ULB

SCIENTIA VINCERE TENEBRAS
ULB

MEMO-F524
MASTER THESIS

Automated classification of stars and systems using machine learning

Author :

TALHAOUI Yassin

Section :

Computer science

promoter :

Defrance Matthieu

May 7, 2025

Contents

1 Introduction **3**

1.1 Context and Motivation 3

1.2 Structure of the Thesis 3

2 Integrating machine learning techniques with traditional astrophysical methods **4**

3 Astrophysical Background **6**

3.1 Some Stellar Classifications 6

3.2 Introduction to sprectra 6

3.3 Red Dwarfs, White Dwarfs, and Main Sequence Stars 8

3.4 Observational Data and Relevance 9

3.5 Astrophysics and Machine Learning Integration 9

4 State of the Art **10**

4.1 Previous studies on stellar spectral analysis 10

4.2 Challenges and limitations 11

4.3 Recent advances and innovations 12

4.4 Future directions and emerging trends 13

5 Hands-on: Datasets And Results **15**

5.1 Dataset 1 : Stellar dataset to predict star types 15

5.1.1 Data analysis and cleaning 15

5.1.2 Models traning 24

5.1.3 Final thoughts 44

5.2 Dataset 2 : Stellar classification dataset - SDSS17 44

5.2.1 Dataset Overview 44

5.2.2 Discussion of Methodological Choices 45

5.2.3 Exploratory Data Analysis (EDA) 46

5.2.4 Feature Engineering 50

5.2.5 Models training 50

5.2.6 Comparison of models 55

5.2.7 Final recommendation 55

6 Discussion **55**

6.1 Model Performance Across Datasets 56

6.2 Interpretation of Feature Importance 56

6.3 Classification Errors and Ambiguities 56

6.4 Methodological Choices and Limitations 57

6.5 Broader Implications and Future Work 57

7 Conclusion 58

A Outliers deletion 59

B Log transformation of Luminosity 59

C Random Forest code 59

D Logistic Regression code 60

E Binary classification code 61

F MLP training logs 63

1 Introduction

1.1 Context and Motivation

The aim of this Master's thesis is to understand and investigate how current machine learning techniques can help us study the stellar properties of single and binary stars. Initially, astrophysics was a very data-poor scientific field, but over time numerous space missions and large astonomical studies have enabled us to collect massive amounts of data from hundreds of millions of astronomical sources. In the GAIA era, the volume of data is set to increase even further, into the petabyte range. Astrophysics has thus become a data-intensive science. All this stellar data has led to the emergence of new algorithmic, computational and statistical challenges. It is in this context that machine learning techniques, used in various fields of scientific study, could prove invaluable in extracting information from observations. By taking advantage of machine learning algorithms, astrophysicists can tackle a wide range of research questions more effectively, enabling us to better understand the cosmos and make new discoveries.

In this work, I focused on applying machine learning models to two astronomical datasets: sdss17 and star-dataset. The first dataset consists of a set of stellar spectral data, used to classify astronomical objects into different categories such as stars, galaxies and quasars. The second dataset contains information on the star systems of a 6-class stellar dataset for the classification of stars, whose key properties we have sought to identify and whose evolutionary stages we have sought to predict using advanced classification techniques. We used various machine learning models, including logistic regression, random forest, gradient boosting (XGBoost) and neural networks, to evaluate their performance in these tasks. Through systematic hyperparameter tuning and model evaluation, we identified the most effective approaches for each dataset, demonstrating the power of machine learning in modern astrophysics.

The results not only highlight the strengths and limitations of various algorithms in astronomical data processing, but also offer insight into how automated classification techniques can contribute to large-scale stellar population studies. By integrating data-driven methods into astrophysical research, we are paving the way for more efficient and accurate analyses of the vast datasets produced by current and future space missions.

1.2 Structure of the Thesis

The thesis is organized into distinct sections: Astrophysical Background, Machine Learning Techniques, State of the Art, Data, Results, and Discussion.

