Machine Learning Methods for Predicting Stellar Parameters in Realistic Molecular Cloud Environments

Master Thesis

Written by

Alejandro Maza Villalpando (HCJ888)

Supervised by

Troels Haugbølle

August 15, 2023

University of Copenhagen

Abstract

This thesis introduces a novel methodology that utilizes data from the RAMSES and MESA frameworks to employ machine learning techniques. The aim is to predict essential stellar parameters, namely temperature, stellar luminosity, radius, and accretion luminosity. The predictions used as inputs are based on values of mass, age, and accretion rate. These input values can be obtained readily in models of a realistic molecular cloud environment. The stellar structure models from MESA diverge from traditional methods that solely rely on mass and age.
The investigation resulted in the implementation of several machine learning models. Using the results from these models, an extensive examination of their strengths and weaknesses was carried out. Furthermore, a comprehensive comparative analysis was conducted, directly contrasting these models with traditional methods like those presented in the literature by DM97 [1]. Based on this comparison, it can be concluded that machine learning methods exhibit high effectiveness as inference algorithms for predicting stellar structure parameters, especially in a dynamic accretion scenario, and exceed the performance of the DM97 model in this context.
An inference module was developed for Python and Fortran, enabling easy integration into simulation frameworks like RAMSES. Additionally, comprehensive documentation was also created to facilitate the incorporation of the models from this thesis into any other programming language. Finally, this thesis investigates potential improvements for the obtained models, outlining different perspectives such as enhancing data quality and investigating alternative machine learning architectures.

Contents

Chapter 1

Introduction

The study of stellar structure is a subject of great interest to astrophysicists. The star's evolution originates with the gravitational attraction that draws gas and dust from vast regions of space into a dense core, in which the life of the star begins. After a star is born, it grows through mass accretion, while maintaining hydrostatic equilibrium. In this thesis, the accretion rate is of particular importance due to its impact on stellar properties like stellar luminosity and temperature. In cases of episodic accretion, the fluctuations in these key properties become inherently complex. The stellar structure can be modeled with modern stellar evolution codes, but the time scale for the evolution of the surface layers is tied to the time scale for significant mass growth and to fluctuations in the accretion rate, making the modeling very costly.
Traditional methods have significantly contributed to our understanding of stellar structure. However, as our knowledge of astrophysical phenomena expands and computational capabilities advance, there is a growing need for modern techniques to tackle the complexities inherent in stellar evolution.

In recent years, machine learning (ML) has emerged as a powerful tool to address complex scientific problems. ML methods offer the potential to unlock new insights and improve our ability to model and predict complex systems. In astrophysics, ML methods have already shown promising results in diverse areas, including galaxy classification [2], exoplanet detection [3], and gravitational wave detection [4], to mention a few. This thesis focuses on studying the capabilities of ML algorithms to predict key stellar properties within realistic molecular cloud environments.

Molecular clouds are the birthplaces of stars, within these, intricate physical processes dictate the formation and evolution of stars, leading to the emergence of a diverse range of stellar structures. Capturing the complex dynamics and interactions in such environments poses significant challenges for traditional modeling techniques. ML algorithms offer a unique opportunity to navigate these complexities, as they excel at identifying non-linear patterns and dependencies in large data sets.

Starting from the collapse of a molecular cloud, to the moment when nuclear fusion begins in the star's core, it can be millions of years. While the direct observation of stars over millions of years is beyond our capabilities, is possible to obtain observation of stars in different stages of their evolution, thus obtaining a precise picture of star evolution. However, stellar evolution simulations are capable to compute the stellar structure of individual stars over millions of years. Making simulations an important tool to understand key properties like accretion. Therefore data obtained from reliable computer simulations was used, in order to train and test the ML models.

The aim of this thesis is to build, train and test ML algorithms using the simulated data, in order to produce reliable results. This will allow astrophysicists to have a modern method to infer the stellar structure. In addition to this, the ML models can be implemented as a part of a simulation, making them more comprehensive and improving their overall utility.

To achieve our goal, we start in chapter 2, where a general overview of the necessary physics to understand the key concepts of star formation is presented, along with ML concepts and terminology. In addition to this, a brief explanation of current techniques used to obtain the stellar structure is explained along with their limitations. In chapter 4 the methods used in order to manipulate the data, train, and test the ML models are described. Moreover, the methods to analyze the ML models, and results, along with the limitations of the ML are described. Chapter 5 presents the results of the ML model's predictions, including a description of the error distribution per input parameter, confidence intervals categorized by star class and age, a comparison of HR diagrams, and an analysis of the machine learning models. In chapter 6 the models were compared between them along with the result from inference using the DM97 model. Furthermore, a discussion on the overall model prediction performance and their particularities was done. Chapter 7 a comprehensive description of the code implementation was introduced through pseudo code, accompanied by detailed explanations to facilitate the integration of the obtained models into any preferred programming language. Chapter 8 outlines an account of the future perspectives for improving the ML model predictions, as well as recommending ML techniques that were not utilized in this thesis. Finally, Chapter 9 presents a summary of the project's findings, along with the corresponding conclusions.

1.1 Data Availability

The data set used for this project, along with various versions resulting from its manipulation, has been saved within the astro cluster at the science high performance computing center, access to these data sets can be granted upon request.

Chapter 2

Background

In order to understand star formation along with the important parameters that form the stellar structure, a brief explanation of these concepts is presented. In addition to this, current methods to model the stellar structure are discussed alongside their downsides as the luminosity spread.

2.1 Star Formation

Stars are formed in the cold, molecular dense parts of the interstellar medium. These regions, known as molecular clouds, have a supersonically turbulent plasma [5], with a constant temperature. As a result, they exhibit a filamentary distribution of matter, with significant variations in density. In the intersection of these filaments, density fluctuations become gravitationally unstable, which leads to star formation [6]. The process of self-gravity effectively condenses material from a core that spans about 10,000 astronomical units (AU) down to a much smaller fraction of an AU. Additionally, due to the conservation of angular momentum, a protoplanetary disk forms around these young stars. These disks are the initial stages of the planetary system, where planets and other celestial bodies may eventually form.
The proto-star evolution depends on possible mass loss or accretion of matter. The process of accretion, which involves the accumulation of matter onto a forming star, does not follow a smooth and continuous pattern. Instead, it is characterized by intermittent and unpredictable variations. Observations of embedded protostars, as well as PMS (pre-main sequence) stars like FU Orionis and Ex-Lupin types, provide compelling evidence of dynamic accretion occurring during both the protostellar and PMS phases [7].

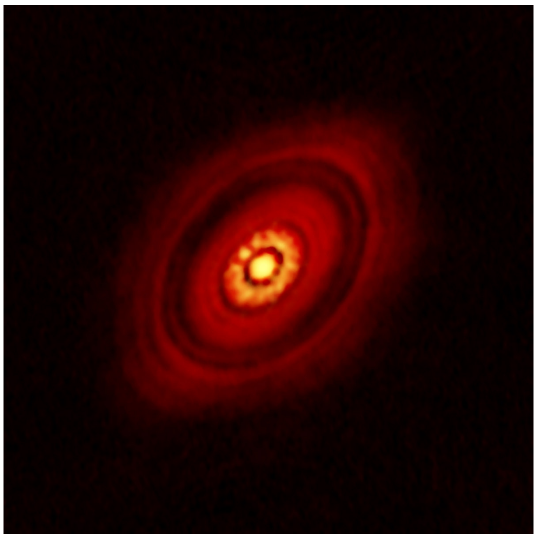

Figure 2.1: VLA capture of HL Tau and its protoplanetary disk [8].

The contraction of the proto-star continues until the temperature in its core gets sufficiently high for nuclear reactions to start in the core. At this point, the star begins to fuse Hydrogen atoms into Helium. Once the nuclear fusion of hydrogen begins, the contraction of the star stops. At this point, stars are in the so-called main sequence. After Hydrogen has been exhausted in the core, the star contracts again, as the temperatures in the core are not sufficient to burn Helium. Due to the pressure change, the temperatures are high enough in the star shells to burn hydrogen. This process repeats until the temperatures are high enough in the core for the fusion of helium to begin, and subsequently to heavier elements. In fact, all heavier elements are produced either during the evolution of the stars, at the end of their life in supernovas, or in the merge of dense celestial objects like neutron stars [9]. The evolution of a star involves a dynamic interplay between gravity and pressure gradients, which arises from an equilibrium between energy generated by nuclear reactions and energy dissipated through radiation at the surface of the star.

2.2 Main Sequence

Stars spend most of their life in the main sequence, specifically in the hydrogen main sequence. Once in the main sequence, the stars are in hydrostatic equilibrium, where the pressure of the gravitational collapse from the outer layers is balanced by the thermal pressure of the core. A distinctive trait of the main sequence stars is that they can be localized in a band of stars that appears on plots of temperature against luminosity. These types of plots are known as Hertzsprung–Russell diagrams (HR diagrams) after their developers, Ejnar Hertzsprung and Henry Norris Russell.

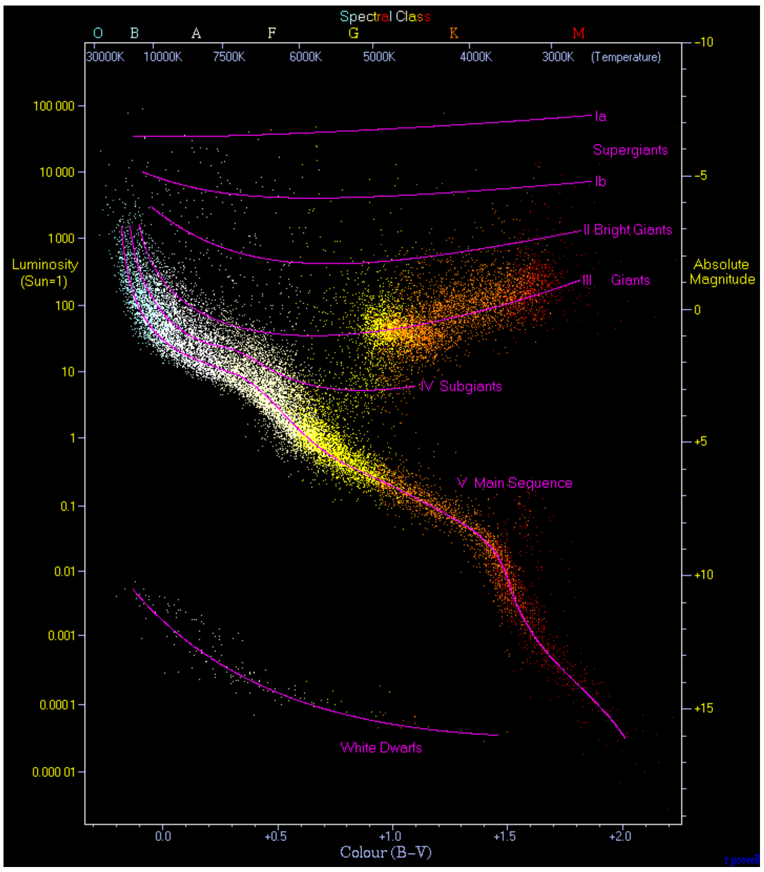

Figure 2.2: HR diagram featuring 22,000 stars sourced from the Hipparcos Catalogue [10], alongside 1,000 low-luminosity stars (red and white dwarfs) obtained from the Gliese Catalogue of Nearby Stars [11]. The Main Sequence, a diagonal band stretching from the top-left to the bottom-right, represents ordinary hydrogen-burning stars like the Sun. On the upper-right side, giant stars form a distinct cluster. Above them, is possible to find the much rarer bright giants and supergiants. The lower-left portion of the diagram is occupied by a band of white dwarfs [12]

HR diagrams depict the primary surface characteristics of stars, the effective temperature T_{eff}, and the stellar luminosity L_s. The basic characteristics of a photon leaving the stellar surface, are given by T_{eff} and L_s. Therefore, these properties play a crucial role in characterizing the observable traits of young stars and their impact on the surrounding environment. One of these properties is the effective stellar radius R by the relation given in equation 2.1.

$$L_s = 4\pi\sigma T_{eff}^4 R^2 \tag{2.1}$$

Where σ is the Boltzmann constant.
The HR diagrams are also known as color-magnitude diagrams, as the color index is predominantly determined by the surface temperature of the star [13]. The apparent magnitude is a measure of the brightness of a star observed from Earth and is related to the apparent magnitude if the distance d is known. When radiation is emitted isotropically and there is no absorption between the star, is possible to relate the magnitude to distance as follows [13]:

$$l = \frac{L_s}{4\pi d^2} \tag{2.2}$$

In order to obtain an HR diagram one can plot the color index and apparent magnitude for a group of stars, in a star cluster, thus obtaining the whole spectrum of star spectrum in the given cluster. Once an HR diagram of a cluster is known is possible to know the age of the cluster. As the cluster ages, the massive stars, exhaust their fuel first turning it into a red giant. Consequently, the star, shifts in its position on the HR diagram. Main sequence stars follow a diagonal line on the HR diagram, whereas red giant stars form a horizontal line known as the red giant branch. As more massive stars enter the red giant branch, the upper portion of the main sequence vanishes, referred to as the main sequence turnoff. By understanding how a star's mass influences its fuel consumption rate and position on the diagram, we can determine the cluster's age using the main sequence turnoff [14]. In addition, the HR diagrams help to determine the reddening towards the star group. Reddening refers to the absorption and scattering of light as it passes through interstellar dust and gas. HR diagrams can also assist in estimating the distance to a star group. By comparing the observed luminosity of stars in the group with their calculated intrinsic luminosity, is possible to determine their distance. Metalicity is another important property that HR diagrams are used to estimate. Metallicity refers to the abundance of heavy elements in a star. As stars with different metallicity exhibit distinct positions on the HR diagram, it becomes possible to assess the metallicity of a group of stars using their positions on the diagram.

2.3 Stellar Structure

Stellar structure models offer a comprehensive insight into a star's internal composition, providing predictions related to its luminosity, temperature, and evolutionary path. As a result of differences in elemental composition and energy transport mechanisms, stars of various classes and ages exhibit unique internal structures.
Stars are classified by analyzing the absorption features in stellar spectra, we can categorize stars into different spectral types based on their temperatures. The current system widely used is the Harvard spectral classification scheme, originally formulated at Harvard College Observatory in the late 1800s. Annie Jump Cannon later refined and published the scheme in its present form in 1924 [15]. The stellar classification with its associated temperature can be seen in table 2.1

Stellar Classification.	
Class	Temperature $10^3(K)$
O	> 25
B	$[11, 25]$
A	$[7.5, 11]$
F	$[6, 7.5]$
G	$[5, 6]$
K	$[3.5, 5]$
M	< 3.5

Table 2.1: Temperature in K for each star class [16].

In order to understand the changes in stellar structure is necessary first to understand the timescales at which this occurs. The first of these is the so-called dynamic timescale ($t_{dynamic}$). One can define the dynamic timescale as the duration it would take for a star to undergo a substantial collapse in the absence of pressure within the star, otherwise known as free-fall. Thus the dynamic timescale can be mathematically described as the time taken for a particle to fall under the gravitational field of the star, therefore after some mathematical manipulation [17] is possible to obtain $t_{dynamic}$ given by equation 2.3.

$$t_{dynamic} = \sqrt{\frac{R^3}{GM}} \tag{2.3}$$

Where G is the gravitational constant, and M is the mass of the star. In the majority of cases, we observe no evidence of motion with such timescales in stars. This suggests that the forces acting on the star are in hydrostatic equilibrium [13].
Furthermore is possible to define the thermal timescale (t_{KH}). Also known as the Helmholtz-Kelvin contraction it represents the duration a star would take to collapse if a star has no internal sources of energy, and it radiates energy at the same rate [18]. The radiated energy is a consequence of the loss of gravitational potential energy [19], thus the thermal timescale is given by the relation 2.4.

$$t_{KH} = \frac{GM^2}{RL_s} \tag{2.4}$$

The period during which a star remains on the main sequence, i.e. the time at which hydrogen is fused to helium is generally referred to as the main-sequence nuclear timescale $t_{nuclear}$. In this context, the energy available from this reaction is determined by the mass difference between the reactants and products of the nuclear reaction, as expressed by $E = Mc^2$, where c is the speed of light. It is estimated that the energy loss from the fusion of hydrogen into helium is $\frac{\Delta E}{E} \approx 0.007$ [20], therefore the nuclear timescale is given by equation 2.5

$$t_{nuclear} \approx 0.007\frac{Mc^2}{L_s} \tag{2.5}$$

In order to get an idea of the range of these timescales we can calculate the case of the Sun with $M_\odot$, $L_\odot$, and $R_\odot$, such timescales are given in table 2.2

Sun Timescales.	
Timescale	Time$\approx$
$t_{dynamic}$	$1600s$
t_{KH}	$3 \times 10^7 yr$
$t_{nuclear}$	$10^{11} yr$

Table 2.2: Calculated timescales for the Sun.

Up to this point, we have established the star's classification and the timescales in which the stellar structure undergoes transformations. Now, we can introduce the equations governing stellar structure. When there are no significant changes occurring in the stellar structure within the timescales defined by the dynamic timescale, it indicates that the star is in hydrostatic equilibrium [13]. Achieving hydrostatic equilibrium requires a delicate balance between the pressure gradient inside the star and the gravitational force pulling inward, therefore is necessary to have a relation describing the change of pressure with respect to its size, given in equation 2.6.

$$\frac{dP}{dR} = -\frac{GM(R)\rho(R)}{R^2} \tag{2.6}$$

where P is the pressure and ρ is the density of the star. Another crucial aspect to understand alterations in the stellar structure within the dynamic timescale involves understanding the variations in the star's mass. In other words, the process when the star is not in hydrostatic equilibrium. This can be referred to as the process of mass transfer from one shell within the star to another, described by the mass continuity equation 2.7.

$$\frac{dM}{dR} = 4\pi R^2 \rho \tag{2.7}$$

In the context of the thermal timescale, the star's thermal energy is transferred by two primary mechanisms: radiation and convection. Radiation involves the transmission of thermal energy through the star. The efficiency of radiation relies on the mean free path of photons and particles [21]. If photons are frequently absorbed, the radiative transport efficiency decreases, and the extent of absorption in the star is characterized by its opacity(κ). We can relate the change of temperature with respect to the radius as seen in equation 2.8.

$$\frac{dT}{dR} = -\frac{4\kappa\rho L}{64\pi R^2 \sigma T^3} \tag{2.8}$$

In equation 2.8 is possible to see that the radiative transfer is directly proportional to the opacity, density, and luminosity flux. Consequently, an increase in any of these factors will lead to a higher temperature gradient, in such case thermal transfer by convection occurs [21]. In an ideal gas, when convection is adiabatic the energy transfer is given by equation 2.9.

$$\frac{dT}{dR} = \left(1 - \frac{1}{\gamma}\right) \frac{T}{P} \frac{dP}{dR} \tag{2.9}$$

where γ is the ratio of the specific heats $\gamma = \frac{c_P}{c_V}$.
Throughout most of the star's evolution, the primary source of energy comes from nuclear reactions. However, additionally, energy is released through gravitational contraction. Thus the general energy production can be derived from the first law of thermodynamics [13], yielding equation 2.10.

$$\frac{dL}{dR} = 4\pi R^2 \left[\rho\epsilon - \rho\frac{d}{dt}\left(\frac{u}{\rho}\right) + \frac{P}{\rho}\frac{d\rho}{dt}\right] \tag{2.10}$$

Where ϵ is the rate of energy production per unit mass, t is time, and u is the internal energy per unit volume.
Equations 2.6, 2.7, 2.8, 2.9, and 2.10 collectively describe the stellar structure. By examining these equations, it becomes evident that crucial parameters influencing the stellar structure include temperature, star luminosity, radius, mass, and naturally, time. Nonetheless, the equations mentioned above assume a constant mass, but in a realistic molecular cloud environment, a proto-star mass increase through accretion. As a result, it becomes necessary to incorporate the dynamics of accretion to accurately describe the evolving stellar systems.

The process of accretion is non-steady and involves multiple variables, such as the properties of the interstellar medium (ISM) in which the star is forming, the angular momentum of the accreting matter [22] among others. To study stellar accretion in a realistic molecular cloud environment, numerical simulations, and models are often used. These simulations take into account factors such as gravitational contraction, feedback from massive stars, and the dynamics of gas accretion onto young stars. These models help in understanding the complex interplay between accretion and feedback in the process of star formation within molecular clouds [7]. The accretion process gives rise to several observable properties in stars, including magnetic activity [23], variations in their observed brightness due to accretion luminosity, and the occurrence of bipolar outflows [24]. For the interest of this thesis, we will focus on the accretion luminosity. The accretion luminosity (L_{acc}), is not precisely part of the stellar structure, however, the observed luminosity is not only coming from the star but from the sum of L_s and L_{acc}:

$$L_{Tot} = L_s + L_{acc} \tag{2.11}$$

Where the accretion luminosity is defined by:

$$L_{acc} = \frac{GM(t)M\dot{M}\alpha}{2R} \tag{2.12}$$

The terminology used in this context aligns with the one proposed by Baraffe et al. (2009). [25] for the energy transfer at the accretion shock. The accretion luminosity is then divided into two terms, one for the energy radiated away from the proto-star and one for the energy that is injected into the proto-star:

$$L_{acc}^{out} = \epsilon(1-\gamma)\frac{GM\dot{M}}{R} \tag{2.13}$$

$$L_{acc}^{in} = \epsilon\gamma\frac{GM\dot{M}}{R} \tag{2.14}$$

where M is the proto-stellar mass, α is the thermal efficiency parameter, G is the gravitational constant, and $\dot{M}$ is the accretion rate. The factor depends on the details of the accretion process, with $\epsilon \leq 1$ for gravitationally bound material and $\epsilon \leq 0.5$ for boundary layer accretion from a thin disc.

Finally is possible to clarify what is meant by stellar structure as the parameters needed to study the evolution of a star in an HR diagram, the observable parameters, alongside the parameters needed to calculate any of these parameters in equations 2.1,and 2.13. Specifically in the rest of this thesis, we refer to the stellar structure parameters as T_{eff}, L_s,R, L_{acc}, M, $\dot{M}$ and star age.

2.4 Luminosity Spread

The luminosity spread refers to the variation in the brightness or luminosity of a group of stars or an individual star. As a simple classical approximation, a group of young stars is assumed to form simultaneously with their final mass. In the context of the main sequence, such a group of stars would trace a distinct line in the HR diagram, a so-called isochrone, which solely relies on the age and metallicity of the group. In reality, as mentioned before proto-stars do not evolve in isolation from the interstellar medium but grow through accretion [7]. Moreover, while age can be defined for a star-forming region, the ages of individual stars in the region will vary. The combined effect of a proto-stellar age spread and time-varying proto-stellar accretion for individual proto-stars is one of the explanations, why stars in young clusters do not follow an isochrone but have a significant luminosity spread [7]. Additionally, the presence of binary or multiple star systems in these clusters can also contribute to the luminosity spread, as their combined light can affect the overall brightness observed [26]. According to the classical understanding, when there are significant differences in luminosity among stars, it suggests an ongoing star formation process that takes place over an extended period of tens of millions of years. This notion favors a slow-paced star formation scenario, which contradicts the observed and modeled lifetimes of star formation regions [27]. Baraffe et al. [25], [28] have suggested that the apparent age spread in star-forming regions could be explained by accreting star-forming models, specifically models featuring time-varying accretion rates.

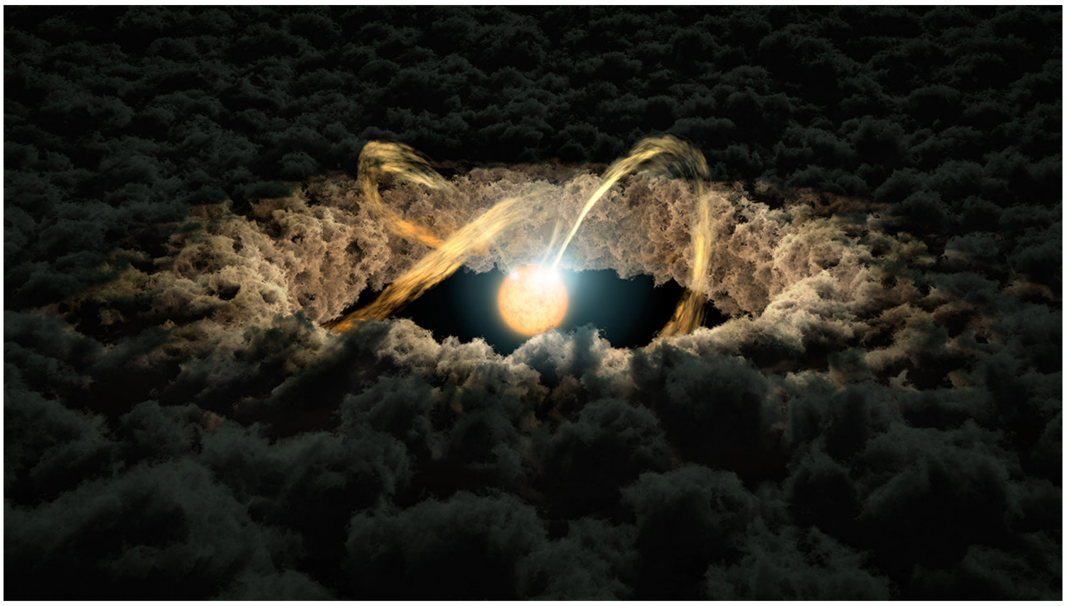

Figure 2.3: Image of a proto-star enclosed by a protoplanetary disk. Within this disk, material flows along the star's magnetic field lines and settles onto the surface of the star. Upon impact, the material illuminates the star. Image credit: NASA/JPL-Caltech [29].

2.5 Protostellar Evolution Models

As mentioned before traditional stellar structure calculations that do not involve accretion result in clearly defined isochrones. These isochrones have been employed for many years to estimate the ages of regions where stars are formed, and in model calculations of star-forming regions to characterize the stellar feedback. However, the significant variation in luminosity observed within these populations has raised concerns [7]. Modeling young stars with a time-varying accretion is possible, but expensive. Including a time-varying accretion rate complicates the endeavor, and makes the problem higher-dimensional in nature. This has been done by Sigurd et al. [7] by creating a simulation using RAMSES [30] [31]. Using adaptive mesh refinement (AMR), RAMSES efficiently solves numerical algorithms for the equations of magnetohydrodynamics [32], a well-suited theory to describe the evolution of ISM and star formation. AMR is a technique used in numerical simulations to dynamically adjust the resolution of a computational mesh based on the local features of the solution. It allows for the efficient allocation of computational resources by concentrating computational effort where it is most needed. The RAMSES code used in this implementation is the modified version by Haugbølle et al. (2017) [33], which includes the incorporation of random turbulence driving, sink particles to serve as a sub-grid model for protostars, technical enhancements enabling efficient scaling to multiple thousand cores, and an upgraded solver that maintains stability even in supersonic flows featuring high Mach numbers.
The results from Sigurd et al. can be seen in figure 2.4, where is possible to see the evolution of the model at different stages, up to $2.5 \times 10^6 yr$. As a result, the model achieves a well-developed stellar population, with a considerable number of pre-main-sequence stars undergoing contraction towards the main sequence along the Hayashi track. This result becomes comparable in age to some of the youngest nearby star-forming regions. Towards the end of the simulation, 214 stars are found to be over a million years old, while 328 stars exhibit an accretion rate of less than $10^{-7} M_{\odot} yr^{-1}$, indicating the completion of their primary accretion phase.

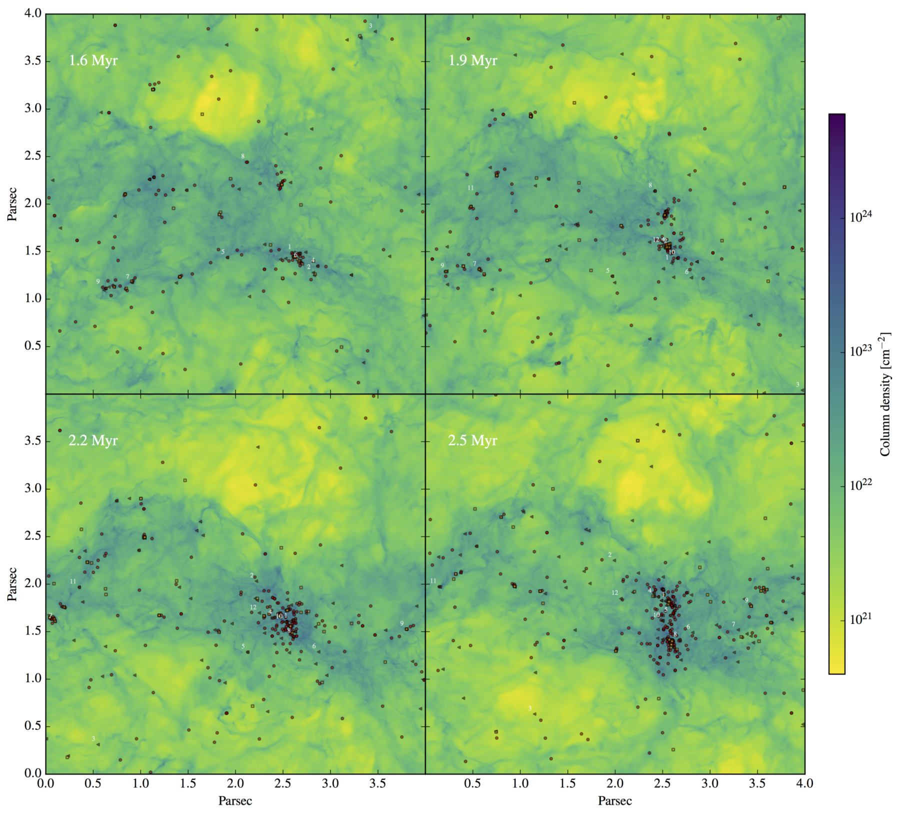

Figure 2.4: Evolution of the model from $[1.6, 2.5] \times 10^6 yr$. Stars are categorized by mass where red circles showcase stars with masses $M < 0.5 M_\odot$, brown triangles stars in the range $[0.5, 1.5] M_\odot$, and orange squares stars with $M > 1.5 M_\odot$. The numbering relates to the accretion profiles shown in figure 2.5. [7].

The accretion rate profile of selected stars is shown in 2.5 where is possible to see the accretion history of 6 stars in the mass range $[0.7, 0.8] M_\odot$ at an age of $2.3 \times 10^6 yr$.

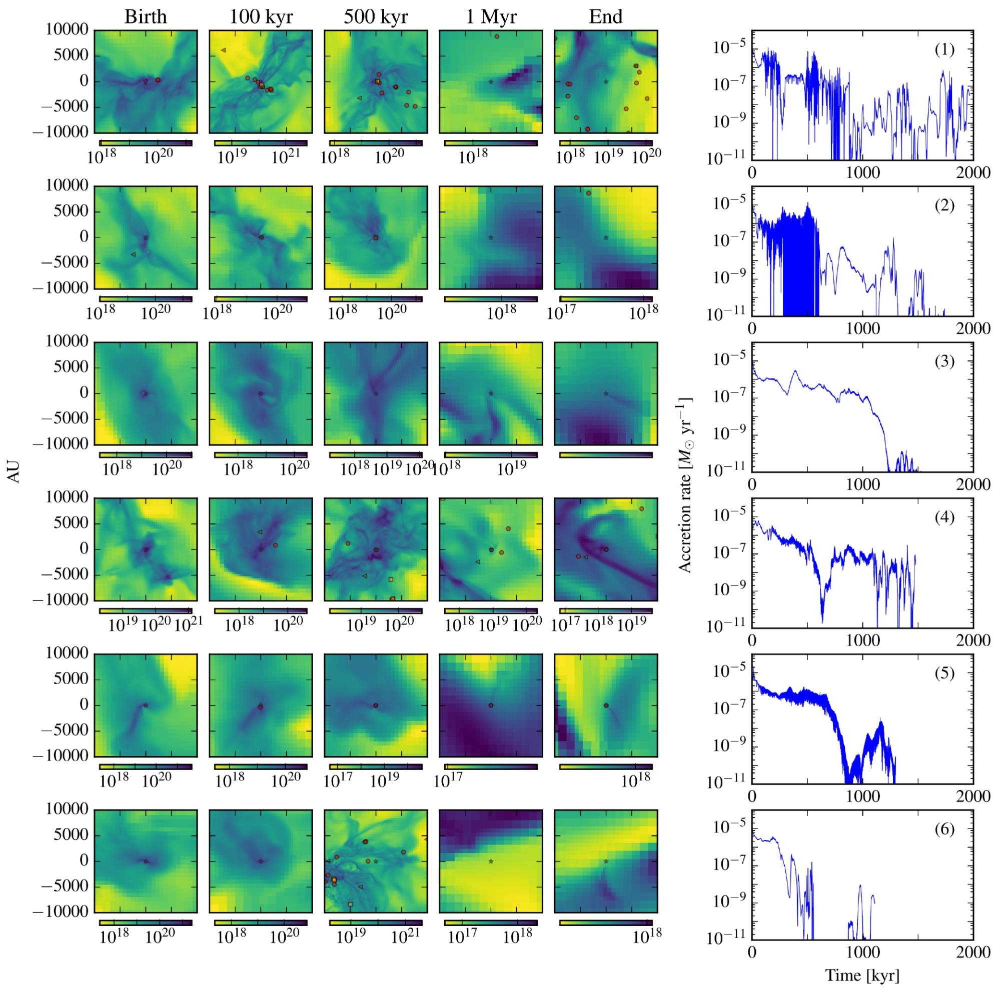

Figure 2.5: Plot showing the gas distribution in the evolution from the birth of the star, up to [$2.3 \times 106yr$]. The right-hand side panel shows the accretion rate history.[7].

As seen in figure 2.5 RAMSES is able to produce the accretion profiles produce along with some of the stellar parameters, namely the mass M, and accretion rate $\dot{M}$, as a function of stellar age. In combination with an initial condition, a protostellar seed with a mass similar to that of Jupiter, these profiles serve as valuable input for stellar structure calculations. This implementation has been integrated into the Module for Experiments in Stellar Astrophysics (MESA) [34], one of the most extensively utilized codes in this field. By utilizing a particular time series of mass accretion, this method is a viable approach in order to gain a better understanding of how accretion affects the protostellar phase. However, its implementation comes with several challenges:

- MESA is a complicated code to run, and allowing for time-varying accretion requires extra customization
- Running a single stellar track requires a substantial amount of computational resources, taking up to a CPU core-year. Because of the limited scalability of MESA, this results in a wall-clock run-time of up to a month for a single star.
- MESA represents an extensive software infrastructure, composed of over a million lines of code and incorporating several gigabytes of tabulated data relevant to various microphysics aspects, including nuclear reaction rates, multi-wavelength opacities, and realistic equations of states.
- Coupling MESA with simulations codes like RAMSES, in order to allow co-evolution of protostars and the surrounding material is highly non-trivial.

In this thesis, we use the data set obtained from 321 realistic large-scale simulations of a molecular cloud fragment (4 pc) with precisely recorded mass accretion rates specifically mass M, accretion rate $M_{\odot}$, and age. In addition to these parameters, we used T_{eff}, L_s, R, and L_{acc} obtained with MESA for their associated RAMSES stellar tracks.

2.6 Existing Infreance Methods

As mentioned in section 2.5 obtaining the stellar structure parameters using MESA comes with complications, however, lightweight stellar structure inference methods do exist. A classic example of this is the one created by D'Antona and Mazziteli (DM97) [1] which interpolates the T_{eff} and L_s, using tables of know values. However, this model considers a classical star evolution with no accretion, producing misleading results due to the luminosity spread among other problems.
In addition to this, the DM97 is outdated, as it doesn't use any modern statistical method to make predictions. In recent years new predicting methods using machine learning have proven to be very successful. Therefore a new modern way of using ML to predict the stellar evolution parameters is indispensable.

The aim of this thesis is to investigate ML methods, along with their implementation in the prediction of T_{eff}, L_s, R, and L_{acc}, given a M, $\dot{M}$ and star age. As discussed in this section modern methods that take into account the full picture of the stellar evolution are needed. Moreover, a successful ML model can be easily implemented in simulation codes like RAMSES, as the inference process is only a few lines of code, that can be translated into practically any programming language. Finally, ML models are fast to execute adding an almost negligible overhead in their implementations.

Chapter 3

Machine Learning

Machine learning (ML) algorithms build models using data samples in order to predict a desired outcome. One of the most important aspects of ML is the quality of the data, therefore it is important to analyze the data, along with modifying it in order to produce a quality data set.
Machine learning algorithms can be classified into supervised, unsupervised, and reinforcement learning, for this thesis, supervised learning is used, in particular a regression algorithm. In supervised learning, the algorithm learns by example. A known data set with the input parameters M, $\dot{M}$ and star age, is given to the algorithm, with a known output T_{eff}, R, L_s, and L_{acc} as a method to train the ML algorithm. The algorithm learns a method to arrive at the outputs given the inputs. Regression is a method of analyzing the connection between input parameters and output parameters. It is often used in machine learning to predict continuous parameters by creating a line of best fit through the data points, where the distance between each point and the line is minimized.

3.1 Exploratory Data Analysis for Machine Learning

Exploratory Data Analysis (EDA) is a crucial step in the data analysis process for ML. EDA involves analyzing data sets to gain insights and identify patterns, anomalies, and relationships. It is helpful to understand the main characteristics of the data before making any assumptions and can guide the selection of appropriate statistical techniques for further analysis or modeling. Some common techniques and best practices for conducting EDA are:

- Identifying the input and output parameters, along with their relationships.
- Visualisation of the data distribution.
- Identifying types of data classes and their distributions.
- Detecting outliers.

Using EDA in this project helped us to understand the distributions of the data set, allowing us to decide how to split the data in order to obtain relevant data set for each phase in the stellar evolution as explained in detail in section 4.1.1.

3.2 Feature Engineering

In ML it is common to modify the data in order to obtain extra parameters, or to make the data better suitable for the ML algorithm. Some common feature engineering techniques are:

- Feature selection: Selecting the relevant input parameters.

- Feature scaling: Common scaling techniques are standardization, subtracting the mean and dividing by the standard deviation, or normalization.
- Feature transformation: A common one is to obtain the $\log 10$ of the feature.
- Feature extraction: Involves creating new data points for example adding previous data points as a way to add time-dependent information on the current state.

In particular in this thesis feature engineering was used, to scale the necessary parameters by normalizing the data, and modifying the data entries to include temporal data as explained in sections 4.1.5 and 4.1.6 respectively.

3.3 Hyper parameter Tuning

In order to obtain the best possible prediction, it is necessary to adjust the ML algorithm parameters, this process is known as hyperparameter tuning (HPT). Hyperparameters are the parameters of the model that are not learned during the training process but rather are set by the user before training. Examples of hyperparameters include the learning rate of an optimization algorithm and the number of hidden layers or neurons in a neural network.
Selecting the optimal values for these hyperparameters is important because they can have a significant impact on the performance of the model. HPT involves systematically testing different values of the hyperparameters and evaluating the performance of the model on a validation set. This process can be done manually, by adjusting the values of the hyperparameters and evaluating the model's performance, or automatically, using algorithms such as grid search [35], random search [36], or Bayesian optimization [37].

3.4 Gradient Boosting

Decision trees are supervised machine learning algorithms used for both classification and regression tasks. The decision tree algorithm works by recursively partitioning the data into subsets based on the values of different input features. It creates a tree-like structure where each internal node represents a decision based on a specific feature, and each leaf node corresponds to a predicted outcome or value. Gradient boosting is a framework that uses an ensemble of weak learners to improve the accuracy of predictions. It works by iteratively training decision tree models on the residuals of the previous trees. The residual is the difference between the predicted value and the true value.
The process begins with an initial model, usually a simple model such as a decision tree with a single node or a constant value, and uses this model to make predictions.

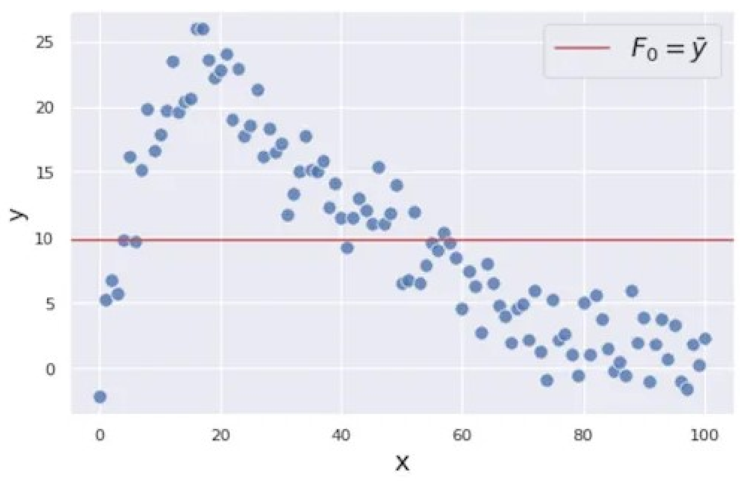

Figure 3.1: The first predictions for a gradient boosting decision tree involves predicting taking the mean indicated by the red line [38].

The errors or residuals of this initial model are then calculated.

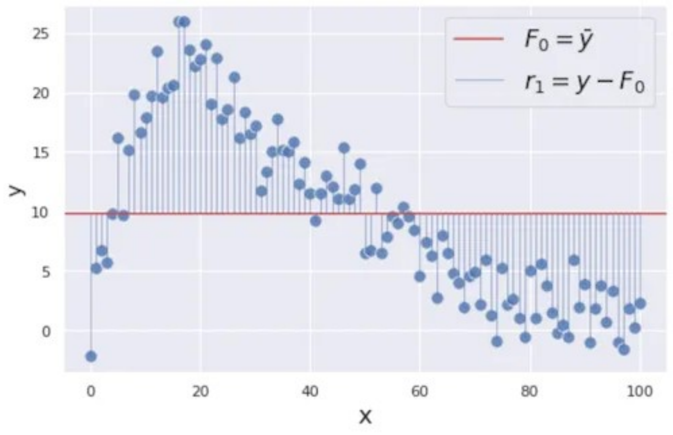

Figure 3.2: Calculation of residuals involves obtaining the difference of the data point with the mean indicated by the blue lines [38].

A new decision tree is trained to predict these residuals.

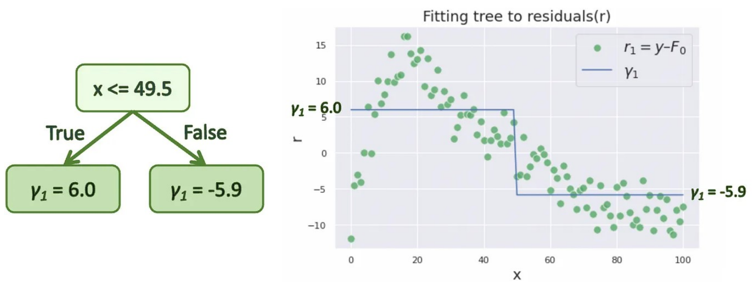

Figure 3.3: New decision tree is created and new predictions are calculated [38].

