielsen. "O streaming atinge um marco histórico na tv e supera a transmissão combinada e a tv a cabo pela primeira vez." Acedido em 09 de Agosto de 2025. [Online]. Available: https://www.nielsen.com/pt/news-center/2025/streaming-reaches-historic-tv-milestone-eclipses-combined-broadcast-and-cable-viewing-for-first-time/?utm_source=chatgpt.com.

[13] V. Sigre-Leirós et al., "Binge-watching in times of covid-19: A longitudinal examination of changes in affect and tv series consumption patterns during lockdown," *Psychology of Popular Media*, vol. 12, pp. 173–185, Feb. 2022. DOI: 10.1037/ppm0000390.

[14] J. Kim, K. Merrill Jr., C. Collins, and H. Yang, "Social tv viewing during the covid-19 lockdown: The mediating role of social presence," *Technology in Society*, vol. 67, p. 101 733, 2021, ISSN: 0160-791X. DOI: https://doi.org/10.1016/j.techsoc.2021.101733. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0160791X21002086.

[15] M. Scopelliti, M. Pacilli, and A. Aquino, "Tv news and covid-19: Media influence on healthy behavior in public spaces," *International Journal of Environmental Research and Public Health*, vol. 18, p. 1879, Feb. 2021. DOI: 10.3390/ijerph18041879.

[16] L. Saulīte and D. Ščeulovs, "The impact on audience media brand choice using media brands uniqueness phenomenon," *Journal of Open Innovation: Technology, Market, and Complexity*, vol. 8, no. 3, p. 128, 2022. DOI: 10.3390/joitmc8030128. [Online]. Available: https://doi.org/10.3390/joitmc8030128.

[17] L. Nixon, K. Ciesielski, and B. Philipp, "Ai for audience prediction and profiling to power innovative tv content recommendation services," in *Proceedings of the 2019 ACM International Conference on Interactive Experiences for TV and Online Video (TVX '19)*, 2019. DOI: 10.1145/3347449.3357485. [Online]. Available: https://doi.org/10.1145/3347449.3357485.

[18] M. Soto-Sanfiel, I. Villegas-Simón, and A. Angulo-Brunet, "Opinion of television managers about their viewers and their interest in science: Audience images and lack of scientific content on television," *El Profesional de la Información*, 2021. DOI: 10.3145/epi.2021.nov.09. [Online]. Available: https://doi.org/10.3145/epi.2021.nov.09.

[19] A. Palomba, "Advancing predictive content analysis: A natural language processing and machine learning approach to television script data," *Journal of Marketing Analytics*, 2025, Published 31 August 2025. DOI: 10.1057/s41270-025-00435-1.

[20] D. Reeth, "Forecasting tour de france tv audiences: A multi-country analysis," *International Journal of Forecasting*, vol. 35, no. 2, pp. 810–821, 2019. DOI: 10.1016/

j.ijforecast.2018.06.003. [Online]. Available: https://doi.org/10.1016/j.ijforecast.2018.06.003.

[21] A. Shestyuk, K. Kasinathan, V. Karapoondinott, R. Knight, and R. Gurumoorthy, "Individual eeg measures of attention, memory, and motivation predict population level tv viewership and twitter engagement," *PLOS ONE*, vol. 14, no. 3, e0214507, 2019. DOI: 10.1371/journal.pone.0214507. [Online]. Available: https://doi.org/10.1371/journal.pone.0214507.

[22] K. El Fayq, S. Tkatek, and L. Idouglid, "Predicting television programs success using machine learning techniques," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 14, pp. 5502–5512, Oct. 2024. DOI: 10.11591/ijece.v14i5.pp5502-5512.

[23] V. Gupta, N. Jain, H. Garg, et al., "Predicting attributes based movie success through ensemble machine learning," *Multimedia Tools and Applications*, vol. 82, no. 7, pp. 9597–9626, 2023. DOI: 10.1007/s11042-021-11553-0.

[24] S. Mahimkar, E. GOEL, and D. KUSHWAHA, "Predictive analysis of tv program viewership using random forest algorithms," Oct. 2021.

[25] I. Hanif and R. Septiani, "Ensemble learning for television program rating prediction," *Indonesian Journal of Statistics and Its Applications*, vol. 5, pp. 377–395, Jun. 2021. DOI: 10.29244/ijsa.v5i2p377-395.

[26] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017. DOI: 10.1109/TNNLS.2016.2582924.

[27] M. E. Cammarano, A. Guarino, D. Malandrino, et al., "Tv shows popularity prediction of genre-independent tv series through machine learning-based approaches," *Multimedia Tools and Applications*, vol. 83, pp. 75 757–75 780, 2024. DOI: 10.1007/s11042-024-18518-z.