2 Integrating machine learning techniques with traditional astrophysical methods

To explore the integration of machine learning techniques with traditional astrophysical methods, we will embark on a challenge where the combination of domain knowledge and data-driven approaches will illuminate the cosmos with great clarity. This collaborative effort harnesses the expertise of astronomers and data scientists, bringing together the knowledge of researchers and computational insights to explore stellar properties. What follows is a presentation of the methodologies used in several studies in this field.

Understanding traditional astrophysical methods

We'll start by exploring traditional astrophysical methods, including spectroscopy, photometry and stellar modeling. Without going too far into the subtleties of stellar classification systems, evolutionary trajectories and chemical abundance analyses, in order to establish a solid knowledge base in the field.

Identify Challenges and Opportunities

By examining the field of astrophysical research, researchers identify areas where traditional methods run into limitations or inefficiencies. Identifying opportunities for improvement such as machine learning techniques can give us something in terms of automating labor-intensive tasks or discovering hidden patterns in the data, to guide our study towards improving the accuracy of predictions and gaining new insights from the data we hold.

Engage in interdisciplinary collaboration

Collaboration between astronomers and data scientists creates an environment where domain experts and practitioners in the field of machine learning converge to exchange ideas, methodologies and points of view. This interdisciplinary dialogue facilitates knowledge sharing and bridges the gap between theoretical astrophysics and computational data analysis.

Data acquisition and processing

Among the most important tasks is the management of diverse and representative data sets comprising stellar spectra, photometric measurements and ancillary information, including stellar classifications, ages and metallicities. Rigorous pre-processing techniques deal with data quality issues such as noise, calibration errors and missing values, guaranteeing the integrity of the analyses.

Feature Engineering and feature selection

Collaboration with astrophysicists enables us to identify relevant astrophysical features that include important stellar properties. Drawing on our domain knowledge, we design informative features that capture spectral line intensities, continuum shapes and other key characteristics. Dimensionality reduction techniques distill high-dimensional spectral data into interpretable feature spaces, improving computational efficiency and interpretability.

Model development and validation

Co-designing machine learning models tailored to specific astrophysical questions or tasks, such as stellar parameter estimation or classification, takes our exploration a step further. Adopting a variety of algorithms, including traditional regression and classification methods, as well as sophisticated deep learning architectures such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), enables us to extract information from complex data.

Interpretability and transparency

With an emphasis on model interpretability and transparency, researchers are collaborating with astronomers to develop techniques for a posteriori interpretability. Feature significance analysis, attention mechanisms and diagnostic explanations of patterns highlight the underlying logic of pattern predictions, facilitating confidence and understanding.

Refinement and iterative evaluation

By adopting an iterative approach to model refinement and evaluation, we solicit feedback from domain experts at every stage of the development process. By continually validating model performance against field observations, we refine algorithms and methodologies based on empirical observations and domain-specific considerations, ensuring robust and reliable analyses.

Added value

Thanks to this combination of traditional astrophysical and machine learning techniques, new perspectives are opening up on the complex field of stellar properties, paving the way for future advances in our understanding of the cosmos.

3 Astrophysical Background

The study of stars forms the foundation of astrophysics. Stars are massive, luminous spheres of plasma held together by gravity and are the primary sources of electromagnetic radiation in the universe. Their classification and analysis have long been essential for understanding stellar evolution, galaxy formation, and the large-scale structure of the cosmos. In this thesis, the goal is to classify stars based on various observable and derived features using machine learning techniques. For this purpose, it is crucial to introduce the basic astrophysical concepts that directly relate to the datasets and methods used.

3.1 Some Stellar Classifications

Stars are commonly classified according to their spectral characteristics and luminosity. The most widely used system is the **Harvard Spectral Classification**, which categorizes stars into spectral types O, B, A, F, G, K, and M (often remembered by the mnemonic "Oh Be A Fine Girl/Guy, Kiss Me"). These classes represent a sequence from the hottest and most massive stars (O-type) to the coolest and least massive (M-type). The spectral class directly corresponds to a star's surface temperature, color, and to some extent, its mass.

In addition to spectral classification, stars are further categorized based on their luminosity using the **Yerkes classification**, which separates stars into luminosity classes such as main sequence stars (dwarfs), giants, and supergiants.