This new decision tree is added to the ensemble, and the predictions from all the trees in the ensemble are combined to make the final prediction.

$$F_1 = F_0 + \nu\gamma_1 \tag{3.1}$$

where F_0 is the initial prediction, γ_1 is the second layer of predictions, based on the first branching of the tree as seen in figure 3.3, and ν is a scaling factor known as the learning rate. The learning rate is used to avoid overfitting and ranges between 0 and 1.

The process is repeated multiple times, with each new tree being trained to correct the mistakes of the previous trees. The final ensemble model is a combination of all the decision trees, where each tree contributes a prediction in the form of a vote or a weighted average.
The trees are grown with the goal of minimizing the overall prediction error, this is done by using an optimization algorithm such as gradient descent to find the best parameters of the tree, this process is also known as boosting.
In this project, the first supervised learning algorithm we have decided to use is a Gradient boosting algorithm called Light Gradient Boosting Machine (LightGBM) released by Microsoft in 2017 [39]. LightGBM is a gradient-boosting framework that uses tree-based learning algorithms. It is designed to be efficient and scalable and is particularly well-suited for large-scale data sets. LightGBM is faster than traditional gradient-boosting libraries, such as XGBoost, giving equal accuracy with up to 10 times less training speed [40].

3.5 Deep Neural Networks

First proposed in 1944 by Warren McCullough and Walter Pitts, neural networks (NN) are a means of doing machine learning, in which a computer learns to perform some task by analyzing training examples [41]. Generally, neural network models consist of layers of neurons densely interconnected. Each neuron might be connected to several neurons in the previous layer, from which it receives data, and several neurons in the next layer, to which it sends data. A diagram of a neural network can be seen in figure 3.4.

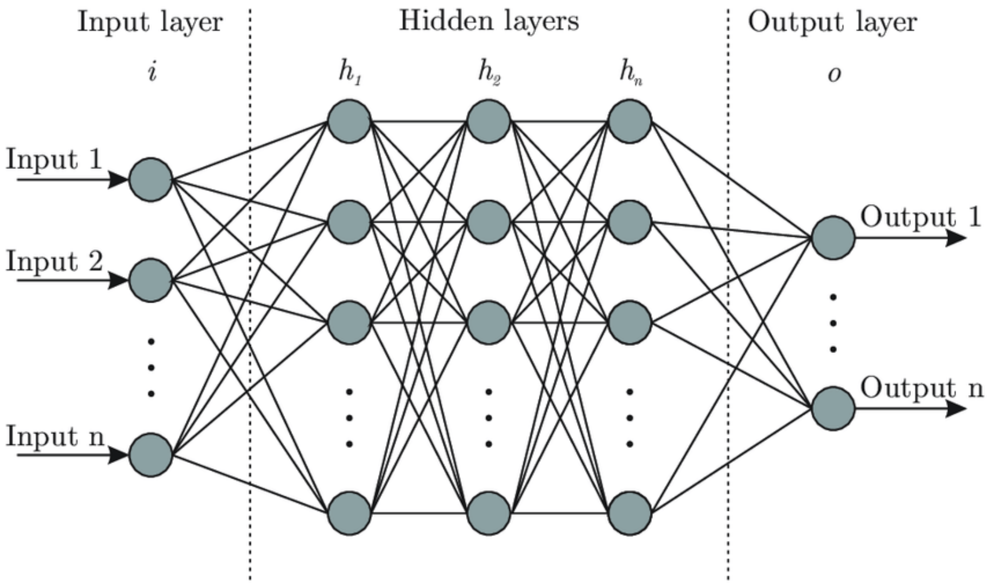

Figure 3.4: Neural network diagram[42].

In figure 3.4 is possible to see the so-called architecture of a simple NN. The architecture terminology is the following:

- Neuron: The basic unit of a NN. A neuron takes inputs, applies weights and biases, computes the weighted sum, and passes the result through an activation function to produce an output.
- Input layer: This is the first layer in a neural network. The number of neurons in the input layer corresponds to the dimensionality of the input data.
- Hidden Layers: They are intermediate layers between the input and output layers. Each hidden layer consists of multiple neurons, and the connections between these layers carry the weighted and biased information from the previous layer.

- Output layer: The final component in the NN architecture, depending on the problem at hand can have one or more neurons, and the type of activation function used may vary. In this layer is where the prediction of the NN is obtained.

The feed-forward process is the basic operation of the NN and, refers to the flow of information through the NN from the input layer to the output layer. The feed-forward process starts with the input layer, which receives the initial data or features that need to be processed. Each connection between the input layer, the hidden layers and the output layer has an associated weight w and a bias b. The w and b are the components of the NN that are adjusted during the learning process. The input data is multiplied by the w and added to the b. Then the incoming data for each neuron is aggregated into a weighted sum as shown in equation 3.2.

$$b + \sum_{i=1}^{n} x_i w_i \tag{3.2}$$

After calculating the weighted sum of w and b at each neuron, an activation function f is applied to introduce non-linearity into the NN. A typical f in regression problems is a Rectified Linear Unit (ReLU) defined mathematically as follows:

$$ReLU(x) = max(0, x) \tag{3.3}$$

where x is the input to the activation function, and the output of ReLU is the maximum of 0 and x. Finally, the mathematical representation of the outputs in each neuron is:

$$f(b + \sum_{i=1}^{n} x_i w_i) \tag{3.4}$$

The output of the NN $\bar{y}$ is compared to the ground truth y using a loss function E, which quantifies the difference between them. In this thesis we use as loss function the mean absolute log error given by:

$$MALE = \frac{1}{N} \sum_{i}^{N} |\log_{10}(y_i) - \log_{10}(\bar{y}_i)| \tag{3.5}$$

In the first forward pass, w and b are assigned a random value, which is then adjusted by the process of backpropagation. This algorithm propagates the gradients of E backward through the network. This process involves computing the gradients of the E with respect to the network parameters, in order to update the parameters accordingly. In other words, computing the gradient allows the algorithm to find the relation between the change in w, b, and f with respect to E. Naturally, the smaller the output of E the better the NN is. Finally, after the gradients are computed an optimization algorithm is used, commonly known as gradient descent, where the w and b are updated using the gradients information, in an attempt to minimize E as shown in figure 3.5.

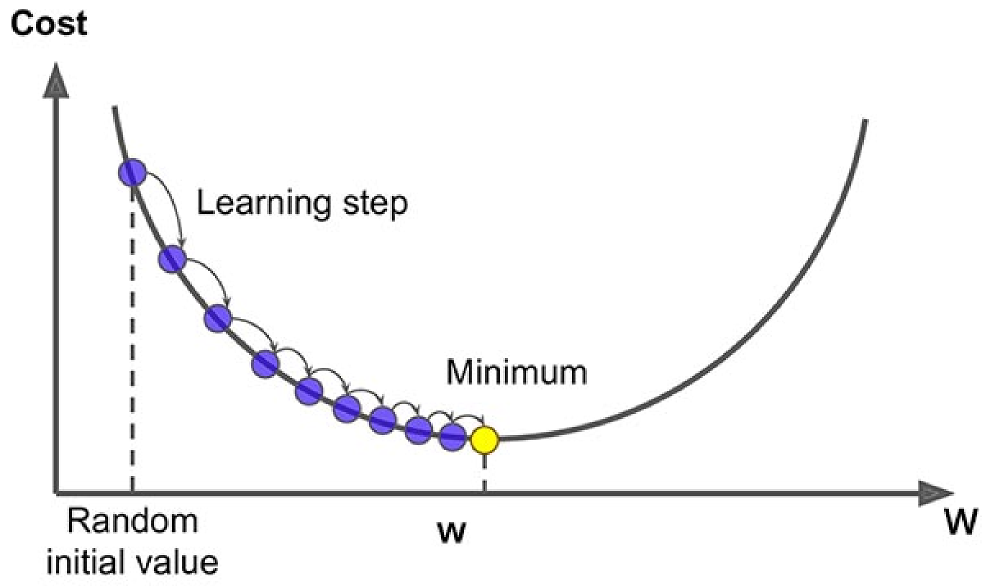

Figure 3.5: Gradient descend through backpropagation, where the purple dots represent the value achieved after each learning step, and the yellow dot represents the local minima. [43].

Stochastic gradient descent is an efficient way to perform gradient descent. This algorithm updates the parameters using a random subset of the data set to approximate the loss function. Therefore, introducing some randomness in the process hence the stochastic. In this project, we use Adam (Adaptive Moment Estimation) to perform the gradient descent. Adam is an extension of the stochastic gradient descent algorithm that aims to overcome some of its limitations, such as the need for manual tuning of the learning rate[44].

3.6 Machine Learning Summary

Machine learning involves several steps, from preparation and curation of the input data, and selection of algorithm to hyperparameter tuning. Each of these steps involves a range of details that can change the performance of the desired outputs. Therefore it is possible to think about the ML process as an experimental one, where a range of different data sets, HPT algorithms, optimizers, etc, are tested. In the following chapter, the process to perform these experiments is outlined in detail, in order to achieve reproducibility. In addition to this results were carefully recorded for each experiment, obtaining useful information to achieve better and faster results in subsequent experiments.

Chapter 4

Machine Learning Methods

4.1 Data Analysis

In machine learning projects, it is usual to analyze data in various ways to comprehend its peculiarities and make informed decisions about how to approach the ML problem. In the following section, we offer an overview of the data analysis methods utilized in this thesis.

4.1.1 Data Distribution

Understanding the data distribution of both the input and predicting parameters is essential. Consequently, an analysis of this distribution was carried out. Additionally, an investigation was conducted to determine the proportion of data with accretion mass $M_{\odot} > 0$ and $M_{\odot} = 0$. The importance of this investigation lies in the fact that when $M_{\odot} = 0$, the mass remains constant, fundamentally changing the input vector's nature, as the star's age becomes the only variable

4.1.2 Splitting Data set

To conduct supervised machine learning, it is necessary to evaluate the ML algorithm using an unseen data set with known truth values. This process helps confirm the accuracy of the model's predictions. Therefore the data set was split into a training set and a testing set. Naturally, the ideal training and testing sets need to be representative of the whole data set, including stars in all ranges of mass, age, and accretion rate. To achieve this goal, the simulation data was categorized into four groups based on the final mass of each simulation. The testing set was subsequently obtained by selecting a percentage of the simulations from each mass category.
For the testing set approximately 16% of the simulations was chosen, with approximately $9.5X10^6$ or 27% of the total data points. After splitting the data sets the distributions of the training and testing sets were compared to get an overview, of the full range of the input parameters.

4.1.3 Balance Training set

In order to prepare the data for the training a common practice for classification problems is to balance the training data set. Balancing a data set in machine learning refers to the process of adjusting the proportion of different classes or categories in a data set to ensure that each class has a similar number of examples. In other words, if we have a data set containing 100 samples, with 90 samples belonging to class A and only 10 samples to class B, it indicates an imbalanced data set. To address this, balancing the data set can be achieved by either removing some samples from class A or adding more samples to class B, in order to ensure a more balanced distribution of samples between the two classes. Balancing a data set is important in ML because many algorithms perform poorly when the data set is imbalanced. For example, a classification algorithm trained on an imbalanced data set may become biased towards the majority class and perform poorly on the minority class. By balancing the data set, we can ensure that the algorithm is trained on a more representative sample of the data and is more likely to generalize well to new examples. Even though this is common practice for classification, is possible to do for regression [45], showing promising results.

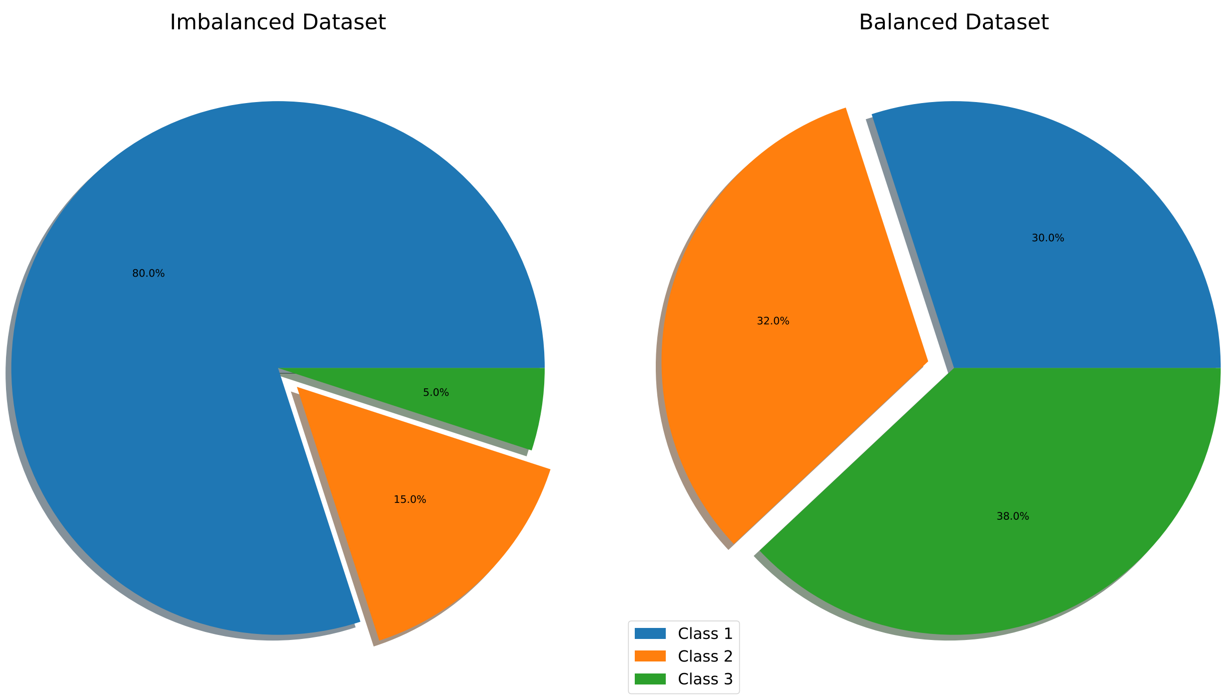

Figure 4.1: The plot on the left-hand side illustrates an example of an imbalanced dataset, while the one on the right-hand side shows a balanced dataset.

Based on the results from the distribution analysis, it became evident that balancing the training set could be advantageous. This balancing process aims to enhance the accuracy of predictions while also reducing the dataset size to improve the training time of the ML algorithms. Therefore a balanced of the data was performed in the following way:

- Binning the data.
- Identifying the size of the second smallest bin, denoted as bin_2. This choice is made to avoid losing valuable data information by selecting the smallest bin.
- Keeping the whole data in the smallest bin.
- Taking a random sample of size bin_2, for the remaining bins.
- Creating a new data set containing the data from the smallest bin and the random samples obtained in the previous step.

4.1.4 Reduce Training set

By analyzing the simulation data, it was observed that there are certain data points where the stellar evolution exhibits minimal change, indicating a stable stellar structure. Consequently, it becomes feasible to reduce the data set by excluding these points where no significant alterations in the stellar structure occur. The process is as follows:

- Obtaining the difference Δ between consecutive points of the prediction parameters.
- Establishing a significant difference for each of the parameters Δsig.
- Removing the point if $\Delta < \Delta sig$.

The following Δ_{sig} for each parameter was used:

- $t_{eff}\Delta_{sig} = 10^{0.001}K$
- $L_{star}\Delta_{sig} = 10^{0.0009}L_{\odot}$
- $R\Delta_{sig} = 10^{0.0005}R_{\odot}$
- $L_{acc}\Delta_{sig} = 10^{0.0007}L_{\odot}$

Following this procedure enables us to obtain a different dataset for experimentation, with almost no loss in information. Additionally, it reduces the training time of the ML algorithms.

4.1.5 Scaling Data

A common practice in ML is to normalize the data and bring it to a common scale. It is particularly useful when the data has different ranges and units. Machine learning algorithms can benefit if all features have similar magnitudes. By scaling the features to a fixed range, is possible to ensure that no particular feature dominates the learning process due to its larger values. Another reason why feature scaling is applied is that few algorithms like Neural network gradient descent converge much faster with feature scaling than without it [46]. Therefore we used a scaling method based in the `MinMaxScaler` from `sklearn` [47], a method that scales the values between $[0, 1]$ using the following equation:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{4.1}$$

This scaling method works well for not Gaussian distributions, and is sensitive to outliers.

4.1.6 Feature Engineering

An important aspect of stellar evolution is its history, this naturally becomes more important due to the fact of accretion. Extended periods of accretion or periods with a high accretion rate, can have a great impact on stellar evolution. Moreover, an episodic accretion can induce changes in the stellar structure, even after the episodic accretion phase has concluded [48].
It was confirmed that it is indeed possible to create this type of data set during a simulation like RAMSES, which is crucial for data preparation. As integrating the ML results into a simulation constitutes one of the primary objectives of this thesis. In order to create a realistic temporal data set the following steps were taken:

- Splitting the data into two subsets: one containing stars with ages $age_0 < 2000yr$ and the other containing stars with ages $age_1 \geq 2000yr$.
- Extracting samples from the age_0 set within the range $[age_{current} - (\Delta t \times 2), age_{current} - \Delta t]$ until the age_0 set was fully processed. The process then continued with the age_1 set. The introduction of the parameter $\Delta t \times 2$ was necessary to reproduce the age variability present in the data obtained from the simulation.
- Choosing a random state vector with mass, age, and accretion rate from the previously obtained sample.

- Using the state vector obtained in the previous step and the current state vector, interpolation was performed to estimate the mass and accretion rate at the desired age if there was no data available in the desired age range in age_0 or age_1 sets.

Following these steps another data set type was created taking the input parameters with a time difference Δt equal to 10, 100, and 1000 years.

4.1.7 Shapley Additive Explanations

Shapley additive explanations (SHAP), is a method based on cooperative game theory used to increase the transparency and interpretability of machine learning models [49]. They provide insights into the contribution of each feature to the final prediction.
Shapley additive explanations quantify the contribution that each feature brings to the prediction made by the model, using the marginal contribution of each feature. SHAP values are based on the idea that the outcome of each possible combination of features needs to be considered in order to determine the importance of a single feature [50]. By considering all possible feature combinations and their contributions, SHAP values provide a consistent way to allocate a value among the features. The number of possible feature combinations is given by 2^F where F is the number of input parameters. The SHAP algorithm trains a predictive model for each of the feature combinations using the same hyperparameters and input data as the final model. Therefore there are 2^F sets of models, trained with every possible combination of features, this includes the empty set with no features, being the "prediction" in this set the average value of all the predictions. Naturally, each set of models is expected to be better the more features are included, in order to quantify how the model improves in every set.
The marginal contribution is defined as the difference between a prediction within a model set with respect to the prediction in the previous model set that does not include the desired input, and can be calculated as follows:

$$Mc_{fs,(fs_k)}(x) = pred_{fs}(x) - pred_{fs_k}(x) \tag{4.2}$$

Where fs is the feature in the current set, fs_k is the feature in the previous set that do not include fs, and $pred(x)$ is the prediction given an input feature x. Each feature has $Mc = \binom{F}{f}$, where f is the number of features in each model set. The marginal contributions are combined through a weighted average, where the sum of the weights (w) is equal to 1. As an illustration for our case, one could choose the mass as shown in figure 4.2.

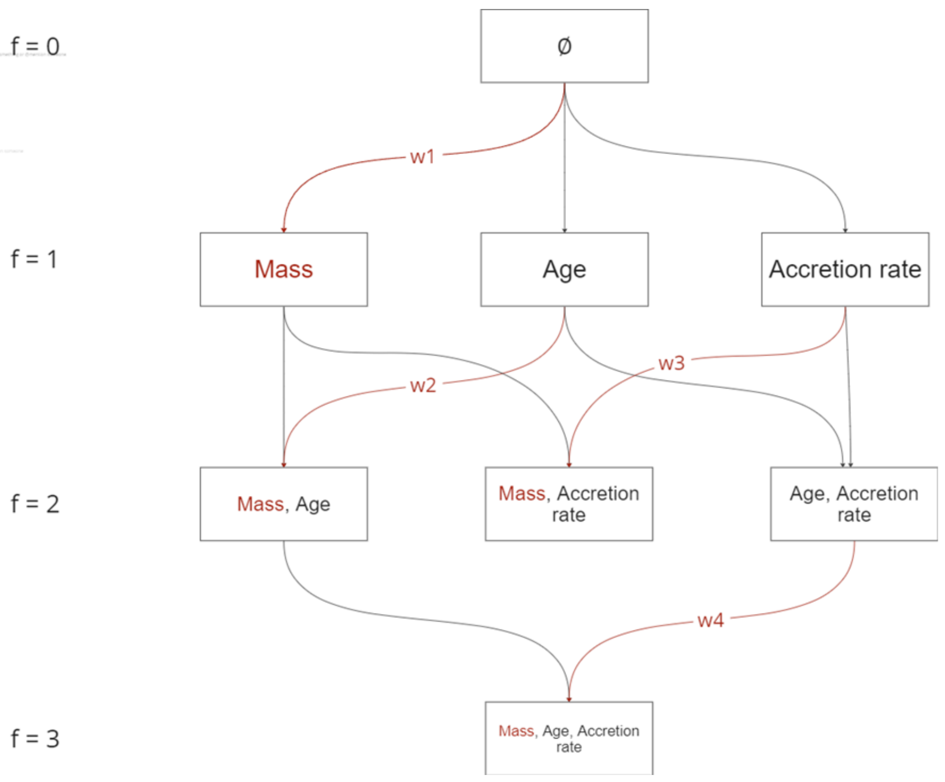

Figure 4.2: Marginal contribution diagram for mass.

Finally, the SHAP values can be obtained using the formula as explained by Slundberg and Lee [51].

$$SHAP_f(x) = \sum_f (f \times \binom{F}{f})^{-1} Mc_{fs,(fs_k)}(x) \tag{4.3}$$

.

By examining the SHAP values for a specific prediction, it is possible to gain insights into which features contribute the most and in what direction. This enables us to explain the model's decision-making process and identify critical factors affecting the prediction. In this work, the `SHAP` framework for Python was used [52] in order to calculate the SHAP values of the LightGBM model.

4.2 Error Metrics

In order to quantify the accuracy of the ML models the predictions can be compared with the known values calculating the absolute log error given by:

$$ALE = |\log_{10}(y) - \log_{10}(\bar{y})| \tag{4.4}$$

Where y is the known value, and $\bar{y}$ is the one obtained by the model.
In order to evaluate the overall performance of the ML model on the test data set, the median of the absolute log errors ($MedALE$) was computed, as it was judged the most informative metric. Furthermore, it was essential to incorporate additional error metrics to gain insights into the behavior of the ML models, especially in extreme cases where the models predict significantly inaccurate values. To accomplish this, the maximum absolute log error ($Max(ALE)$) was determined for each individual simulation, and subsequently, the median of these values ($MedMaxALE$) was computed as a secondary error metric.

In the case when the machine learning predictions for accretion luminosity, fall below $10^{-5.99}$, an adjustment is made to set $L_{acc} = 10^{-6.1} L_{\odot}$. This is necessary because accretion luminosity values smaller than $L_{acc} = 10^{-5.99} L_{\odot}$ do not exist and therefore, this manual correction can be applied.

Consequently, it becomes essential to align the simulation values accordingly. This method guarantees a more precise quantification of accuracy for the accretion luminosity.
This is done because there is no accretion luminoisty smaller than $L_{acc} = 10^{-5.99} L_{\odot}$ therefore is possibkle to make a manual correction in the ML predictions by considering all the predicted values lower than $L_{acc} = 10^{-5.99} L_{\odot}$ as a constant value $L_{acc} = 10^{-6.1} L_{\odot}$ allowing us to accuratley quantify the accuracy in a correct maner.

4.3 Bayesian Optimisation

Bayesian optimization is a powerful technique for optimizing functions, where the objective function's analytical form or gradients are unknown. It is particularly useful in machine learning for hyperparameter tuning (HPT).
Bayesian optimization combines Bayesian inference and optimization to iteratively explore the search space efficiently. It employs probabilistic models, such as Gaussian processes, to model the unknown objective function and guide the search toward regions of interest,[53].
In this project, the following steps were performed:

- Specifying the range and type of hyperparameters to be optimized.
- Defining the function to be optimized.
- Initializing the surrogate model by selecting a probabilistic model to represent the objective function.
- Choosing an acquisition function that balances exploration to guide the search.
- Selecting new points to evaluate based on the acquisition function and updating the surrogate model.
- Iteratively sampling new points, evaluating, and updating the surrogate model until the end of the training process.

The algorithm was implemented using the `Optuna` framework [54], which offers a high-level interface for efficient implementation and execution of Bayesian optimization.

4.3.1 Optuna Features

Using `Optuna` following features the HPT was performed:

- Automatic algorithm selection: `Optuna` automatically selects and configures appropriate algorithms based on the defined problem.
- Pruning: It supports the early stopping of unpromising trials to allocate computational resources effectively.
- Visualization: It provides visualization tools to analyze and understand the optimization process.

`Optuna` based Bayesian optimization provides several advantages, including efficient exploration of the search space and a reduction in the number of function evaluations required.

4.3.2 Cross Validation

The best ML model is a model that can generalize, and adapt to new unseen data, one common technique to make this possible is cross-validation. The basic idea behind cross-validation is to divide the data set into two parts: a training set, which is used to train the model, and a validation set, which is used to evaluate the model's performance. Nonetheless, a single split of the data set into training and validation sets may not fully represent the model's performance on unseen data. To overcome this limitation, cross-validation requires multiple splits of the data set into training and validation sets, with each split utilizing a distinct subset of the data as the validation set.
The most common form of cross-validation is k-fold cross-validation [55], which involves dividing the data set into k equally sized folds. In the first iteration, the model is trained on the k_1 fold and evaluated on the remaining fold. This process is repeated k times, with each fold serving as the validation set once. The results of each fold are then averaged to obtain an overall estimate of the model's performance.

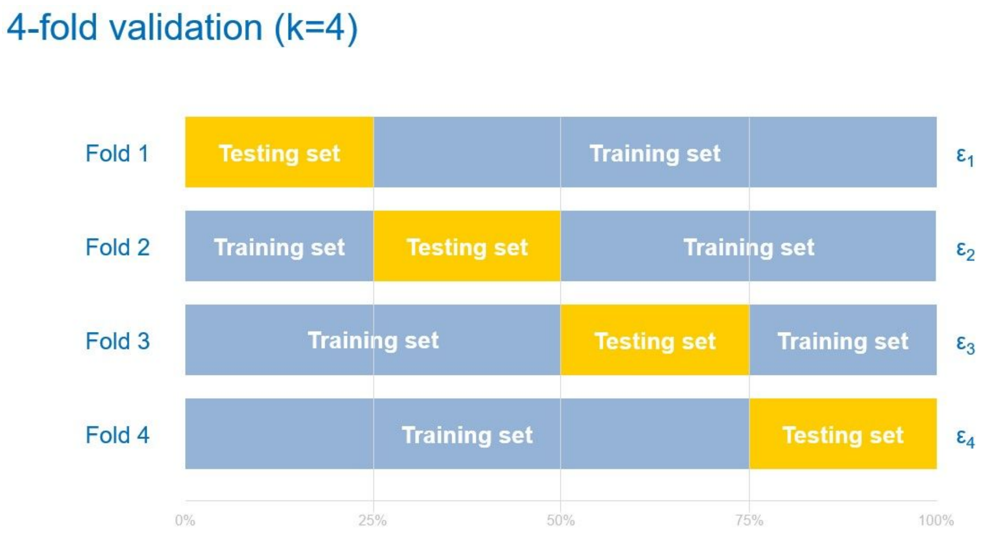

Figure 4.3: Cross validation example where 25% different partitions of the data are taken as a testing set for every fold [56].

In this thesis, the cross-validation was implemented with $K = 3$, during the Bayesian optimization algorithm. The mean of the results of the cross-validations was taken in order to return a single minimum value to minimize.

4.4 Model Selection

After completing the hyperparameter tuning process, it becomes possible to train a machine learning model using the best hyperparameters. This training was done for 400 epochs on each of the data sets. To identify the best model, the following data processing pipeline was constructed:

- Splitting the data into accretion $M_\odot yr^{-1} > 0$, and non-accretion $M_\odot yr^{-1} = 0$.
- Scaling both training data sets to obtain a scaler for normalization.
- Using the accretion data set to create three additional data sets: one with the balanced method, another with the reduced method, and a third with the full data set.
- Combining the no accretion set with the previously mentioned data sets to create a mixed set.
- Performing HPT using Bayesian optimization and cross-validation with all the data sets.

- Using the best HP obtained with the Bayesian optimization to train a model.
- Scaling the test data set using the training scaler.
- Making predictions on the test set using the trained model.

The accretion pipeline method can be summarized in figure 4.4

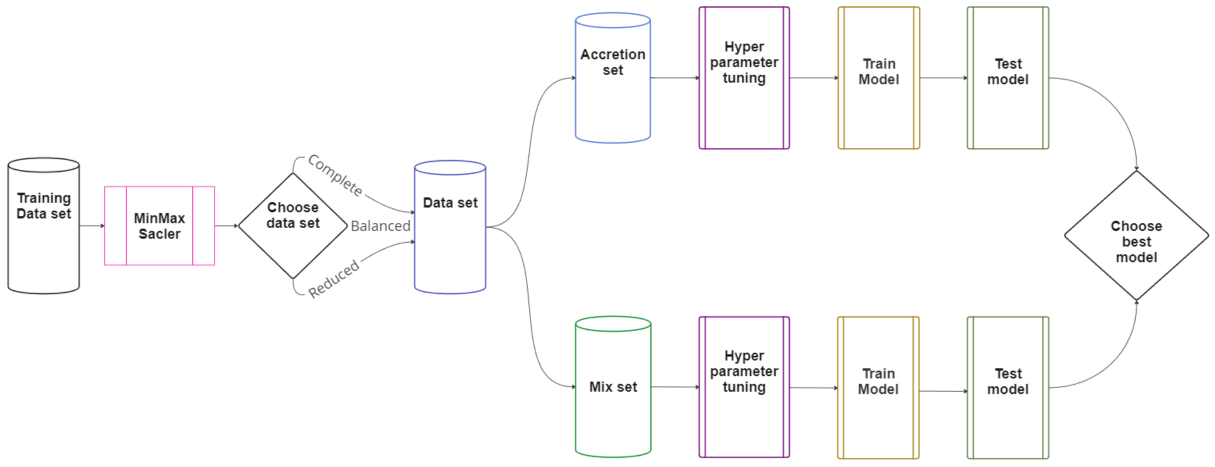

Figure 4.4: Data pipeline, to obtain the best machine learning models for the accretion set .

In order to choose the best model the ALE was computed for each y $\bar{y}$ pair for both the whole data set and for the individual simulations. Then the $MedALE$ was calculated for the whole test data set. It was noted that relying solely on the $MedALE$ metric led to instances where models performing well overall were chosen, but they exhibited higher values in the $max(ALE)$ metric. Therefore, for the 53 individual test simulations, the $MedMaxALE$ was calculated in order to obtain models that perform well overall and try to minimize the maximum errors.

4.5 LightGBM

LightGBM is a great framework to use as a starting point, given that is simple to use, is fast, and doesn't require GPUs. Consequently, it presents a favorable option for establishing a workflow that can be readily applied to future models.

4.5.1 LightGBM Parameters

There are several parameters that affect the model performance as it is possible to see in the official LightGBM documentation [57]. For this project, the HPT was performed with the following parameters:

- `num_leaves`: The parameter that controls the number of decision leaves in a single tree. The decision leaf of a tree is the node where the decision happens.

- max_depth: The parameter that controls the maximum distance between the root node of each tree and a leaf node, is a key parameter to avoid overfitting.
- learning_rate: Step size parameter of the gradient descend, Controls the learning speed. Typical values lie between 0.01 and 0.3 [53].
- n_estimators: The parameter that controls the number of decision trees. Set to constant n_estimators = 3000.
- subsample: Specify the percentage of rows used per tree-building iteration. That means some rows will be randomly selected for fitting each tree. This improved generalization and training speed [58].

Using this parameter in combination with Bayesian optimization the best parameters of each model were found.

4.6 Pytorch Neural Network

Neural networks are a powerful tool for regression tasks, allowing us to predict continuous numeric values based on input features. In a regression neural network, the network learns to approximate the underlying mapping between input variables and the target variable.
The pipeline to finding the best models for the NN is the same as outlined in section 4.4 and in figure 4.4, with the difference that only the reduced and balanced data sets were used, and not the complete data sets. This could be a harmful factor to find the best model using all the available resources, however, results from the LightGBM model indicate that it is not necessary to use the complete set to obtain the best results. Thanks to this result it was possible to optimize the time and usage of the GPUs.

4.6.1 Neural Network Parameters

The neural network architecture is determined by the HPT resulting in slight variations in each NN's architecture. However, the general architecture consists of a sequence of input and hidden layers, with a customizable number of neurons in each of these. The tunable parameters include the following:

- First layer size: The number of nodes or neurons that the input vector is multiplied by.
- Hidden Layer Size: The number of nodes or neurons in each hidden layer of a neural network.
- Learning Rate: A hyperparameter that controls the step size at each iteration during the training of a neural network. It determines how quickly the model adjusts its weights based on the calculated gradients. A higher learning rate may result in faster convergence but risks overshooting the optimal solution, while a lower learning rate may lead to slower convergence or getting stuck in local minima.
- Batch Size: The number of training samples or data points processed in a single forward and backward pass during one iteration of training. It affects the speed and efficiency of the learning process. Smaller batch sizes allow for more frequent weight updates but may lead to noisy gradients, while larger batch sizes provide smoother gradients but require more memory.
- Gamma: Decrease the learning rate after a certain amount of epochs have been completed. It reduces overshooting and allows the algorithm to converge.

- `Number of Hidden Layers`: The total count of hidden layers in a neural network architecture.

Apart from these variables, the neural network architecture also includes certain fixed parameters. Specifically, the activation function utilized is a `ReLU` function. The selected loss function for this project is the $mean(ALE)$ ($MALE$). Finally, during the backpropagation phase, the optimization algorithm employed is the Adam optimizer.

4.6.2 Number of Parameters Convergence

The number of parameters N_{params} in a neural network can be quantified by counting the weights and biases. To calculate the number of w, the following approach is used:

$$N_{params}(w) = I \times H_0 + \sum_{i=0}^{n-1} H_i \times H_{i+1} + H_n \times O \tag{4.5}$$

where I represents the number of neurons in the input layer, H_0 is the number of neurons in the first hidden layer, n is the number of hidden layers and O is the number of neurons in the output layer. Then the number of b is calculated:

$$N_{params}(b) = I + \sum_{i=0}^{n} H_i \tag{4.6}$$

Finally adding both:

$$N_{params} = N_{params}(w) + N_{params}(b) \tag{4.7}$$

An analysis of the optimal number of parameters was performed by simply adding `Number of Hidden Layers` to the best models obtained in section 4.6, and calculating the $MedALE$ and $MaxMedALE$ on the testing set.

4.7 Temporal Neural Network

The temporal neural network (TNN) was trained using the feature-engineered data and the state vectors described in section 4.1.6. The same workflow as described in section 4.6 was employed for this training process. However, the no accretion set was used without incorporating the temporal state vectors, as it was concluded that adding these vectors would not provide any supplementary information. Instead, for the no accretion predictions, the `Number of Hidden Layers` was increased in the Bayesian optimization, restricting to values within the range of $[10, 20]$. Consequently, due to the disparity between the accretion and no accretion sets in this experiment, the mix set was not employed in addition to the already not used complete set. Therefore the only two remaining sets to use were the balanced and the reduced set.

4.8 Models Analysis

In addition to obtaining a trustworthy model to predict the stellar structure, it results interesting to analyze the ML models, in order to understand what is the importance of the input parameters, and state vectors.

4.8.1 Confidence Intervals

To establish a reliable model and obtain confidence intervals, a method that takes a $\frac{1}{1000}$ random sample of the testing set with replacement 2000 times was employed. Subsequently, predictions are made for each sample, and the $MALE$ of the predicted sample is computed.

More detailed confidence intervals separated by star age and mass class using the Morgan–Keenan (MK) system, were done. In order to perform this analysis, the same procedure as before was used, but splitting the simulations first by mass class, and then by age. The age spread of the sample is not uniform between simulations, therefore in order to obtain the mean a normalization by age was done as follows:

$$\bar{X} = \frac{\sum_{i=1}^{n-1} x_i \Delta t_i}{\sum_{i=1}^{n-1} \Delta t_i} \tag{4.8}$$

where $\bar{x}$ is the mean of the sample, Δt is the age difference of each point defined by $\Delta t = \frac{t_{n+1} - t_{n-1}}{2}$.The first and last data points were excluded from the calculation, causing negligible impact. To obtain $\bar{X}$, 500 random samples were acquired from each data split with a sample size of 0.01. Each sample's mean $\bar{X}$ was then calculated.

4.8.2 HR Diagrams

In addition to the earlier error analysis, the predicted values of T_{eff} and L_s were plotted on HR diagrams alongside their corresponding simulation counterparts. This comparative analysis was performed for four stars, each belonging to one of the star classes AB, GF, K, and M. This was a necessary part of the analysis as obtaining the HR diagrams is arguably the most important aspect of predicting the stellar structure. In addition to this, the previous metrics to quantify the errors could be deceiving, and the true behavior of the predictions can be observed in these diagrams.

4.8.3 Hyperparametrs Importance

Using Bayesian optimization with `Optuna`, it became possible to analyze the importance of hyperparameters. This analysis is valuable for understanding the mechanism of the ML model, enabling a focus on the parameters that the neural network relies on more heavily to make predictions. As a result, the hyperparameter importance of each network was determined and visualized through plots.

Chapter 5

Results

The objective of this thesis is to develop a machine learning model that can predict temperature, star luminosity, and accretion luminosity based on the star's mass, age, and accretion rate. In order to show the model's results, several analyses were done. This includes the outcomes obtained through data manipulation, the results from various machine learning models, including confidence intervals, and the model analysis using SHAP values and hyperparameter importance. All results were obtained by following the method outlined in section 4.

5.1 Exploratory Data Analysis

The stellar structure parameters utilized as inputs for the machine learning model and obtained from RAMSES include mass, accretion rate, and star age. Figure5.1 illustrates their corresponding distributions.

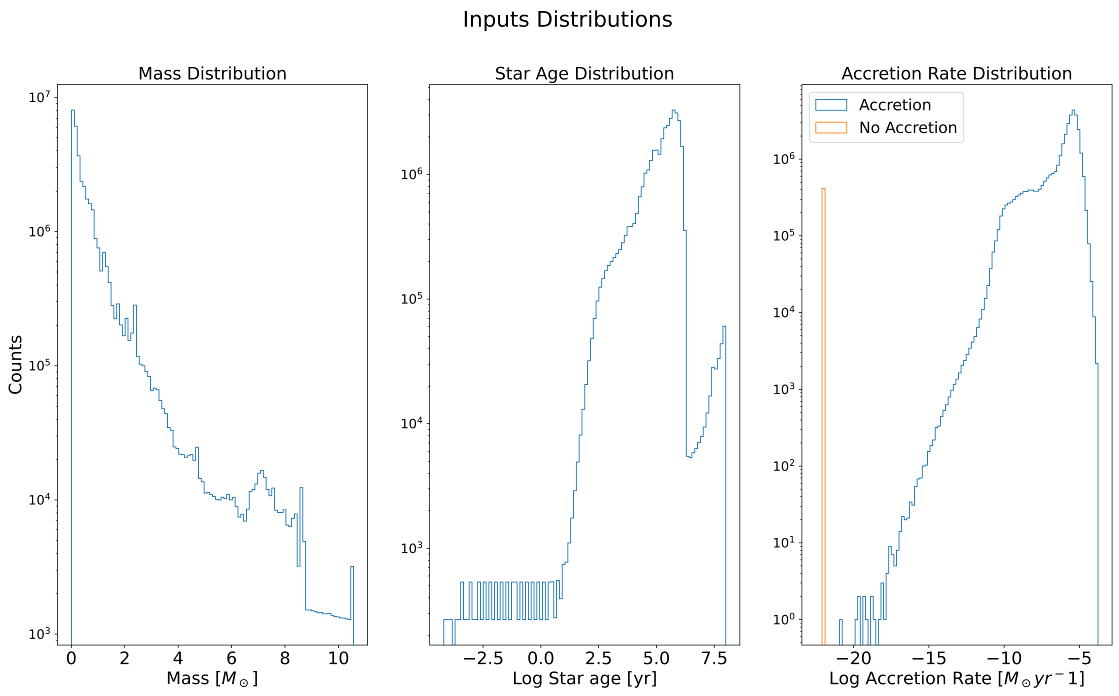

Figure 5.1: Distribution of mass, age, and accretion rate obtained from RAMSES.

As shown in figure 5.1, the majority of data points correspond to stars with masses ranging from 0 to $2M_\odot$. Additionally, the distribution of star ages indicates a concentration in the range of $10^{2.5}$

to 10^6 years, followed by a significant decline. Moreover, the plot reveals a prevalent high accretion rate, reaching its peak at $10^{-5} M_{\odot} yr^{-1}$. In addition to this, there is an uneven distribution of data, with a disproportionate amount of instances having accretion compared to those without it. This particularity of the data set can be seen clearly in figure 5.2. In order to deal with this problem the data was split into accretion and non-accretion sets. For each of these sets a ML model was trained.

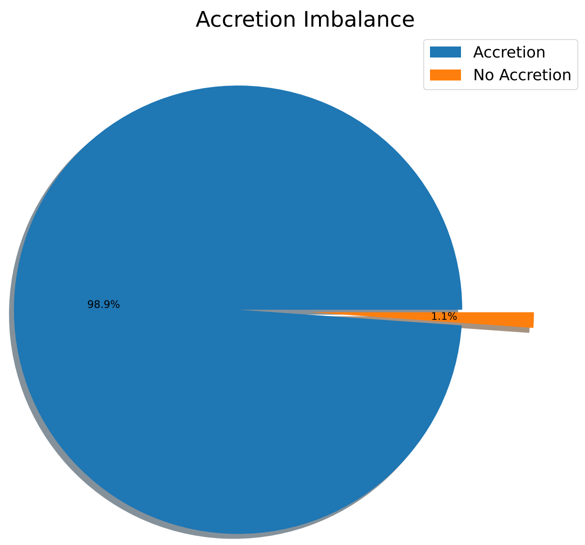

Figure 5.2: Proportion of data with accretion rate vs no accretion rate. The blue section, which contains 99% of the data, represents star evolution points with an accretion rate greater than $M_{\odot} yr^{-1} > 0$, while the orange section represents points with an accretion rate of $M_{\odot} yr^{-1} = 0$

In order to familiarize ourselves with the data set it is important to understand the distributions shown in figure 5.3, where it is possible to see interesting distinct features of each parameter.