[28] P. Danaher, M. Smith, and T. Danaher, "Forecasting television ratings," *International Journal of Forecasting*, vol. 27, pp. 1215–1240, Oct. 2011. DOI: 10.1016/j.ijforecast.2010.08.002.

[29] Y. Shao, "Research on prediction model of audience viewing behavior of new media driven by big data," in *2024 IEEE 2nd International Conference on Image Processing and Computer Applications (ICIPCA)*, 2024, pp. 1882–1886. DOI: 10.1109/ICIPCA61593.2024.10709133.

[30] I. H. Sarker, "Machine learning: Algorithms, real-world applications and research directions," *SN Computer Science*, vol. 2, no. 3, p. 160, 2021. DOI: 10.1007/s42979-021-00592-x. [Online]. Available: https://doi.org/10.1007/s42979-021-00592-x.

[31] M. Mohammed, M. Khan, and E. Bashier, *Machine Learning: Algorithms and Applications*. CRC Press, Jul. 2016, ISBN: 9781498705387. DOI: 10.1201/9781315371658.

[32] S. Kotsiantis, "Supervised machine learning: A review of classification techniques.," *Informatica (Slovenia)*, vol. 31, pp. 249–268, Jan. 2007.

[33] M. Jordan and T. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science (New York, N.Y.)*, vol. 349, pp. 255–60, Jul. 2015. DOI: 10.1126/science.aaa8415.

[34] C. Bishop, "Pattern recognition and machine learning," in Jan. 2006, vol. 16, pp. 140–155. DOI: 10.1117/1.2819119.

[35] X. Zhu, "Semi-supervised learning literature survey," *Comput Sci, University of Wisconsin-Madison*, vol. 2, Jul. 2008.

[36] R. Rao, "Reinforcement learning: An introduction; r.s. sutton, a.g. barto (eds.)," *Neural Networks*, vol. 13, pp. 133–135, Jan. 2000. DOI: 10.1016/S0893-6080(99)00098-2.

[37] X. Su, X. Yan, and C. Tsai, "Linear regression," *Wiley Interdisciplinary Reviews Computational Statistics*, vol. 4, pp. 275–294, 3 2012. DOI: 10.1002/wics.1198.

[38] N. Roustaei, "Application and interpretation of linear-regression analysis," *Medical Hypothesis, Discovery & Innovation Ophthalmology Journal*, vol. 13, no. 3, pp. 151–159, 2024. DOI: 10.51329/mehdiophthal1506. [Online]. Available: https://doi.org/10.51329/mehdiophthal1506.

[39] A. Hoerl and R. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, pp. 55–67, Apr. 2012. DOI: 10.1080/00401706.1970.10488634.

[40] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics)*. Feb. 2009, ISBN: 0387848576.

[41] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, Oct. 2001. DOI: 10.1023/A:1010950718922.

[42] A. Liaw and M. Wiener, "Classification and regression by randomforest," *Forest*, vol. 23, Nov. 2001.

[43] J. Friedman, "Greedy function approximation: A gradient boosting machine," *The Annals of Statistics*, vol. 29, Nov. 2000. DOI: 10.1214/aos/1013203451.

[44] J. Friedman, "Stochastic gradient boosting," *Computational Statistics Data Analysis*, vol. 38, pp. 367–378, Feb. 2002. DOI: 10.1016/S0167-9473(01)00065-2.

[45] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," Aug. 2016, pp. 785–794. DOI: 10.1145/2939672.2939785.

[46] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–44, May 2015. DOI: 10.1038/nature14539.

[47] W. S. McCulloch and W. H. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943. DOI: 10.1007/BF02478259.

[48] S. Chavlis and P. Poirazi, *Drawing inspiration from biological dendrites to empower artificial neural networks*, Jun. 2021. DOI: 10.48550/arXiv.2106.07490.

[49] I. Jones and K. Kording, *Can single neurons solve mnist? the computational power of biological dendritic trees*, Sep. 2020. DOI: 10.48550/arXiv.2009.01269.

[50] American Institute of Physics (AIP), "Brain-inspired learning in artificial neural networks: A review," *Applied Machine Learning*, vol. 2, no. 2, p. 021 501, 2024.

[51] J. C. Ye, "Biological neural networks," in *Geometry of Deep Learning: A Signal Processing Perspective*. Singapore: Springer Nature Singapore, 2022, pp. 79–90, ISBN: 978-981-16-6046-7. DOI: 10.1007/978-981-16-6046-7_5. [Online]. Available: https://doi.org/10.1007/978-981-16-6046-7_5.