3.2 Introduction to sprectra

The importance of spectra

Our understanding of the physical properties of stars depends largely on the analysis of their spectra. By examining absorption lines, we can determine the mass, temperature and composition of stars. The shape of the lines provides information about atmospheric processes.

Composition

Stellar spectra consist of a continuous spectrum on which narrow spectral lines are superimposed, mainly dark absorption lines, but sometimes also bright emission lines.

Continuous spectra

The continuous spectrum comes from the star's hot surface. The atmosphere absorbs specific wavelengths, creating dark zones in the spectrum, indicating different chemical compositions.

Classification

Stellar spectra are classified according to the intensity of these spectral lines. This classification system was initiated by Isaac Newton, then perfected by Joseph Fraunhofer and others.

Measurement methods

Stellar spectra are generally obtained using objective prisms or slit spectrographs. These methods enable detailed analysis of individual spectral lines.

Analysis

Spectra are converted into intensity plots, revealing flux density as a function of wavelength. The shape of the spectral lines provides valuable information on stellar atmospheres, while the intensity of the lines can be used to determine chemical compositions.

Harvard spectral classification

Developed at Harvard Observatory, this classification system ranks stars according to their spectral characteristics, mainly temperature. It includes letters for spectral types and numbers for subclasses.

Yerkes spectral classification

A more precise classification system introduced by the Yerkes Observatory takes into account both temperature and luminosity. It classifies stars into six luminosity categories, providing a better understanding of their properties.

Particular spectra

Some stars exhibit particular spectra due to factors such as strong stellar winds, rotation or binary interactions. Examples include Wolf-Rayet stars, Be stars and shell stars.

The benefits of stellar spectra

In addition, the analysis of stellar spectra enables us to understand the nature and evolution of celestial bodies like stars. They provide us with information such as :

- **Temperature**: The surface temperature of a star and that of its outer envelope can be deduced from the color of the light it emits. Indeed, a hotter star will appear bluer, as the higher temperature favors the emission of light at shorter wavelengths, in accordance with the law of thermal radiation. By analyzing its spectrum, it is possible to estimate an “effective” temperature for the star, taking into account the transfer of radiation through the different layers of its stellar atmosphere.

- **Chemical composition**: The particular frequencies of spectral lines provide distinct information on which elements absorb or emit photons. Spectroscopic databases have been developed from the study of laboratory-produced spectra, making it easier to identify the origin of observed absorption or emission lines in astronomical spectra. By analyzing the relative intensity of the characteristic lines of the elements detected, and based on theoretical models, it is possible to infer the chemical composition of each star’s atmosphere.
- **The velocity**: $\Delta\lambda$, the shift of observed spectral lines, is commonly used to measure velocities. It is used to calculate the radial velocity $\mathbf{v}$ of a celestial object, which corresponds to its velocity component along the line of sight. This velocity is expressed as

  $v = c\frac{\Delta\lambda}{\lambda}$.

In short, the analysis of stellar spectra is a powerful tool for probing the secrets of stars. By revealing their temperature, chemical composition and velocity, these spectra offer us a fascinating window into the nature and evolution of celestial stars. They are thus an essential pillar of astrophysical research, enabling us to better understand the mysteries of the universe.

3.3 Red Dwarfs, White Dwarfs, and Main Sequence Stars

In the datasets analyzed in this thesis, several types of stars appear, including **Red Dwarfs**, **White Dwarfs**, and **Main Sequence Stars**. These categories are essential to interpret model predictions and feature importance results.

- **Red Dwarfs (M-type)**: These are small, cool, and faint stars, with surface temperatures below 3,500 K. They are the most common type of star in the Milky Way but are difficult to observe due to their low luminosity. Red Dwarfs are particularly relevant in the datasets used because of their prevalence and distinctive features such as low temperature and color indices.
- **White Dwarfs**: These are stellar remnants of low to medium mass stars that have exhausted their nuclear fuel. They are characterized by high density, low luminosity, and high surface temperature. Their identification in spectral data is important due to their unique signatures in color and magnitude.
- **Main Sequence Stars**: This category includes stars that are in the stable hydrogen-burning phase of their evolution, such as our Sun (a G-type star). These stars span a wide range of temperatures and luminosities and form a clear pattern in the Hertzsprung-Russell diagram, a key tool for understanding stellar properties.