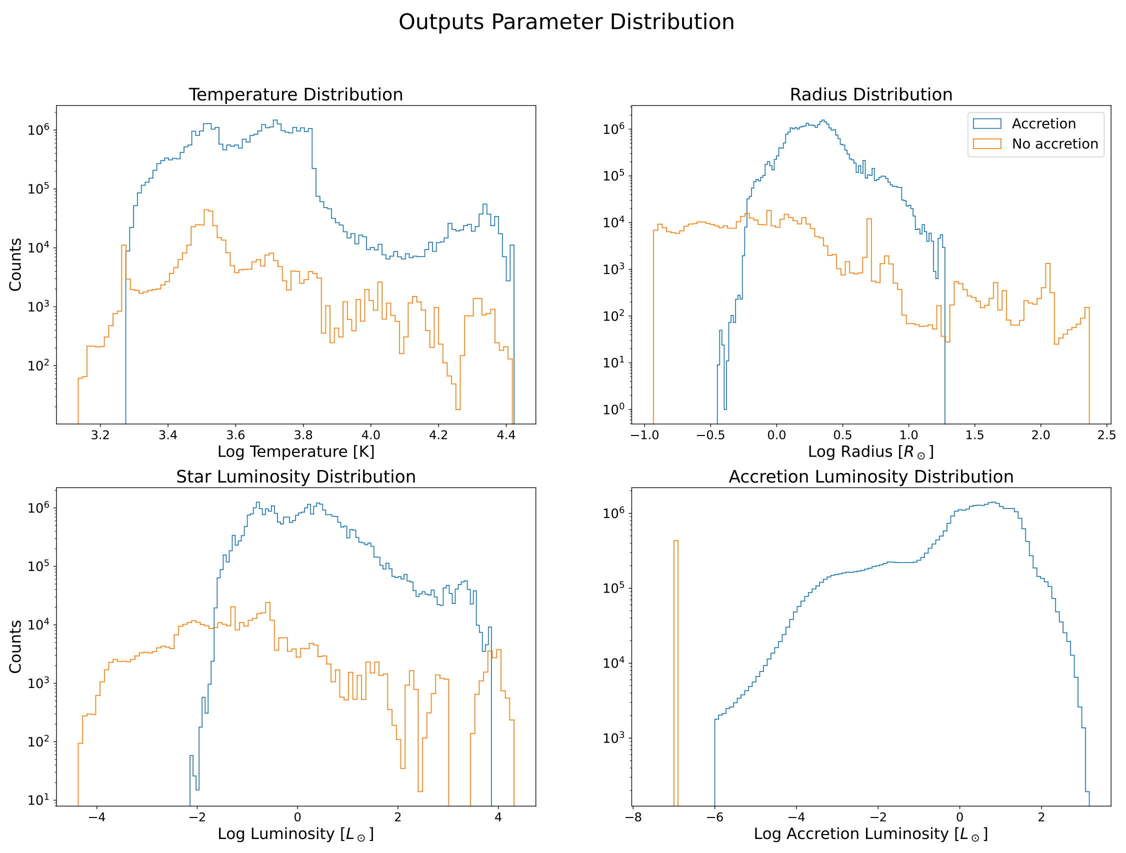

Figure 5.3: The distributions of temperature, star luminosity, radius, and accretion luminosity are represented in the plot. The blue curve shows the distribution of accretion data, while the orange curve represents the distribution of data with no accretion. For visualization purposes, the no accretion set, which has an accretion luminosity $L_{acc} = 0L_\odot$, is positioned at $L_{acc} = 10^{-7}L_\odot$ in the bottom-right panel of the accretion luminosity distribution.

In the case of the accretion set in the four cases, there is a clear imbalance in the lower and upper limits of the distributions. An important feature to take into account is the minimum accretion luminosity with $L_{acc} = 10^{-5.99}L\odot$, this value is important in order to accurately calculate the errors in the L_{acc} as mentioned in section 4.2. In the case of the no-accretion data set, it is interesting to observe that the range of values for T_{eff}, L_s, and R is larger compared to the accretion data sets, with R showing the most significant difference, being approximately twice as large as the accretion data set. Furthermore, the no-accretion data set exhibits a better balanced and uniform distribution when compared to the accretion data set. Due to this characteristic and the smaller size of the no-accretion set, no additional data manipulation was necessary for this subset.

5.1.1 Spliting the Data Set

In an attempt to split the training and testing set as evenly as possible the data set was divided into 4 categories as seen in figure 5.4. From each of these subsets, a proportion of the simulations was randomly chosen as training and testing sets.

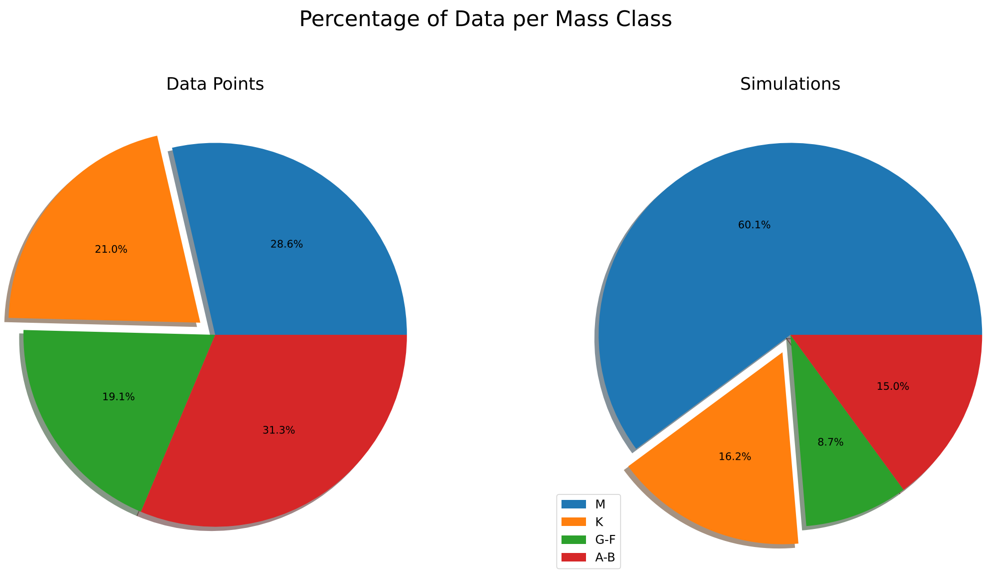

Figure 5.4: Percentage of data per star categories. The left-hand side shows the percentage of data points per class, while the right-hand side represents the percentage of simulations in the respective class.

After the splitting a comparison of the full training and testing data set can be seen in figure 5.5.

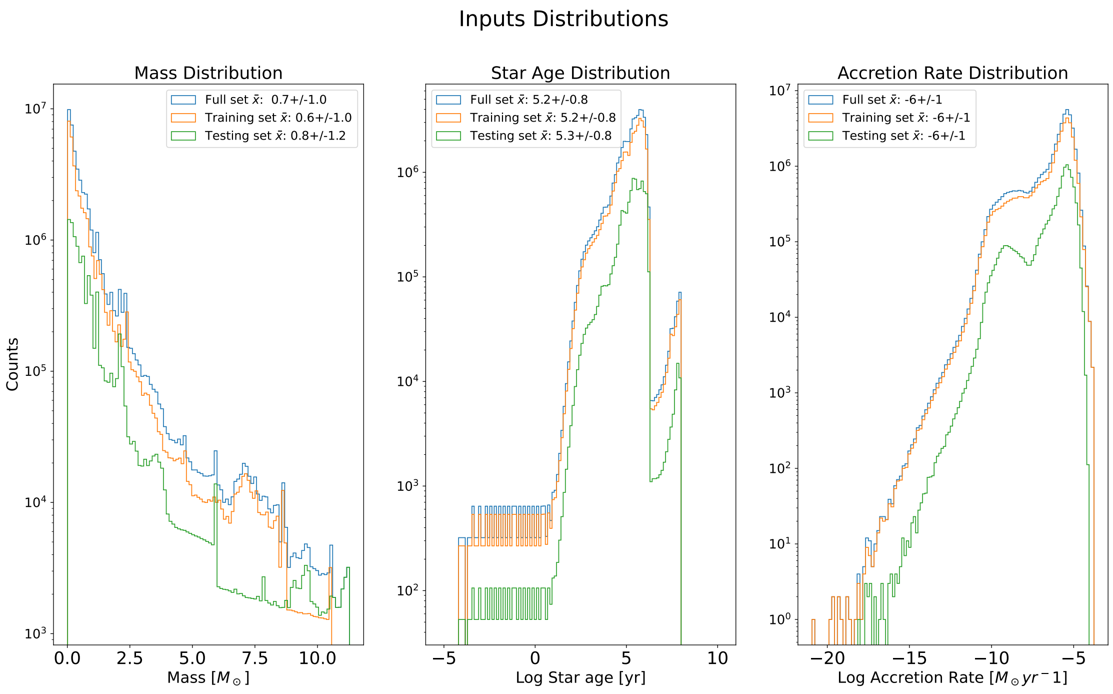

Figure 5.5: The full, training, and testing distributions are depicted in blue, orange, and green, respectively. In the left panel, the plot shows the mass distribution, the center panel represents the star age distribution, and the right-hand side shows the accretion rate distribution. For each of these plots, the mean and standard deviation were calculated to compare the splittings.

Comparing the data sets in figure 5.5 is possible to see that the mass in the training set does not represent the full range of masses, this is because, there is only one simulation in the highest mass

range with $M = 11.3M\odot$, and by the randomness of the data splitting, this simulation is in the testing set. In the case of star age, it is noticeable that the training and testing sets are more comparable between them, with the testing set containing values within the same range as the training set. Finally, in the case of the accretion rate test set, it can be observed that there are a few missing values in both the upper and lower ends of the range. However, the number of these missing points is so minimal that their impact is negligible.
The means ($\bar{x}$) of all the distributions were calculated to provide a quantitative basis for comparing them with the respective values displayed in figure 5.5. By using $\bar{x}$, we can observe a strong agreement between the different splits.

5.1.2 Balancing and Reducing Training set

Figure 5.3 illustrates that the accretion data set displays a noticeable underrepresentation of certain output values, particularly at the extremes. Therefore, balancing the data set seems to be the appropriate approach, as explained in Section 4.1.3. Additionally, a second modified dataset for the accretion set was created by eliminating data points where no significant change in the stellar structure was observed, as detailed in Section 4.1.4. From this point onward in the thesis, we will refer to these data sets as the balanced and reduced datasets, respectively.
The resulting training data set sizes are shown in table 5.1.

Accretion Data Sets Size by Output Parameter.				
Data Type	T_{eff}	L_s	R	L_{acc}
Complete	34.1×10^6	34.1×10^6	34.1×10^6	34.1×10^6
Reduced	5.8×10^6	7.6×10^6	3×10^6	6.2×10^6
Balanced	1.4×10^6	1.8×10^6	2.5×10^6	1.9×10^6

Table 5.1: Resulting accretion data sizes before and after applying reduction and balance methods.

As mentioned earlier, the no accretion set did not require any further manipulation, hence, the training no accretion set remains unchanged, containing approximately 0.41×10^6 data points.

5.1.3 Parameter Importance SHAP Analysis

Using the models obtained from LightGBM, a SHAP analysis was performed following the method outlined in section 4.1.7. The results of this analysis are presented in figure 5.6.

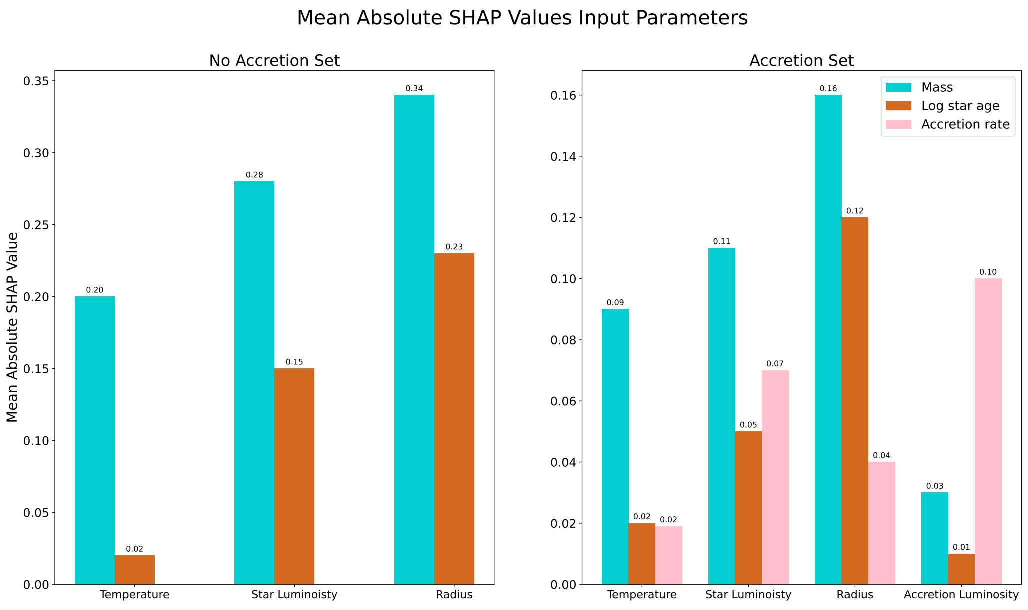

Figure 5.6: The SHAP values analysis was conducted for the predictions of temperature, star luminosity, radius, and accretion rate. In the left-hand side panel, the importance of mass is indicated in blue, and log star age is represented in brown for the no accretion set. In the right-hand side panel displays the accretion set parameter importance with Mass in blue, star age in brown, and accretion rate in pink.

The magnitude of a Mean Absolute SHAP Value represents the importance of the feature influence on the model's outputs. It is crucial to understand that SHAP values are computed independently for each feature, leading to variations in scales and magnitudes. Additionally, the total sum of mean absolute SHAP values across all features does not hold any predetermined relationship to a specific value or percentage [59].
Equations 2.8, 2.9, and 2.10 establish the relationships between luminosity, radius, and temperature during the radiative transfer, convection, and energy production processes in a star. These relationships highlight the significance of mass as a crucial factor influencing changes in these parameters. Therefore, as seen in figure 5.6 it is not surprising that once again, in both the accretion and no accretion sets, the SHAP analysis identifies mass as the most important input parameter for T_{eff}, L_s, and R.
In the case of the L_{acc} case, the accretion rate was the most important feature, again the explanation of this can be seen in the relation on equation 2.12. This analysis confirms the known physics facts about the stellar structure and provides confidence in the validity of the data sets.

5.1.4 Benchmark HR diagrams

Among the test simulations, one of each star class was chosen as a benchmark to compare with the ML predictions. These stars have the following final masses:

Final Mass Benchmark Stars.	
Star class	Mass ($M\odot$)
AB	5.9
GF	1.6
K	0.6
M	0.4

Table 5.2: Star classes final mass in solar masses $M\odot$.

Each of these stars exhibits interesting characteristics that will aid us in identifying consistently challenging areas for the ML models to predict. Therefore an HR diagram of these stars was obtained and shown in figure 5.7.

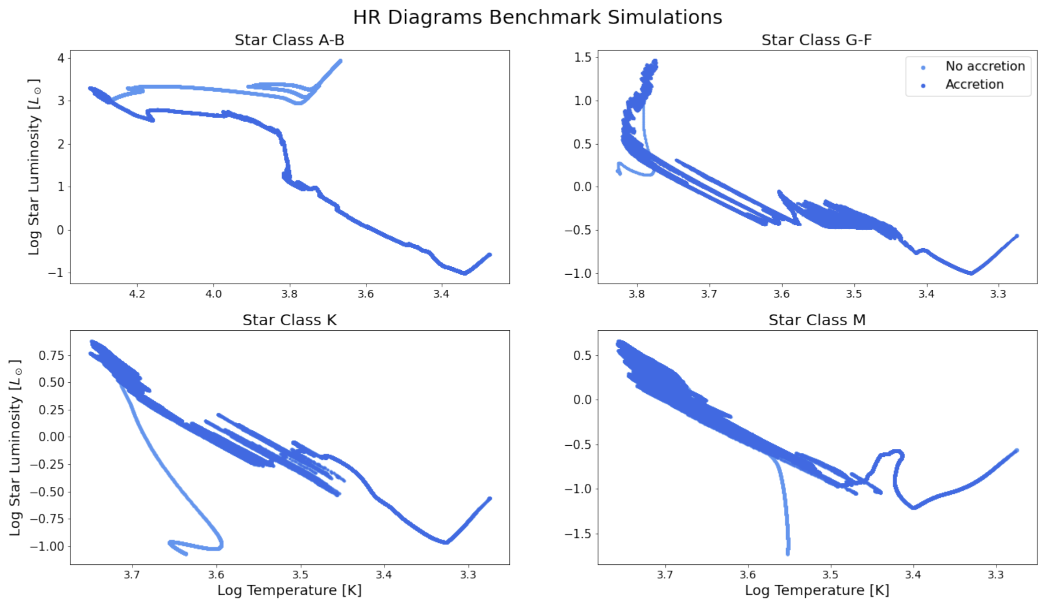

Figure 5.7: The top left-hand side panel of the HR diagram displays an AB star as one of the four benchmark stars. The top right-hand side panel shows a GF star, while the bottom left-hand side panel depicts a K star, and finally, the bottom right-hand side panel shows an M star. In each panel, the light blue points correspond to stellar evolution without accretion, whereas the darker blue points represent stellar evolution with accretion.

The stars of class GF, K, and M seen in figure 5.7 were selected because of their significant variability in accretion, as evident from the spread in their luminosity. Predicting these star types is particularly interesting, as it aligns with the main objective of this thesis.
In the case of the AB star, the accretion rate is more uniform during its evolution. However, the no accretion section indicates a long evolution after the star has entered the main sequence, moving past the hydrogen fusion phase. The uniqueness of the AB star's evolution makes it intriguing since there is no other star with a similar profile in the training set. Consequently, it becomes interesting to examine whether the ML models can generalize beyond the boundaries of the training set.

5.2 LightGBM

The machine learning results presented in this section were obtained using LightGBM, as described in sections 3.4 and 4.5. This model serves as an initial step to explore the capabilities of ML models, as it is a powerful yet straightforward method to implement.

5.2.1 Models Selection LightGBM

To select the optimal model for each parameter, we utilized the best models obtained from hyper-parameter tuning, as seen in appendix B.1.
The best models for the different data types were chosen among all the tests as seen in appendix A.1 using the previously defined $MedALE$ and $MedMaxALE$ metrics. The results for the best models in the accretion section and the no accretion section are presented in tables 5.3 and 5.4, respectively.

Best Parameters NN Error Analysis Accretion Set				
Errors	T_{eff}	L_s	R	L_{acc}
$MedALE$	0.0098	0.0976	0.0279	0.0326
$MedMaxALE$	0.085	0.587	0.159	0.300

Table 5.3: $MedALE$ and $MedMaxALE$ for TNN all prediction parameters.

In the accretion section, for T_{eff}, L_s, and L_{acc}, the best models were obtained using the balanced set. Specifically, the mix set was the best for T_{eff} and L_{acc}. As for L_s, the best model was the one from the accretion set. The decision to select this particular model was based on the fact that the combination of $MedALE$ and $MedMaxALE$ falls at an intermediate point among the rest accuracy results. Finally, the R, best model was obtained using the complete accretion set.

Best Parameters NN Error Analysis No Accretion Set.			
Errors	T_{eff}	L_s	R
$MedALE$	0.0035	0.0256	0.0141
$MedMaxALE$	0.046	0.330	0.152

Table 5.4: $MedALE$ and $MedMaxALE$ for TNN all prediction parameters.

In the case of the no accretion section the R best model was obtained using the reduced mix set, whereas the T_{eff}, and L_s the no accretion sets.

5.2.2 Errors Distributions per Input Parameter LightGBM

After selecting the best models, the initial step involves examining the error distribution concerning the input parameters. To achieve this, a 2D histogram was created, as illustrated in the figures.5.8, 5.9, 5.10, and 5.11.

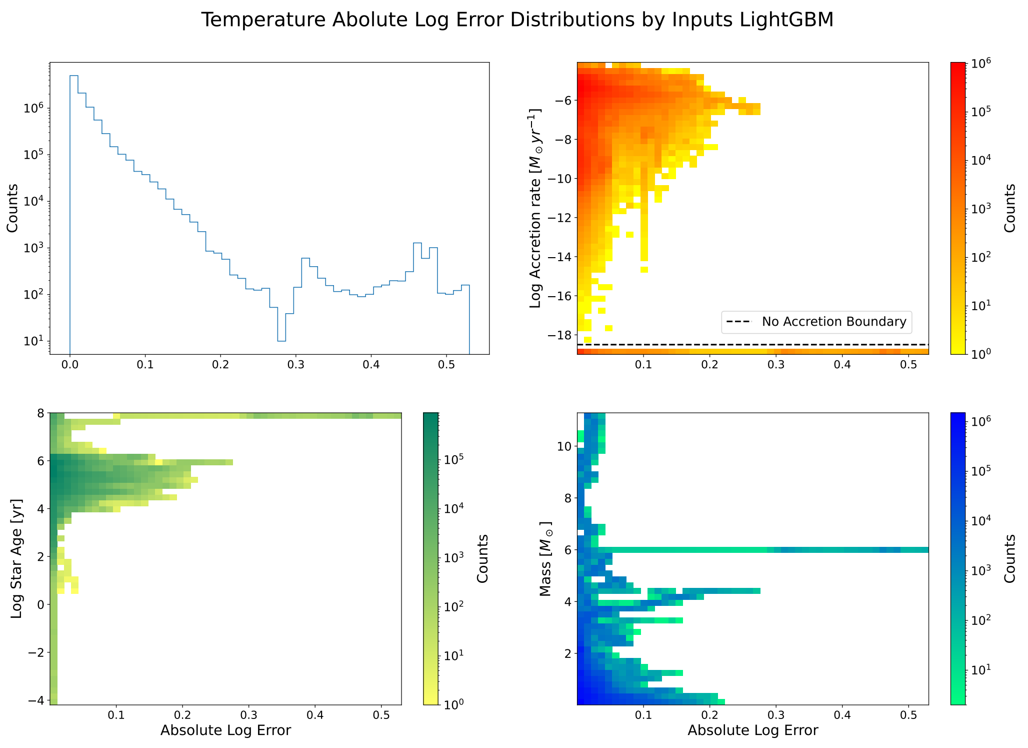

Figure 5.8: Temperature absolute error distribution comparison between input parameters obtained using the LightGBM. The top left-hand side panel illustrates the distribution of ALE. In the top right-hand side panel, the ALE distribution per accretion rate is displayed, with the section below the dashed line representing the region where $M_\odot yr^{-1} = 0$. The bottom left-hand side panel showcases the distribution of ALE per star age, and lastly, the bottom right-hand side panel demonstrates the distribution of ALE per mass.

Regarding temperature, figure 5.8 reveals that the worst errors in the accretion section occur in areas with high accretion rates ranging between $[10^{-10}, 10^{-5}]M_\odot \mathrm{yr}^{-1}$. However, the overall decrease in accuracy is observed in the no accretion section. In the accretion section, the errors tend to concentrate between ages of $[10^3, 10^6]$ years, with the largest errors occurring at 10^8 years, which corresponds to the no accretion set. Concerning errors by mass, it is interesting to note three distinct areas with errors around $[0, 1]M_\odot$, $4M_\odot$, and the most substantial errors around $6M_\odot$.

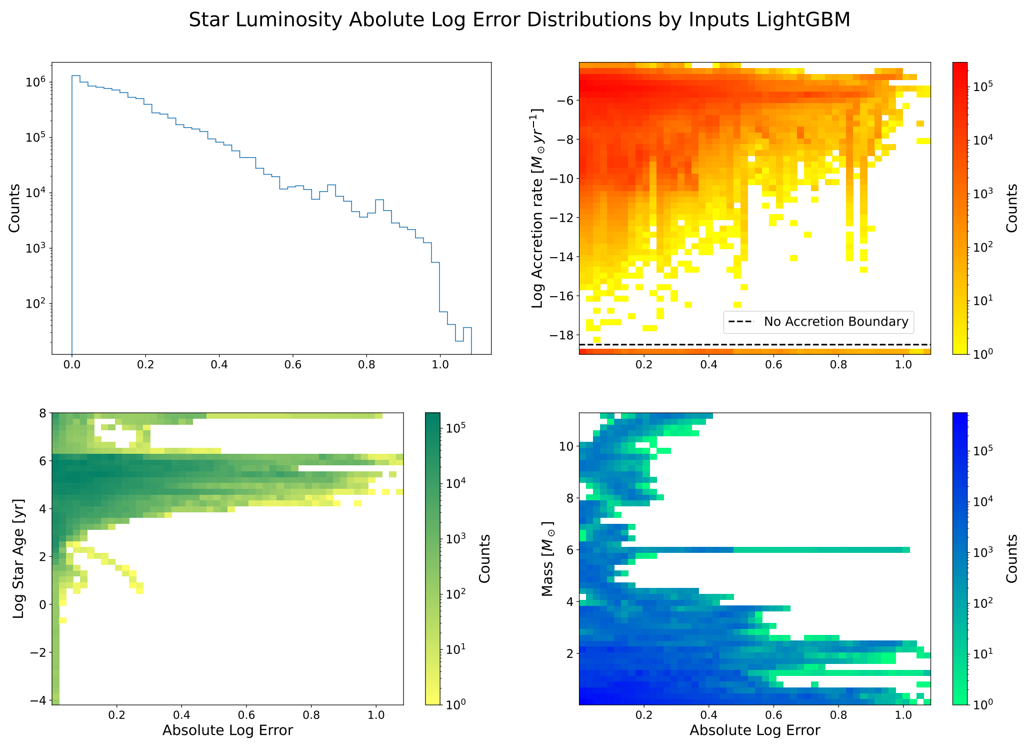

Figure 5.9: Star luminosity absolute error distribution comparison between input parameters obtained using the LightGBM. The top left-hand side panel illustrates the distribution of ALE. In the top right-hand side panel, the ALE distribution per accretion rate is displayed, with the section below the dashed line representing the region where $M_\odot yr^{-1} = 0$. The bottom left-hand side panel showcases the distribution of ALE per star age, and lastly, the bottom right-hand side panel demonstrates the distribution of ALE per mass.

For luminosity, figure 5.9 shows a more significant spread of errors. The worst errors concerning accretion rate exhibit a distributed pattern, covering all values for both the accretion and no accretion sections. However, there is a high concentration of errors around $10^{-6} M_\odot \text{yr}^{-1}$.
Regarding errors by age, they tend to concentrate between $[10^4, 10^6]$ years, with these errors being larger than those in the no accretion section at 10^8 years.
Concerning errors by mass, the most significant errors are observed in the low mass range of $[0, 4]M_\odot$. Additionally, there is a notable concentration of errors around $6M_\odot$.

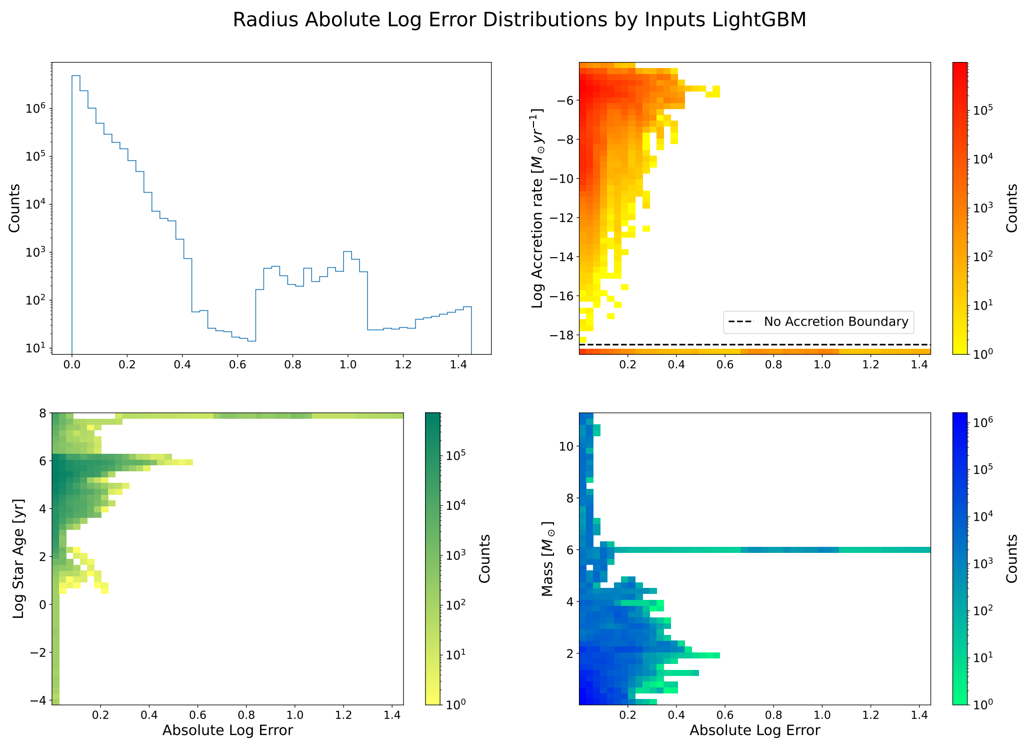

Figure 5.10: Radius absolute error distribution comparison between input parameters obtained using the LightGBM. The top left-hand side panel illustrates the distribution of ALE. In the top right-hand side panel, the ALE distribution per accretion rate is displayed, with the section below the dashed line representing the region where $M_\odot yr^{-1} = 0$. The bottom left-hand side panel showcases the distribution of ALE per star age, and lastly, the bottom right-hand side panel demonstrates the distribution of ALE per mass.

In the case of the radius, figure 5.10 shows a concentration of errors occurring in the area with high accretion rates around $10^{-6} M_\odot \text{yr}^{-1}$. However, the model less accurate predictions are overwhelmingly present in the area with no accretion, spanning more than twice the number of errors observed in the accretion section.
The concentration of errors by age begins to increase between $[10^3, 10^6]$ years, with the most significant errors appearing at the higher end, around 10^8 years, corresponding to the no accretion section.
Similar to previous observations, the less accurate predictions are focused on the $10^6 M_\odot$ range, with an additional area of errors spanning $[0, 4] M_\odot$.

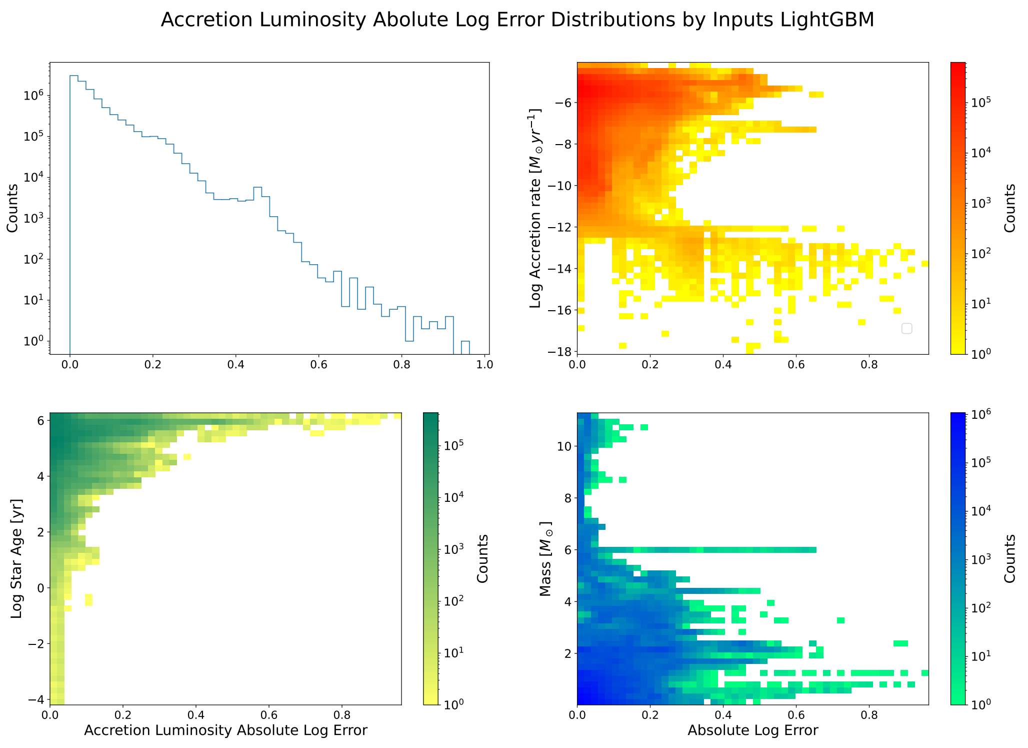

Figure 5.11: Accretion luminosity absolute error distribution comparison between input parameters obtained using the LightGBM. The top left-hand side panel illustrates the distribution of ALE. In the top right-hand side panel, the ALE distribution per accretion rate is displayed, with the section below the dashed line representing the region where $M_\odot yr^{-1} = 0$. The bottom left-hand side panel showcases the distribution of ALE per star age, and lastly, the bottom right-hand side panel demonstrates the distribution of ALE per mass.

Finally, for the accretion luminosity in Figure 5.11, there is a specific concentration of significant inaccuracies in the regions with low accretion rates, particularly around $[10^{-18}, 10^{-12}]M_\odot\text{yr}^{-1}$. However, in the area with high accretion rates, errors exhibit an increase with two distinct peaks around $10^{-8}M_\odot\text{yr}^{-1}$ and $10^{-6}M_\odot\text{yr}^{-1}$.
Moreover, it is noticeable that the maximum errors occur when the star reaches the main sequence at 10^6 years. Interestingly, the biggest inaccuracies in mass occur at different intervals, with areas spanning $[0, 5]M_\odot$, and a distinct peak at $6M_\odot$.

In general, the distribution of the errors is expected as the majority happens in sections with high accretion rates, which are the areas with less predictability. However the no accretion section presents the worst errors overall, this result was not expected, as stars with no accretion corresponding to main sequence stars, have well-known behavior due to the mass-luminosity relation [13]. A particularity present in all the parameters is the distribution around $10^6M_\odot$. At first glance, this could indicate that the model has problems making predictions in one or a few simulations from the testing set, due to a lack of data in that range. In the next sections, a more in-depth analysis of this behavior will be done.

5.2.3 LightGBM Confidence Intervals

Following the method described in section 4.8.1, the confidence intervals of each result were obtained. The results of this analysis is shown in figure 5.12.

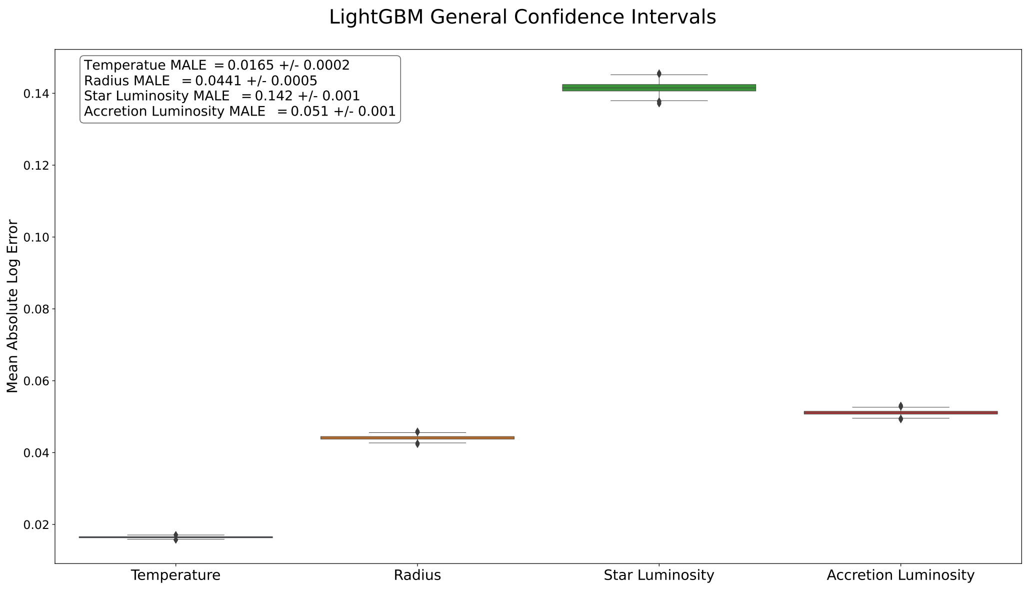

Figure 5.12: Confidence intervals for temperature, star radius luminosity, and accretion luminosity, were obtained by utilizing the absolute log error in the LightGBM predictions on the test set.

As seen in figure 5.12, the results align with the patterns observed in tables 5.3 and 5.4. The best performing prediction parameter remains T_{eff}, while L_s exhibits the poorest performance.

Furthermore, the confidence intervals are categorized based on mass and age. The results of these analyses are displayed in figures 5.13, 5.14, 5.15, and 5.16.

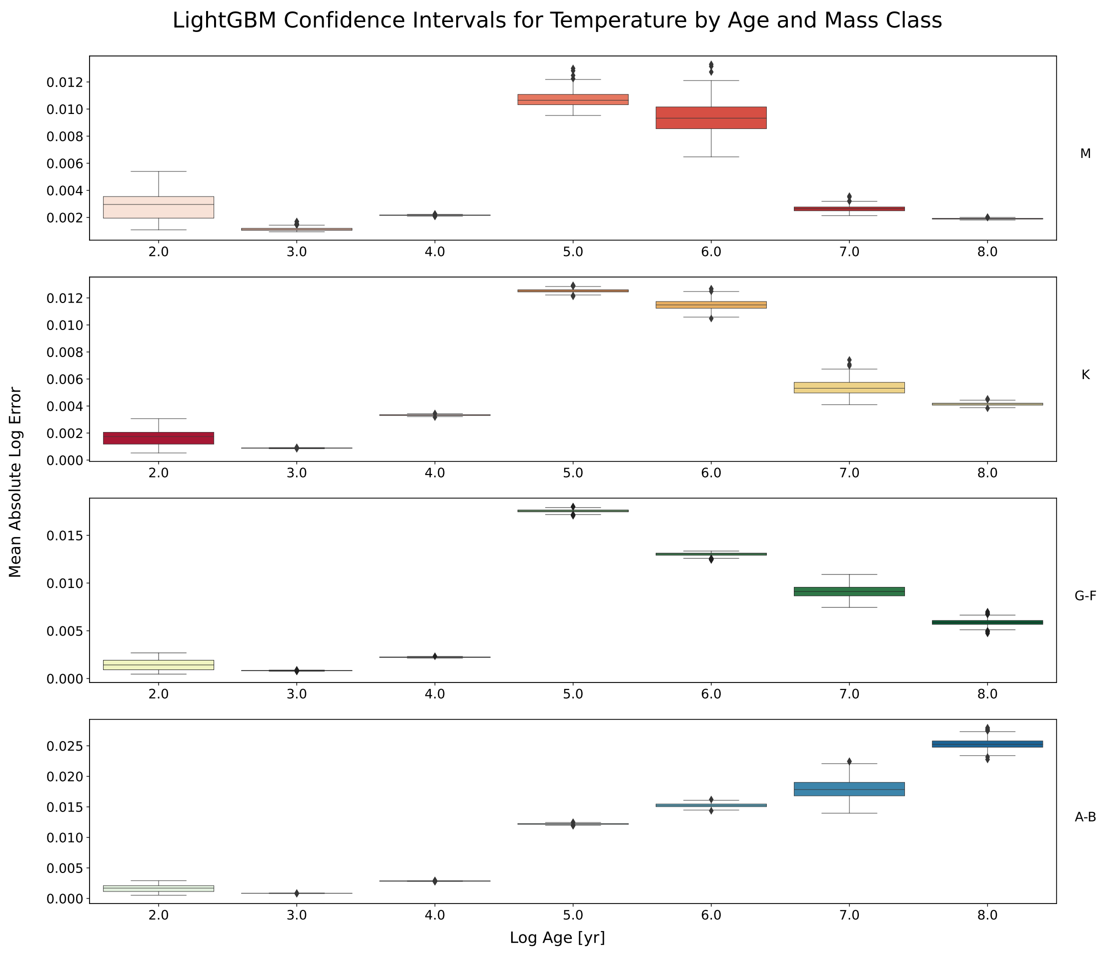

Figure 5.13: The confidence intervals of temperature, grouped by star class and age brackets, are shown in the figure. These intervals were derived using the absolute log error in the LightGBM predictions on the test set. The y-axis shows the age brackets, with the last number representing the upper limit of each bracket. For instance, the first bracket represented by 2 corresponds to stars with ages between 10^0 and 10^2 years.

The findings from figure 5.13 reveal that, concerning temperature, the most significant errors occur in the GF, K, and M stars within the age range of $[10^5, 10^6]yr$. This aligns with the accretion set, particularly towards the end of this phase. However, the most substantial errors are observed in stars of class AB with significant errors spanning from $[10^6, 10^8]yr$. These results corroborate the findings obtained in section 5.2.2.