[52] S. Smys, J. Chen, and S. Shakya, "Survey on neural network architectures with deep learning," *Journal of Soft Computing Paradigm*, vol. 2, pp. 186–194, Jul. 2020. DOI: 10.36548/jscp.2020.3.007.

[53] A.-N. Sharkawy, "The effect of increasing hidden layers on the performance of the deep neural network: Modelling, investigation, and evaluation," *Research on Engineering Structures and Materials*, pp. 1–19, Dec. 2024. DOI: 10.17515/resm2024.442st0909tn.

[54] X. Zhao, L. Wang, Y. Zhang, et al., "A review of convolutional neural networks in computer vision," *Artificial Intelligence Review*, vol. 57, p. 99, 2024. DOI: 10.1007/s10462-024-10721-6. [Online]. Available: https://doi.org/10.1007/s10462-024-10721-6.

[55] C. Jin, S. Jang, X. Sun, J. Li, and R. Christenson, "Damage detection of a highway bridge under severe temperature changes using extended kalman filter trained neural network," *Journal of Civil Structural Health Monitoring*, vol. 6, pp. 545–560, Jul. 2016. DOI: 10.1007/s13349-016-0173-8.

[56] N. Kulathunga, N. Ranasinghe, D. Vrinceanu, Z. Kinsman, L. Huang, and Y. Wang, *Effects of the nonlinearity in activation functions on the performance of deep learning models*, Oct. 2020. DOI: 10.48550/arXiv.2010.07359.

[57] M. Murray, V. Abrol, and J. Tanner, *Activation function design for deep networks: Linearity and effective initialisation*, May 2021. DOI: 10.48550/arXiv.2105.07741.

[58] A. Jagtap and G. Karniadakis, "How important are activation functions in regression and classification? a survey, performance comparison, and future directions," *Journal of Machine Learning for Modeling and Computing*, vol. 4, pp. 21–75, Jan. 2023. DOI: 10.1615/JMachLearnModelComput.2023047367.

[59] GeeksforGeeks. "Backpropagation in neural network." Retrieved from GeeksforGeeks. [Online]. Available: https://www.geeksforgeeks.org/backpropagation-in-neural-network/.

[60] R. Scoble. "Neural networks: Simulating the human brain in ai." [Online]. Available: https://www.unaligned.io/p/neural-networks-simulating-human-brain-ai.

[61] Z. Lipton, "A critical review of recurrent neural networks for sequence learning," May 2015.

[62] R. Schmidt, *Recurrent neural networks (rnns): A gentle introduction and overview*, Nov. 2019. DOI: 10.48550/arXiv.1912.05911.

[63] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, Nov. 1997. DOI: 10.1162/neco.1997.9.8.1735.

[64] B. Lindemann, T. Müller, H. Vietz, N. Jazdi, and M. Weyrich, "A survey on long short-term memory networks for time series prediction," *Procedia CIRP*, vol. 99, pp. 650–655, 2021, 14th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 15-17 July 2020, ISSN: 2212-8271. DOI: https://doi.org/10.1016/j.procir.2021.03.088. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212827121003796.

[65] G. Box, "Box and jenkins: Time series analysis, forecasting and control," in Jan. 2013, pp. 161–215, ISBN: 978-1-349-35027-8. DOI: 10.1057/9781137291264_6.

[66] P. Liu, "Time series forecasting based on arima and lstm," Jan. 2022. DOI: 10.2991/aebmr.k.220603.195.

[67] A. Permanasari, I. Hidayah, and I. A. Bustoni, "Sarima (seasonal arima) implementation on time series to forecast the number of malaria incidence," Oct. 2013, pp. 203–207, ISBN: 978-1-4799-0423-5. DOI: 10.1109/ICITEED.2013.6676239.

[68] V. Plevris, G. Solorzano, N. Bakas, and M. Ben Seghier, "Investigation of performance metrics in regression analysis and machine learning-based prediction models," Jun. 2022. DOI: 10.23967/eccomas.2022.155.

[69] E. Lewinson, *A comprehensive overview of regression evaluation metrics*, NVIDIA Technical Blog, Apr. 2023. [Online]. Available: https://developer.nvidia.com/blog/a-comprehensive-overview-of-regression-evaluation-metrics/.

[70] J. López, M. Vaz-Álvarez, and C. Ceide, "Covid-19 and public service media: Impact of the pandemic on public television in europe," *Profesional de la Información*, 2020. DOI: 10.3145/epi.2020.sep.18.