3.4 Observational Data and Relevance

The datasets used in this thesis are derived from large sky surveys, where stars are observed across multiple photometric bands. Each observation records a star's brightness in various filters (e.g., u, g, r, i, z), from which physical parameters such as temperature, radius, and luminosity can be estimated or inferred. These observational features are directly influenced by the underlying astrophysical properties described above.

For instance, Red Dwarfs are easily distinguishable from Main Sequence stars by their color index and absolute magnitude. White Dwarfs, on the other hand, may have similar magnitudes to main sequence stars but can be identified by their spectral class and high temperature. These astrophysical distinctions underpin the machine learning classification task and guide feature engineering and model interpretation.

3.5 Astrophysics and Machine Learning Integration

Connecting this background to the methodological approach, the astrophysical properties guide the feature selection and help evaluate the model outputs. Understanding the physical meaning of features such as temperature, absolute magnitude, or spectral type allows for more meaningful interpretations of feature importance and classification errors. Moreover, astrophysical context is vital in interpreting clusters or classification boundaries in the results.

Thus, this section forms a necessary bridge between the observational data and the computational models employed in the following chapters. By grounding the data in physical reality, we ensure that the machine learning models not only perform well statistically but also yield scientifically valid insights.

4 State of the Art

This section reviews existing works on stellar classification using machine learning, including key methodologies and recent advances.

4.1 Previous studies on stellar spectral analysis

Analysis of chemical composition

The researchers used machine learning techniques to analyze stellar spectra and infer their chemical composition. By training models on spectral data with known chemical abundances, the algorithms can predict the elemental composition of stars based on their spectra. Feature extraction methods such as **principal component analysis (PCA)** were used to reduce the dimensionality of the spectral data while retaining relevant information. Machine learning algorithms such as **support vector machines (SVM)** or **random forests** were employed to classify stars into different chemical abundance classes based on their spectra.

Temperature estimation

Machine learning techniques were used to estimate the effective temperature of stars from their spectral characteristics. Researchers used regression techniques such as linear regression or neural networks to predict the temperature of stars from their spectral characteristics. Feature engineering methods, including wavelength selection or continuum normalization, were applied to improve the predictive performance of temperature estimation models.

Brightness prediction

Machine learning algorithms have been used to predict stellar brightness, which is a measure of their intrinsic luminosity. Luminosity estimation is essential for understanding the properties of stars and their evolutionary stages. Regression algorithms such as Gaussian processes or deep learning models such as convolutional neural networks (CNNs) have been used to predict stellar brightness from spectral data. Feature extraction techniques, such as line intensity indices or flux ratios, have been used to capture relevant information from stellar spectra for brightness prediction.

Methodologies and algorithms

Feature extraction : Methods such as PCA have been used to extract relevant features from stellar spectra while reducing dimensionality. **Classification** : Algorithms such as SVM, Random Forests or k-nearest neighbors were used to classify stars into different categories based on their spectral characteristics.

Regression : Techniques such as linear regression, Gaussian processes or neural networks have been used to predict continuous stellar parameters such as temperature or luminosity from spectral data.

4.2 Challenges and limitations

When applying machine learning techniques to stellar astrophysics, several challenges and limitations must be considered.

Data quality

Stellar spectra data samples may contain artifacts, instrumental effects, or calibration errors that can affect data quality. In addition, variations in data quality between different observational sources or instruments can introduce biases or inconsistencies into the analysis.

Samples size

Obtaining large and diverse data samples of stellar spectra for training machine learning models can be challenging, especially for rare or exotic stellar objects. Limited sample sizes can lead to insufficient coverage of the parameter space, affecting the model's ability to generalize to unseen data.

Noise Reduction

Stellar spectra are often subject to noise from various sources, including photon noise, background noise, and instrumental effects. Developing robust noise reduction techniques that effectively filter out noise while preserving the underlying signal is crucial for accurate spectral analysis.

Interpretation of the models

Machine learning models, especially complex ones such as deep learning models, can lack interpretability, making it difficult to understand how they arrive at their predictions. Interpretable models are essential for understanding the physical processes underlying stellar phenomena and for validating the reliability of model predictions.