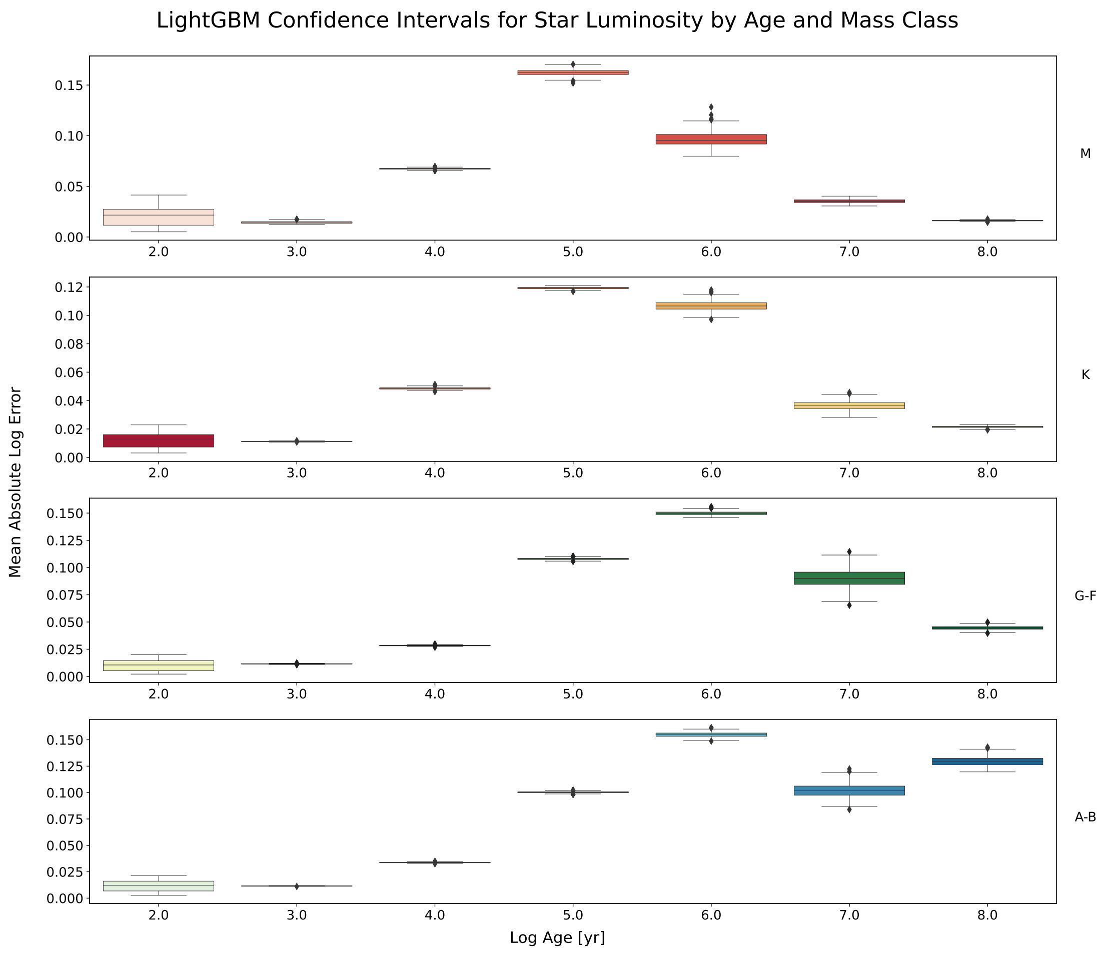

Figure 5.14: The confidence intervals of star luminosity, grouped by star class and age brackets, are shown in the figure. These intervals were derived using the absolute log error in the LightGBM predictions on the test set. The y-axis shows the age brackets, with the last number representing the upper limit of each bracket. For instance, the first bracket represented by 2 corresponds to stars with ages between 10^0 and 10^2 years.

Figure 5.14 shows the star luminosity, where it is evident that the predicted $MALE$ begins to increase at a star age of 10^4 years. Similarly, for T_{eff}, the peak errors are concentrated in the age range $[10^5, 10^6]$ years. Once again, in the no accretion section, the age range of $[10^7, 10^8]$ years displays poor accuracy, primarily observed in the star class AB.

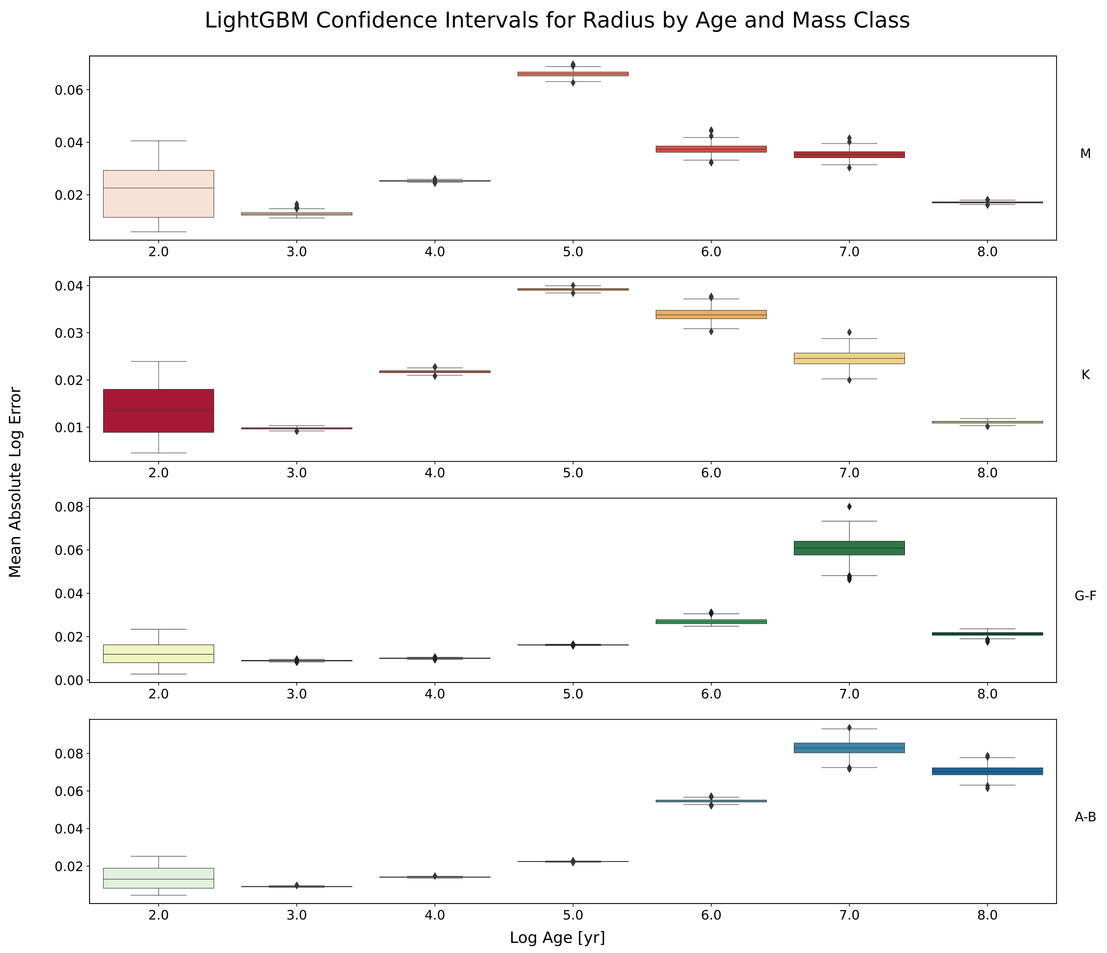

Figure 5.15: The confidence intervals of radius, grouped by star class and age brackets, are shown in the figure. These intervals were derived using the absolute log error in the LightGBM predictions on the test set. The y-axis shows the age brackets, with the last number representing the upper limit of each bracket. For instance, the first bracket represented by 2 corresponds to stars with ages between 10^0 and 10^2 years.

Regarding the radius confidence intervals, as shown in figure 5.15, the $MALE$ of stars M and K exhibit the least accuracy in the age range of $[10^5, 10^6]yr$. Additionally, they display a greater variance at a young stage of $[10^2]yr$. For stars GF and AB, the largest errors occur at an age of 10^7yr, once again corresponding to the no accretion section, with these errors representing the most significant discrepancies.

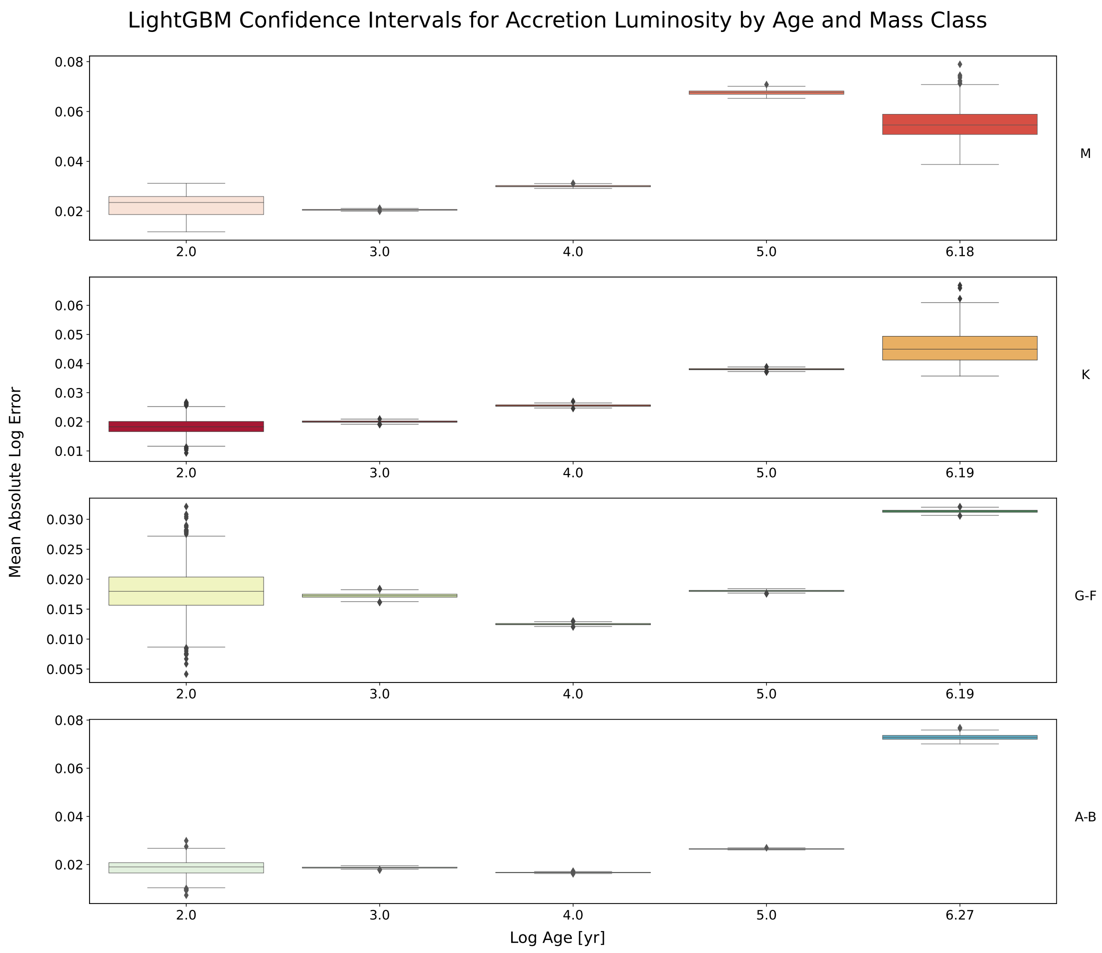

Figure 5.16: The confidence intervals of accretion luminosity, grouped by star class and age brackets, are shown in the figure. These intervals were derived using the absolute log error in the LightGBM predictions on the test set. The y-axis shows the age brackets, with the last number representing the upper limit of each bracket. For instance, the first bracket represented by 2 corresponds to stars with ages between 10^0 and 10^2 years.

Finally, in the case of accretion luminosity, as shown in figure 5.16, it is evident that the least accurate predictions are observed in stars AB and M, with the worst errors occurring at an age of 10^6 years. Similarly, for stars GF and K, the most substantial errors for these classes are also present at an age of 10^6 years. Additionally, the stars GF display a wide variance at lower ages.

An intriguing aspect of this analysis is the consistency in errors at the end of the accretion set. Moreover, the worst errors in stars AB are consistent across all parameters in the no accretion section.

5.2.4 HR Diagram Analysis LightGBM

As mentioned in the preceding section, the most significant errors occur within the star age range of $[10^5, 10^7]yr$. Additionally, figures 5.9, 5.10, and 5.8 suggest the possibility of some problematic simulations. Therefore, an analysis of the individual simulations was conducted using the benchmark HR diagrams, displayed in figures 5.17, 5.18, 5.19, and 5.20.

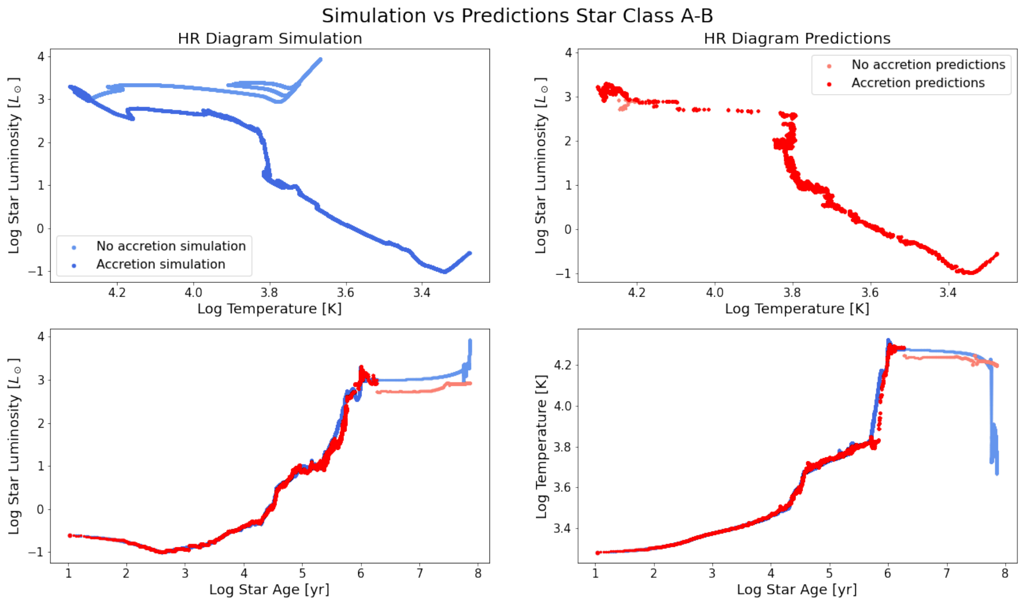

Figure 5.17: Test simulation AB star, with final mass $M = 5.9M_\odot$. The top panels display HR diagrams, with the left-hand side representing the diagram obtained from MESA data and the right-hand side showing the one obtained through LightGBM predictions. The bottom left-hand side presents the luminosity as a function of age, showing both the MESA values and the predictions for comparison. Similarly, the bottom right-hand side displays the temperature evolution of the star, with both MESA and predicted values overlaid for comparison.

As seen in figure 5.17, the HR diagrams reveal that the model faces challenges in making predictions when the star reaches the main sequence. The lower right plot highlights the reason behind this error, as the ML model fails to predict the drop in temperature once the star reaches the main sequence. For L_s, the predictions in this range are relatively better, but they still struggle to accurately model all the features present in the simulated data. In the accretion section, the ML model manages to reproduce the overall shape of the simulated HR diagram. However, the resolution of the predictions is poor, as it fails to accurately reproduce sections of the curve, which is evident in the temperature range $[10^{3.8}, 10^{4.1}]K$. This is probably caused by a lack of training data but it is easily addressed because the post-main sequence evolution is exactly the part that is accessible with classical stellar evolution models.

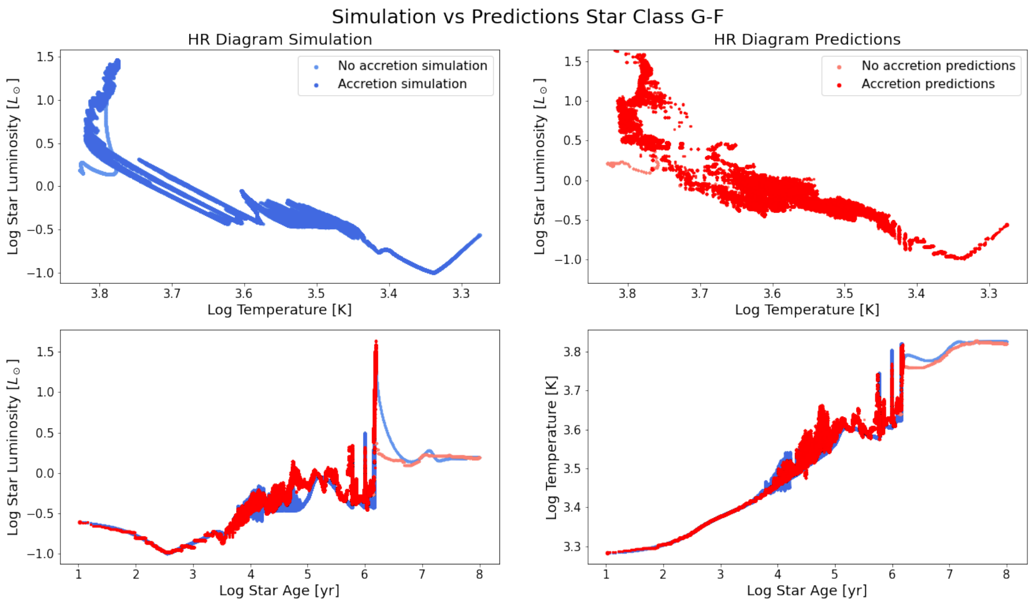

Figure 5.18: Test simulation GF star, with final mass $M = 1.6M_\odot$. The top panels display HR diagrams, with the left-hand side representing the diagram obtained from MESA data and the right-hand side showing the one obtained through LightGBM predictions. The bottom left-hand side presents the luminosity as a function of age, showing both the MESA values and the predictions for comparison. Similarly, the bottom right-hand side displays the temperature evolution of the star, with both MESA and predicted values overlaid for comparison.

For the GF star, illustrated in figure 5.20, we observe a similar behavior where the model encounters challenges in accurate predictions once the star reaches the main sequence. However, the error in this case occurs in the luminosity prediction, as evident in the lower left panel. Although the ML model manages to provide a general overview of the HR diagram, the resolution remains poor.

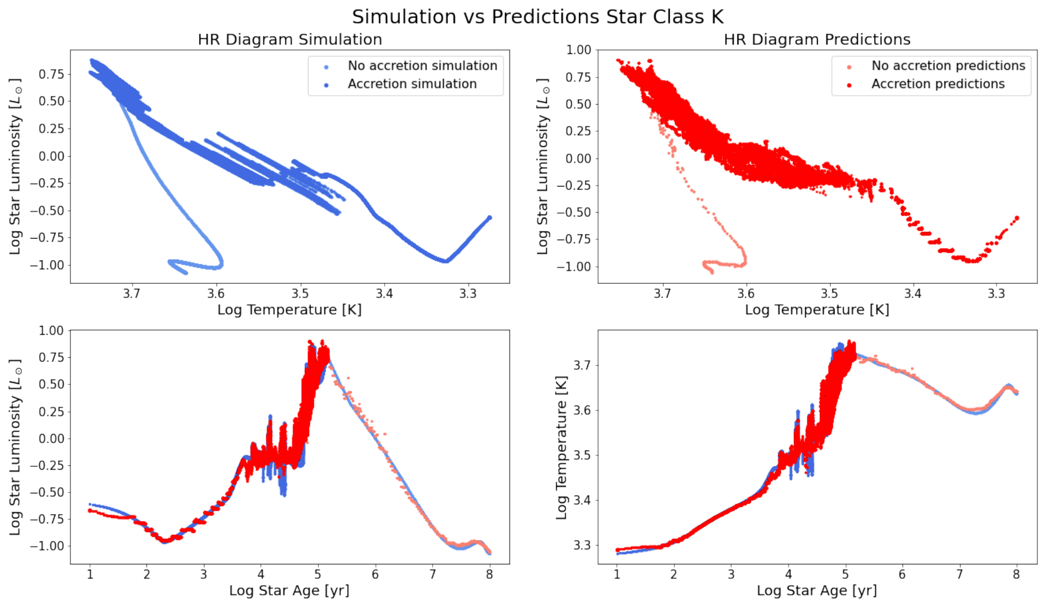

Figure 5.19: Test simulation K star, with final mass $M = 0.6M_{\odot}$.The top panels display HR diagrams, with the left-hand side representing the diagram obtained from MESA data and the right-hand side showing the one obtained through LightGBM predictions. The bottom left-hand side presents the luminosity as a function of age, showing both the MESA values and the predictions for comparison. Similarly, the bottom right-hand side displays the temperature evolution of the star, with both MESA and predicted values overlaid for comparison.

Figure 5.19 displays the HR diagram for the K star. It is clear that the ML model can roughly predict all features of the curve, yet the resolution remains poor. Specifically, in the accretion section, there is a good agreement with the overall shape of the T_{eff} and L_s curves. However, in the age range of $[10^4, 10^5]yr$, fluctuations in both T_{eff} and L_s occur, which the ML model was not able to predict.

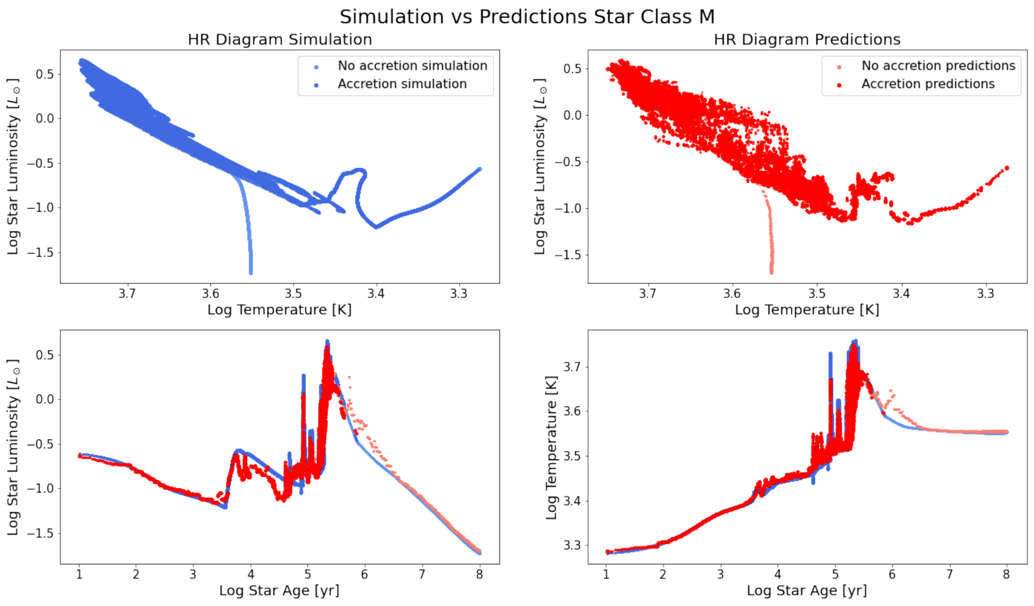

Figure 5.20: Test simulation of a star with final mass $M = 0.358M_\odot$. The top panels display HR diagrams, with the left-hand side representing the diagram obtained from MESA data and the right-hand side showing the one obtained through LightGBM predictions. The bottom left-hand side presents the luminosity as a function of age, showing both the MESA values and the predictions for comparison. Similarly, the bottom right-hand side displays the temperature evolution of the star, with both MESA and predicted values overlaid for comparison.

Finally, for star class M in figure 5.20, the predictions are similar to the previous cases. We observe that the predictions align with the overall curves of T_{eff} and L_s. In this case, the predictions manage to follow, to some extent, the increase in T_{eff} and L_s observed at $[10^{4.7}, 10^5]$ years.

Despite failing to capture some of the most interesting features in certain cases, overall, the ML model results manage to predict the general shape of the curves. However, an important observation is the lack of resolution present in the results of all four tests.

5.2.5 Hyper Parameters Importance LightGBM

As discussed in section 4.5.1, the hyperparameters for the LightGBM model include `subsample`, `num_leaves`, `learning rate`, and `max depth`. To determine the significance of these hyperparameters in making predictions, a Bayesian optimization method with `Optuna` was employed. The results of this analysis are presented in figure 5.21.

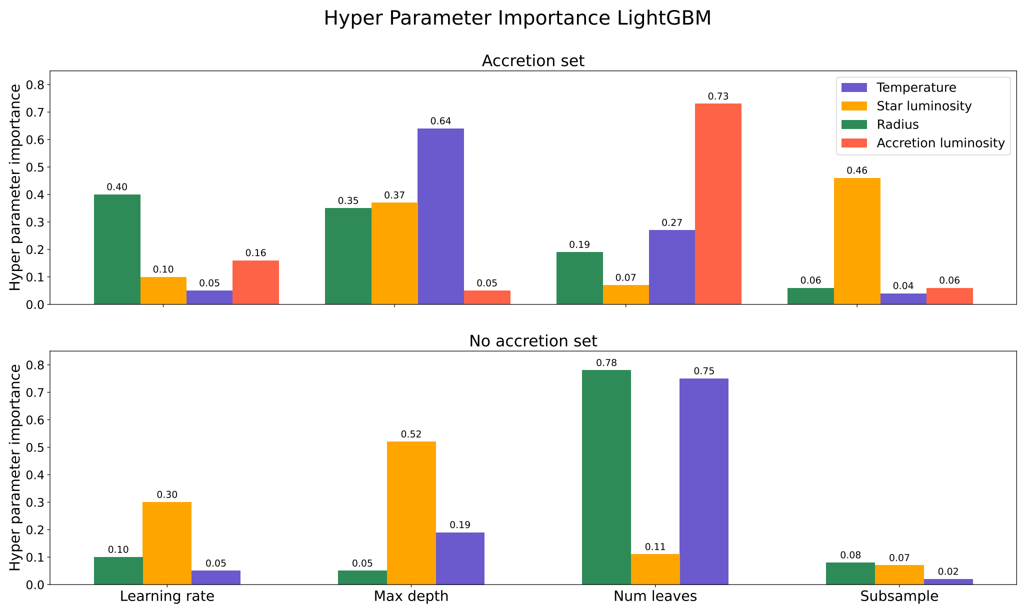

Figure 5.21: The importance of hyperparameters for temperature, star luminosity, radius, and accretion luminosity LightGBM models. The top panel displays the parameter importance of accretion models, while the bottom panel represents the parameter importance for no accretion models.

Figure 5.21 shows that for most parameters, the most crucial hyperparameter varied. However, the `num_leaves` consistently emerged as the most important for T_{eff} and R in the no accretion set, and for L_{acc} in the accretion set. Additionally, the analysis shows that `subsample` has little importance for most parameters, except for L_s in the accretion set, where it gains significance.

5.3 Neural Network

The following section presents the results from an implementation of a neural network, following the methodology described in section 4.6. The obtained results were subsequently analyzed using the same procedure as in section 5.2.

5.3.1 Models Selection Neural Network

One crucial finding from the previous analysis is that, while using the complete data set yields the best results in some instances it is not the case for most scenarios. Moreover, even in cases where the complete data set shows slight superiority, the difference is not substantial enough to justify the necessity of using the complete data set. This aspect is particularly significant for the hyper parameter tunning (HPT) and training of the NN. Given that the complete data set can be up to 24 times larger in certain cases, the training and HPT overhead becomes significantly high for the minimal or negligible benefit it provides. Therefore, for the NN, only the reduced and balanced sets were employed.
The results obtained from the optimization with `Optuna` are presented in the appendix B.2. Additionally, all models were evaluated using our error metrics $MedALE$ and $MedMaxALE$ on the test set, as shown in appendix A.2. The best results of this analysis are displayed in tables 5.5, and 5.6

Best Parameters NN Error Analysis Accretion Set.				
Errors	T_{eff}	L_s	R	L_{acc}
$MedALE$	0.0069	0.0736	0.0237	0.0289
$MedMaxALE$	0.077	0.555	0.160	0.199

Table 5.5: $MedALE$ and $MedMaxALE$ for NN accretion set, for all prediction parameters.

In all the cases in the accretion set the best NN models come from the balanced set, with the exception of L_{acc} where the reduced mix set yields the best accuracy. Interestingly, the results of this experiment exhibit a similar pattern to the ones conducted in section 5.2. Specifically, the model achieving the best performance is the one designed for T_{eff} predictions, followed by the model for R predictions, then the model for L_{acc} predictions. Lastly, the model for L_s predictions appears to be the least successful performer.

Best Parameters NN Error Analysis No Accretion Set.			
Errors	T_{eff}	L_s	R
$MedALE$	0.0033	0.01073	0.0028
$MedMaxALE$	0.021	0.171	0.082

Table 5.6: $MedALE$ and $MedMaxALE$ for NN no accretion set, for all prediction parameters.

Within the group of models not involving accretion, no successful model was achieved when employing the mixed set. The most accurate outcomes were consistently derived from the utilization of the no-accretion set. An important fact in this result is that the $MedALE$ of R is actually smaller than for T_{eff}. However, if we look at the $MedMaxALE$, the pattern of the results resembles the one obtained in section 5.2.1.

5.3.2 Errors Distributions per Input Parameter Neural Network

Using the best NN models obtained by the analysis done in section 5.3.1, the analysis of the error distribution per input parameter was done. Results of this analysis can be seen in figures 5.22,5.23, 5.24, and 5.25.

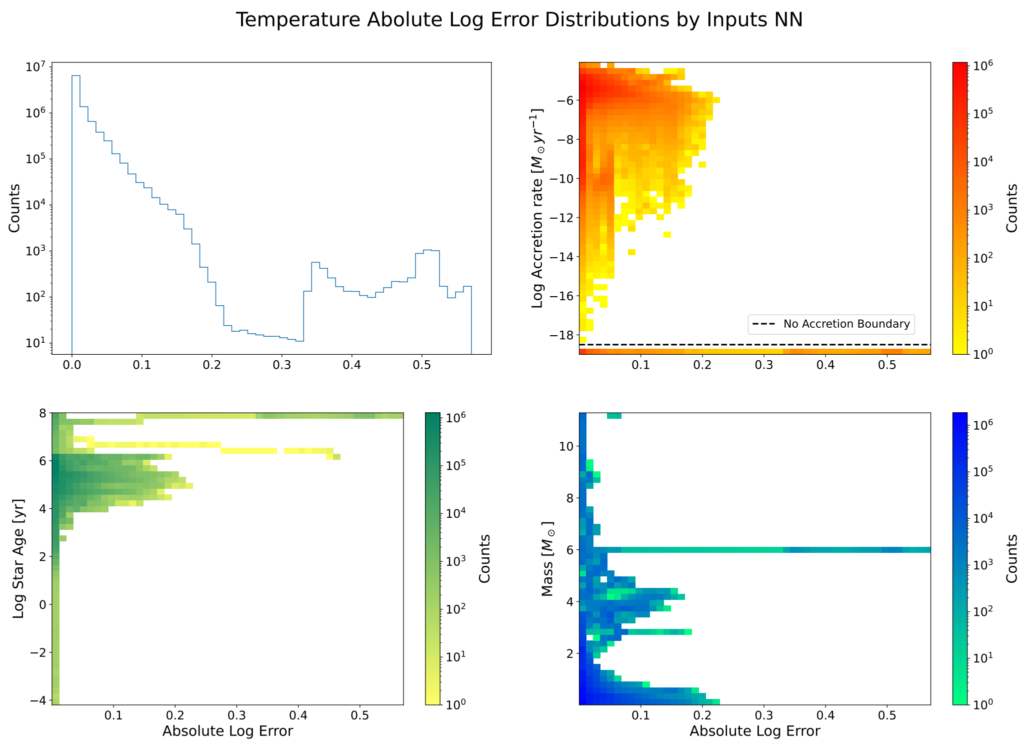

Figure 5.22: Temperature absolute error distribution comparison between input parameters obtained using the NN model. The top left-hand side panel illustrates the distribution of ALE. In the top right-hand side panel, the ALE distribution per accretion rate is displayed, with the section below the dashed line representing the region where $M_\odot yr^{-1} = 0$. The bottom left-hand side panel showcases the distribution of ALE per star age, and lastly, the bottom right-hand side panel demonstrates the distribution of ALE per mass.

In the temperature ALE distribution, seen in figure 5.22, it is evident that the poorest accuracy occurs in the non-accretion set, particularly in stars with a mass of $6M_\odot$ and towards the end of the simulation, within the range of $[10^6, 10^8]yr$. This section of the distribution corresponds to the no accretion set. In the accretion section, the area with the highest density of errors concentrates within the age range of $[10^4, 10^6]yr$ and in areas with a high accretion rate, approximately within $[10^{-8}, 10^{-6}]M_\odot yr^{-1}$.
Regarding the ALE distribution per mass, shown in the lower right-hand side plot of figure 5.22, there are two distinct peaks that correspond to the accretion set. The peak with the poorest accuracy is associated with low masses, around $[0, 1]M_\odot$, while the second peak, with relatively better accuracy, corresponds to masses around $[3, 5]M_\odot$.

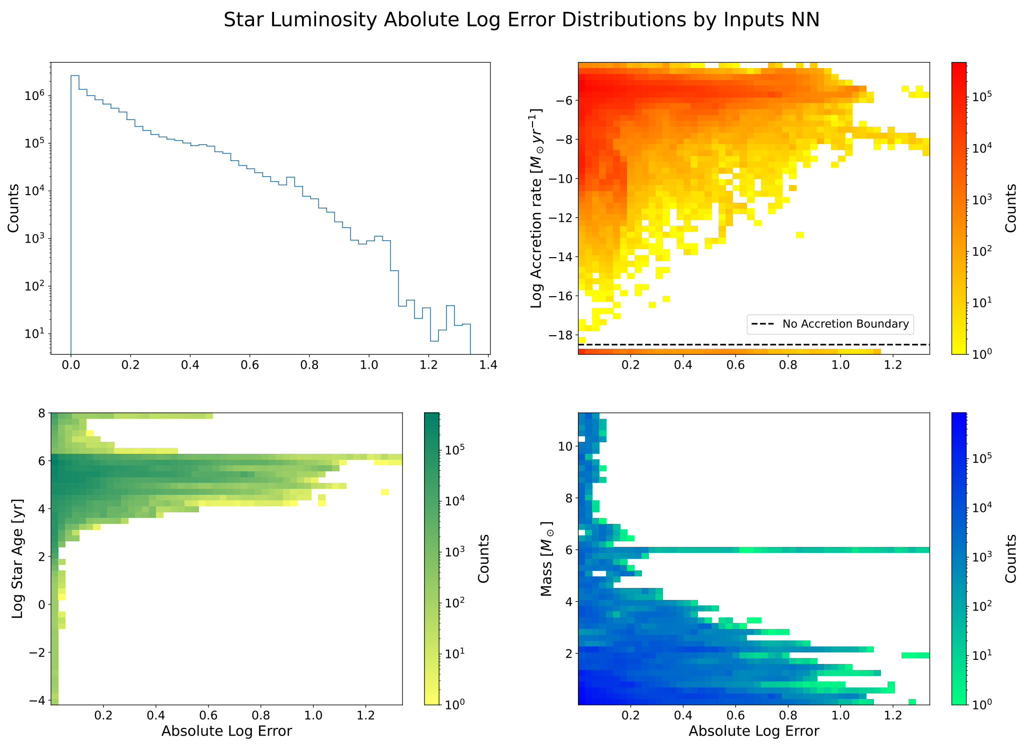

Figure 5.23: Star luminosity absolute error distribution comparison between input parameters obtained using the NN model. The top left-hand side panel illustrates the distribution of ALE. In the top right-hand side panel, the ALE distribution per accretion rate is displayed, with the section below the dashed line representing the region where $M_\odot yr^{-1} = 0$. The bottom left-hand side panel showcases the distribution of ALE per star age, and lastly, the bottom right-hand side panel demonstrates the distribution of ALE per mass.

Analyzing the distribution of star luminosity errors in figure 5.23, it becomes evident that there is a high concentration of predictions with poor accuracy, particularly at a high accretion rate, with a distinct peak around $10^{-6} M_\odot yr^{-1}$. Additionally, the age range with the worst errors is towards the end of the accretion set, exhibiting a distinct high concentration of errors within the area of $[10^4, 10^6] yr$. Interestingly, most of the errors with high values are observed in stars with low mass. However, the familiar peak of errors at $6M_\odot$ can still be observed.
In the case of the no accretion section, the L_s predictions exhibit a minor concentration of errors beyond $ALE > 0.6$.

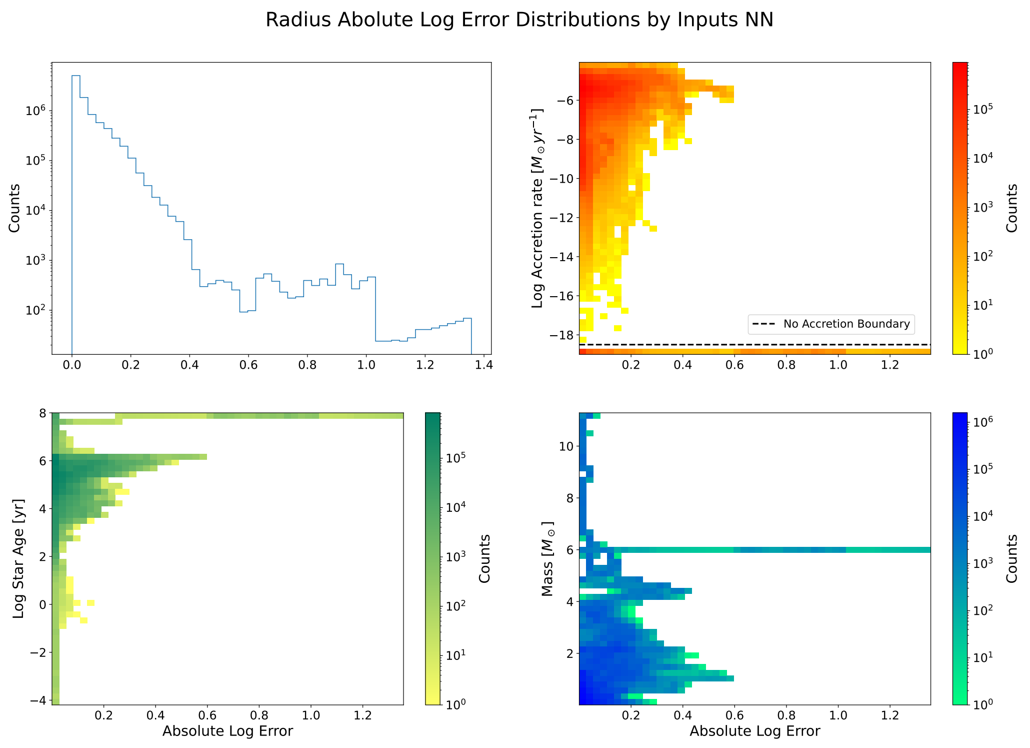

Figure 5.24: Radius absolute error distribution comparison between input parameters obtained using the NN model. The top left-hand side panel illustrates the distribution of ALE. In the top right-hand side panel, the ALE distribution per accretion rate is displayed, with the section below the dashed line representing the region where $M_\odot yr^{-1} = 0$. The bottom left-hand side panel showcases the distribution of ALE per star age, and lastly, the bottom right-hand side panel demonstrates the distribution of ALE per mass.

The Radius ALE distribution exhibits a similar pattern to the temperature, with the poorest accuracy occurring in the no accretion set, particularly for stars with a mass of $6M_\odot$ and towards the end of the simulation around $10^8 yr$. Moreover, significant errors are observed in stars with low mass, falling within the range of $[1, 2]M_\odot$. We can see that these errors occur towards the end of the accretion section at ages between 10^5 and 10^6 years. These error peaks correspond with areas of high accretion rate, in particular, the peak at $10^{-6} M_\odot yr^{-1}$.

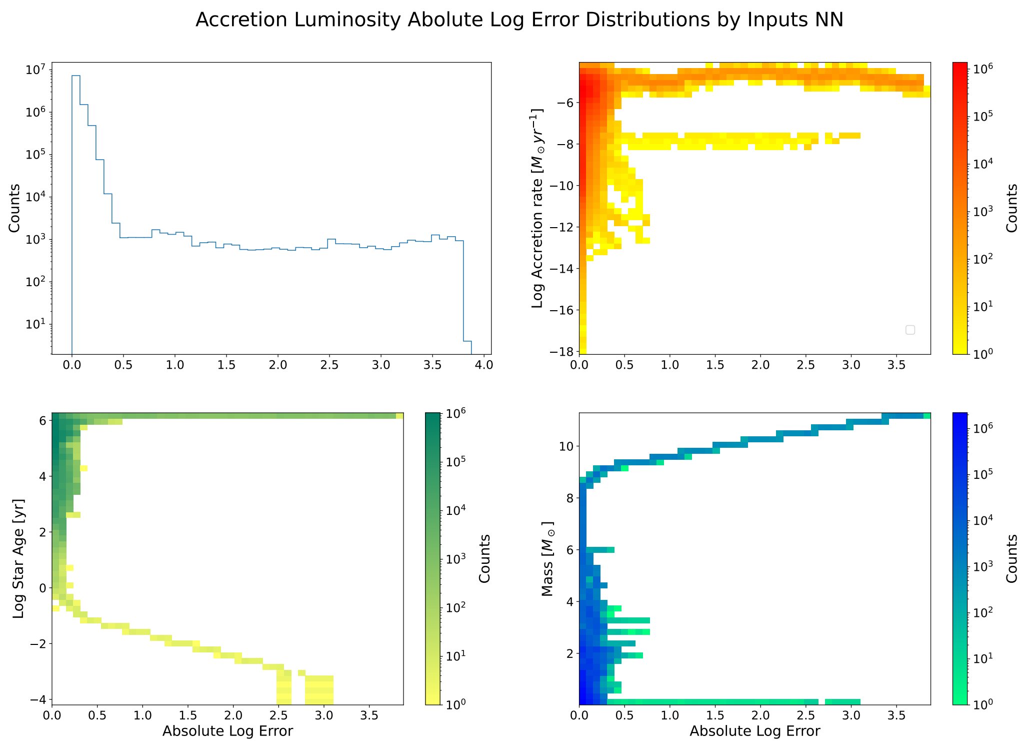

Figure 5.25: Accretion luminosity absolute error distribution comparison between input parameters obtained using the NN model. The top left-hand side panel illustrates the distribution of ALE. In the top right-hand side panel, the ALE distribution per accretion rate is displayed, with the section below the dashed line representing the region where $M_{\odot}yr^{-1} = 0$. The bottom left-hand side panel showcases the distribution of ALE per star age, and lastly, the bottom right-hand side panel demonstrates the distribution of ALE per mass.

Finally, the accretion luminosity seen in figure 5.25 presents distributions of poor accuracy that are highly unpredictable. It's clear from the age per ALE plot on the lower left side that the poorest accuracy is observed at the end of the star's accretion phase, although a minor concentration of poor accuracy is noticeable at the very start. Regarding the mass per ALE, the worst ALE is found in stars exceeding $8M_{\odot}$, along with a minor concentration of high errors in low mass stars, corresponding to the values observed in the lower ages. Furthermore, the accretion rate exhibits two distinct peaks, one at $10^{-8}M_{\odot}yr^{-1}$ with shows evidence of poor accuracy, and another at $10^{-5.9}M_{\odot}yr^{-1}$. The uniqueness of these findings could suggest that there are a few simulations where the NN model has difficulty making precise predictions.
An in-depth analysis was done confirming that in fact there were six test simulations out of the 53 presenting these errors, the maximum errors of these precise simulations can be seen in appendix C.

As an overview of these results, we saw the worst errors happening in the no-accretion phase of the star, with the particular peak in a star with mass $6M_{\odot}$ present in T_{eff}, R, and L_s. In addition to this, at first glance, the L_{acc} errors distribution shows signs of concern. However, upon closer examination and a detailed analysis of the predictions, it becomes clear that overall, the NN model performance was satisfactory.