[71] J. Yang, S. Rahardja, and P. Fränti, "Outlier detection: How to threshold outlier scores?" In *Proceedings of the International Conference on Artificial Intelligence, Information Processing and Cloud Computing*, ser. AIIPCC '19, Sanya, China: Association for Computing Machinery, 2019, ISBN: 9781450376334. DOI: 10.1145/3371425.3371427. [Online]. Available: https://doi.org/10.1145/3371425.3371427.

[72] Oracle, *IQR (Intervalo entre Quartis) — docs.oracle.com*, https://docs.oracle.com/cloud/help/pt_BR/pbcs_common/PFUSU/insights_metrics_IQR.htm, [Accessed 01-10-2025].

[73] C. Santos and C. Maurer, "Reconfigurações do consumo de telejornalismo no cenário convergente," *Intexto*, no. 55, p. 130 338, 2023. DOI: 10.19132/1807-8583.55.130338. [Online]. Available: https://doi.org/10.19132/1807-8583.55.130338.

[74] V. Becker and K. Alves, "Análise da queda da audiência do jornal nacional e os impactos no telejornalismo," *Comunicação & Inovação*, vol. 16, no. 32, pp. 87–102, 2015. DOI: 10.13037/ci.vol16n32.3348. [Online]. Available: https://doi.org/10.13037/ci.vol16n32.3348.

[75] Marktest. "Avaliação ao consumo de televisão em portugal." Acedido em 15 de setembro de 2025. [Online]. Available: https://www.marktest.com/wap/a/n/id~2bac.aspx.

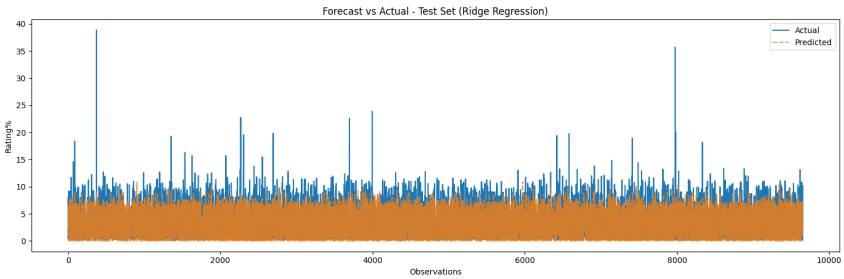
[76] C. D. Burnay, P. Lopes, M. N. de Sousa, J. Félix, and A. L. Carvalho, "Portugal: Diferentes idades, diferentes formas de consumo, a mesma televisão," in *¿ Qué está pasando con las narrativas en la ficción iberoamericana?*, Observatorio iberoamericano de ficción televisiva, 2024, pp. 249–273.

[77] N. Ribeiro and C. Duff Burnay, *As Novas Dinâmicas do Consumo Audiovisual em Portugal*. ERC, Jan. 2016, ISBN: 978-989-20-6593-9.

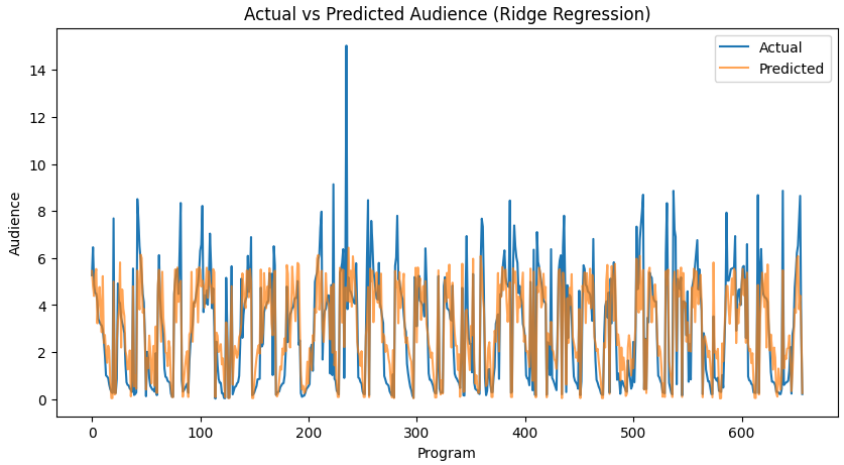
[78] F. Pedregosa et al., "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, Jan. 2012.