Generalisation Performance

Overfitting occurs when a model learns to capture noise or irrelevant patterns in the training data, leading to poor generalization performance on unseen data. Regularization techniques, cross-validation, and model complexity control are essential to mitigate overfitting and ensure model robustness.

Introduction of Bias

This can occur when the training dataset is not representative of the underlying population of interest, leading to biased model predictions. Care must be taken to ensure that the training dataset adequately covers the full diversity of stellar properties and avoids biases introduced by observational or sampling methods.

Generalization to unseen data

Machine learning models trained on one observational dataset may not generalize well to unseen data from different telescopes, instruments, or observing conditions. Transfer learning techniques, which leverage knowledge gained from one dataset to improve performance on another, can help address generalization issues.

Motivation

Addressing these challenges and limitations is essential to successfully applying machine learning techniques to stellar astrophysics. By developing robust methodologies, incorporating domain knowledge, and carefully evaluating model performance, researchers can overcome these obstacles and unlock the full potential of machine learning techniques to advance our understanding of the cosmos.

4.3 Recent advances and innovations

Recent advances in machine learning methodologies have significantly improved the analysis of stellar spectra, offering innovative approaches for feature engineering, dimensionality reduction, and model optimization. Here are some examples:

Dimensionality reduction

Variational autoencoders (VAEs) and generative adversarial networks (GANs) have been used for unsupervised dimensionality reduction of spectral data. These techniques learn low-dimensional representations of spectra while preserving essential information, facilitating more efficient processing and analysis.

Models Optimization

Bayesian optimization methods, such as Gaussian processes and Bayesian neural networks, have been used for hyperparameter tuning and model optimization. These techniques enable more efficient exploration of the hyperparameter space and better convergence of machine learning models.

Deep learning Architectures

- Convolutional neural networks (CNNs) have been applied to spectral data for tasks such as stellar object classification, spectral feature identification, and stellar parameter estimation. Convolutional neural networks can automatically learn spatial patterns in spectral data, making them well-suited for tasks that involve analyzing spatially structured information.
- Recurrent neural networks (RNNs) have been used to model temporal dependencies in time series of spectral data, enabling prediction of stellar variability and transient events.
- Long Short-Term Memory (LSTM) networks, a type of RNN, have shown promise in capturing long-term dependencies in sequential spectral data, enabling more accurate modeling of stellar dynamics over time.

These recent advances in machine learning methodologies have revolutionized the analysis of stellar spectra, enabling more accurate and efficient processing of observational data. By leveraging deep learning architectures and innovative feature engineering and model optimization techniques, researchers can gain new insights into the complex physical processes occurring in stars and galaxies.

4.4 Future directions and emerging trends

As the possibilities of machine learning continue to expand and our understanding of stellar astrophysics deepens, future directions in research promise to open new perspectives and advance our knowledge of the cosmos. Emerging trends in both fields offer exciting opportunities for innovation and discovery.

Future prospects

In the future, research in machine learning and stellar astrophysics is expected to explore increasingly complex and interdisciplinary questions. Integrating advanced machine learning techniques with traditional astrophysical methods will enable researchers to tackle the fundamental mysteries of astrophysics with greater precision and efficiency.

Identify emerging trends

- **Multimodal data analysis** : With the advent of multi-wavelength, multi-messenger astronomy, future research will focus on integrating data from diverse sources, such as optical, infrared, radio, and gravitational-wave observations. Machine learning algorithms capable of analyzing multimodal data streams will play a critical role in uncovering synergies and correlations between different wavelengths and cosmic messengers.

- **Transfer learning** : Transfer learning techniques, which leverage knowledge gained in one domain to improve performance in another, will become increasingly common in stellar astrophysics. By transferring learned representations of well-studied stellar populations to underexplored regions of parameter space, transfer learning enables more efficient exploration and characterization of diverse stellar populations.

- **Ensemble methods** : Ensemble learning approaches, which combine predictions from multiple models to improve accuracy and robustness, will be leveraged to address the uncertainties and complexities inherent in astrophysical phenomena. Ensemble methods provide a powerful framework for integrating diverse models, data sources, and observational uncertainties, enabling more reliable predictions and inferences.