5.3.3 Confidence Intervals Neural Network

An alternative approach for analyzing the results obtained by the NN is to calculate confidence intervals, similar to the ones obtained in section 5.2.3. The results of this analysis are shown in figure 5.26.

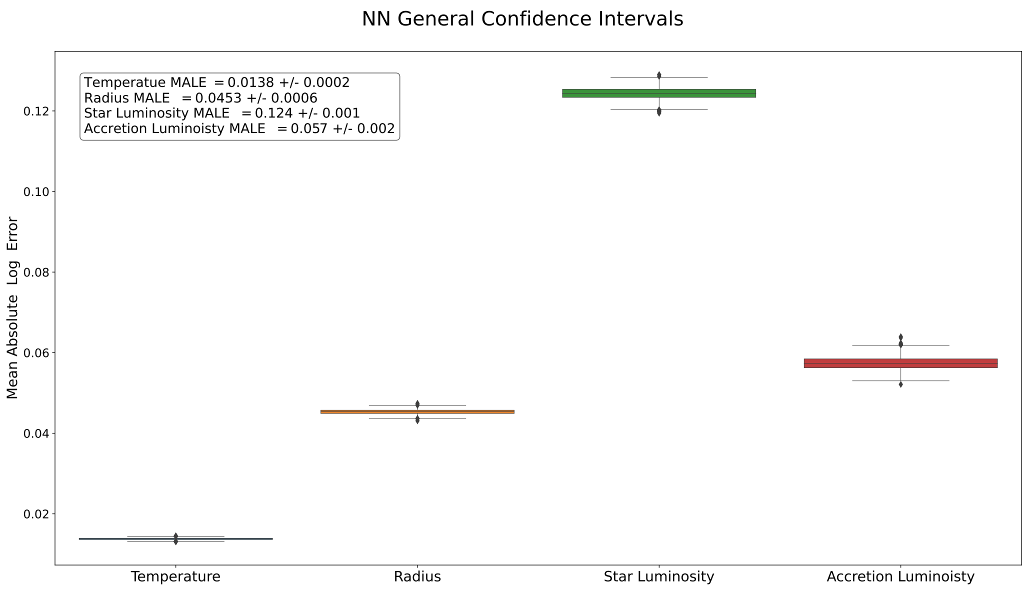

Figure 5.26: Confidence intervals for temperature, star radius luminosity, and accretion luminosity, were obtained by utilizing the absolute log error in the NN predictions on the test set.

The results seen in figure 5.26 follow a pattern that aligns with the findings obtained in figure 5.12. A way to obtain more detailed information about the NN model's performance is to perform the confidence intervals by age, and star class described in section 4.8.1. The results of these analyses are shown in figures 5.27 5.28, and 5.29.

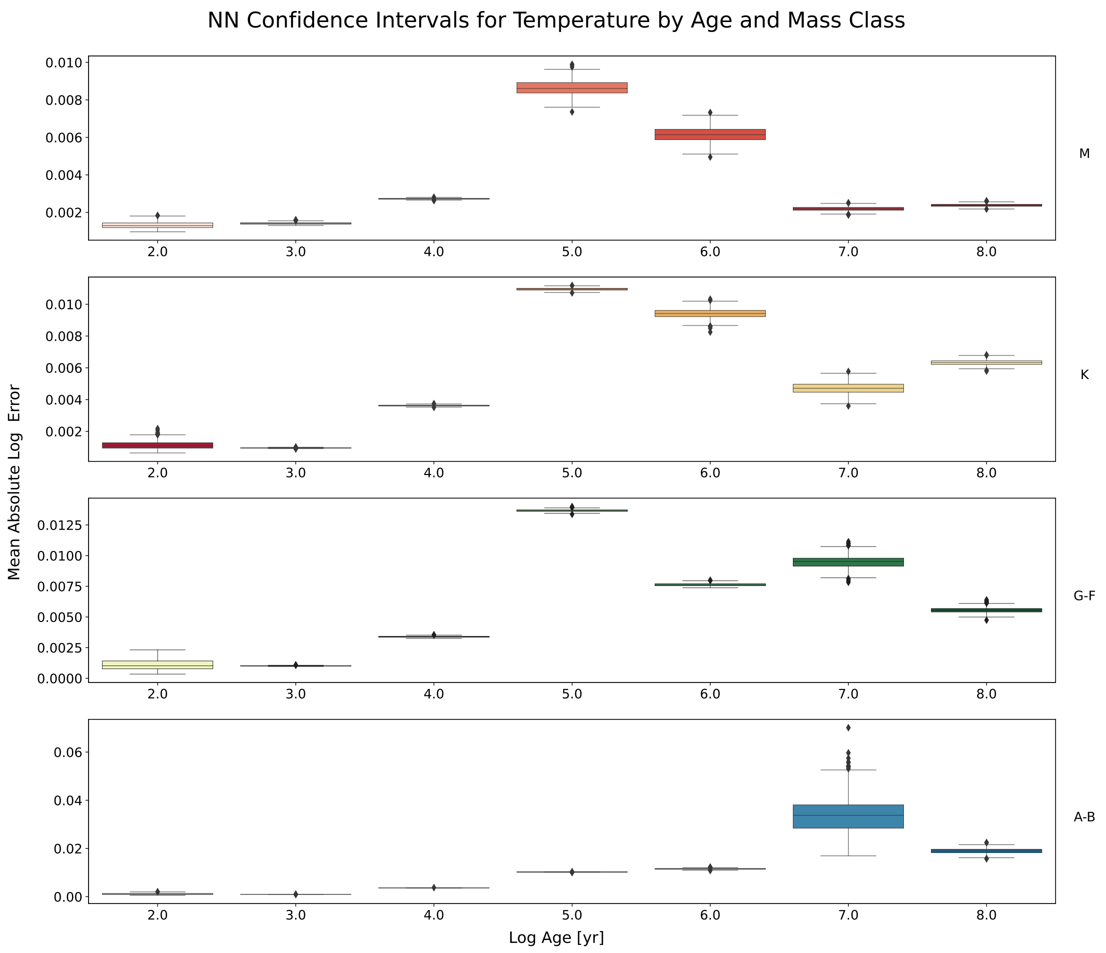

Figure 5.27: The confidence intervals of temperature, grouped by star class and age brackets, are shown in the figure. These intervals were derived using the absolute log error in the NN predictions on the test set. The y-axis shows the age brackets, with the last number representing the upper limit of each bracket. For instance, the first bracket represented by 2 corresponds to stars with ages between 10^0 and 10^2 years.

In figure 5.27, it is evident that the NN predictions for T_{eff} are most inaccurate for AB stars, particularly in the no accretion set, with errors reaching up to 0.06 at ages around $10^7 yr$. Similarly, for the other classes, the weakest performance occurs at the end of the accretion phase within the range of $[10^5, 10^6]yr$, which aligns with the findings previously observed in section 5.3.2. Interestingly, the M class exhibits relatively low errors during the no accretion phase, while classes K, GF, and AB show higher errors in this particular stage of stellar evolution.

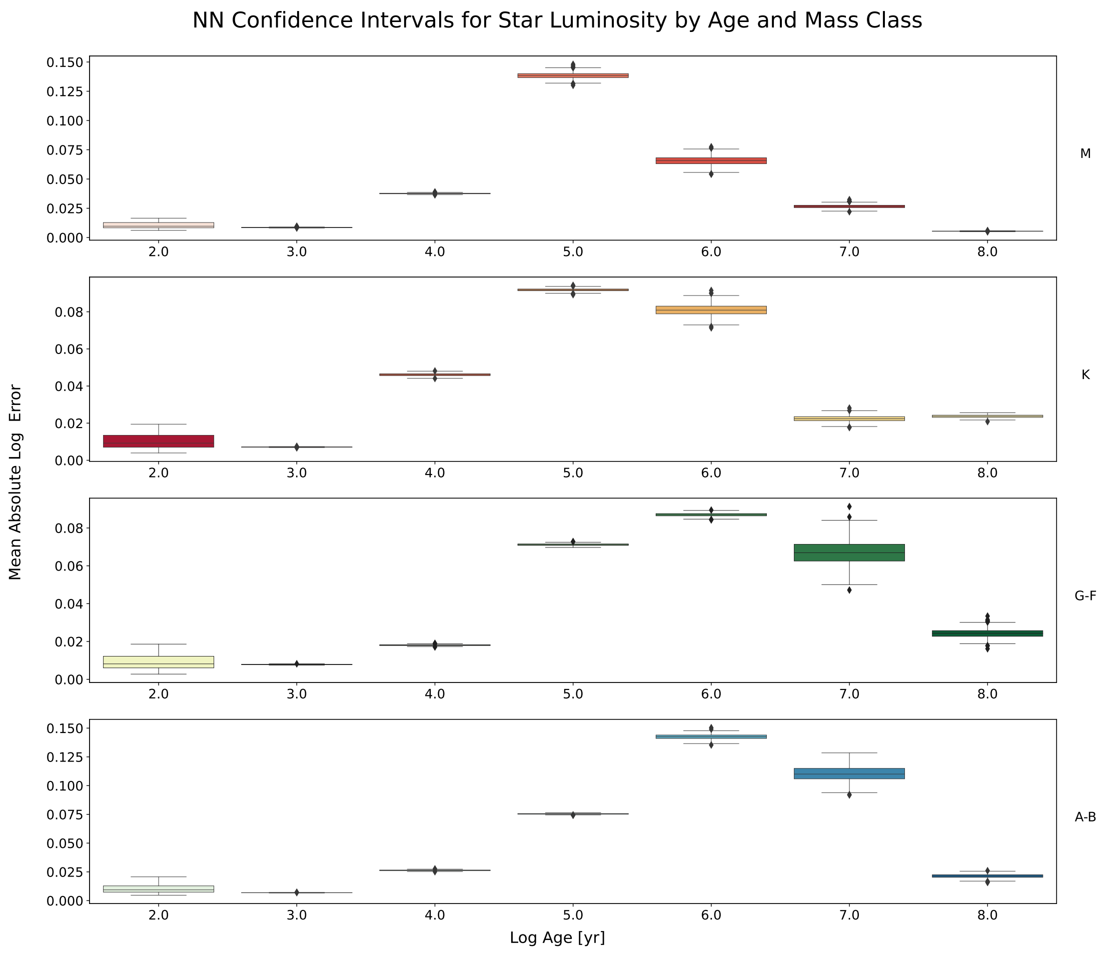

Figure 5.28: The confidence intervals of star luminosity, grouped by star class and age brackets, are shown in the figure. These intervals were derived using the absolute log error in the NN predictions on the test set. The y-axis shows the age brackets, with the last number representing the upper limit of each bracket. For instance, the first bracket represented by 2 corresponds to stars with ages between 10^0 and 10^2 years.

Figure 5.28 illustrates that the least accurate predictions for L_s occur in AB and M stars. An interesting aspect of these results is the consistent pattern observed across all star classes. Initially, during the early stages of star evolution, errors are minimal and concentrated at the lower end of the spectrum. However, as age progresses, errors start to increase at around $10^4 yr$ years, reaching their peak within the age range of $[10^5, 10^6] yr$. In other words, the $MALE$ exhibits a bell-shaped distribution skewed to the left for all the star classes.

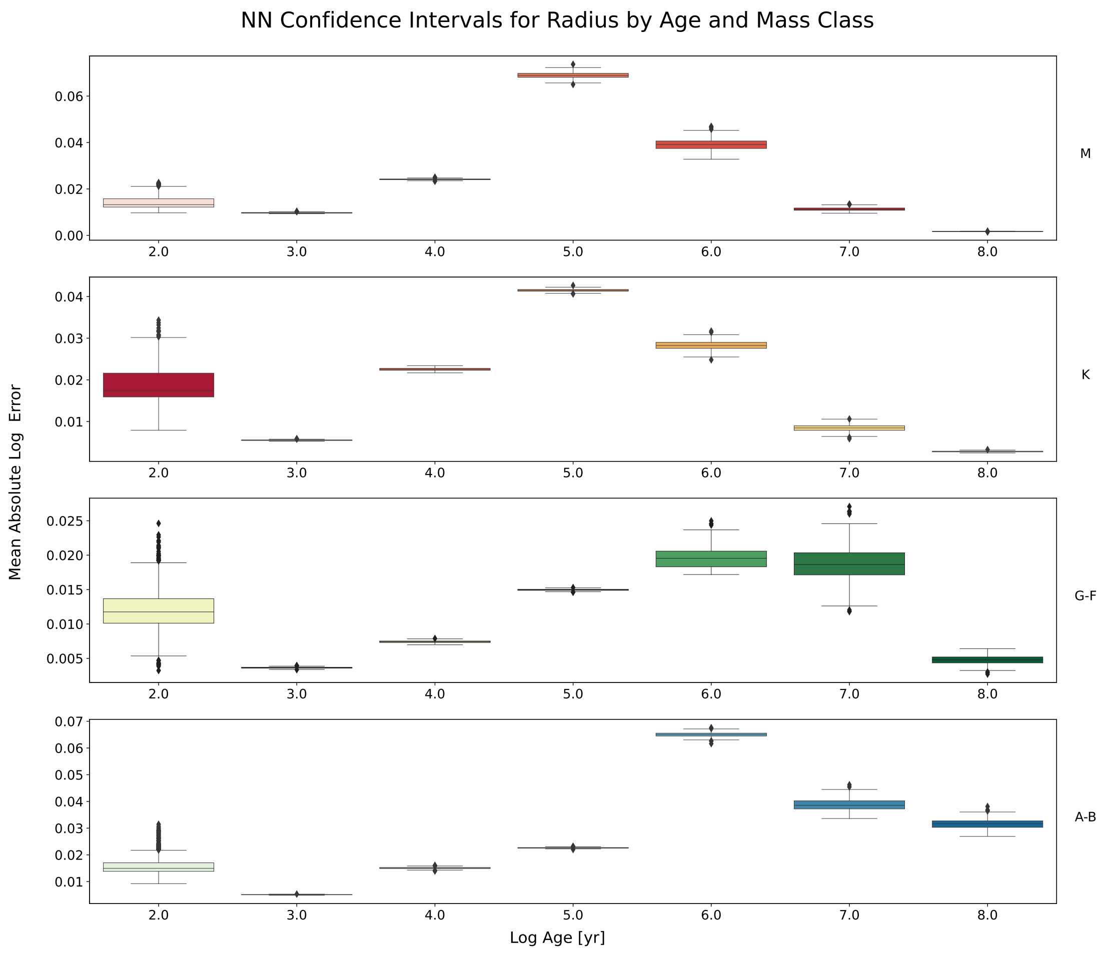

Figure 5.29: The confidence intervals of radius, grouped by star class and age brackets, are shown in the figure. These intervals were derived using the absolute log error in the NN predictions on the test set. The y-axis shows the age brackets, with the last number representing the upper limit of each bracket. For instance, the first bracket represented by 2 corresponds to stars with ages between 10^0 and 10^2 years.

In the case of the Radius, seen in figure 5.29, the worst errors are in star classes M and AB, again when the stars begin to reach the end of the accretion phase. In this case, we can see a spread of the $MALE$ distributions in the age range $[0, 10^2]yr$. This is especially true in the stars GF and K, given that, in both cases the outliers spread out to their relative maximum $MALE$. In this case, the no accretion section has relatively low errors, with the exception of the GF stars, in which the spread of the distribution at age 10^7, reaches the maximum $MALE$ in its class.

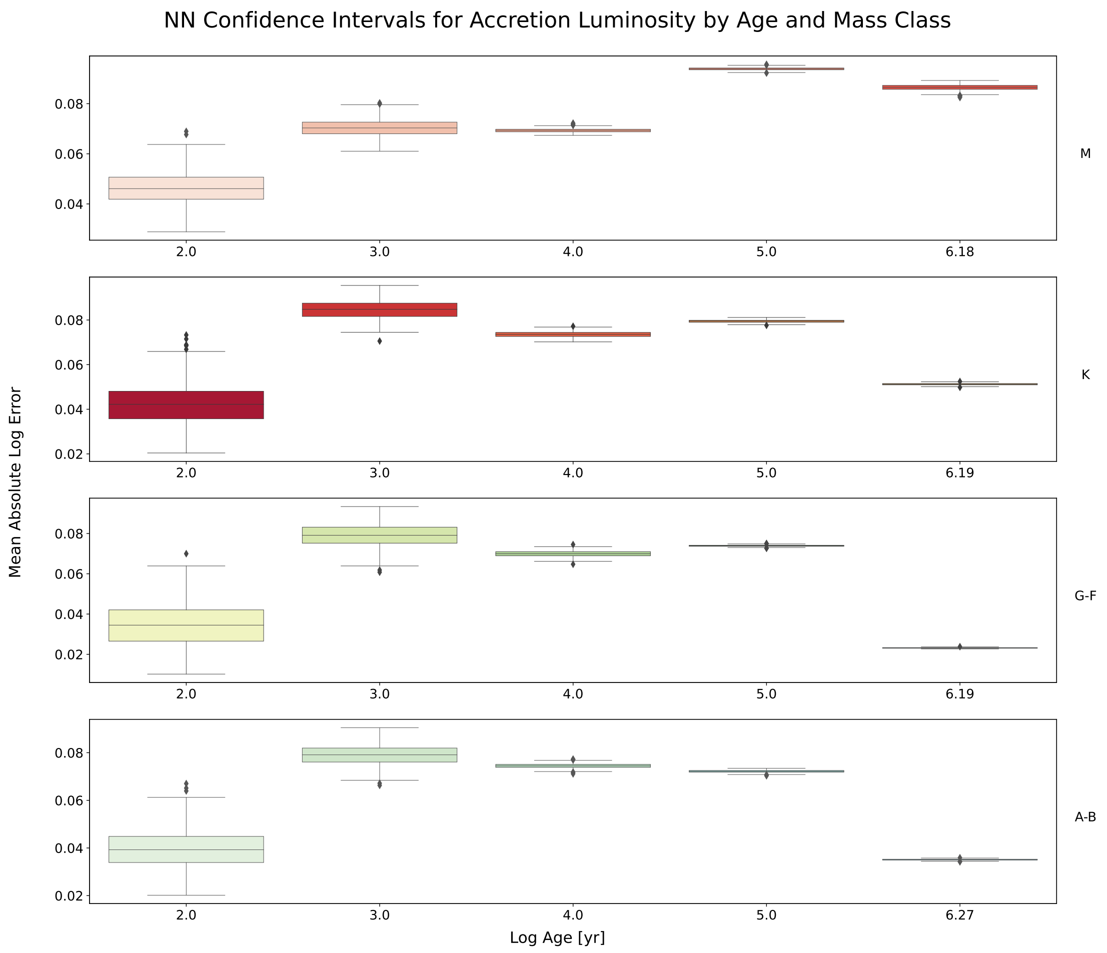

Figure 5.30: The confidence intervals of accretion luminosity, grouped by star class and age brackets, are shown in the figure. These intervals were derived using the absolute log error in the NN predictions on the test set. The y-axis shows the age brackets, with the last number representing the upper limit of each bracket. For instance, the first bracket represented by 2 corresponds to stars with ages between 10^0 and 10^2 years.

Finally, the L_{acc} results, shown in figure 5.30 are very consistent along all star classes and ages. An expected outcome is the spread of the distributions at the beginning of the star evolution up to $10^2 yr$ due to the results seen in figure 5.25. However, the accuracy achieved in the case of AB stars, especially toward the end of the accretion phase differs significantly from the previous predictions in this section.

In this section, we observed that the results generally align with the anticipated patterns, but they also provided a comprehensive validation of the limitations of the NN models. Notably, during the initial stages of star evolution ($[0, 10^4] yr$), all parameters exhibit relatively good predictive performance across all stars. Moreover, the predictions during the no-accretion phase are generally accurate, except for a few exceptions, particularly in the case of AB stars.

5.3.4 HR Diagram Analysis for the Neural Network

Arguably the most interesting analysis is the HR diagram predictions due to the importance of the star evolution analysis. In addition to this obtaining good results in the combined predictions of T_{eff}

and L_s is one of the great challenges of this thesis. Following the procedure done in section 5.2.4 using the benchmark simulations the results of this analysis are presented in figures 5.31, 5.34, and 5.32.

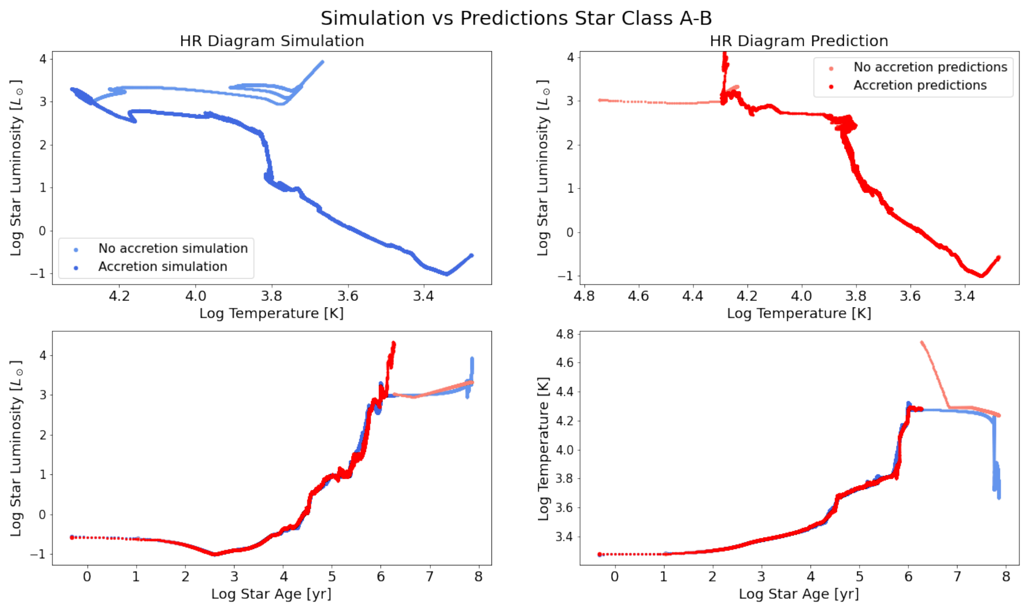

Figure 5.31: Test simulation of a star with final mass $M = 5.951 M_\odot$. The top panels display HR diagrams, with the left-hand side representing the diagram obtained from MESA data and the right-hand side showing the one obtained through the NN predictions. The bottom left-hand side presents the luminosity as a function of age, showing both the MESA values and the predictions for comparison. Similarly, the bottom right-hand side displays the temperature evolution of the star, with both MESA and predicted values overlaid for comparison.

Examining the predictions for the AB star in figure 5.31, we observe that the accretion set successfully tracks both the T_{eff} and L_s curves until the conclusion of this phase. However, at the beginning of the no accretion section, the predicted T_{eff} values are entirely inaccurate. Additionally, the NN model fails to capture the temperature decrease that occurs towards the end of the simulation at $10^8 yr$. Despite these challenges, the predictions still exhibit good resolution, showing a continuous curve along the star's evolution.

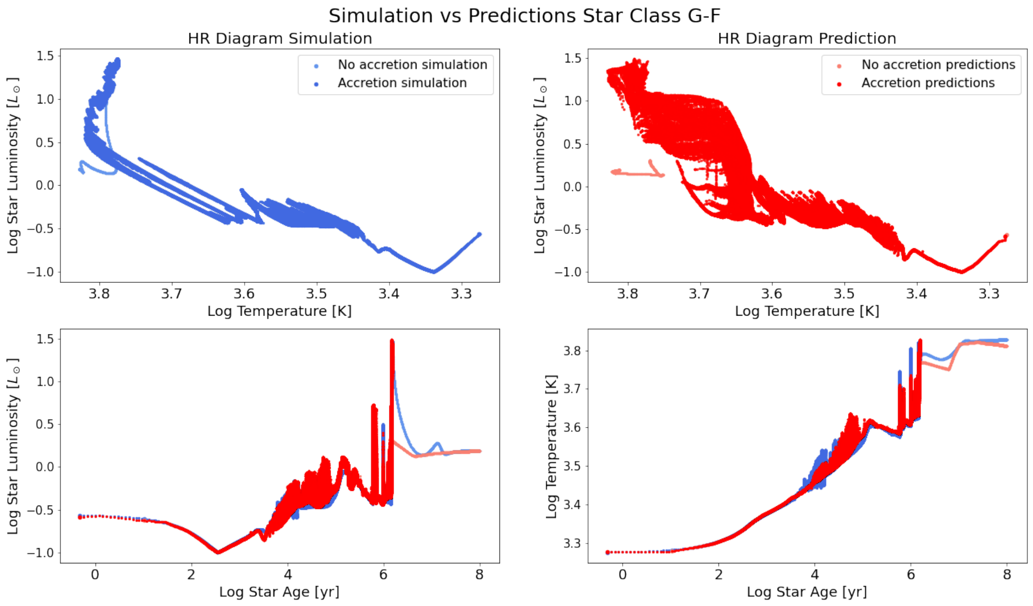

Figure 5.32: Test simulation of a star with final mass $M = 1.165M_\odot$. The top panels display HR diagrams, with the left-hand side representing the diagram obtained from MESA data and the right-hand side showing the one obtained through the NN predictions. The bottom left-hand side presents the luminosity as a function of age, showing both the MESA values and the predictions for comparison. Similarly, the bottom right-hand side displays the temperature evolution of the star, with both MESA and predicted values overlaid for comparison.

For the GF star, as shown in figure 5.32, the T_{eff} predictions in the age range of $[10^{3.8}, 10^{4.8}]yr$ do not accurately capture the fluctuations present in the simulated data. Additionally, in the predicted HR diagram, we notice an area around $[10^{3.65}, 10^{3.75}]K$ and $[10^0, 10^{1.3}]L_\odot$, where the combination of parameter errors creates a feature that is not present in the simulation.
During the no accretion phase, the predicted T_{eff} follows the simulation curve to some extent. However, the predicted L_s does not match the curve pattern at the beginning of this phase. Moreover, the L_s predictions by the NN model only manage to reproduce a linear curve at the end of this phase, failing to capture the detailed features of the simulated curve. Despite these discrepancies, it is notable that the overall shapes and value ranges are roughly reproduced by the NN models, even though the finer details of the curves are not accurately reproduced.

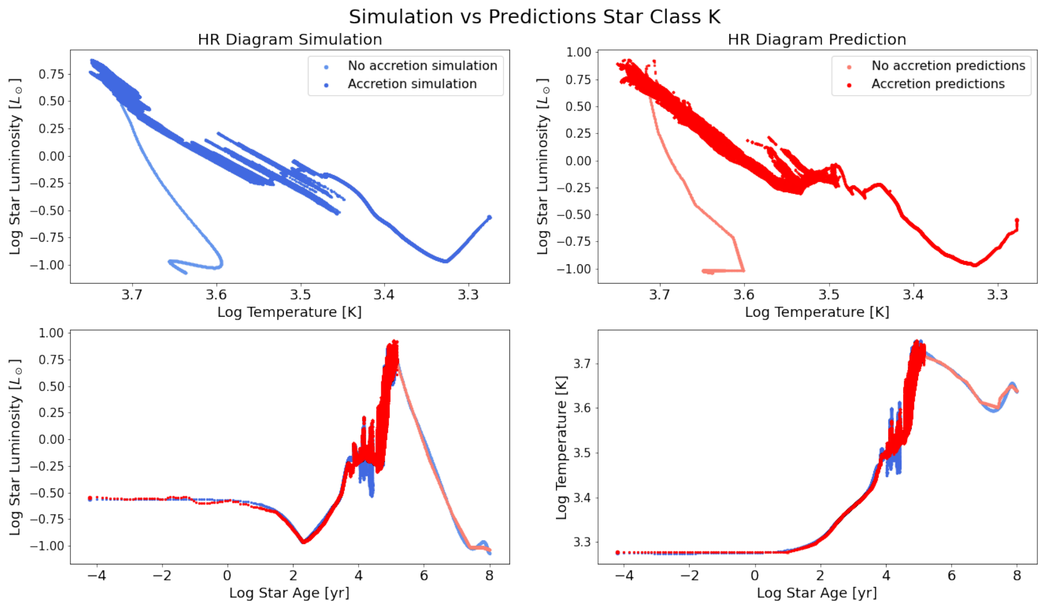

Figure 5.33: Test simulation of a star with final mass $M = 0.613M_\odot$. The top panels display HR diagrams, with the left-hand side representing the diagram obtained from MESA data and the right-hand side showing the one obtained through the NN predictions. The bottom left-hand side presents the luminosity as a function of age, showing both the MESA values and the predictions for comparison. Similarly, the bottom right-hand side displays the temperature evolution of the star, with both MESA and predicted values overlaid for comparison.

The predictions for the K star, as depicted in figure 5.33, generally capture the overall shape of the T_{eff} and L_s curves. However, they fail to faithfully reproduce the fluctuations observed in both T_{eff} and L_s after $10^4 yr$. Consequently, this discrepancy is reflected in the predicted HR diagram, where the NN models struggle to replicate some of the most interesting features around $[10^{3.5}, 3^6]K$ and $[10^{-0.5}, 10^{0.25}]L_\odot$.
Furthermore, in the no accretion section of the predictions, both curves are somewhat reproduced by the NN model. However, in the predicted HR diagram, this curve is composed of linear sections, resulting in a failure to replicate the smoothness present in the actual curve.

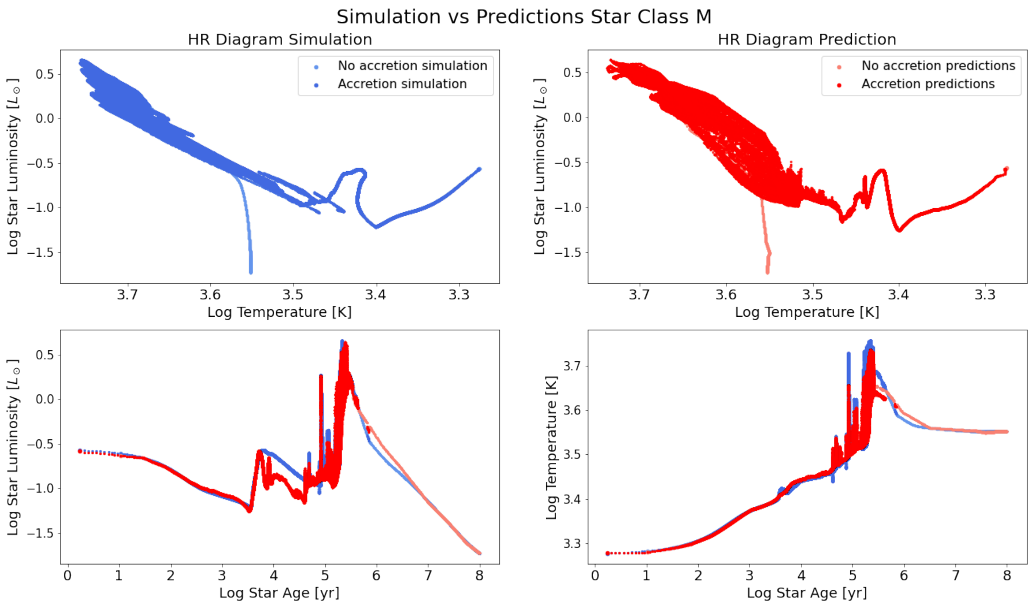

Figure 5.34: Test simulation of a star with final mass $M = 0.358M_\odot$. The top panels display HR diagrams, with the left-hand side representing the diagram obtained from MESA data and the right-hand side showing the one obtained through the NN predictions. The bottom left-hand side presents the luminosity as a function of age, showing both the MESA values and the predictions for comparison. Similarly, the bottom right-hand side displays the temperature evolution of the star, with both MESA and predicted values overlaid for comparison.

Finally, the predictions in the star M seen in figure 5.34, manage to reproduce the overall star evolution. However, we can see that in the L_s curve there is a continuous error at the age range $[10^4, 10^5]yr$. In addition to this, in the T_{eff} curve, we can see that the decrease in temperature seen around $10^5 yr$ is not reproduced by the NN model results. Furthermore in the same age range, the rapid increase in T_{eff} is not reproduced fully. In this case, the no accretion phase is a simple one, therefore the model managed to reproduce the simulation results without a problem.
In general, this section demonstrates that the predicted HR diagrams provide a reasonable approximation of the overall evolution of the stars. The predictions show continuous curves that follow the natural progression of stellar structure evolution.
An interesting observation is that, across all stars, the predictions struggle to reproduce areas with significant fluctuations. This is particularly evident in the case of the decrease in T_{eff} and L_s following an accretion burst, which is not accurately reproduced in most cases.

5.3.5 Hyper Parameter Importance Neural Network

In the case of the NN the importance of the hyper parameters mentioned in section 4.6.1 was obtained using `optuna.visualization.plot_param_importances` function [60]. Results can be seen in figure 5.35.

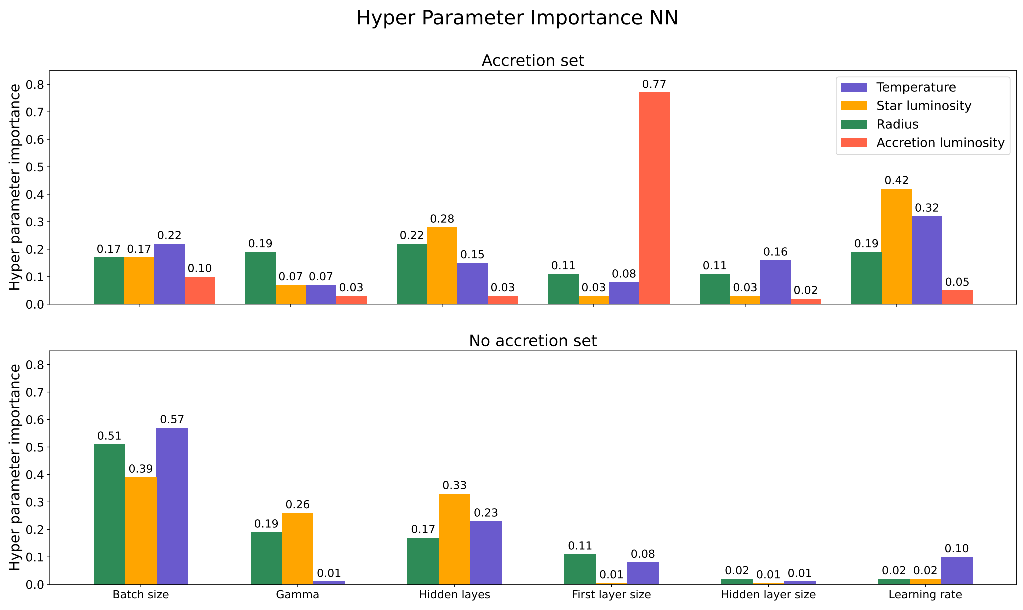

Figure 5.35: The importance of hyperparameters for temperature, star luminosity, radius, and accretion luminosity NN models. The top panel displays the parameter importance of accretion models, while the bottom panel represents the parameter importance for no accretion models.

In figure 5.35, we can observe that there is no single parameter that is generally more important for all the stellar structure parameters in the accretion set. However, the `Learning rate` generally stands out as the best performer for most parameters, except for L_{acc}, where the `First layer size` takes the lead with a hyper parameter importance (HPI) of 0.77. The `Hidden layers` parameter comes in second place in importance for this set.
On the other hand, in the no accretion set, it is evident that the most critical parameter is the `Batch size`, followed by the `Hidden layers` parameter as the second most influential. Interestingly, the `Hidden layers size` appears to be the least important parameter for both sets.

5.3.6 Number of Parameters Analysis

Given the results seen in figure 5.35, exploring the increase in hidden layers and by default the number of parameters results in interesting. Therefore an increase in the number of parameters as explained in section 4.6.2 was performed iteratively on the best models in order to see the influence that this might have in the results. results of this analysis are shown in figures 5.36, 5.37, 5.38, 5.39, 5.40, 5.41,and 5.42

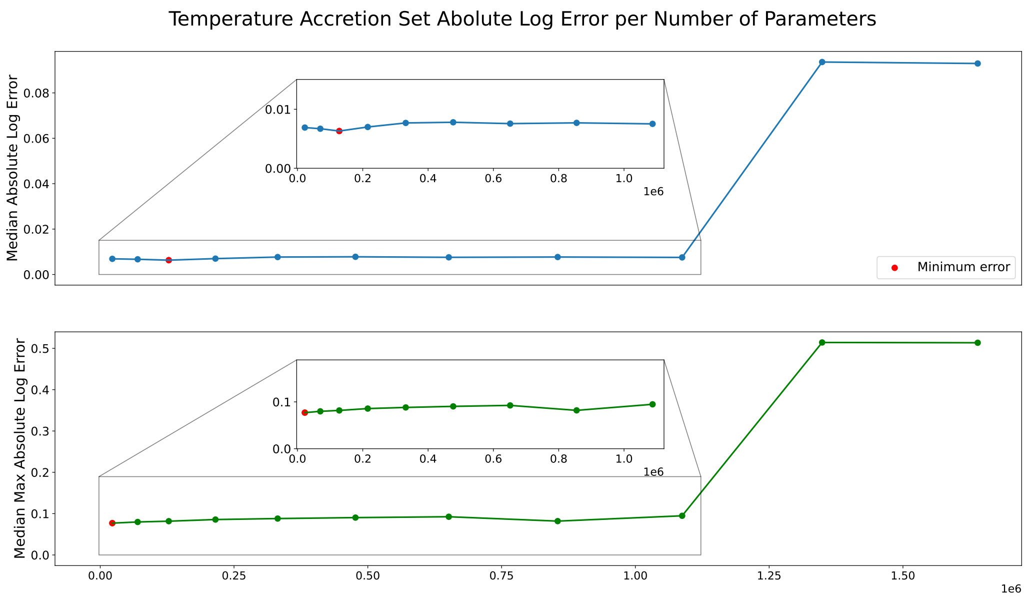

Figure 5.36: The errors of the temperature accretion set are shown as a function of the number of parameters. The top panel illustrates $MedALE$ plotted against the number of parameters, while the bottom panel displays $MaxMedALE$ as a function of the number of parameters. To capture the finer details that may have been lost due to plot dimensions, a zoom-in was performed. Within this zoomed view, the red point represents the minimum error achieved in the experiment.

Figure 5.36 shows that the optimal number of parameters for minimizing the $MedALE$ is 0.13×10^6, while the default number of parameters, 0.02×10^6, works best for minimizing $MedMaxALE$. Notably, the model's performance decreases significantly when the number of parameters exceeds 1.3×10^6.

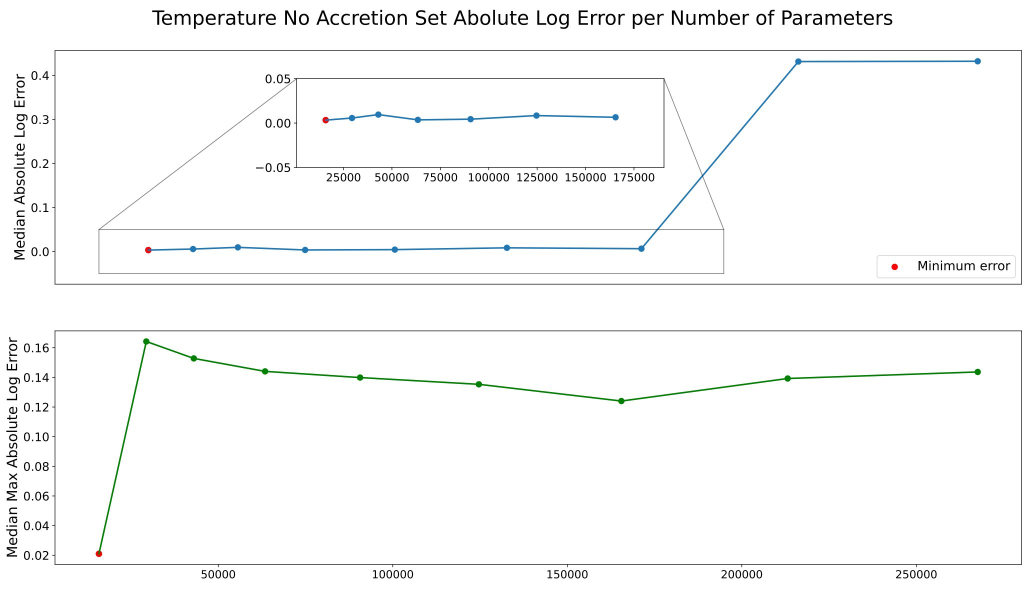

Figure 5.37: The errors of the temperature no accretion set are shown as a function of the number of parameters. The top panel illustrates $MedALE$ plotted against the number of parameters, while the bottom panel displays $MaxMedALE$ as a function of the number of parameters. To capture the finer details that may have been lost due to plot dimensions, a zoom-in was performed. The red point represents the minimum error achieved in the experiment.

Regarding T_{eff} without accretion, the most effective model was the default one with 0.016×10^6 parameters. When testing new models, their accuracy in terms of maximum errors decreased significantly, being on average 7 times worse than the minimum error. As for $MedALE$, the errors start to diverge at around 0.15×10^6 number of parameters increasing significantly after this.

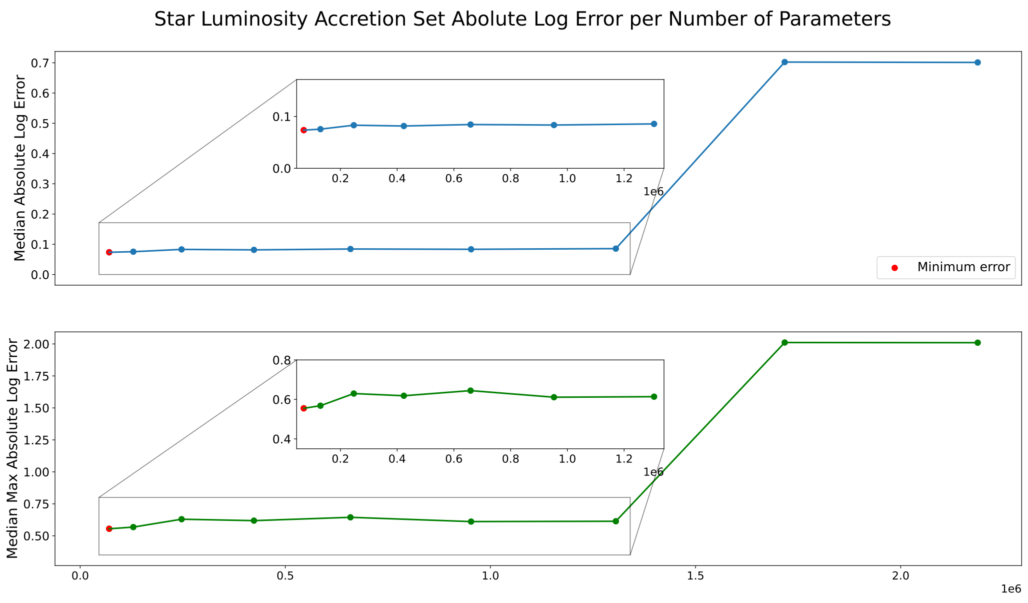

Figure 5.38: The errors of the star luminosity accretion set are shown as a function of the number of parameters. The top panel illustrates $MedALE$ plotted against the number of parameters, while the bottom panel displays $MaxMedALE$ as a function of the number of parameters. To capture the finer details that may have been lost due to plot dimensions, a zoom-in was performed. Within this zoomed view, the red point represents the minimum error achieved in the experiment.