Appendix A

Appendix

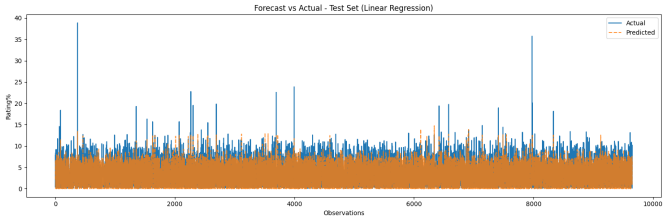


(a) Predicted vs Actual plot for the test subset

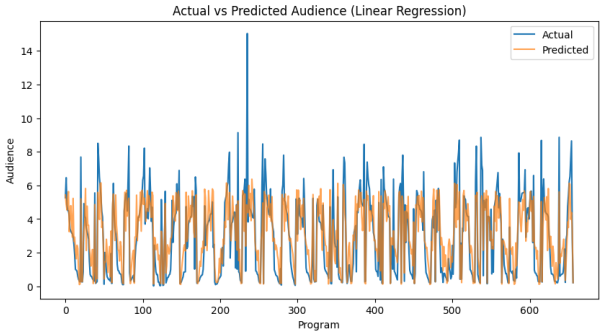


(b) Predicted vs Actual plot for external dataset

Figure A.1: Predicted vs Actual Dataset A - Ridge Regression

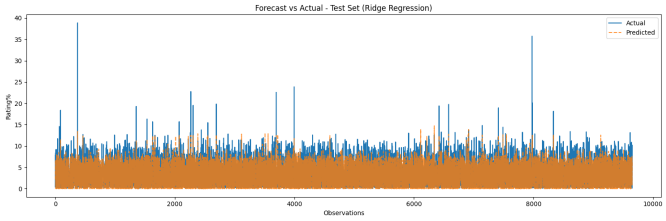


(a) Predicted vs Actual plot for the test subset

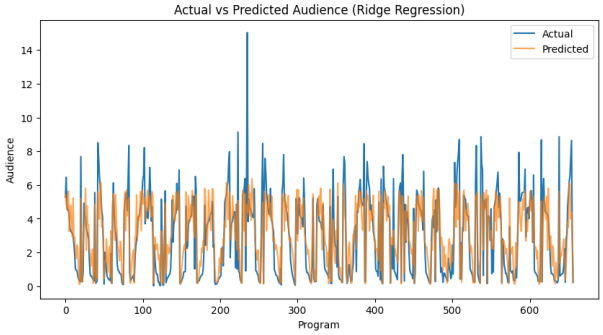


(b) Predicted vs Actual plot for external dataset

Figure A.2: Predicted vs Actual Dataset B - Linear Regression

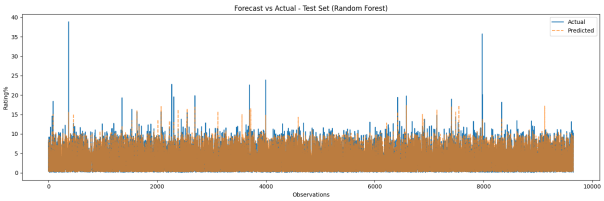


(a) Predicted vs Actual plot for the test subset

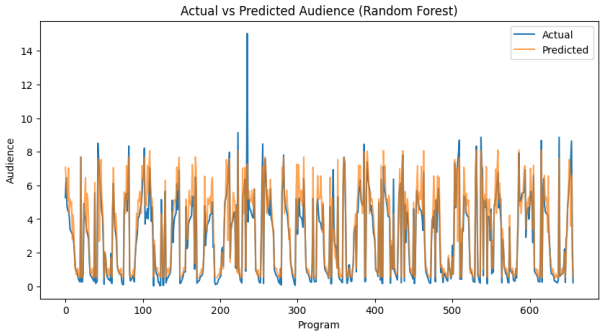


(b) Predicted vs Actual plot for external dataset

Figure A.3: Predicted vs Actual Dataset B - Ridge Regression

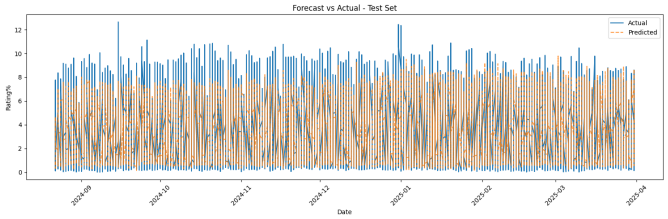


(a) Predicted vs Actual plot for the test subset

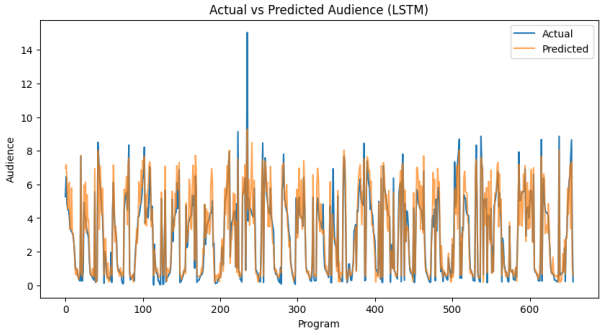