Discussion of potential applications

Machine learning offers immense potential to revolutionize future astronomical surveys and missions, by proposing new approaches for data analysis, interpretation and discovery:

- **Automated analysis of observations** : Machine learning algorithms will streamline the analysis of large-scale astronomical observations, enabling the automated detection and characterization of celestial objects, transient events, and astrophysical phenomena. Real-time data processing and event classification will improve our ability to identify rare and elusive cosmic phenomena.

- **Precision cosmology** : Machine learning techniques will facilitate precision cosmological analyses by extracting subtle signals from cosmological datasets, such as maps of the cosmic microwave background, observations of large-scale structures, and gravitational wave observations. Advanced statistical methods and model selection techniques will enable more accurate parameter estimation and hypothesis testing in cosmological models.

- **Characterization of exoplanets** : Machine learning algorithms will advance the field of exoplanet characterization by enabling the detection and classification of exoplanetary systems from stellar spectra and photometric observations. New feature extraction methods and data-driven models will improve our ability to identify exoplanets, characterize their atmospheres, and assess their habitability potential.

In summary, future research directions in machine learning techniques and stellar astrophysics will explore emerging trends such as multimodal data analysis, transfer learning, and ensemble methods, paving the way for transformative advances in our understanding of the universe. By harnessing the power of machine learning techniques, astronomers will unlock new perspectives on the cosmos.

5 Hands-on: Datasets And Results

In the following, I will apply all the machine learning techniques acquired during my master's degree in computer science to datasets containing data related to stars and celestial objects. The objective will be to use classification via selected machine learning models in order to classify stars/celestial objects into different categories. To do this, I will follow the following steps:

- Explore the dataset (EDA: check missing values, distributions, correlations)
- Feature selection (correlation matrix, importance ranking)
- Binary classification (start with two classes, e.g., Main Sequence vs. White Dwarf)
- Multi-class classification (gradually moving to six classes). Test multiple classifiers (logistic regression, SVM, random forest, neural networks, etc.
- Evaluate performance (precision, accuracy, recall, F1 score, confusion matrix)
- Write explanations (justify choices, compare results)

5.1 Dataset 1 : Stellar dataset to predict star types

The goal of this project is to build a machine learning model capable of predicting the type of a star based on its physical properties, such as temperature, luminosity, radius, absolute magnitude, color, and spectral class. This classification task follows the Hertzsprung-Russell (HR) diagram, a fundamental tool in astrophysics that categorizes stars based on their temperature and luminosity.

5.1.1 Data analysis and cleaning

This dataset comes to us from Kaggle and includes 240 stars, classified into six different star types:

- 0 → Brown Dwarf
- 1 → Red Dwarf
- 2 → White Dwarf
- 3 → Main Sequence
- 4 → Supergiant
- 5 → Hypergiant

The properties of each star are measured relative to the Sun, here are the input features :

- Temperature (K) - Surface temperature in kelvins

- Luminosity (L/Lo) - Brightness relative to the sun
- Radius (R/Ro) - Radius relative to the sun
- Absolute magnitude (Mv) - Intrinsic luminosity
- Star color - Color observed after spectral analysis
- Spectral class - Classification based on spectral lines (O, B, A, F, G, K, M)

Data types are numeric, except Star color and Spectral Class which are categorical.

Data collection and preparation

The dataset was created using real astrophysical equations and data sources such as:

- Stefan-Boltzmann law - to calculate brightness
- Wien's Displacement Law - To estimate surface temperature
- Absolute Magnitude Relationships - To Determine Intrinsic Brightness
- Parallax Methods - To Derive Radius Values

The dataset was compiled from multiple online sources, taking about three weeks to collect and preprocess, ensuring that missing data were calculated using astrophysical formulas.

Importance of the study

Understanding how stars are classified and how they evolve over time is fundamental to astronomy and astrophysics. This study helps to:

- Validate the HR diagram using machine learning.
- Developing predictive models for star classification.
- Explore the importance of features in differentiating stars.