In figure 5.38, we observe the most favorable outcomes in predicting L_s within the accretion segment, it is achieved by the default model with 0.07×10^6 parameters. Furthermore, it becomes evident that beyond 1.3×10^6 parameters, the model errors experience a significant increase.

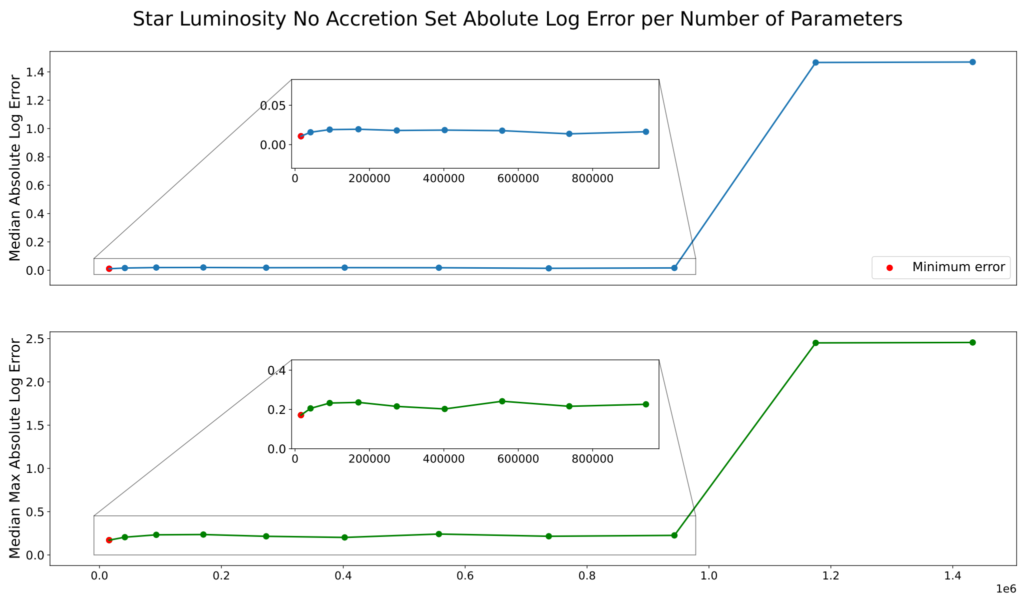

Figure 5.39: The errors of the star luminosity no accretion set are shown as a function of the number of parameters. The top panel illustrates $MedALE$ plotted against the number of parameters, while the bottom panel displays $MaxMedALE$ as a function of the number of parameters. To capture the finer details that may have been lost due to plot dimensions, a zoom-in was performed. Within this zoomed view, the red point represents the minimum error achieved in the experiment.

Likewise, when considering L_s no accretion section, the optimal model remains the original one, featuring 0.016×10^6 parameters. In both error analyses, the outcomes begin to diverge after reaching 0.94×10^6 parameters. As a result, the $MedALE$ experienced a substantial increase of up to 89 times, while the $MedMaxALE$ escalated up to 10 times compared to the minimum values.

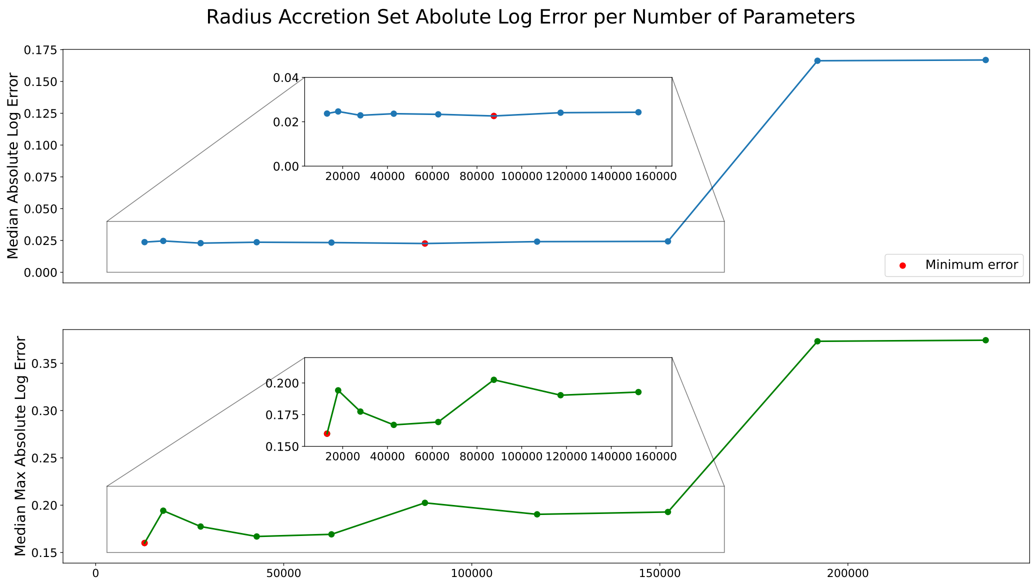

Figure 5.40: The errors of the radius accretion set are shown as a function of the number of parameters. The top panel illustrates $MedALE$ plotted against the number of parameters, while the bottom panel displays $MaxMedALE$ as a function of the number of parameters. To capture the finer details that may have been lost due to plot dimensions, a zoom-in was performed. Within this zoomed view, the red point represents the minimum error achieved in the experiment.

In the figure 5.40, it is possible to see the R accretion section $MedALE$ minimum was found using 0.06×10^6 parameters. Conversely, the $MedMaxALE$ minimum was observed in the default model, having 0.01×10^6 parameters. Upon examining both error metrics, it becomes evident that the models start to overfit after reaching 0.2×10^6 parameters.

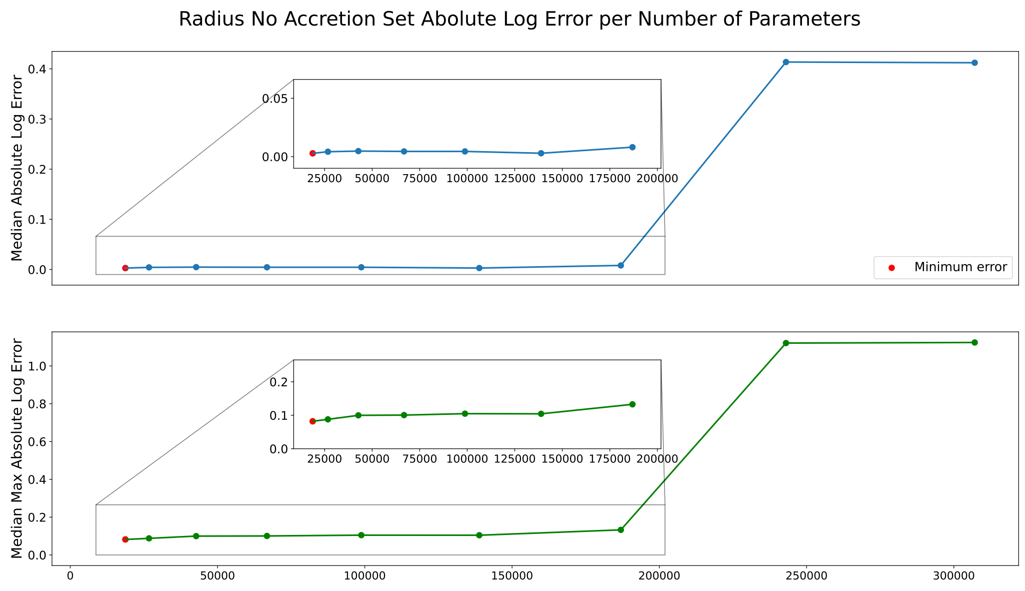

Figure 5.41: The errors of the radius no accretion set are shown as a function of the number of parameters. The top panel illustrates $MedALE$ plotted against the number of parameters, while the bottom panel displays $MaxMedALE$ as a function of the number of parameters. To capture the finer details that may have been lost due to plot dimensions, a zoom-in was performed. Within this zoomed view, the red point represents the minimum error achieved in the experiment.

According to our error metrics, the examination of the R no accretion section indicates that the default model performed the best, being this the model with 0.02×10^6 parameters. Similarly, both metrics diverge from the minimum area after around 0.19×10^6 parameters. For the $MedALE$ metric, we observed an average increase of up to 91 times compared to the minimum errors in this metric. Meanwhile, in the case of $MedMaxALE$, there was an increase of approximately 11 times compared to its minimum value.

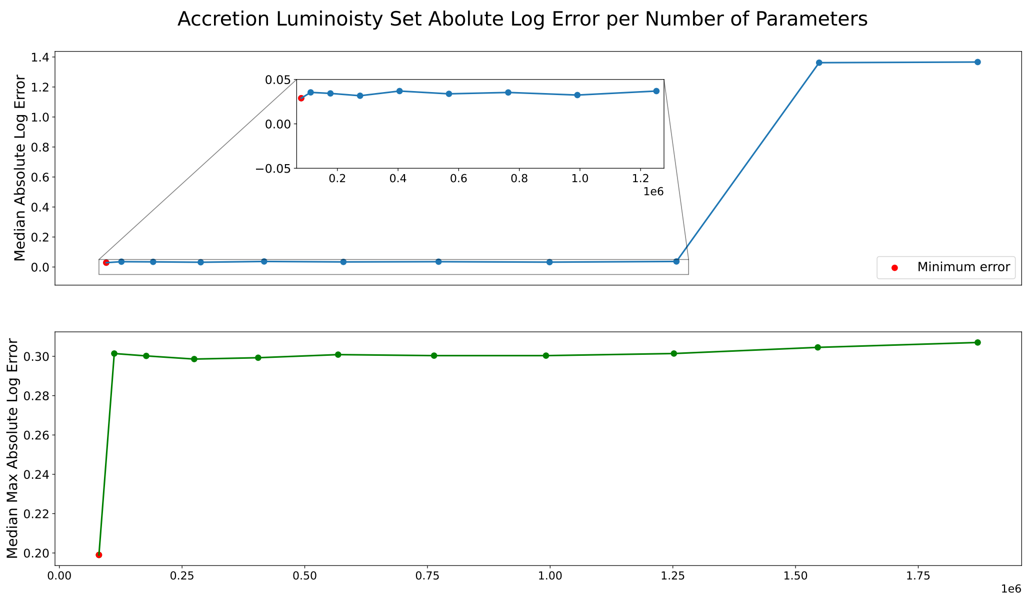

Figure 5.42: The errors of the accretion luminosity are shown as a function of the number of parameters. The top panel illustrates $MedALE$ plotted against the number of parameters, while the bottom panel displays $MaxMedALE$ as a function of the number of parameters. To capture the finer details that may have been lost due to plot dimensions, a zoom-in was performed. The red point represents the minimum error achieved in the experiment.

In Figure 5.42, it is evident that the original model excelled in the accretion luminosity test, achieving the best performance with 0.08×10^6 parameters, as indicated by both error metrics, $MedALE$ and $MedMaxALE$. Additionally, we can see that in the $MedALE$ metric, the errors start to increase drastically after reaching 1.2×10^6 parameters, while in the $MedMaxALE$ metric, a noticeable difference is observed right after the original model.

The results in this section indicate that, in general, the best predictions are obtained using the original models. This could be attributed to the fact that increasing the number of parameters requires tuning the other parameters as well. Moreover, it was observed that adding too many parameters leads to overfitting, suggesting that along with increasing the number of parameters, the other hyperparameters also need to be adjusted accordingly.

5.4 Temporal Neural Network

Implementing the procedure outlined in section 4.1.6, we meticulously prepared the data by including state vectors that capture the star's evolution over the last 10, 100, and 1000 years at the current point. In addition to this, the `Hidden layers` parameter was increased to allow a space exploration of up to 20 `Number of Hidden Layers`. The following results are the ones obtained using this data set.

5.4.1 Models Selection Temporal Neural Network

Similar to section 5.3.1, the complete dataset was not utilized in this experiment. Additionally, the mix set was excluded due to discrepancies in the data format caused by the absence of state vectors in the no accretion set. Nevertheless, for the no accretion set, an exploration was conducted by varying the `Number of Hidden Layers` with an additional constraint in the lower bound as explained in section 5.3.1. The results obtained from the HPT are displayed in appendix B.3. From these results,

the best models were evaluated using the test data set and the $MedALE$ and $MedMAxALE$ metrics with the results shown in appendix A.3.
The error calculations for the best models obtained using the temporal data are presented in table 5.7.

Best Parameters TNN Error Analysis.				
Errors	T_{eff}	L_s	R	L_{acc}
$MedALE$	0.0076	0.084	0.029	0.025
$MedMaxALE$	0.1	0.63	0.25	0.23

Table 5.7: $MedALE$ and $MedMaxALE$ for TNN all prediction parameters.

In this machine learning experiment, the most optimal models were achieved by employing the balanced set for T_{eff}, L_s, and R, whereas for L_{acc}, the best model was obtained using the reduced set.

The result of the no accretion test with the modified `Number of Hidden Layers` in the HPT optimization are presented in table 5.8.

Best Parameters Large NN No Accretion Error Analysis.			
Errors	T_{eff}	L_s	R
$MedALE$	0.0018	0.0083	0.004
$MedMaxALE$	0.033	0.35	0.11

Table 5.8: $MedALE$ and $MedMaxALE$ for Large NN No Accretion Error prediction parameters.

5.4.2 Errors Distributions per Input Parameter Temporal Neural Network

Using the best models obtained in section 5.4.1 the analysis of the errors distributions per input parameters was done see in figures 5.43, 5.44, 5.45, and 5.46.

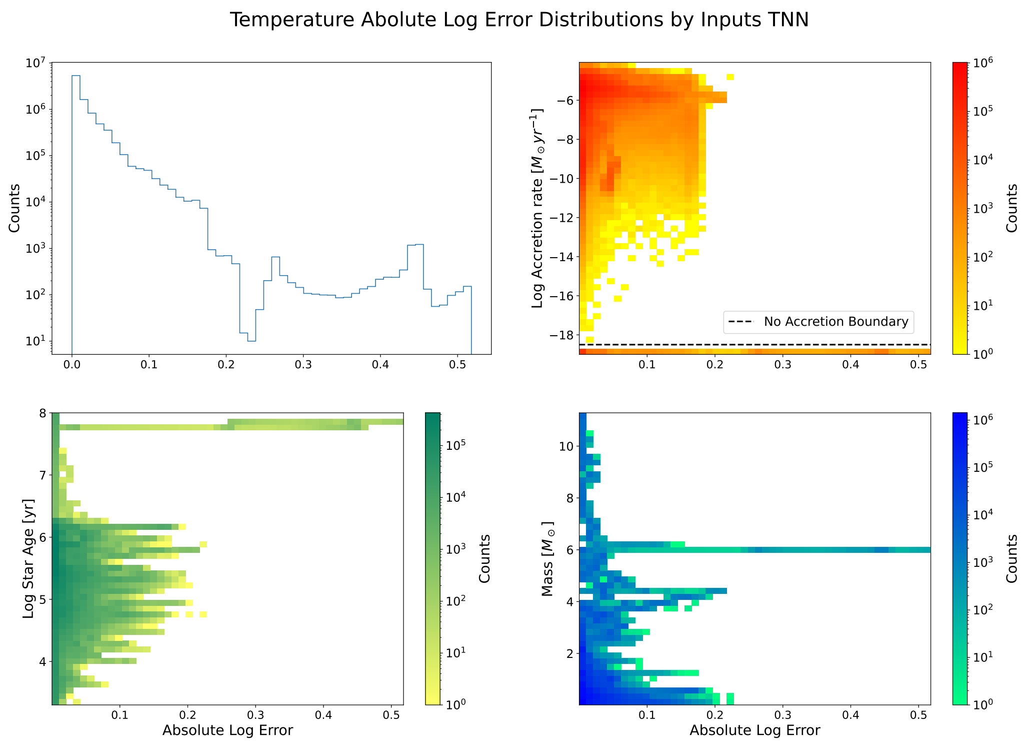

Figure 5.43: Temperature absolute error distribution comparison between input parameters obtained using the TNN model. The top left-hand side panel illustrates the distribution of ALE. In the top right-hand side panel, the ALE distribution per accretion rate is displayed, with the section below the dashed line representing the region where $M_\odot yr^{-1} = 0$. The bottom left-hand side panel showcases the distribution of ALE per star age, and lastly, the bottom right-hand side panel demonstrates the distribution of ALE per mass.

In figure 5.43, the temperature predictions exhibit a clustering of errors distributed across various ages, except in the area of $[10^{3.3}, 10^4]yr$, where the errors are notably smaller. Regarding the accretion rate against ALE, a dense cluster of errors is noticeable at a high accretion rate of $10^{-6}M_\odot yr^{-1}$. As for the mass per ALE predictions, two peaks with elevated errors are observed, one at lower masses in the range of $[0, 1]M_\odot$ and another around $4M_\odot$. When examining the no accretion phase, it becomes evident that the predictions were the least accurate in this region, particularly due to one simulation with a mass of $6M_\odot$. Nevertheless, it is worth noting that the majority of these errors are concentrated within a low ALE range.

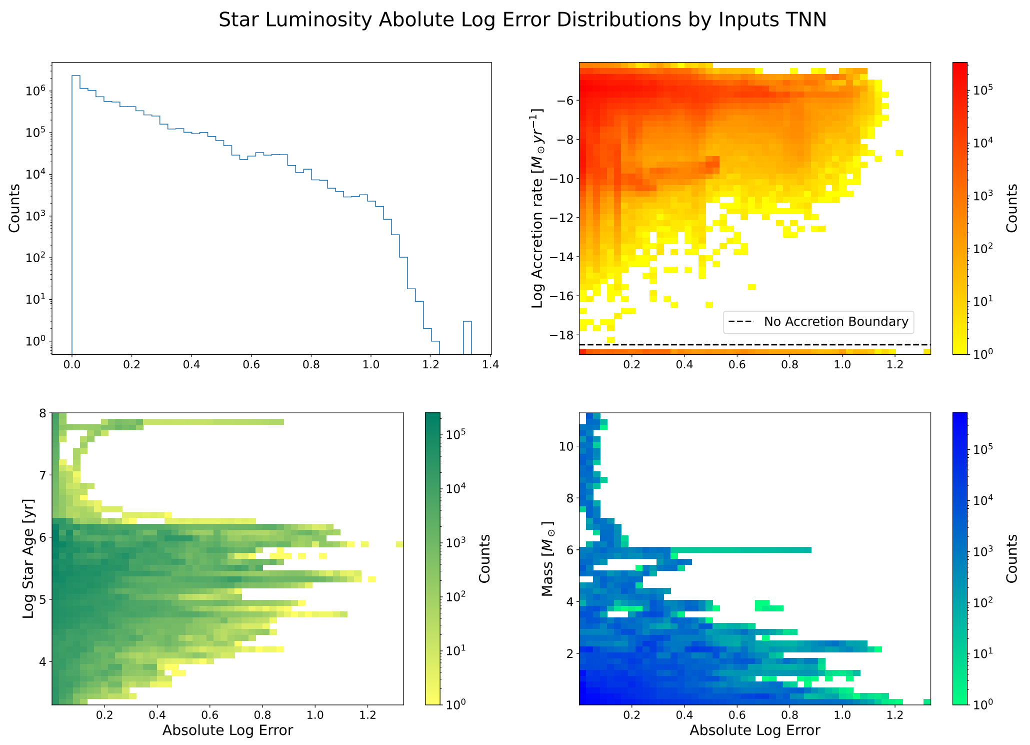

Figure 5.44: Star luminosity absolute error distribution comparison between input parameters obtained using the TNN model. The top left-hand side panel illustrates the distribution of ALE. In the top right-hand side panel, the ALE distribution per accretion rate is displayed, with the section below the dashed line representing the region where $M_\odot yr^{-1} = 0$. The bottom left-hand side panel showcases the distribution of ALE per star age, and lastly, the bottom right-hand side panel demonstrates the distribution of ALE per mass.

In the star luminosity case, as seen in figure 5.44, we observe a similar behavior to the T_{eff} in terms of age, where the density of errors spans across all ages, except for the initial stages of evolution and a sudden drop at the end of the temporal set. Regarding the accretion rate per ALE plot, a high density of errors is noticeable within the range of $[10^{-10}, 10^{-6}]M_\odot yr^{-1}$. As for the mass per ALE plot, it is evident that the most significant errors occur in stars with lower masses, specifically within the range of $[0, 3]M_\odot$. In the no accretion set, we observe that the majority of errors are concentrated in the lower range of values.

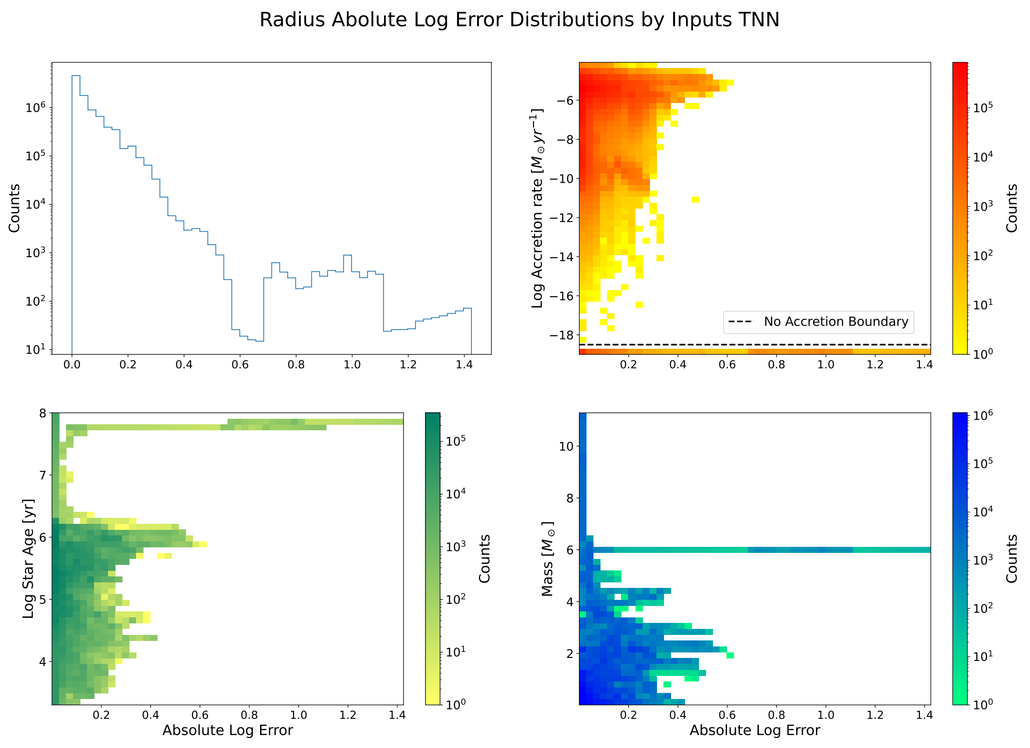

Figure 5.45: Radius absolute error distribution comparison between input parameters obtained using the TNN model. The top left-hand side panel illustrates the distribution of ALE. In the top right-hand side panel, the ALE distribution per accretion rate is displayed, with the section below the dashed line representing the region where $M_\odot yr^{-1} = 0$. The bottom left-hand side panel showcases the distribution of ALE per star age, and lastly, the bottom right-hand side panel demonstrates the distribution of ALE per mass.

The distribution of radius errors seen in figure 5.45 reveals that the most significant errors occur in regions with high accretion rates, approximately around $10^{-6}M_\odot yr^{-1}$. Concerning the ALE per age the highest density of errors in the accretion set emerges towards the end of this set, with the least accurate predictions concentrated around the $10^6 yr$. As for the ALE per mass, the errors demonstrate a decreasing trend as the mass increases, with the lowest errors observed in the mass range of $[7, 11]M_\odot$.
Similar to the temperature predictions, the absence of accretion leads to poor accuracy, particularly for the star with a mass of $M = 6M_\odot$ and towards the end of the simulation. Nonetheless, it is important to note that most errors fall within the lower range of the ALE value.

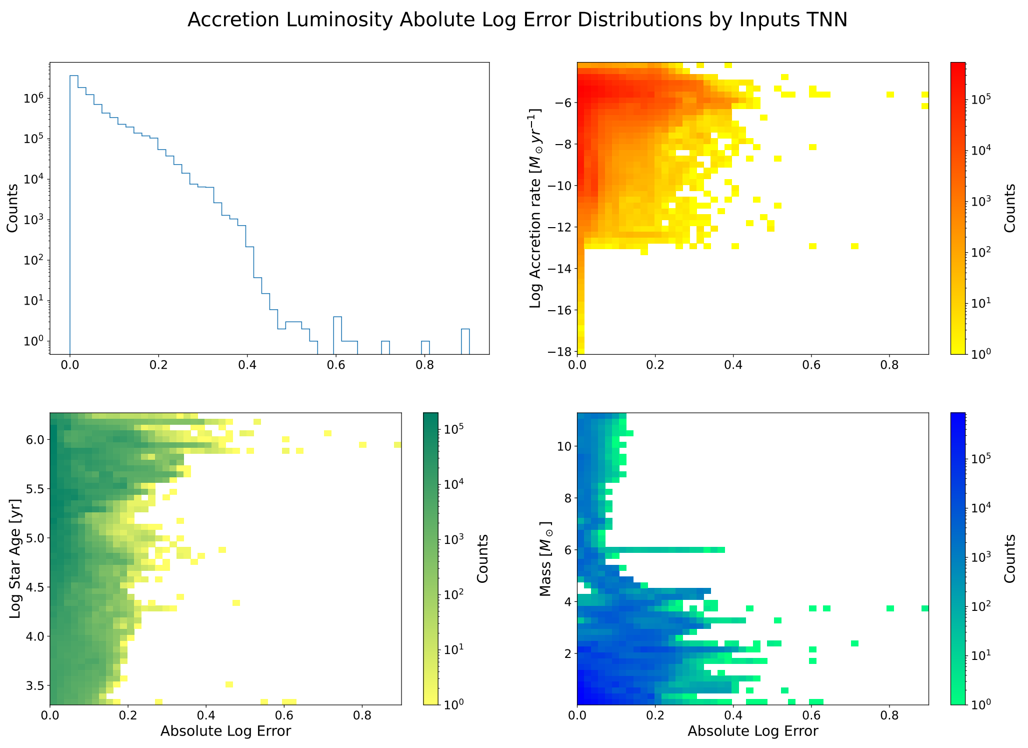

Figure 5.46: Accretion luminosity absolute error distribution comparison between input parameters obtained using the TNN model. The top left-hand side panel illustrates the distribution of ALE. In the top right-hand side panel, the ALE distribution per accretion rate is displayed, with the section below the dashed line representing the region where $M_\odot yr^{-1} = 0$. The bottom left-hand side panel showcases the distribution of ALE per star age, and lastly, the bottom right-hand side panel demonstrates the distribution of ALE per mass.

Finally, the plot shown in figure 5.46 illustrates the ALE distribution of the accretion luminosity. It exhibits a prominent concentration of errors at high accretion rates, specifically at $10^{-6} M_\odot yr^{-1}$. Regarding the age ALE relation, the errors appear to be somewhat evenly distributed across all age ranges, with only a few high errors in the $10^6 yr$ region. As for the ALE mass analysis, it reveals that the density with the least accuracy falls within the range of $[0, 4] M_\odot$, with a prominent peak observed at $6M\odot$.

During this analysis, we observed a significant decrease in accretion luminosity errors compared to the previous two experiments. As expected, regions with high accretion rates exhibited poor accuracy, and the low-mass stars also showed the most substantial errors.

5.4.3 Confidence Intervals Temporal Neural Netowrk

Following the usual analysis, confidence intervals were calculated for the outcomes derived from the TNN experiment seen in figure 5.47.

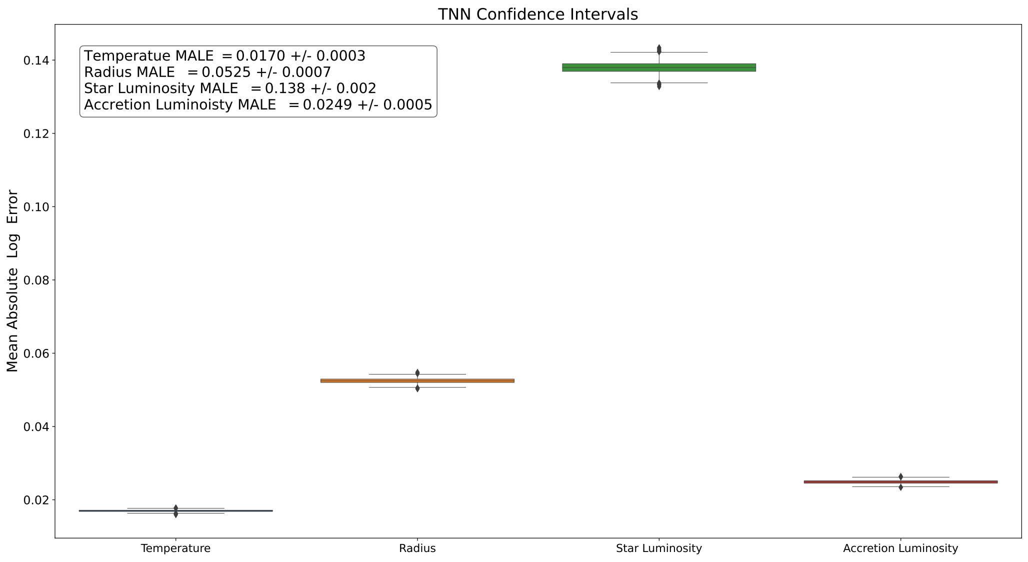

Figure 5.47: Confidence intervals for temperature, star radius luminosity, and accretion luminosity, were obtained by utilizing the absolute log error in the TNN predictions on the test set.

During the examination of the confidence intervals results, we observe the anticipated rise in $MALE$ for L_s and R when compared to T_eff as seen in the previous experiments. An interesting finding from this analysis is related to L_{acc}. Specifically, we can observe that $MALE = 0.0249 \pm 0.0005$, which represents an approximate decrease of 46% from the results LightGBM and NN results shown in figures 5.12 and 5.26 respectively.
Continuing with the familiar analysis, confidence intervals were calculated for different mass classes and age groups. An important consideration to bear in mind during this analysis is that the age range was limited due to the predictions starting after $10^{3.3}yr$. As a consequence, the age range for this experiment was approximately reduced to $[10^3, 8]yr$. The results of these calculations can be seen in figures 5.48, 5.49, 5.50, and 5.51.

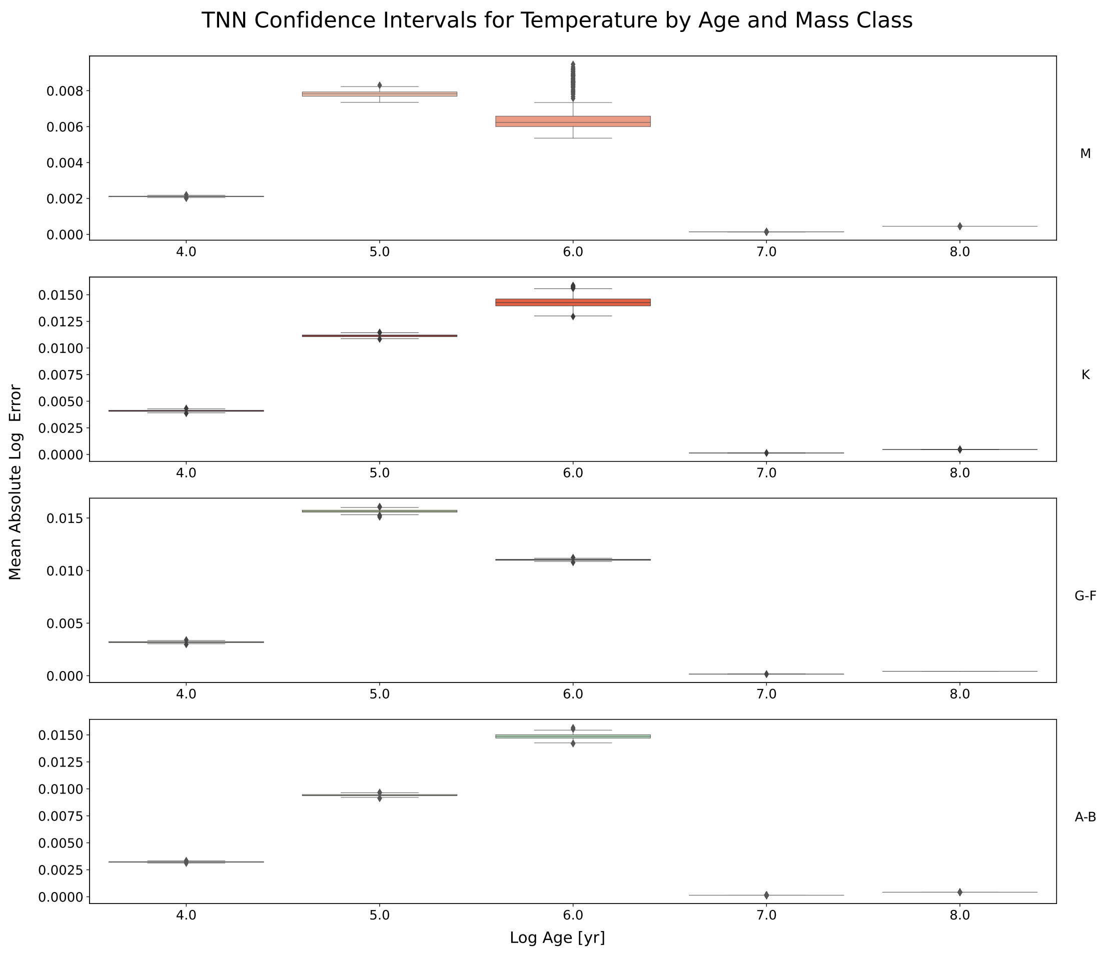

Figure 5.48: The confidence intervals of temperature, grouped by star class and age brackets, are shown in the figure. These intervals were derived using the absolute log error in the TNN predictions on the test set. The y-axis shows the age brackets, with the last number representing the upper limit of each bracket. For instance, the first bracket represented by 4 corresponds to stars with ages between $10^{3.3}$ and 10^4 years.

With regards to temperature figure 5.48 reveals that the star types K and AB display the least accurate predictions, particularly towards the end of the accretion simulation. Additionally, GF stars exhibit notably poor errors within the age range of $[10^4, 10^5]yr$. On the contrary, the M class demonstrates the best performance with its $MALE$ range being up to half that of the other star types. Consistently, across all cases, stars within the age range of $[10^3, 10^4]yr$ and at the end of the main sequence, specifically after 10^6yr, exhibit the best performance.

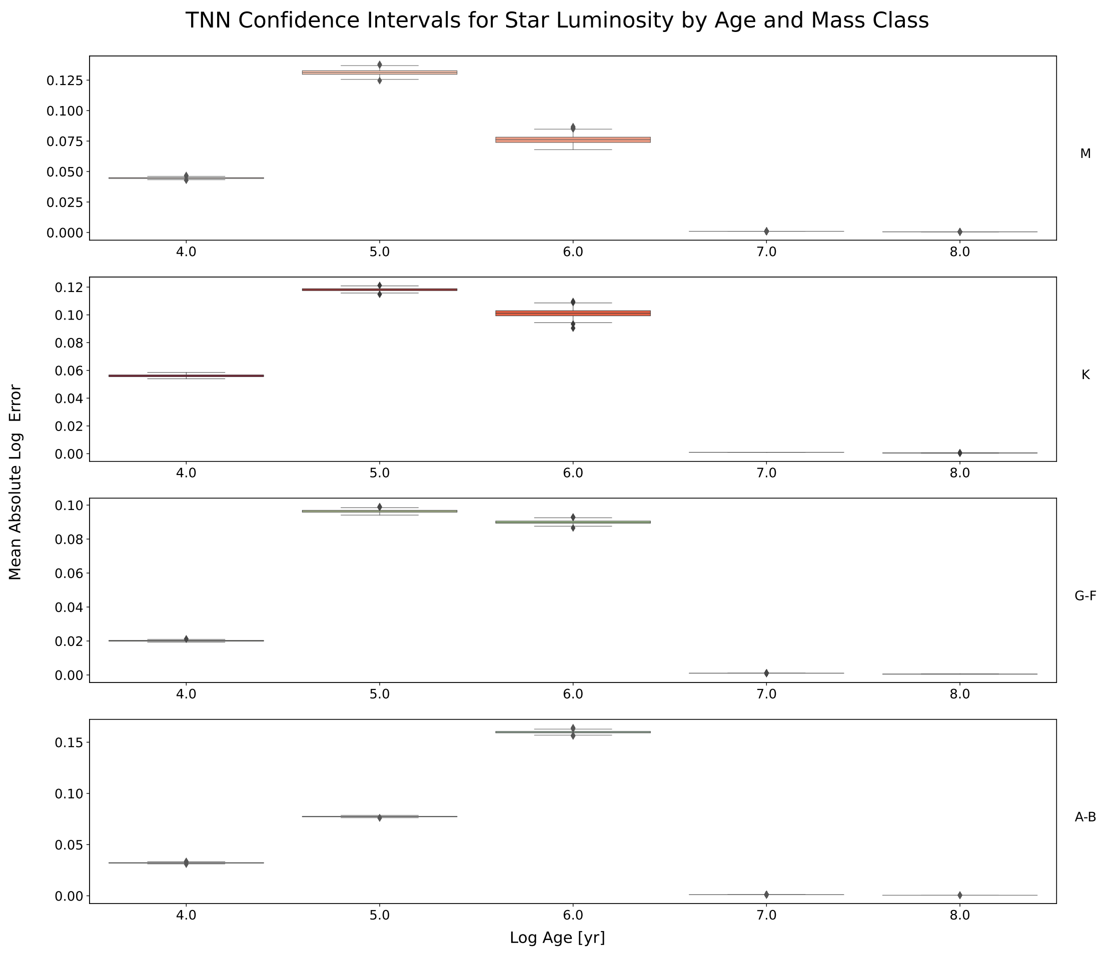

Figure 5.49: The confidence intervals of star luminosity, grouped by star class and age brackets, are shown in the figure. These intervals were derived using the absolute log error in the TNN predictions on the test set. The y-axis shows the age brackets, with the last number representing the upper limit of each bracket. For instance, the first bracket represented by 4 corresponds to stars with ages between $10^{3.3}$ and 10^4 years.

The confidence intervals for star luminosity shown in figure 5.49 indicate that the most significant errors occur in stars of types AB and M. Specifically, stars of type AB exhibit poor accuracy toward the end of the accretion phase. While stars of type M displayed poor accuracy within the ages ranging from 10^4 to 10^5 years. Similar to the T_{eff} analysis, the age range that yields the best performance across all star classes is $[10^3, 10^4]$ years for the accretion phase. However, the best performance is observed in the no-accretion phase in all cases.

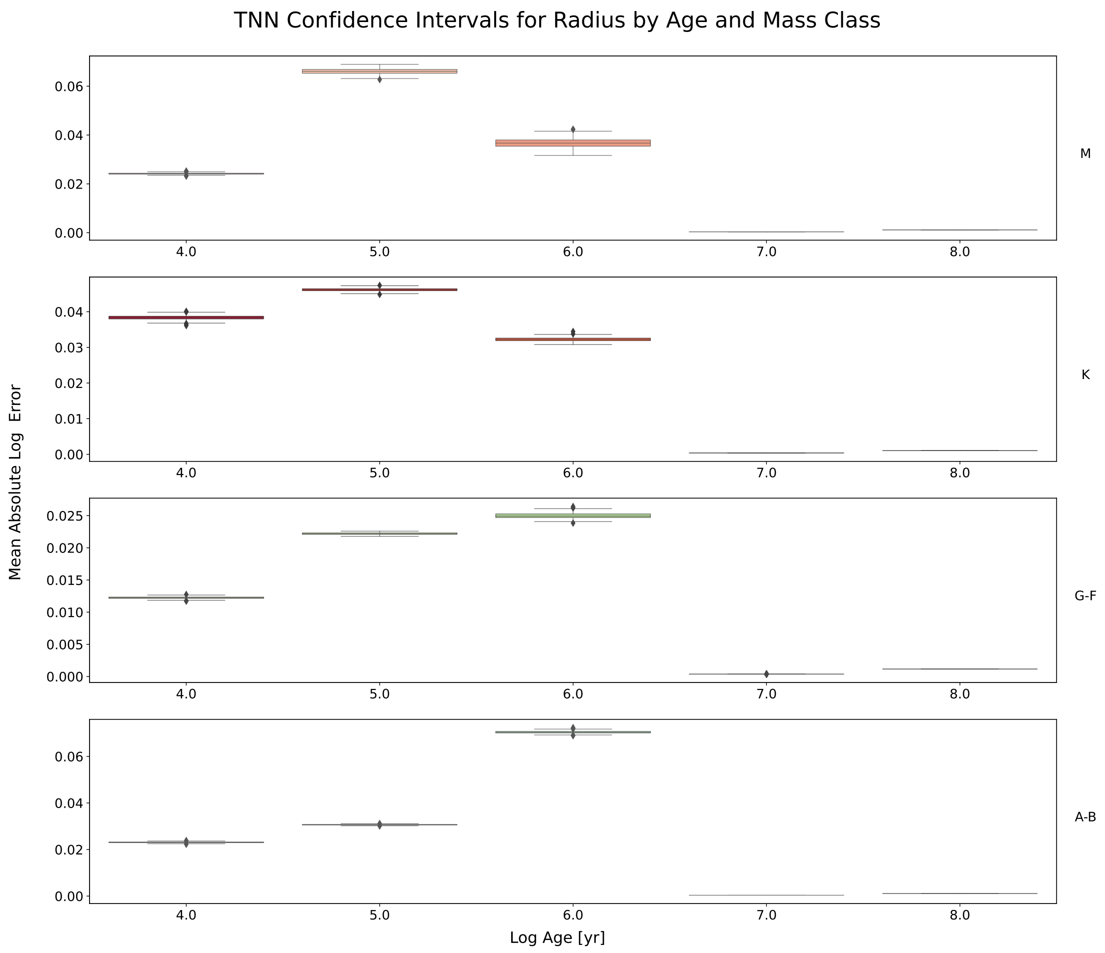

Figure 5.50: The confidence intervals of radius, grouped by star class and age brackets, are shown in the figure. These intervals were derived using the absolute log error in the TNN predictions on the test set. The y-axis shows the age brackets, with the last number representing the upper limit of each bracket. For instance, the first bracket represented by 4 corresponds to stars with ages between $10^{3.3}$ and 10^4 years.