(b) Predicted vs Actual plot for external dataset

Figure A.4: Predicted vs Actual Dataset B - Random Forest

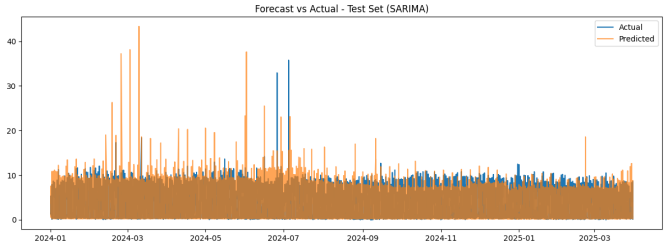


(a) Predicted vs Actual plot for the test subset

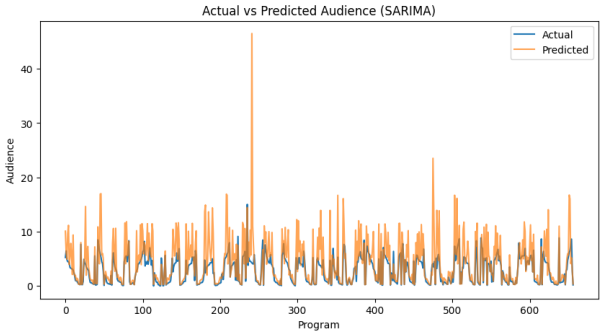


(b) Predicted vs Actual plot for external dataset

Figure A.5: Predicted vs Actual Dataset B - LSTM

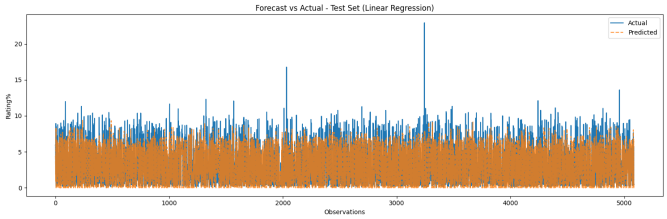


(a) Predicted vs Actual plot for the test subset

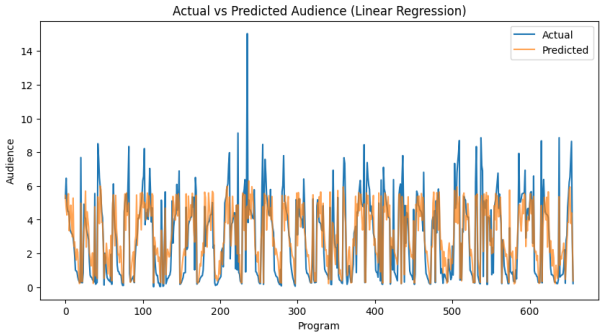


(b) Predicted vs Actual plot for external dataset

Figure A.6: Predicted vs Actual Dataset B - SARIMA

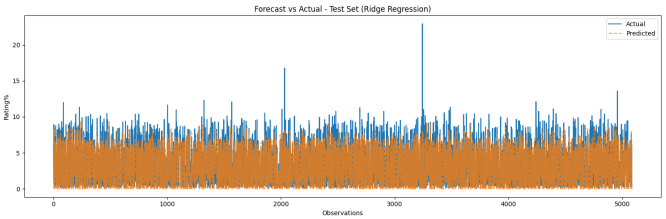


(a) Predicted vs Actual plot for the test subset

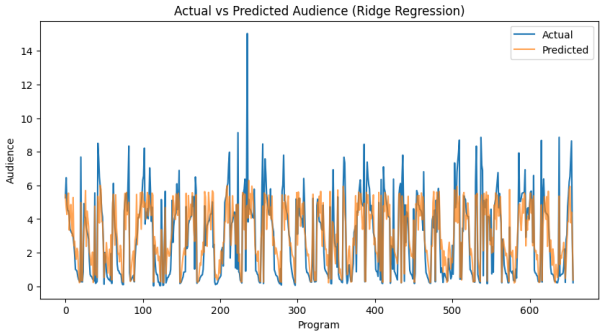


(b) Predicted vs Actual plot for external dataset

Figure A.7: Predicted vs Actual Dataset C - Linear Regression

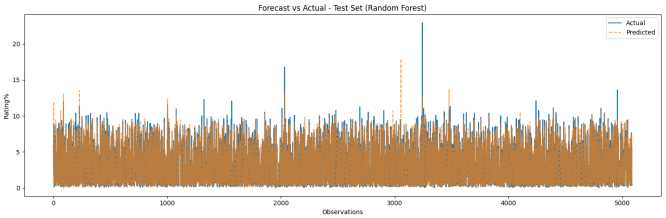