This project uses machine learning classifiers to analyze the performance of different models in stellar classification, from simple binary classification to 6-class multi-class classification. The results can provide valuable insights into the relationships between stellar properties and how stars fit into stellar evolution.

Exploration and analysis of the dataset

Before building a classification model, it is essential to explore and understand the dataset to identify patterns, relationships, and potential preprocessing needs. This section focuses on analyzing the structure of the dataset, checking for missing values, visualizing distributions, and understanding correlations between features.

Checking for missing values

Missing data can impact model performance. We check for missing values to determine if preprocessing steps, such as imputation or deletion, are necessary.

Output ;

```
Temperature (K)           0
Luminosity(L/Lo)          0
Radius(R/Ro)              0
Absolute magnitude(Mv)    0
Star type                 0
Star color                0
Spectral Class            0
dtype: int64
```

We have no missing values.

Statistical summary

Generating summary statistics allows you to know the range, mean and distribution of numeric variables. It also allows to detect outliers, and understand the scale and variance of features.

```
        Temperature (K) Luminosity(L/Lo)        Radius(R/Ro)    Absolute magnitude(Mv)  Star type
count   240.000000      240.000000      240.000000      240.000000      240.000000
mean    10497.462500    107188.361635   237.157781      4.382396        2.500000
std     9552.425037     179432.244940   517.155763      10.532512       1.711394
min     1939.000000     0.000080        0.008400        -11.920000      0.000000
25%     3344.250000     0.000865        0.102750        -6.232500       1.000000
50%     5776.000000     0.070500        0.762500        8.313000        2.500000
75%     15055.500000    198050.000000   42.750000       13.697500       4.000000
max     40000.000000    849420.000000   1948.500000     20.060000       5.000000
```

The dataset is highly diverse, capturing a broad spectrum of stellar properties from cold, dim dwarfs to massive, bright supergiants. Distributions are skewed for luminosity and radius — appropriate transformations (e.g., log scaling) may help modeling.

Class distribution (balance check)

To ensure that our dataset is not severely imbalanced, we visualize the distribution of different star types.

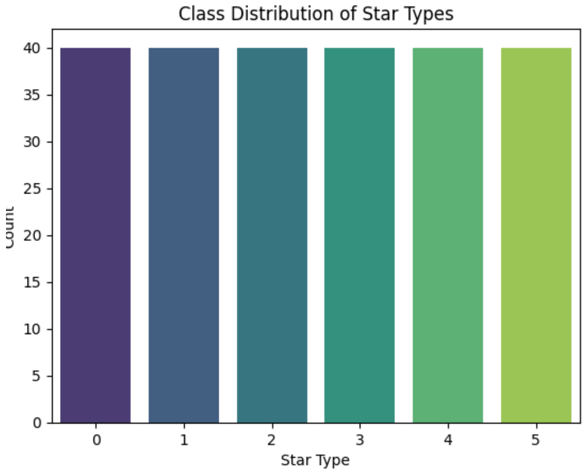


Figure 1: Classes Distribution - This figure illustrates a key aspect of the analysis.

The dataset is perfectly balanced between the different classes, with 40 samples per class.

Analysis of the Correlation Matrix

This correlation matrix (see figure 2) provides insight into the relationships between different features of the dataset. Here are some key observations:

Strong correlation with star type:

Luminosity (0.68), radius (0.66) and temperature (0.41) show a strong positive correlation with star type.

Absolute magnitude (-0.96) is strongly negatively correlated with star type, which makes sense since brighter stars (lower magnitude values) tend to be higher in the classification hierarchy.

Variables Dépendancies :

Luminosity and Radius (0.53): Larger stars tend to be brighter.

Absolute magnitude and luminosity (-0.69): Higher luminosity results in lower absolute magnitude values (inverse relationship by definition).

Color and spectral class of stars:

Star Color and Temperature (-0.7): This negative correlation makes sense because hot stars tend to appear blue, while cool stars appear red.

Star color and absolute magnitude (0.33): The brightest stars tend to exhibit specific color characteristics.