The confidence intervals concerning the predictions of the radius as seen in figure 5.50, show a similar pattern to the L_s case, with regards to poor performance in stars of types AB during ages $[10^5, 10^6]yr$, and stars of type M during ages $[10^4, 10^5]yr$. On the other hand, stars of class GF demonstrate the best performance, with the maximum $MALE$ at the end of the accretion phase being approximately 2.8 times smaller than the maximum errors observed in AB and GF stars. In this analysis, the best accuracy does not occur at the beginning of the test set for all stars. As observed, stars of type K display better errors at the end of the accretion phase than in the two previous age brackets. In the absence of accretion, we consistently observe the best results, as in the T_{eff} ans L_s cases.

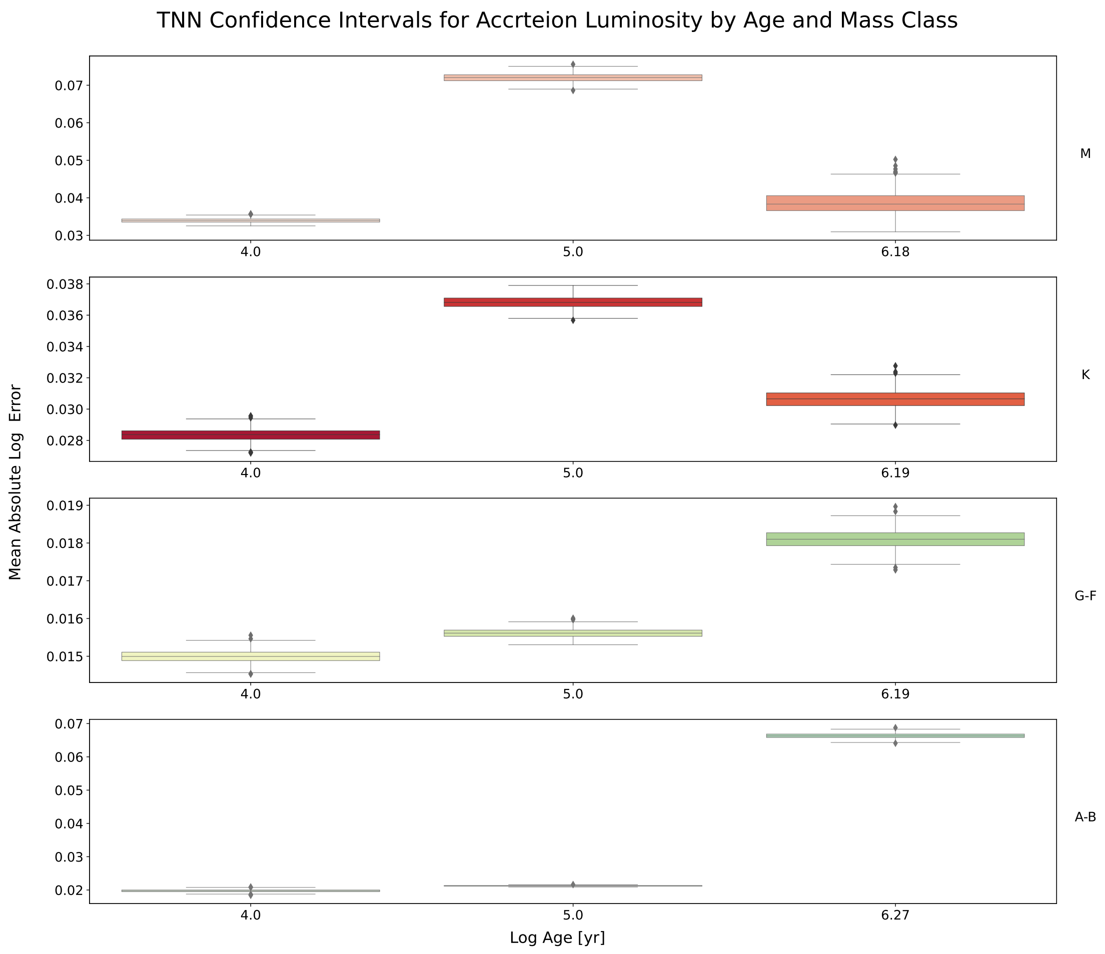

Figure 5.51: The confidence intervals of accretion luminosity, grouped by star class and age brackets, are shown in the figure. These intervals were derived using the absolute log error in the TNN predictions on the test set. The y-axis shows the age brackets, with the last number representing the upper limit of each bracket. For instance, the first bracket represented by 4 corresponds to stars with ages between $10^{3.3}$ and 10^4 years.

The final set of confidence intervals belong to the accretion luminosity predictions, displayed in figure 5.51. Similarly to the cases of L_s and R, we observe that stars of type M and AB exhibit the poorest performance, while stars of type GF demonstrate the best performance. Once again, it is evident that the most favorable results for all star classes are found within the age range of $[10^{3.3}, 10^4]yr$.

5.4.4 HR Diagram Analysis Temporal Neural Network

Up until now, the weaknesses, and strengths of the trained models in this experiment have been evident. Nevertheless, to assess the applicability of this models, it is necessary to examine the HR diagrams. The HR diagram analysis from the benchmark simulations is illustrated in figures 5.52, 5.53, 5.54, and 5.55.

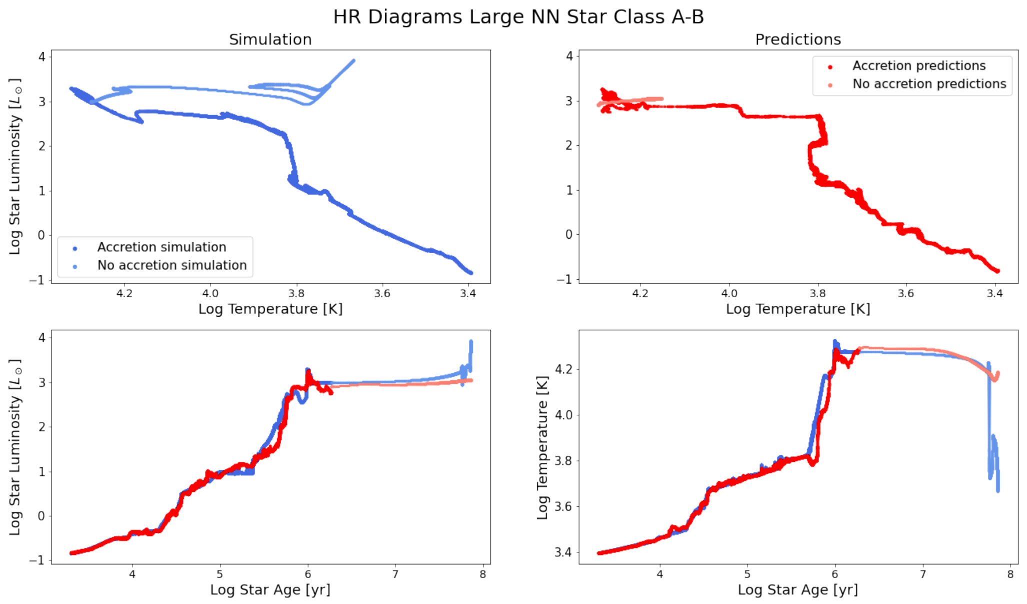

Figure 5.52: Test simulation of a star with final mass $M = 5.951M_\odot$. The top panels display HR diagrams, with the left-hand side representing the diagram obtained from MESA data and the right-hand side showing the one obtained through the TNN predictions. The bottom left-hand side presents the luminosity as a function of age, showing both the MESA values and the predictions for comparison. Similarly, the bottom right-hand side displays the temperature evolution of the star, with both MESA and predicted values overlaid for comparison.

The predictions for the benchmark AB star are shown in figure 5.52, which generally follows the HR simulation curve. However, in the luminosity as a function of age plot, we notice that although the predictions roughly follow the simulation curve up to the $10^5 yr$ mark, they deteriorate thereafter, failing to capture some of the interesting features present in the simulation. There's a similar behavior observed after $10^{5.5} yr$ in the temperature as a function of the age plot, where the simulation demonstrates a sharp temperature increase with a linear shape, but the predictions exhibit an erratic behavior. Due to the failure of both L_s and T_{eff} predictions within a similar age range, the HR diagram predictions fail to faithfully reproduce the features depicted by the simulation. In the same manner, as the previous AB HR diagrams seen in figures 5.17 and 5.31, the no accretion model demonstrates an inability to anticipate the temperature drop that occurs after the star depletes all the hydrogen in its core, even if it works somewhat better than the previous models.

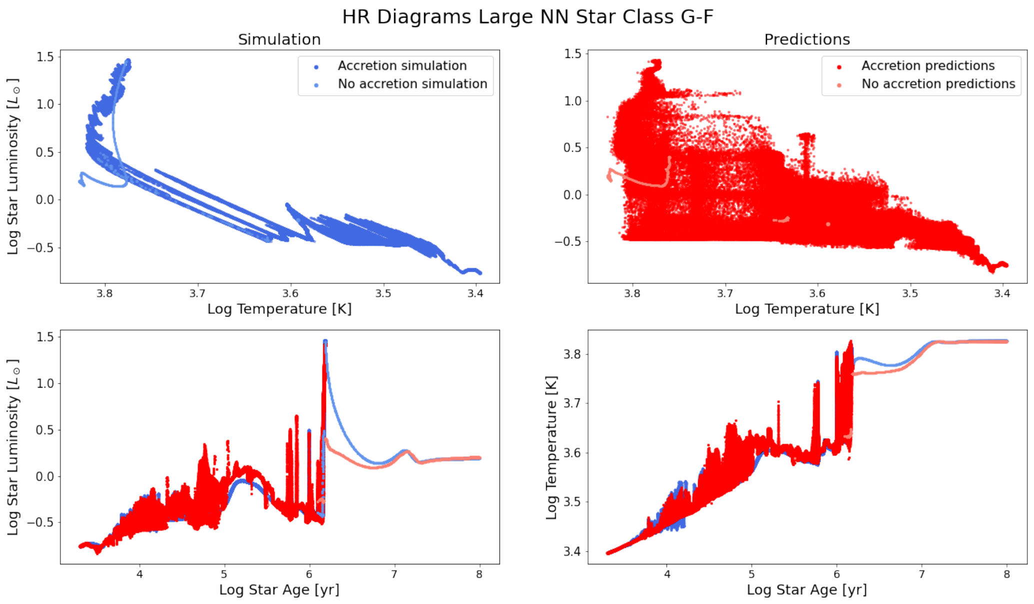

Figure 5.53: Test simulation of a star with final mass $M = 1.165 M_\odot$. The top panels display HR diagrams, with the left-hand side representing the diagram obtained from MESA data and the right-hand side showing the one obtained through the TNN predictions. The bottom left-hand side presents the luminosity as a function of age, showing both the MESA values and the predictions for comparison. Similarly, the bottom right-hand side displays the temperature evolution of the star, with both MESA and predicted values overlaid for comparison.

The HR predictions for the star GF, displayed in figure 5.53, demonstrate a notably erratic pattern. Although the precise cause behind this erratic behavior is not apparent, it seems that the errors occurring towards the end of the temperature against age curve, within the range of $10^{3.6}K$ to $10^{3.8}K$, are correlated with errors in the luminosity at the same age. Specifically, there is a sharp increase in luminosity within the range of $[10^{-0.4}, 10^{1.5}]L_\odot$, which leads to less satisfactory HR predictions. Regarding the no accretion section, it becomes apparent that the error originates from the luminosity predictions. At the beginning of this phase, the luminosity predictions are lower than they should be, leading to a discrepancy when combined with the temperature values on the Hertzsprung-Russell diagram.

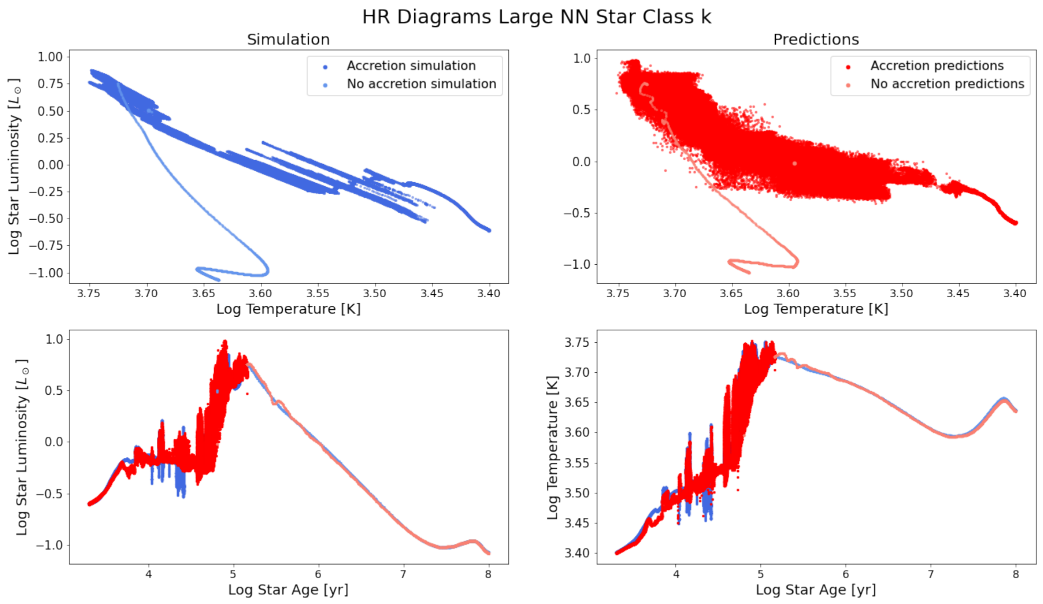

Figure 5.54: Test simulation of a star with final mass $M = 0.613 M_\odot$. The top panels display HR diagrams, with the left-hand side representing the diagram obtained from MESA data and the right-hand side showing the one obtained through the TNN predictions. The bottom left-hand side presents the luminosity as a function of age, showing both the MESA values and the predictions for comparison. Similarly, the bottom right-hand side displays the temperature evolution of the star, with both MESA and predicted values overlaid for comparison.

Regarding the predictions of the benchmark K star, as seen in figure 5.55, there is a noticeable widening of the HR curve. This issue becomes evident when examining the individual temperature and luminosity curves, as the combination of errors leads to a loss of resolution, resulting in the broadening of the sharp lines present in the simulation HR diagram. Moreover, discrepancies are observed in both curves, a clear example of this happens between $10^{4.25} yr$ and $10^{4.5} yr$, where the predictions fail to accurately capture the decrease in temperature and luminosity after present after a sharp increase of these. In this star, the predictions during the accretion phase effectively follow the HR simulation curve.

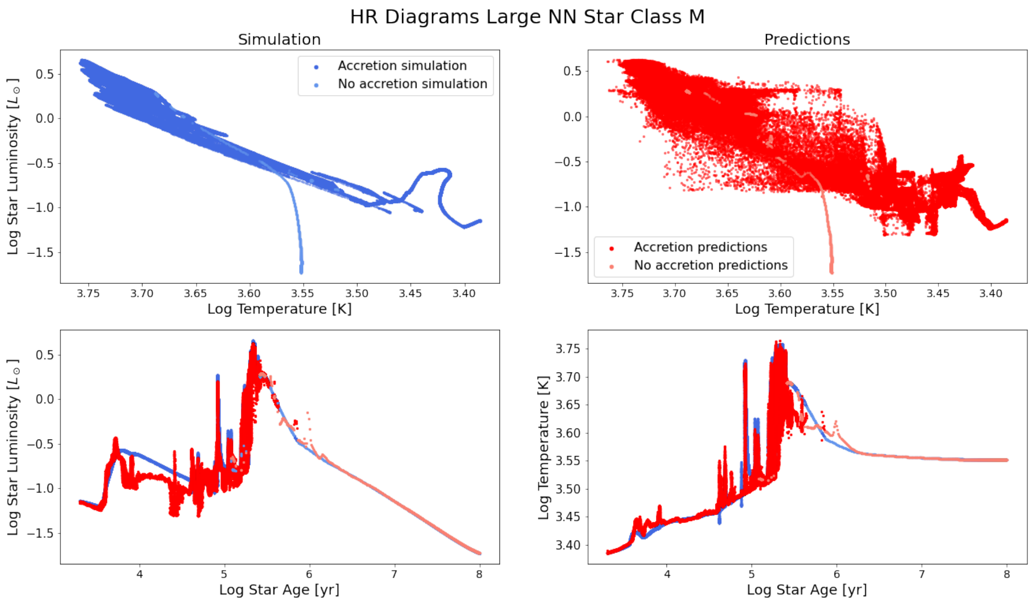

Figure 5.55: Test simulation of a star with final mass $M = 0.358M_\odot$. The top panels display HR diagrams, with the left-hand side representing the diagram obtained from MESA data and the right-hand side showing the one obtained through the TNN predictions. The bottom left-hand side presents the luminosity as a function of age, showing both the MESA values and the predictions for comparison. Similarly, the bottom right-hand side displays the temperature evolution of the star, with both MESA and predicted values overlaid for comparison.

In figure 5.55, the predictions for the M star exhibit highly inaccurate luminosity estimations, particularly within the age range of $[10^{3.7}, 10^{4.5}]yr$. In this range, the predicted luminosity is shifted downward and displays complex patterns that are not present in the simulation. Moreover, the HR diagram figure clearly illustrates the issue of combined errors. For instance, if we focus on the region with $T_{eff} \approx 10^{3.55}K$ and $L_s \approx 10^{0.25}L_\odot$, right after 10^5yr, we can observe a sharp and wide increase in T_{eff} and L_s. Although not immediately apparent, it is possible to deduce that an erroneous prediction in this specific region for both parameters can lead to the pronounced widening seen in the predictions HR diagram. The predictions for the later evolution of this star effectively follow the simulation curve, exhibiting a few exceptions that are particularly evident in the temperature against age plot, where the curve displays some erratic patterns at around 10^6yr that are not present in the simulation.
In addition to this, a notable aspect observed in these results is the presence of a discontinuity on both the temperature and luminosity curves. This discontinuity arises from the decrease in accretion mass during this particular stage of the star.

5.4.5 Hyper Parameter Importance Temporal Neural Network

The final aspect of this experiment involves evaluating the importance of parameters, as described in section 4.8.3, and the results are depicted in figure 5.56.

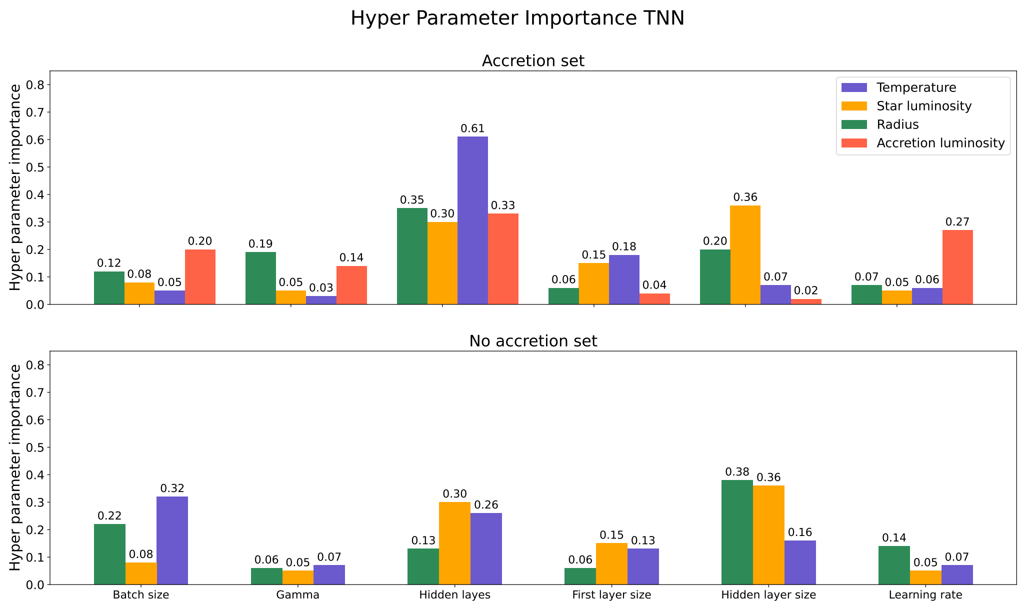

Figure 5.56: The importance of hyperparameters for temperature, star luminosity, radius, and accretion luminosity TNN models. The top panel displays the parameter importance of accretion models, while the bottom panel represents the parameter importance for no accretion models.

Figure 5.56 indicates that the most critical factors are the hidden layers and their size. Moreover, the batch size, specifically concerning the radius and temperature, plays a crucial role in the no-accretion models. It is worth noting that the learning rate holds relatively low importance across all cases, except for the accretion luminosity prediction, where it becomes the second most important factor.

Chapter 6

Models Comparison and Discussion

The models were compared using the error metrics $MedALE$ and $MaxMedALE$, along with an analysis of the HR diagram. Additionally, the DM97 model was included in the comparison to assess how well the obtained models matched an existing model, which was a primary goal of the thesis. It's worth noting that the tables used in the DM97 model do not cover all possible mass and age ranges present in the test data. Consequently, approximately 9.8% of the data points were excluded for this model. However, the error metrics could still be applied to the DM97 model for the existing data points. Similarly, the HR diagrams, though not complete, provided insights into how our models compared to the DM97 model.

6.1 Machine Learning Models Comparison

To choose the best model from those obtained in the experiment, a comparison was done using the familiar error metrics, namely $MedALE$ and $MaxMedALE$. Additionally, the DM97 model was compared with the models obtained in this project, where star luminosity and temperature were interpolated. It is worth noting that while one could argue that R and L_{acc} can be obtained for the DM97 model using equations 2.1 and 2.12 respectively, we decided to compare the predictive capabilities of the statistical models for the specific desired parameter. Hence, the calculations were not performed, and the comparison was made using the parameters obtained directly from the models. The results of this analysis can be seen in figures 6.1, and 6.2 for the accretion set and no accretion set respectively.

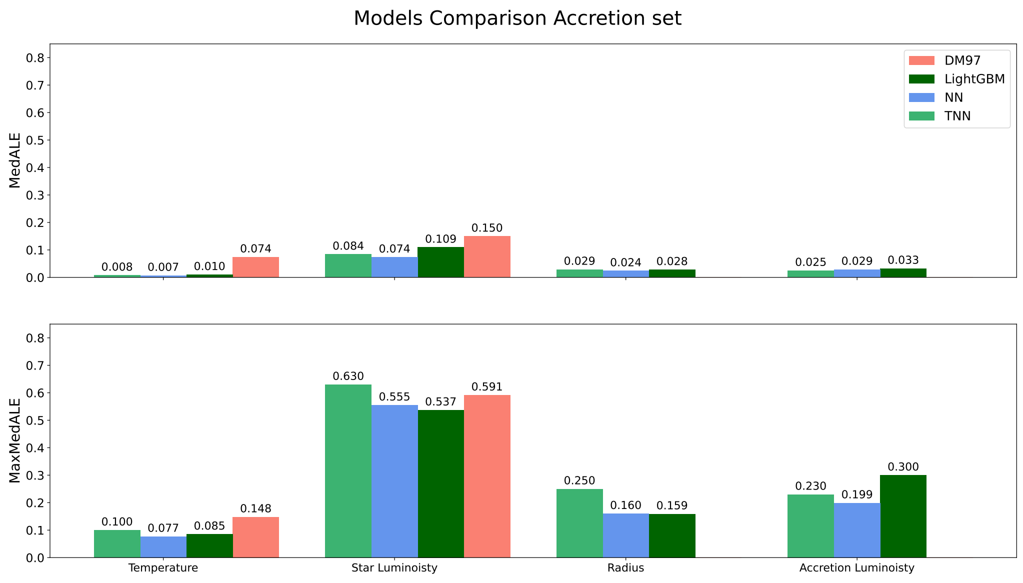

Figure 6.1: The comparison of the accretion models for temperature, star luminosity, radius, and accretion luminosity is presented in the figure. The top panel displays the median absolute log error, while the bottom panel shows the maximum median absolute log error.

Figure 6.1 shows the comparative results of various models in the accretion set. Overall, the NN experiment produced the best model, particularly in temperature prediction, where it outperformed DM97 by approximately 11 times in the $MedALE$ metric and about 2 times in the $MaxMedALE$ metric. Regarding the radius and star luminosity, the improvements achieved by the NN model compared to the other tests were not as remarkable, though still better in the $MedALE$ metric. However, in the $MaxMedALE$ metric, there is a slight advantage for the LightGBM model over the NN model, but the NN model is generally considered superior, as the $MedALE$ indicates that the model will produce better predictions overall. When it comes to accretion luminosity, the TNN model, as already demonstrated in section 5.4, performed the best. While the $MaxMedALE$ metric shows slightly better results for the NN model in this specific case, this difference is not significant enough to open a discussion about which model is the best in general.

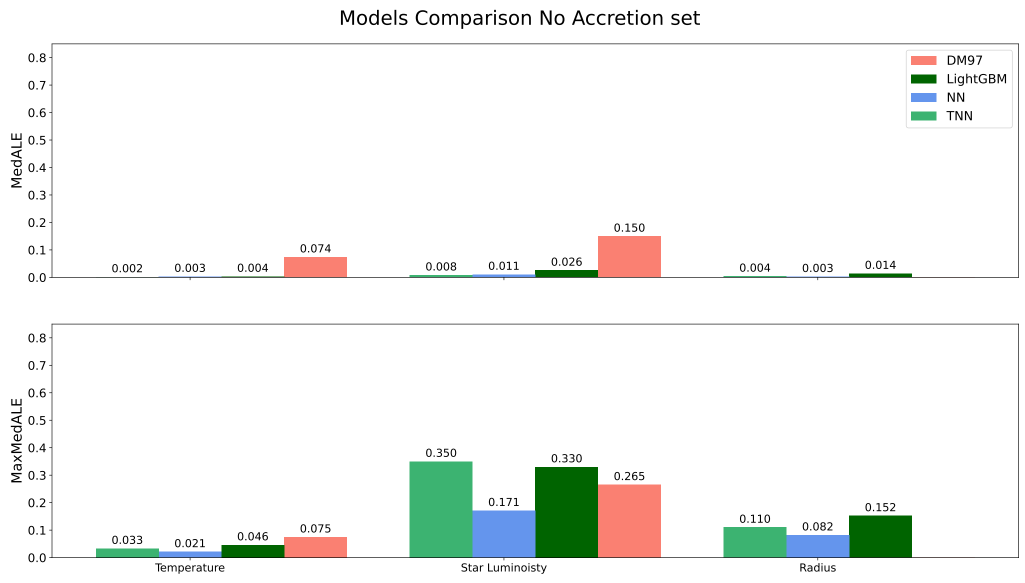

Figure 6.2: The comparison of the no accretion models for temperature, star luminosity, radius, and accretion luminosity is presented in the figure. The top panel displays the median absolute log error, while the bottom panel shows the maximum median absolute log error.

In the comparison of the no-accretion models, as seen in figure 6.2, the top-performing models for temperature and star luminosity are those obtained in the TNN experiment according to the $MEdALE$ metric. When comparing these results with the DM97 model, we observe an improvement of approximately 97% and 94%, respectively. However, for the $MaxMedALE$ metric, the NN model performs the best for these two parameters. The difference in temperature prediction between the NN and TNN models is not significant, while for star luminosity, the NN model is approximately twice as good. If we take into consideration the results from the $MedALE$ metric, where the difference between the NN and TNN models is only 0.003, it becomes apparent that the NN model stands out as the better choice for the star luminosity. As for the radius prediction, both the $MedALE$ and $MaxMedALE$ metrics indicate that the NN model is the superior one.

6.2 HR Diagram Comparison

As observed in the previous sections, judging the models only on error metrics is insufficient to fully evaluate the models performance. The combination of errors can sometimes lead to erratic behavior in the context of the HR diagram. In addition to this the resolution of the HR diagram is one of the factors that the errors metric fail to quantify. Therefore, in this section, we present the HR diagrams of the benchmark simulations obtained from all the models discussed in this thesis, including the DM97 model. As mentioned before the DM97 model does not include the entire test dataset, making it challenging to assess its true performance. Nevertheless, it remains an interesting aspect for analysis. The results of this HR diagram comparison are depicted in figures 6.3, 6.4, 6.5, and 6.6.

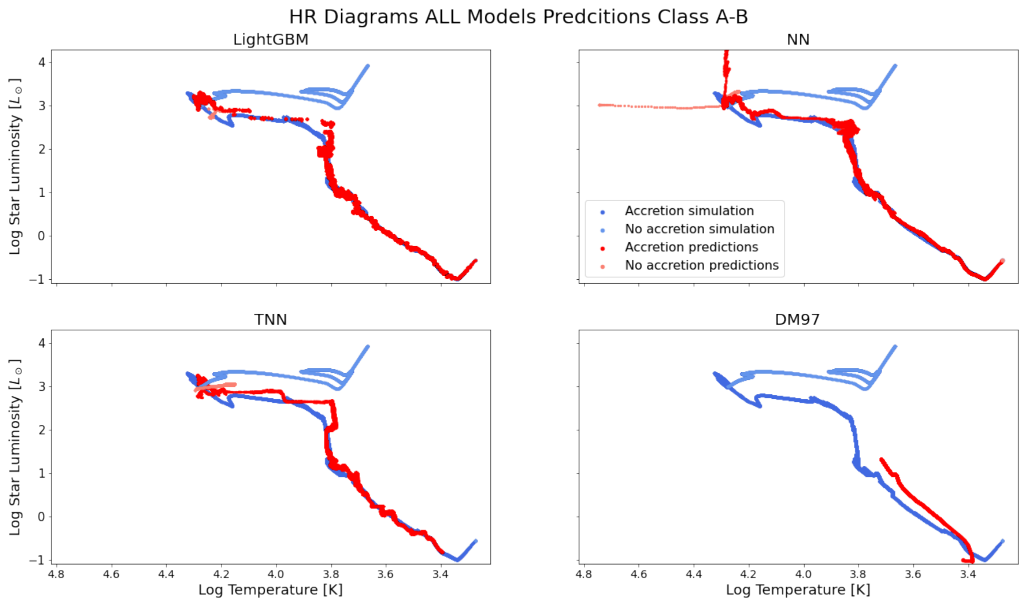

Figure 6.3: The HR diagrams of the *AB* benchmark star are displayed with an overlay of MESA values and model predictions. The top left panel corresponds to the HR diagram obtained using LightGBM predictions. The top right-hand side panel displays the HR diagram from the NN model predictions. Moving to the bottom left-hand side panel, it shows the HR diagram from the TNN model predictions. Finally, the bottom right-hand side panel presents the HR diagram from the DM97 inference.

Figure 6.3 shows the HR diagram comparison for the *AB* simulation. It is evident that the predictions from the NN experiment in the accretion section exhibit greater resolution than the other models, furthermore, it is the more consistent with regards to the simulation. However, all predictions in the no accretion section fail to provide accurate results. As for the DM97 model, it only replicates a small section of the HR curve, making it hard to draw any conclusive remarks.

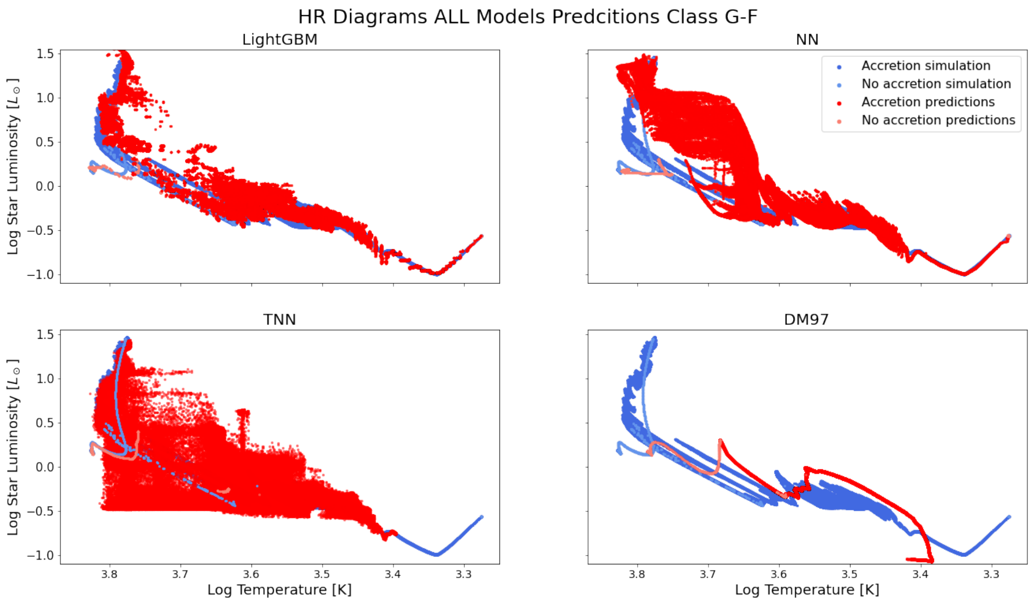

Figure 6.4: The HR diagrams of the *GF* benchmark star are displayed with an overlay of MESA values and model predictions. The top left panel corresponds to the HR diagram obtained using LightGBM predictions. The top right-hand side panel displays the HR diagram from the NN model predictions. Moving to the bottom left-hand side panel, it shows the HR diagram from the TNN model predictions. Finally, the bottom right-hand side panel presents the HR diagram from the DM97 inference.

For the *GF* star, as shown in figure 6.4, the NN accretion predictions once again exhibit a higher level of resolution compared to other ML models. While it doesn't precisely capture the temperature and luminosity behavior, it does indicate the presence of fluctuations caused by accretion. The LightGBM model, despite producing results with low resolution, manages to predict the overall structure better than all other models. On the contrary, the DM97 model reproduces a smooth curve with no clear signs of fluctuations due to accretion, which can be misleading for this specific star. Such behavior is expected from this type of model, as it doesn't account for the accretion rate in its predictions. Regarding the no accretion section, the best ML model is the one obtained in the TNN experiment, as it provides better resolution than the LightGBM model and produces a smooth curve unlike the NN model. As for the DM97 no accretion set inference, it does reproduce a smooth curve with good resolution but with the wrong temperature, causing it to be shifted to the left in the diagram.

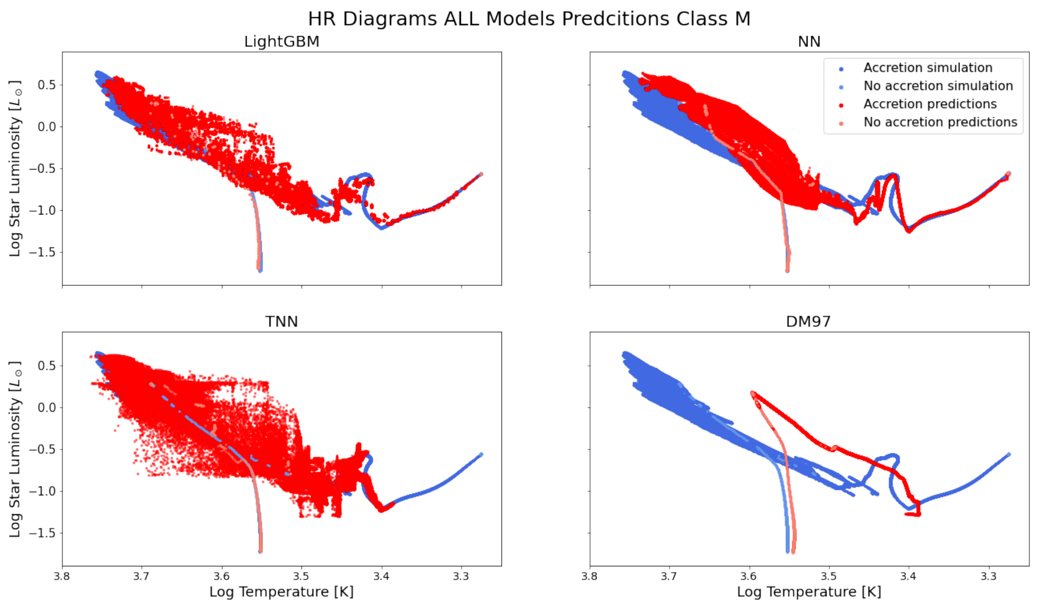

Figure 6.5: The HR diagrams of the M benchmark star are displayed with an overlay of MESA values and model predictions. The top left panel corresponds to the HR diagram obtained using LightGBM predictions. The top right-hand side panel displays the HR diagram from the NN model predictions. Moving to the bottom left-hand side panel, it shows the HR diagram from the TNN model predictions. Finally, the bottom right-hand side panel presents the HR diagram from the DM97 inference.

Figure 6.5 shows the HR curves of the M star. In this figure, it becomes evident that the LightGBM model reproduces the star's evolution more accurately than other ML models. However, the trade-off is the lack of resolution in this model, which is notably high. In terms of resolution, the NN model stands out as the best among all ML models. Nevertheless, it exhibits a shift upwards at the end of the diagram, resulting from a luminosity shift. For the no accretion set, the NN model reproduces the most accurate curve compared to all other models. As with previous stars, the DM97 model once again generates a featureless curve, ignoring the fluctuations in luminosity and temperature.

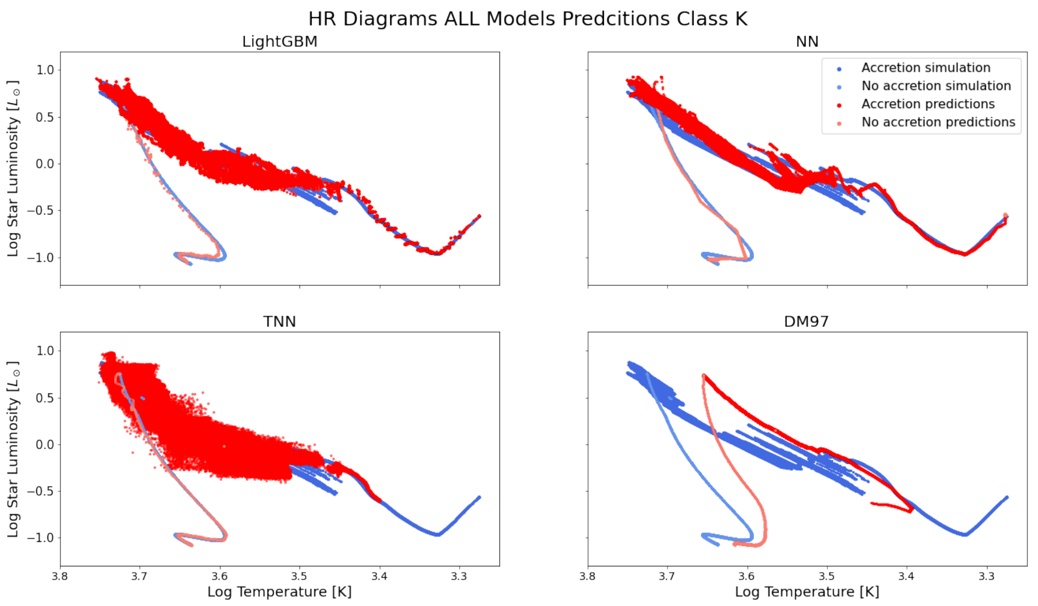

Figure 6.6: The HR diagrams of the K benchmark star are displayed with an overlay of MESA values and model predictions. The top left panel corresponds to the HR diagram obtained using LightGBM predictions. The top right-hand side panel displays the HR diagram from the NN model predictions. Moving to the bottom left-hand side panel, it shows the HR diagram from the TNN model predictions. Finally, the bottom right-hand side panel presents the HR diagram from the DM97 inference.

The final star is the K benchmark star depicted in figure 6.6, where we can see that the best model predictions are the ones obtained with the NN model as it reproduces the overall shape and has the best resolution. Regarding the no accretion section, the TNN model performs the best in accurately reproducing the curve. As for the DM97 model, while it roughly follows the shape of the HR curve, the results are shifted towards lower temperatures, and the characteristic absence of intricate structures is quite noticeable.

6.3 Data

In sections 5.2.2, 5.3.2, and 5.4.2, we observe a consistent occurrence of high errors in a specific simulation namely the AB benchmark simulation. This outcome was anticipated due to the findings depicted in figure 5.4. The figure illustrates that only 15% of the simulations fall into this particular category. Furthermore, the AB star is significantly advanced beyond the hydrogen-burning phase, and no other star in the data set has reached this stage of evolution. This result is expected, as supervised machine learning models learn from examples, and since there are no comparable examples in the data set, the model faces challenges in handling such cases.
Another crucial point to consider is that the performance of the balanced dataset consistently outperformed both the complete set and the reduced set, despite being consistently smaller in size. This serves as confirmation that the widely accepted practice of balancing data sets in machine learning is effective.

6.4 Confidence Intervals

Regarding the confidence intervals depicted in figures 5.12, 5.26, and 5.47, it is intriguing to observe that the outcomes for T_{eff}, L_s, and R exhibit a consistent pattern described by equation 2.1. Where the error propagation formula given by equation 6.1 approximately aligns with the obtained results.

$$\left(\frac{\delta L}{L}\right)^2 = 2\left(\frac{\delta R}{R}\right)^2 + 4\left(\frac{\delta T}{T}\right)^2 \tag{6.1}$$

This result is interesting since the predictions for each of these parameters are made independently. Despite this independence, it appears that the relationship described by equation 2.1 is maintained.

With respect to the confidence intervals based on age and class, the temperature $MALE$ distributions in figures 5.13, 5.27, and 5.48 exhibit a consistent pattern across the models. Notably, the predictions are least accurate for ages between 10^5 and 10^6 years in the accretion set, with classes AB and GF consistently scoring the poorest results. This could potentially be attributed to insufficient data for these types of stars, as depicted in figure 5.4. Additionally, it is worth noting that the M stars perform best across all three models, confirming the idea that the amount of data plays a crucial role in accuracy. Supporting the findings in section 6.1 the no accretion set, shows a remarkable improvement observed in the TNN experiment compared to the previous two models.
Analyzing the figures 5.14, 5.28, and 5.49, we observe the accuracy of star luminosity concerning age and class. A consistent pattern emerges, indicating a decline in model accuracy within the age range of $[10^5, 10^6]$ years across all three experiments. Notably, all star classes display comparable errors, except in the NN case where the GF and K stars exhibit superior performance. This result contradicts the hypothesis that the quantity of data was the cause of the significant errors observed.
In the predictions of radius based on class and age, as seen in figures 5.15, 5.29, and 5.50, we consistently observe poor accuracy within the age range of $[10^5, 10^6]$ years. However, in the case of LightGBM and NN experiments, we notice a higher variance in the first few hundred years of the stars. In the accretion section, the results demonstrate remarkable consistency across all experiments, revealing the most substantial errors in the AB stars and comparatively lower errors in the K stars. This finding challenges the hypothesis that the increase in errors is primarily due to a lack of data. Notably, these two classes have very similar percentages of data, specifically around 15% and 16% of the simulations, respectively.
Lastly, the analysis of accretion luminosity is presented in figures 5.16, 5.30, and 5.51. These figures reveal inconsistent results across all models, with the NN predictions displaying the highest level of inconsistency between the models, but relatively higher consistency within its own predictions per class mass. In terms of the LightGBM model and the NN, the GF class exhibits the best accuracy, followed by the K class, and lastly, the AB and M classes.