(a) Predicted vs Actual plot for the test subset

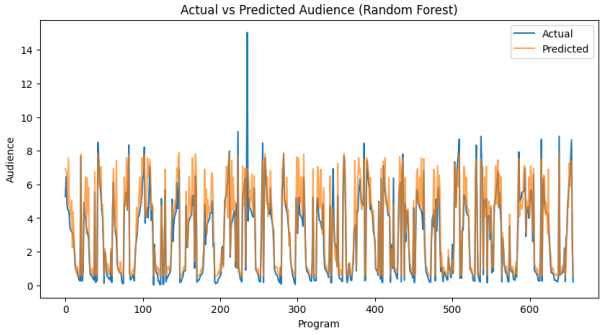


(b) Predicted vs Actual plot for external dataset

Figure A.8: Predicted vs Actual Dataset C - Ridge Regression

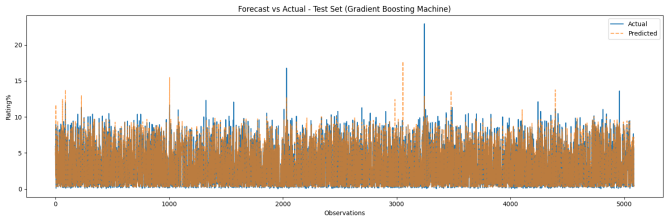


(a) Predicted vs Actual plot for the test subset

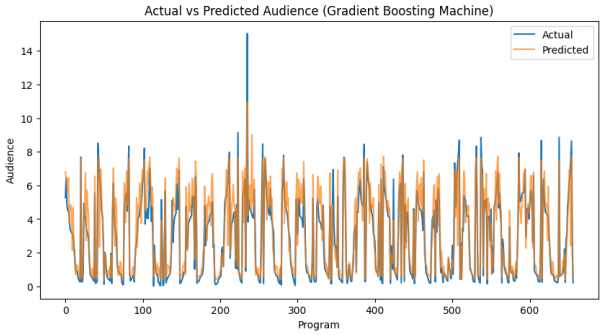


(b) Predicted vs Actual plot for external dataset

Figure A.9: Predicted vs Actual Dataset C - Random Forest

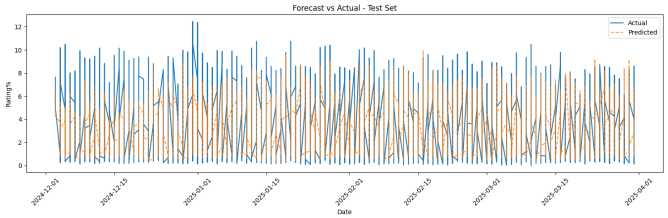


(a) Predicted vs Actual plot for the test subset

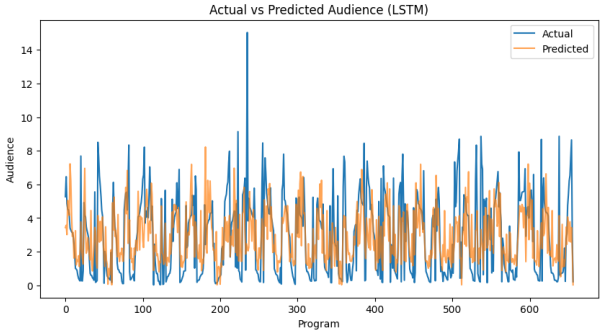


(b) Predicted vs Actual plot for external dataset

Figure A.10: Predicted vs Actual Dataset C - Gradient Boosting Machine

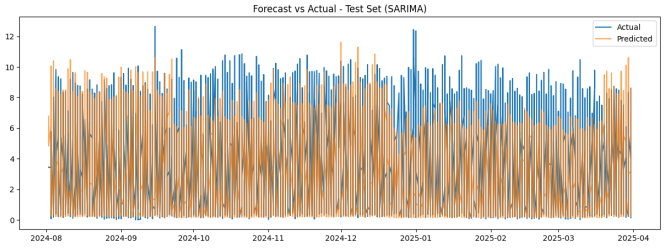


(a) Predicted vs Actual plot for the test subset

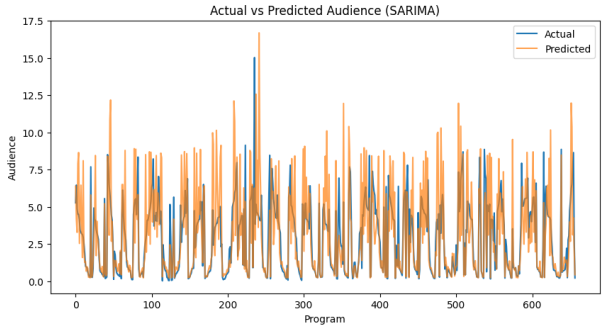


(b) Predicted vs Actual plot for external dataset

Figure A.11: Predicted vs Actual Dataset C - LSTM

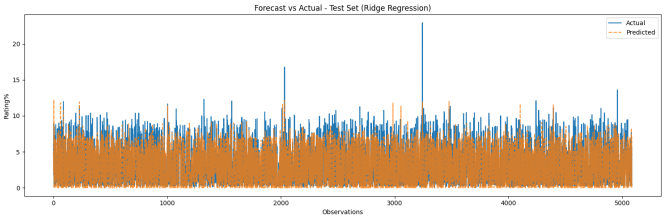


(a) Predicted vs Actual plot for the test subset

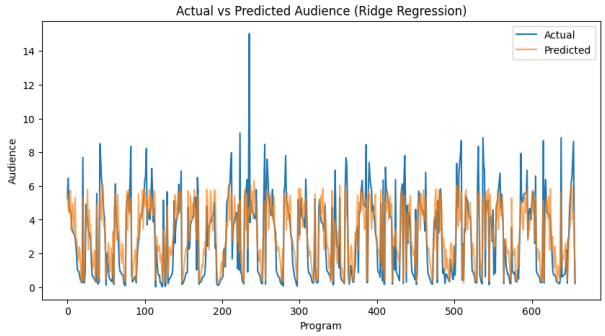


(b) Predicted vs Actual plot for external dataset

Figure A.12: Predicted vs Actual Dataset C - SARIMA

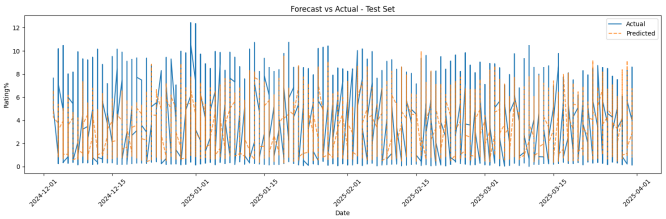


(a) Predicted vs Actual plot for the test subset

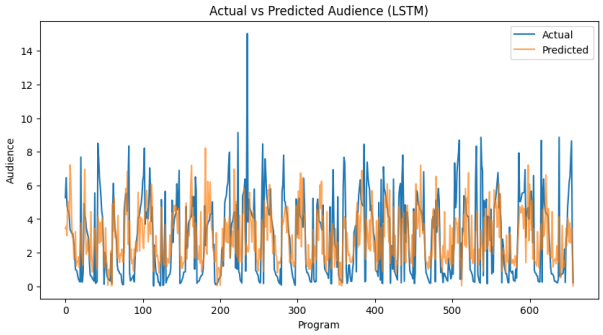


(b) Predicted vs Actual plot for external dataset

Figure A.13: Predicted vs Actual Dataset D - Ridge Regression

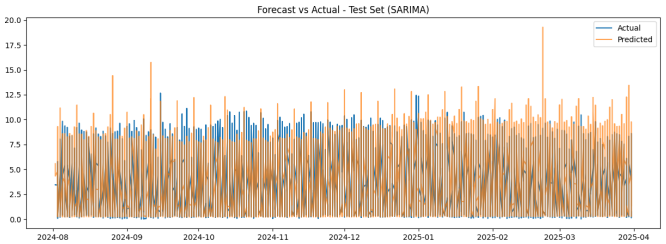


(a) Predicted vs Actual plot for the test subset

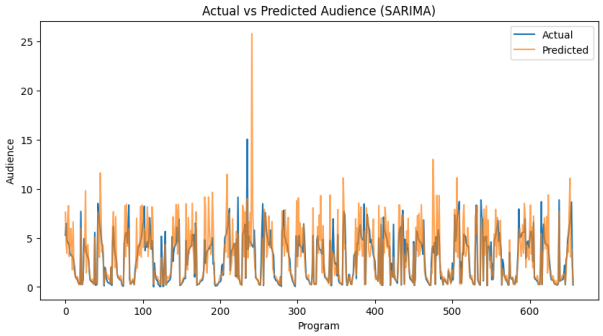


(b) Predicted vs Actual plot for external dataset

Figure A.14: Predicted vs Actual Dataset D - LSTM



(a) Predicted vs Actual plot for the test subset



(b) Predicted vs Actual plot for external dataset

Figure A.15: Predicted vs Actual Dataset D - SARIMA