6.5 Hyper Parameter Importance

As the LightGBM Model and the neural network ones differ from each other it is not possible to directly compare them. However, the NN and TNN models can be compared, as a significant change was made in the architecture, specifically an increase in the `Number of Hidden Layers`. Although the results in section 5.3.6 did not show any significant improvements by increasing the `Number of Hidden Layers`, the optimization algorithm's range for the `Number of Hidden Layers` was expanded to allow broader exploration. By doing this, the no accretion set in the TNN experiment improved its performance in terms of the error metric and the smoothness of the curve, as discussed in sections 6.1 and 6.2. Moreover, it is interesting to note that the importance of, both `Number of Hidden Layers` and `Hidden Layer Size`, in the TNN experiment, increased compared to the other factors. An intriguing aspect to note in the analysis of figure 5.35 is the significant inconsistency in results between the accretion and no accretion sections. However, in figures 5.56, we can observe a better agreement between these two sections. This suggests that increasing the number of parameters in the models, in addition to increasing accuracy, strengthens the consistency between parameter importance.

Chapter 7

Integration with Star Formation Models

The obtained models were published in the `https://github.com/almazagit1002/Stellar-Parameters-RMCE-ML.git` GitHub repository, as unformatted sequential files. In addition to this, an implementation of the NN predictions was published in Python and Fortran. Moreover, in the attempt to make this work available for any implementation, a pseudo code along with a brief description of its functionality is outlined in the section 7.1. To evaluate the functionality, one can use a reference input vector $[1866331.3016338805, 5.95185857, 5.196092096677678e{-}08]$, representing age, mass, and accretion rate. The resulting output would be $[0.2260622624173494, 4.076734103841037, 4.287999266204293, 0.3875$ which corresponds to the logarithmic values of radius, stellar luminosity, temperature, and accretion luminosity, respectively.

7.1 Pseudo Code

In order to implement the NN inference, the code takes an array (`input_vector`) with the familiar input parameters [age,M,$\dot{M}$], and units [$yr, M_{\odot}, M_{\odot}yr^{-1}$]. The first step is to transform the `input_vector` into the necessary format for the neural network. In order to achieve this the `INPUT_MANIPULATION` function is used.
`INPUT_MANIPULATION` compute the logarithms of the star age, and the accretion luminosity when necessary, and scaling the `input_vector` using the `MIN_MAX_SCALER` function described in equation 4.1.

```
 FUNCTION INPUT_MANIPULATION(input_vector:array)-> input_vector:array,
acc_type:string, pred_params:array, len_pred_params:int
    input_vector[0] <- log10(input_vector[0])
    acc_type <- 'Acc'
    IF input_vector[2] < 1e-40 THEN
        input_vector <- input_vector[:2]
        acc_type <- 'N_acc'
        pred_params <- ['Radius', 'Lum', 'T_eff']
        len_pred_params <- length(pred_params)
    ELSE
        input_vector[2] <- log10(input_vector[2])
        PRED_PARAMS <- ['Radius', 'Lum', 'T_eff', 'Acc_Lum']
        len_pred_params <- length(pred_params)
    END IF
    input_vector <- MIN_MAX_SCALER(input_vector, acc_type)
    RETURN input_vector, acc_type, pred_params, len_pred_params
 FUNCTION MIN_MAX_SCALER(input_vector:array, acc_type:string)
-> scaled_vec:array
    mins <- array([-4.1981, 0.01, -99])
    maxs <- array([8.0, 10.567092, -3.7644])
    IF acc_type == 'N_acc' THEN
        mins <- mins[:2]
        maxs <- maxs[:2]
    END IF
    scaled_vec <- (input_vector - mins) / (maxs - mins)
    RETURN scaled_vec
```

Once the `input_vector` values have been transformed, is possible to make the predictions using `PREDICT_STELLAR_PROPERTIES`. This function takes the transformed `input_vector`, and iterates over `pred_params`, in order to read the necessary NN weights, bias, and architecture, using the function `NEURAL_NETWORK_`. At this stage, the `FORWARD_PASS` described in equation 3.4, takes the NN parameters and the transformed `input_vector`, in order to return the predicted output. Finally, the `INVERSE_MIN_MAX_SCALING` is applied when necessary.

```
 FUNCTION PREDICT_STELLAR_PROPERTIES(input_vector:array, acc_type:string,
pred_params:array, len_pred_params:int) -> predictions:array
    predictions <- array of zeros with length len_pred_params
    FOR i IN range(len_pred_params):
        layers, weights, bias <-NEURAL_NETWORK_(pred_params[i], acc_type)
        pred <- FORWARD_PASS(input_vector, layers, weights, bias)
        predictions[i] <- pred
    IF len_pred_params > 3 THEN
        predictions[3] <-
        INVERSE_MIN_MAX_SCALING(predictions[3])
    END IF
    RETURN predictions

FUNCTION NEURAL_NETWORK_(pred_param:string, acc_type:string) -> layers:int,
weights:array, bias:array
     file <- read(f'path_to_records/{pred_param}_{acc_type}_records.unf')
     layers <- read_ints(int32)
     Shapes <- read_ints(int32).reshape(layers, 2)
     weights <- create empty array with size layers
     bias <- create empty array with size layers
     FOR i IN range(layers):
         weights[i] = read_reals(float).reshape((Shapes[i][0], Shapes[i][1]))
     FOR i IN range(layers):
         bias[i] =read_reals(float)
     file.close()
     RETURN layers, weights, bias

FUNCTION RELU(x:array) -> array
     RETURN max(0, x)


FUNCTION FORWARD_PASS(input_vector:array, layers:int, weights:array,
bias:array) -> output:float
```

```
    FOR i IN range(layers-1):
        Layer_pass <- RELU(matrix_multiplication(input_vector,
       weights[i])+bias[i])
        input_vector <- Layer_pass
    output <- matrix_multiplication(input_vector,weights[-1]) + bias[-1]
    RETURN output

FUNCTION INVERSE_MIN_MAX_SCALING(scaled_val:float)
-> inverse_scaled:float
    min_value <- -99.0
    max_value <- 3.170073
    inverse_scaled <- scaled_val * (max_value - min_value) + min_value
    RETURN inverse_scaled
```

The mentioned code can be run in any programming language, as it doesn't require any external libraries. A possible section of the code where it is necessary to be particularly careful is the matrix multiplication in the `FORWARD_PASS` function. In order to be able to perform the matrix multiplication correctly, one needs to be sure that the `weights` and `bias` have the right shape when they are read from the `.unf` files.

Chapter 8

Future Work

So far, we have explored the usefulness of using machine learning techniques to predict stellar parameters within a realistic cloud environment. These predictions have provided valuable insights into the star's evolution, allowing for seamless integration with existing software to gain a comprehensive understanding of the stellar structure. Nevertheless, it is essential to recognize that the current predictions are not flawless, and there are several aspects, ranging from the training data to the model architecture, that offer opportunities for improvement. In this section, we present some suggestions aimed at enhancing the accuracy of the models and expanding their capability to effectively predict a broader range of star types.

8.1 Machine Learning Architectures

Although our attempt to introduce temporal structure into our data, in the TNN model, did not show any significant improvement compared to the NN or LightGBM models, there is evidence suggesting that including such information could be beneficial. A clear example of this can be observed in the luminosity as a function of age plot for the GF star, shown in figures 5.18, 5.32, and 5.53. At the boundary between the accretion and no accretion sections in these plots, there is a noticeable discontinuity, which indicates that some prior knowledge of the stellar parameters could be valuable in this scenario. Hence, investigating ML architectures such as transformers might offer a solution to this issue.
Transformers, which incorporate an attention mechanism, were first introduced in 2017 (Vaswani et al., 2017) [61] and have since proven highly effective, forming the foundation for large language models like GPT-3. The attention mechanism works by allowing the model to focus on different parts of the input sequence during processing. For each parameter in a state vector, the model calculates its importance or relevance with respect to all other parameters in the state vector. This attention mechanism enables the model to efficiently capture long-range dependencies and relevant relationships within the input data, making it effective in understanding sequential information. Such an attention mechanism could be beneficial in order to predict the complicated fluctuations in the parameters given by stellar evolution.
During this project, another option that received limited exploration involved creating a two-neural network system. In this approach, the first neural network is a simpler system, similar to the ones investigated in this thesis, while the second neural network takes predictions from the first network alongside the regular parameters to enhance the predictions. In this manner, the second NN in the system has an initial starting point to work with. However, it is crucial to note that such an architecture could potentially lead to more significant errors, as any inaccuracies in the first NN layer might propagate and worsen in the subsequent stages, potentially spiraling out of control.

A more straightforward alternative involves investigating and improving the state vectors provided to the TNN. As discussed in section 4.1.6, these state vectors are currently composed of data from around 10, 100, and 1000 years prior to the target point. In a more comprehensive study, additional temporal information could be considered, either within the same range or by extending the time window further.

There exists a disparity between the error metrics employed for model selection and the loss function integrated into the ML architectures. The primary accuracy source for model selection is the $MedALE$, while the ML algorithms loss function is the $MALE$. Referencing appendix B, it is apparent that the ML validation scores do not always align with the inference accuracy in the test set found in appendix A. Furthermore, the utilization of distinct metrics introduces complexity to maintaining a coherent methodology. Aligning the error metrics is then the most effective approach moving forward.

Finally, a simple improvement to the ML architecture could involve the incorporation of a dropout layer. This dropout layer randomly deactivates a specified proportion of neurons in the preceding layer during training. Consequently, for each training example, a distinct set of neurons is dropped out, introducing randomness into the network's computations. The primary benefit of dropout is its ability to mitigate overfitting, where the model becomes excessively specialized to the training data, leading to poor performance on new, unseen data [62].

8.2 Data

In this thesis, the ML models were trained using data from stars up to approximately $10M_\odot$. To broaden the predictive range and include more massive stars, it becomes crucial to retrain new models with stars having masses $M > 10_\odot$. As our current data set is constrained in the high mass range, and it lacks any O stars.
Additionally, it would be beneficial to train machine learning models using data from each category of stars with an equal percentage representation. In section 6, we have discussed the hypothesis that more data is required for certain star classes. While there is not enough evidence to fully support this hypothesis, there is also no evidence to refute it. Furthermore, in this thesis, star classes B, A, F, and G were grouped into AB and GF classes to simplify the analysis. However, an ideal training data set should consist of approximately equal amounts of star simulations across all categories. This balanced representation might enhance the model's performance and accuracy in predicting star properties across the entire stellar mass spectrum.

We have employed an effective ML technique by balancing the data set. However, there are other widely used techniques for imbalanced data sets that involve weighted loss functions. With this approach, higher weights are assigned to the minority class, while lower weights are assigned to the majority class during the training process. As a consequence, the model focuses more on the minority class, aiming to minimize errors in those instances [63]. Adopting this approach in training a ML model has the potential to improve generalization and, consequently, the overall accuracy.

Chapter 9

Conclusions

In this thesis, we have explored the possibility of employing machine learning techniques to predict stellar parameters such as temperature, stellar luminosity, radius, and accretion luminosity based on a given mass, age, and accretion rate within a realistic molecular cloud environment. We used data obtained from RAMSES and MESA, aiming to modernize outdated methods that only consider mass and age, like the DM97. Our research involved comparing the results of various machine learning models developed in this study with each other and with the DM97 model. All the models created during this research are publicly available on GitHub, along with an implementation in Python and Fortran that can be easily integrated into stellar evolution software, such as RAMSES. Additionally, we provided a pseudocode along with a description of its functionality for easy implementation in any programming language.

Furthermore, we developed a methodology for comparing different machine learning models. The study revealed that accurately predicting stellar evolution with intense accretion rates is indeed challenging. Consequently, we provided insights for improving the accuracy of ML models, suggesting various options, ranging from data preparation to machine learning architecture. We find that the Neural Network model performed the best according to the error metrics. The temporal neural network showed some potential but the predictions had a large degree of fuzzyness due to the error combination between temperature and stellar luminosity. In terms of star types the most accurate results were found for low stellar masses, where also most training data is available. We speculate, that if significantly larger data, within particular higher mass stars, could be created, the resulting networks based on such data may perform better.

In conclusion, machine learning methods prove to be powerful tools for inferring stellar structure parameters. The results surpass those obtained from the DM97 model, especially in detecting complicated fluctuations caused by the accretion rate. However, while the results are promising, they are not flawless, and further work is needed to create a model that can better align with MESA's results. Despite the fact that our predictions did not precisely capture the complex fluctuations in temperature and luminosity seen in the HR diagrams, they still provide valuable insights into star evolution, revealing areas of interest. These findings serve as good indicators of important regions to explore further. Finally, the integration of these methods into existing codes like RAMSES was achieved, resulting in a more robust and complete system.

9.1 Acknowledgment

I want to express my gratitude to my supervisor, Troels Haugbølle, whose encouragement has been a constant source of motivation. His keen attention to detail has compelled me to dive deeper into areas of interest, while our discussions have helped me to structure my own thinking process in order to advance this research. Moreover, our engaging one-on-one conversations about star evolution have undoubtedly enriched my understanding of these concepts. As I conclude this project, I do so with a broader and more profound comprehension of both machine learning methods and star evolution, thanks to his guidance and our stimulating discussions.

Appendix A

Appendix Test Results

A.1 LightGBM Models Results

A.1.1 Accretion

Error Analysis LightGBM Temperature Accretion Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.009	0.10
Reduced	0.017	0.13
Complete	0.010	0.09

Table A.1: $MedALE$ and $MedMaxALE$ for temperature predictions using the balanced and reduced data set with the LightGBM model.

Error Analysis LightGBM Temperature Mix Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.009	0.085
Reduced	0.018	0.14
Complete	0.010	0.1

Table A.2: $MedALE$ and $MedMaxALE$ for temperature predictions using the balanced and reduced data set with the LightGBM model.

Error Analysis LightGBM Star Luminosity Accretion Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.098	0.59
Reduced	0.109	0.54
Complete	0.100	0.60

Table A.3: $MedALE$ and $MedMaxALE$ for star luminosity predictions using the balanced and reduced data set with the LightGBM model.

Error Analysis LightGBM Star Luminosity Mix Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.094	0.67
Reduced	0.112	0.63
Complete	0.092	0.63

Table A.4: $MedALE$ and $MedMaxALE$ for star luminosity predictions using the balanced and reduced data set with the LightGBM model.

Error Analysis LightGBM Radius Accretion Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.028	0.2
Reduced	0.045	0.18
Complete	0.028	0.16

Table A.5: $MedALE$ and $MedMaxALE$ for radius predictions using the balanced and reduced data set with the LightGBM model.

Error Analysis LightGBM Radius Mix Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.033	0.24
Reduced	0.036	0.22
Complete	0.030	0.23

Table A.6: $MedALE$ and $MedMaxALE$ for radius predictions using the balanced and reduced data set with the LightGBM model.

Error Analysis LightGBM Accretion Luminosity		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.033	0.42
Reduced	0.052	1.72
Complete	0.031	1.50

Table A.7: $MedALE$ and $MedMaxALE$ for accretion luminosity predictions using the balanced and reduced data set with the LightGBM model.

Error Analysis LightGBM Accretion Luminosity Mix Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.031	0.40
Reduced	0.048	2.24
Complete	0.031	1.53

Table A.8: $MedALE$ and $MedMaxALE$ for accretion luminosity predictions using the balanced and reduced data set with the LightGBM model.

A.1.2 No Accretion

Error Analysis LightGBM Temperature No Accretion Set		
Data type	*MedALE*	*MedMaxALE*
No accretion	0.003	0.07

Table A.9: *MedALE* and *MedMaxALE* for temperature predictions using the balanced and reduced data set with the LightGBM model.

Error Analysis LightGBM Temperature Mix Set		
Data type	*MedALE*	*MedMaxALE*
Balanced	0.005	0.06
Reduced	0.018	0.09
Complete	0.010	0.06

Table A.10: *MedALE* and *MedMaxALE* for temperature predictions using the balanced and reduced data set with the LightGBM model.

Error Analysis LightGBM Star Luminosity No Accretion Set		
Data type	*MedALE*	*MedMaxALE*
No accretion	0.026	0.33

Table A.11: *MedALE* and *MedMaxALE* for star luminosity predictions using the balanced and reduced data set with the LightGBM model.

Error Analysis LightGBM Star Luminosity Mix Set		
Data type	*MedALE*	*MedMaxALE*
Balanced	0.037	0.45
Reduced	0.068	0.56
Complete	0.114	0.63

Table A.12: *MedALE* and *MedMaxALE* for star luminosity predictions using the balanced and reduced data set with the LightGBM model.

Error Analysis LightGBM Radius Accretion Set		
Data type	*MedALE*	*MedMaxALE*
No accretion	0.014	0.15

Table A.13: *MedALE* and *MedMaxALE* for radius predictions using the balanced and reduced data set with the LightGBM model.

Error Analysis LightGBM Radius Mix Set		
Data type	*MedALE*	*MedMaxALE*
Balanced	0.013	0.19
Reduced	0.013	0.16
Complete	0.041	0.30

Table A.14: *MedALE* and *MedMaxALE* for radius predictions using the balanced and reduced data set with the LightGBM model.

A.2 NN Models Results

A.2.1 Accretion set

Error Analysis NN Temperature Accretion Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.007	0.08
Reduced	0.017	0.22

Table A.15: $MedALE$ and $MedMaxALE$ for temperature predictions using the balanced and reduced data set with the NN model.

Error Analysis NN Temperature Mix Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.007	0.08
Reduced	0.016	0.93

Table A.16: $MedALE$ and $MedMaxALE$ for temperature predictions using the balanced and reduced data set with the NN model.

Error Analysis NN Star Luminosity Accretion Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.074	0.55
Reduced	0.110	0.67

Table A.17: $MedALE$ and $MedMaxALE$ for star luminosity predictions using the balanced and reduced data set with the NN model.

Error Analysis NN Star Luminosity Mix Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.078	0.58
Reduced	0.113	0.65

Table A.18: $MedALE$ and $MedMaxALE$ for star luminosity predictions using the balanced and reduced data set with the NN model.

Error Analysis NN Radius Accretion Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.024	0.16
Reduced	0.044	0.19

Table A.19: $MedALE$ and $MedMaxALE$ for radius predictions using the balanced and reduced data set with the NN model.

Error Analysis NN Radius Mix Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.028	0.18
Reduced	0.042	0.22

Table A.20: $MedALE$ and $MedMaxALE$ for radius predictions using the balanced and reduced data set with the NN model.

Error Analysis NN Accretion Luminosity		
Data type	*MedALE*	*MedMaxALE*
Balanced	0.034	0.22
Reduced	0.035	0.37

Table A.21: *MedALE* and *MedMaxALE* for accretion luminosity predictions using the balanced and reduced data set with the NN model.

Error Analysis NN Accretion Luminosity Mix Set		
Data type	*MedALE*	*MedMaxALE*
Balanced	0.032	0.31
Reduced	0.029	0.2

Table A.22: *MedALE* and *MedMaxALE* for accretion luminosity predictions using the balanced and reduced data set with the NN model.

A.2.2 No Accretion set

Error Analysis NN Temperature No Accretion Set		
Data type	*MedALE*	*MedMaxALE*
No accretion	0.003	0.02

Table A.23: *MedALE* and *MedMaxALE* for temperature predictions using the balanced and reduced data set with the NN model.

Error Analysis NN Temperature Mix Set		
Data type	*MedALE*	*MedMaxALE*
Balanced	0.006	0.051
Reduced	0.012	0.75

Table A.24: *MedALE* and *MedMaxALE* for temperature predictions using the balanced and reduced data set with the NN model.

Error Analysis NN Star Luminosity Accretion Set		
Data type	*MedALE*	*MedMaxALE*
No accretion	0.011	0.17

Table A.25: *MedALE* and *MedMaxALE* for star luminosity predictions using the balanced and reduced data set with the NN model.

Error Analysis NN Star Luminosity Mix Set		
Data type	*MedALE*	*MedMaxALE*
Balanced	0.020	0.22
Reduced	0.022	0.26

Table A.26: *MedALE* and *MedMaxALE* for star luminosity predictions using the balanced and reduced data set with the NN model.

Error Analysis NN Radius No Accretion Set		
Data type	$MedALE$	$MedMaxALE$
No accretion	0.0028	0.082

Table A.27: $MedALE$ and $MedMaxALE$ for radius predictions using the balanced and reduced data set with the NN model.

Error Analysis NN Radius Mix Set		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.0042	0.11
Reduced	0.0072	0.44

Table A.28: $MedALE$ and $MedMaxALE$ for radius predictions using the balanced and reduced data set with the NN model.

A.3 TNN Models Results

A.3.1 Accretion set

Error Analysis TNN Temperature		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.0076	0.10
Reduced	0.0084	0.09

Table A.29: $MedALE$ and $MedMaxALE$ for temperature predictions using the balanced and reduced data set with the TNN model.

Error Analysis TNN Star Luminosity		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.084	0.62
Reduced	0.099	0.60

Table A.30: $MedALE$ and $MedMaxALE$ for star luminosity predictions using the balanced and reduced data set with the TNN model.

Error Analysis TNN Radius		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.029	0.25
Reduced	0.034	0.25

Table A.31: $MedALE$ and $MedMaxALE$ for radius predictions using the balanced and reduced data set with the TNN model.

Error Analysis TNN Accretion Luminosity		
Data type	$MedALE$	$MedMaxALE$
Balanced	0.026	0.25
Reduced	0.025	0.23

Table A.32: $MedALE$ and $MedMaxALE$ for accretion luminosity predictions using the balanced and reduced data set with the TNN model.

Appendix B

Appendix Optimization Results

B.1 LightGBM Models Parameters

B.1.1 Temperature

LightGBM Optuna Best Parameters Temperature Complete Set					
Data set	Val score	Num leaves	Learning rate	Max depth	Subsample
Accretion	0.22	114	0.098	10	0.99
No accretion	0.20	17	0.008	6	0.80
Mix	0.44	85	0.53	7	0.66

Table B.1: Optuna best parameters for temperature optimization using the complete set

LightGBM Optuna Best Parameters Temperature Balanced Set					
Data set	Val score	Num leaves	Learning rate	Max depth	Subsample
Accretion	0.19	87	0.099	8	0.83
No accretion	0.20	17	0.008	6	0.80
Mix	0.5	119	0.021	11	0.60

Table B.2: Optuna best parameters for temperature optimization using the balanced set

LightGBM Optuna Best Parameters Temperature Reduced Set					
Data set	Val score	Num leaves	Learning rate	Max depth	Subsample
Accretion	0.19	13	0.05	10	0.8
No accretion	0.20	17	0.008	6	0.80
Mix	0.4	16	0.07	7	0.67

Table B.3: Optuna best parameters for temperature optimization using the reduced set

B.1.2 Star Luminosity

LightGBM Optuna Best Parameters Star Luminosity Complete Set					
Data set	Val score	Num leaves	Learning rate	Max depth	Subsample
Accretion	1.53	92	0.06	6	0.61
No accretion	1.3	14	0.04	3	0.65
Mix	1.53	84	0.05	6	0.66

Table B.4: Optuna best parameters for star luminosity optimization using the complete set

LightGBM Optuna Best Parameters Star Luminosity Balanced Set					
Data set	Val score	Num leaves	Learning rate	Max depth	Subsample
Accretion	1.4	90	0.073	6	0.63
No accretion	1.3	14	0.04	3	0.65
Mix	1.5	87	0.093	9	0.84

Table B.5: Optuna best parameters for star luminosity optimization using the balanced set

LightGBM Optuna Best Parameters Star Luminosity Reduced Set					
Data set	Val score	Num leaves	Learning rate	Max depth	Subsample
Accretion	1.3	85	0.0089	6	0.92
No accretion	1.3	14	0.04	3	0.65
Mix	1.4	50	0.08	4	0.75

Table B.6: Optuna best parameters for star luminosity optimization using the reduced set

B.1.3 Radius

LightGBM Optuna Best Parameters Radius Complete Set					
Data set	Val score	Num leaves	Learning rate	Max depth	Subsample
Accretion	0.57	9	0.095	4	0.60
No accretion	0.55	3	0.087	11	0.62
Mix	1.085	60	0.95	11	0.80

Table B.7: Optuna best parameters for radius optimization using the complete set

LightGBM Optuna Best Parameters Radius Balanced Set					
Data set	Val score	Num leaves	Learning rate	Max depth	Subsample
Accretion	0.58	76	0.025	11	0.99
No accretion	0.55	3	0.087	11	0.62
Mix	0.94	120	0.095	6	0.99

Table B.8: Optuna best parameters for radius optimization using the balanced set.

LightGBM Optuna Best Parameters Radius Reduced Set					
Data set	Val score	Num leaves	Learning rate	Max depth	Subsample
Accretion	0.46	7	0.007	4	0.95
No accretion	0.55	3	0.087	11	0.62
Mix	0.66	53	0.009	11	0.69

Table B.9: Optuna best parameters for radius optimization using the reduced set

B.1.4 Accretion Luminosity

LightGBM Optuna Best Parameters Accretion Luminosity Complete Set					
Data set	Val score	Num leaves	Learning rate	Max depth	Subsample
Accretion	10	7	0.006	4	0.85
Mix	10.1	10	0.006	9	0.98

Table B.10: Optuna best parameters for accretion luminosity optimization using the complete set.

LightGBM Optuna Best Parameters Accretion Luminosity Balanced Set					
Data set	Val score	Num leaves	Learning rate	Max depth	Subsample
Accretion	6.5	82	0.024	9	0.81
Mix	6.5	51	0.033	6	0.90

Table B.11: Optuna best parameters for accretion luminosity optimization using the balanced set.

LightGBM Optuna Best Parameters Accretion Luminosity Reduced Set					
Data set	Val score	Num leaves	Learning rate	Max depth	Subsample
Accretion	3.0	87	0.057	9	0.9
Mix	2.8	5	0.023	11	0.75

Table B.12: Optuna best parameters for accretion luminosity optimization using the reduced set

B.2 NN Models Parameters

B.2.1 Temperature

NN Optuna Best Parameters Temperature Balanced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.013	68	170	0.0001	300	0.40	1
No accretion	0.02	66	235	7.2×10^{-5}	94	0.64	1
Mix	0.026	107	82	0.0024	429	0.17	2

Table B.13: Optuna best parameters for temperature optimization using the balanced set with NN.

NN Optuna Best Parameters Temperature Reduced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.016	108	133	0.0004	370	0.098	2
No accretion	0.02	66	235	7.2×10^{-5}	94	0.64	1
Mix	0.0155	45	200	0.00021	141	0.7	2

Table B.14: Optuna best parameters for temperature optimization using the reduced set with NN.

B.2.2 Star Luminosity

NN Optuna Best Parameters Star Luminosity Balanced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.065	48	242	0.0004	403	0.7	1
No accretion	0.021	98	160	4.6×10^{-5}	238	0.77	0
Mix	0.044	66	234	0.0009	181	0.6	2

Table B.15: Optuna best parameters for star luminosity optimization using the balanced set with NN.

NN Optuna Best Parameters Star Luminosity Reduced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.073	109	177	0.0001	82	0.3	1
No accretion	0.021	98	160	4.6×10^{-5}	238	0.77	0
Mix	0.0625	49	179	0.0004	380	0.16	2

Table B.16: Optuna best parameters for star luminosity optimization using the reduced set with NN.

B.2.3 Radius

NN Optuna Best Parameters Radius Balanced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.085	112	70	2.0×10^{-5}	287	0.92	1
No accretion	0.017	118	89	0.0001	78	0.94	1
Mix	0.016	75	168	0.0004	150	0.27	2

Table B.17: Optuna best parameters for radius optimization using the balanced set with NN.

NN Optuna Best Parameters Radius Reduced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.017	87	107	0.0002	163	0.23	2
No accretion	0.017	118	89	0.0001	78	0.94	1
Mix	0.015	52	225	0.0005	491	0.16	2

Table B.18: Optuna best parameters for radius optimization using the reduced set with NN.

B.2.4 Accretion Luminosity

NN Optuna Best Parameters Accretion Luminosity Balanced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.0007	33	167	7.3×10^{-5}	468	0.8	2
Mix	0.0003	44	88	2.5×10^{-5}	168	0.54	1

Table B.19: Optuna best parameters for accretion luminosity optimization using the balanced set with NN.

NN Optuna Best Parameters Accretion Luminosity Reduced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.002	116	230	4.3×10^{-5}	107	0.36	2
Mix	0.016	77	180	0.0002	196	0.56	2

Table B.20: Optuna best parameters for accretion luminosity optimization using the reduced set with NN.

B.3 TNN Models Parameters

B.3.1 Temperature

TNN Optuna Best Parameters Temperature Balanced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.003	186	356	0.0001	741	0.32	10

Table B.21: Optuna best parameters for temperature optimization using the balanced set with TNN.

TNN Optuna Best Parameters Temperature Reduced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.005	116	396	0.0001	410	0.76	10

Table B.22: Optuna best parameters for temperature optimization using the reduced set with TNN.

TNN Optuna Best Parameters Temperature No accretion Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
No accretion	0.006	350	590	4.6×10^{-5}	345	0.78	14

Table B.23: Optuna best parameters for temperature optimization using the no accretion set with TNN.

B.3.2 Star Luminosity

NN Optuna Best Parameters Star Luminosity Balanced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.02	176	356	9.6×10^{-5}	509	0.21	9

Table B.24: Optuna best parameters for star luminosity optimization using the balanced set with TNN.

TNN Optuna Best Parameters Star Luminosity Reduced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.022	176	376	6.3×10^{-5}	277	0.3	7

Table B.25: Optuna best parameters for star luminosity optimization using the reduced set with TNN.

TNN Optuna Best Parameters Star Luminosity No Accretion Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
No accretion	0.003	250	750	0.00015	285	0.56	20

Table B.26: Optuna best parameters for star luminosity optimization using the no accretion set with TNN.

B.3.3 Radius

TNN Optuna Best Parameters Radius Balanced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.005	186	426	$4.3\ \times10^{-5}$	222	0.1	12

Table B.27: Optuna best parameters for radius optimization using the balanced set with TNN.

TNN Optuna Best Parameters Radius Reduced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.004	116	446	$4.3\ \times10^{-5}$	245	0.4	8

Table B.28: Optuna best parameters for radius optimization using the reduced set with TNN.

TNN Optuna Best Parameters Radius No Accretion Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
No accretion	0.004	420	610	$9.6\ \times10^{-5}$	296	0.45	12

Table B.29: Optuna best parameters for radius optimization using the no accretion no accretion set with TNN.

B.3.4 Accretion Luminosity

TNN Optuna Best Parameters Accretion Luminosity Balanced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.0009	106	166	4.3×10^{-5}	450	0.32	1

Table B.30: Optuna best parameters for accretion luminosity optimization using the balanced set with TNN.

TNN Optuna Best Parameters Accretion Luminosity Reduced Set							
Data set	Val score	Input	Hidden L	Lr	Batch size	Gamma	Num Hidden
Accretion	0.0002	136	496	8.4×10^{-6}	187	0.87	5

Table B.31: Optuna best parameters for accretion luminosity optimization using the reduced set with TNN.

Appendix C

Appendix Accretion Luminoisty NN Errors

L_{acc} Problematic Simulations for NN Model		
Star final mass ($M\odot$)	$MaxALE$	$MedALE$
11.28	3.9	0.092
3.87	3.0	0.067
0.40	2.9	0.006
0.13	2.8	0.112
0.61	2.6	0.022
0.11	2.5	0.118

Table C.1: $MaxALE$ by final star mass in L_{acc} predictions using NN model.

Bibliography

[1] FRANCESCA D'Antona and ITALO Mazzitelli. Evolution of low mass stars. *Memorie della Societa Astronomica Italiana*, 68:807, 1997.

[2] PO Baqui, V Marra, L Casarini, R Angulo, LA Diaz-Garcia, C Hernández-Monteagudo, PAA Lopes, C López-Sanjuan, D Muniesa, VM Placco, et al. The minijpas survey: star-galaxy classification using machine learning. *Astronomy & Astrophysics*, 645:A87, 2021.

[3] Abhishek Malik, Benjamin P Moster, and Christian Obermeier. Exoplanet detection using machine learning. *Monthly Notices of the Royal Astronomical Society*, 513(4):5505–5516, 2022.

[4] Tom Marianer, Dovi Poznanski, and J Xavier Prochaska. A semisupervised machine learning search for never-seen gravitational-wave sources. *Monthly Notices of the Royal Astronomical Society*, 500(4):5408–5419, 2021.

[5] Mark R Krumholz. Star formation in molecular clouds. In *AIP Conference Proceedings*, volume 1386, pages 9–57. American Institute of Physics, 2011.

[6] Norbert S Schulz. *From dust to stars: studies of the formation and early evolution of stars.* Springer Science & Business Media, 2007.

[7] Sigurd S. Jensen and Troels Haugbølle. Explaining the luminosity spread in young clusters: proto and pre-main sequence stellar evolution in a molecular cloud environment. *Monthly Notices of the Royal Astronomical Society*, 474(1):1176–1193, 11 2017.

[8] Sean M Andrews, David J Wilner, Enrique Macías, Carlos Carrasco-González, and Andrea Isella. Resolved substructures in protoplanetary disks with the ngvla. *Science with a next generation very large array*, 517:137, 2018.

[9] Hsin-Yu Chen, Salvatore Vitale, and Francois Foucart. The relative contribution to heavy metals production from binary neutron star mergers and neutron star–black hole mergers. *The Astrophysical Journal Letters*, 920(1):L3, 2021.

[10] The hipparcos and tycho catalogues. `https://www.cosmos.esa.int/web/hipparcos/catalogues`. Accessed: 2023-07-18.

[11] Gliese catalogue of nearby stars. `http://astro.vaporia.com/start/gl.html`. Accessed: 2023-07-18.

[12] The hertzsprung russell diagram. `http://www.atlasoftheuniverse.com/hr.html`. Accessed: 2023-07-18.

[13] Jørgen Christensen-Dalsgaard. *Lecture notes on stellar structure and evolution.* Aarhus Universitet. Institute for Fysik og Astronomi, 2008.

[14] How do we measure the age of a globular cluster? `https://www.astronomy.com/science/how-do-we-measure-the-age-of-a-globular-cluster/`. Accessed: 2023-07-18.

[15] Harvard spectral classification. https://astronomy.swin.edu.au/cosmos/h/harvard+spectral+classification. Accessed: 2023-07-18.

[16] Harvard spectral classification. http://physicsanduniverse.com/harvard-spectral-classification/. Accessed: 2023-07-18.

[17] Stellar timescales. https://www.astro.princeton.edu/~gk/A403/timescales.pdf. Accessed: 2023-07-22.

[18] Kelvin-helmholtz timescale. http://astro.vaporia.com/start/khtimescale.html. Accessed: 2023-07-18.

[19] Giora Shaviv. Why the kelvin–helmholtz timescale is not really their timescale. *New Astronomy Reviews*, 51(10-12):803–813, 2008.

[20] 10 timescales in stellar interiors. http://www.mit.edu/~iancross/8901_2019A/lec011.pdf. Accessed: 2023-07-18.

[21] Energy transport within a star. https://people.ast.cam.ac.uk/~pettini/Stellar%20Structure%20Evolution/Lecture08.pdf. Accessed: 2023-07-22.

[22] G Siegfried Kutter and Warren M Sparks. Stellar accretion of matter possessing angular momentum. *Astrophysical Journal, Part 1 (ISSN 0004-637X), vol. 321, Oct. 1, 1987, p. 386-393.*, 321:386–393, 1987.

[23] Philip J Armitage. Magnetic activity in accretion disc boundary layers. *Monthly Notices of the Royal Astronomical Society*, 330(4):895–900, 2002.

[24] John Bally. Protostellar outflows. *Annual Review of Astronomy and Astrophysics*, 54:491–528, 2016.

[25] I Baraffe, G Chabrier, and J Gallardo. Episodic accretion at early stages of evolution of low-mass stars and brown dwarfs: A solution for the observed luminosity spread in h–r diagrams? *The Astrophysical Journal*, 702(1):L27, 2009.

[26] The luminosity of eclipsing binary systems. https://astro.unl.edu/naap/ebs/luminosity.html. Accessed: 2023-07-18.

[27] Paolo Padoan, Liubin Pan, Troels Haugbølle, and Åke Nordlund. Supernova driving. i. the origin of molecular cloud turbulence. *The Astrophysical Journal*, 822(1):11, 2016.

[28] Isabelle Baraffe, Eduard Vorobyov, and Gilles Chabrier. Observed luminosity spread in young clusters and fu ori stars: a unified picture. *The Astrophysical Journal*, 756(2):118, 2012.

[29] Light echoes give clues to protoplanetary disk. https://exoplanets.nasa.gov/news/1339/light-echoes-give-clues-to-protoplanetary-disk/. Accessed: 2023-05-25.

[30] Romain Teyssier. Cosmological hydrodynamics with adaptive mesh refinement-a new high resolution code called ramses. *Astronomy & Astrophysics*, 385(1):337–364, 2002.

[31] Sébastien Fromang, Patrick Hennebelle, and Romain Teyssier. A high order godunov scheme with constrained transport and adaptive mesh refinement for astrophysical magnetohydrodynamics. *Astronomy & Astrophysics*, 457(2):371–384, 2006.

[32] Introduction to magnetohydrodynamics. https://lweb.cfa.harvard.edu/~namurphy/Presentations/Introduction_to_MHD.pdf. Accessed: 2023-07-22.

[33] Troels Haugbølle, Paolo Padoan, and Åke Nordlund. The stellar imf from isothermal mhd turbulence. *The Astrophysical Journal*, 854(1):35, 2018.

[34] Modules for Experiments in Stellar Astrophysics. https://docs.mesastar.org/en/release-r22.11.1/. Accessed: 2023-02-27.

[35] sklearn.model_selection.gridsearchcv. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html. Accessed: 2023-03-08.

[36] sklearn.model_selection.randomizedsearchcv. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html. Accessed: 2023-03-03.

[37] How to implement bayesian optimization from scratch in python. https://machinelearningmastery.com/what-is-bayesian-optimization/. Accessed: 2023-03-08.

[38] All you need to know about gradient boosting algorithm part 1. regression. https://towardsdatascience.com/all-you-need-to-know-about-gradient-boosting-algorithm-part-1-regression-2520a34a502. Accessed: 2023-01-25.

[39] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30:3146–3154, 2017.

[40] You are missing out on lightgbm. it crushes xgboost in every aspect. https://towardsdatascience.com/how-to-beat-the-heck-out-of-xgboost-with-lightgbm-comprehensive-tutorial-5eba52195997. Accessed: 2023-01-25.

[41] Explained: Neural networks. https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414. Accessed: 2023-01-25.

[42] Facundo Bre, Juan M Gimenez, and Víctor D Fachinotti. Prediction of wind pressure coefficients on building surfaces using artificial neural networks. *Energy and Buildings*, 158:1429–1441, 2018.

[43] What is Gradient Descent in machine learning? https://saugatbhattarai.com.np/what-is-gradient-descent-in-machine-learning/. Accessed: 2023-01-25.

[44] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[45] Strategies and tactics for regression on imbalanced data. https://towardsdatascience.com/strategies-and-tactics-for-regression-on-imbalanced-data-61eeb0921fca. Accessed: 2023-03-08.

[46] All about feature scaling. https://medium.com/p/bcc0ad75cb35. Accessed: 2023-06-11.

[47] sklearn.preprocessing.minmaxscaler. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html. Accessed: 2023-06-11.

[48] Michael Kuffmeier, Søren Frimann, Sigurd S Jensen, and Troels Haugbølle. Episodic accretion: the interplay of infall and disc instabilities. *Monthly Notices of the Royal Astronomical Society*, 475(2):2642–2658, 2018.

[49] Using shap values to explain how your machine learning model works. https://towardsdatascience.com/using-shap-values-to-explain-how-your-machine-learning-model-works-732b3f40e137#:~:text=SHAP%20values%20(SHapley%20Additive%20exPlanations,interpretability%20of%20machine%20learning%20models. Accessed: 2023-03-08.

[50] Shap values explained exactly how you wished someone explained to you. `https://towardsdatascience.com/shap-explained-the-way-i-wish-someone-explained-it-to-me-ab81cc69ef30`. Accessed: 2023-06-11.

[51] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[52] Welcome to the shap documentation. `https://shap.readthedocs.io/en/latest/index.html`. Accessed: 2023-06-11.

[53] Kaggler's guide to lightgbm hyperparameter tuning with optuna in 2021. `https://towardsdatascience.com/kagglers-guide-to-lightgbm-hyperparameter-tuning-with-optuna-in-2021-ed048d9838b5`. Accessed: 2023-03-02.

[54] Optuna a hyperparameter optimization framework. `https://optuna.org/`. Accessed: 2023-06-11.

[55] A gentle introduction to k-fold cross-validation. `https://machinelearningmastery.com/k-fold-cross-validation/`. Accessed: 2023-03-03.

[56] Cross-validation. `https://es.mathworks.com/discovery/cross-validation.html`. Accessed: 2023-03-13.

[57] Lightgbm-parameters tuning. `https://lightgbm.readthedocs.io/en/v3.3.2/Parameters-Tuning.html`. Accessed: 2023-03-02.

[58] Understanding lightgbm parameters (and how to tune them). `https://neptune.ai/blog/lightgbm-parameters-guide`. Accessed: 2023-03-03.

[59] An Introduction to SHAP Values and Machine Learning Interpretability. `https://www.datacamp.com/tutorial/introduction-to-shap-values-machine-learning-interpretability`. Accessed: 2023-07-31.

[60] optuna.visualization.plot_param_importances. `https://optuna.readthedocs.io/en/stable/reference/visualization/generated/optuna.visualization.plot_param_importances.html`. Accessed: 2023-05-25.

[61] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[62] Dropout layer. `https://keras.io/api/layers/regularization_layers/dropout/`. Accessed: 2023-07-31.

[63] K Ruwani M Fernando and Chris P Tsokos. Dynamically weighted balanced loss: class imbalanced learning and confidence calibration of deep neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33(7):2940–2951, 2021.