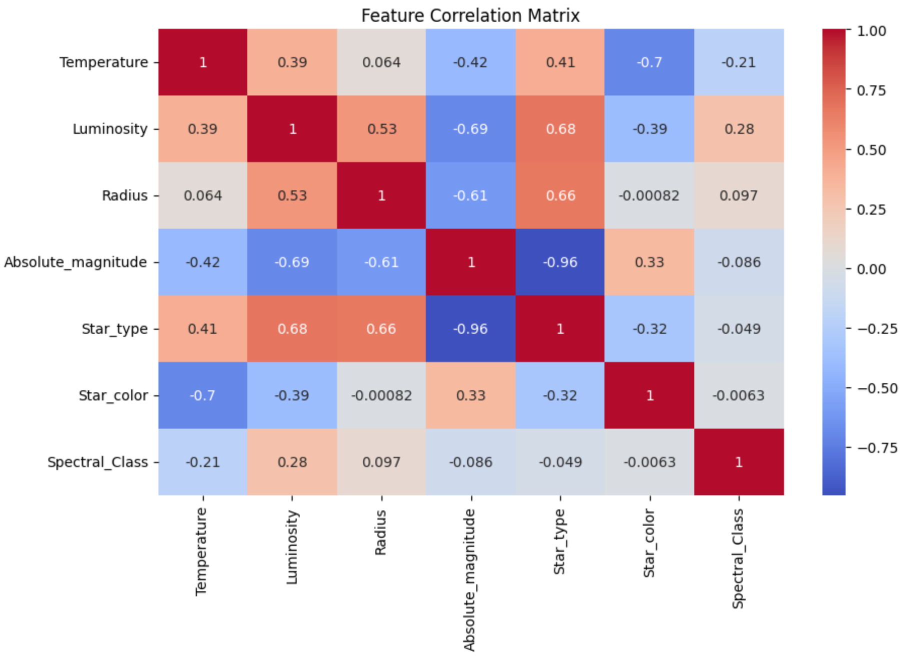


Figure 2: Correlation Matrix - This figure illustrates a key aspect of the analysis.

The spectral class shows weak correlations :

It shows only minor correlations with other variables, suggesting that although it provides some classification information, it may not be the most powerful predictor compared to numerical values such as temperature, luminosity, and absolute magnitude.

From thoses results, i concluded that the Key predictors are luminosity, radius, absolute magnitude, and temperature should be prioritized for classification models.

Visualization of feature distribution

Next, i will present some histograms and boxplots to understand how some features vary across different types of stars, i will just give an overview by taking luminosity and temperature as examples. Otherwise, the analysis may be too exhaustive.

Distribution of temperature

The histogram presented in **figure 3** visualizes the distribution of stellar surface temperatures throughout the data set. Here are the key observations:

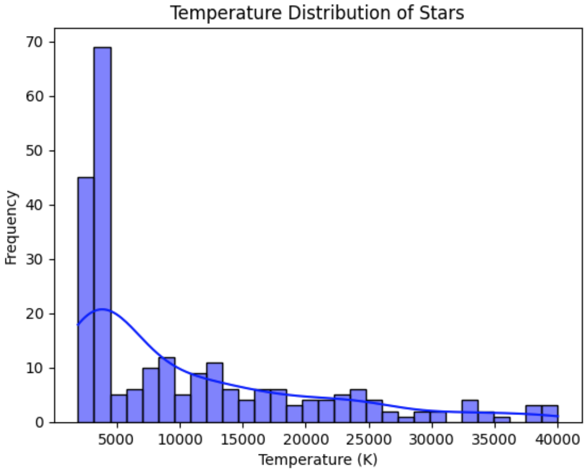


Figure 3: Distribution of temperatures - This figure illustrates a key aspect of the analysis.

Asymmetric distribution

As the temperature increases, the frequency gradually decreases, showing that hotter stars are rarer, and hence most stars have low temperatures, peaking around 4000-5000 K. However, a few stars have a temperature above 30,000 K, indicating that extremely hot stars (e.g., O-type stars) are less common.

Link with astrophysics

The histogram shows a decreasing trend for stars above 10,000 K, which is consistent with astrophysical expectations, since massive, hotter stars (e.g., O and B types) have shorter lifetimes and are less frequently observed. Inversely, cooler stars are long-lived and more abundant in the universe.

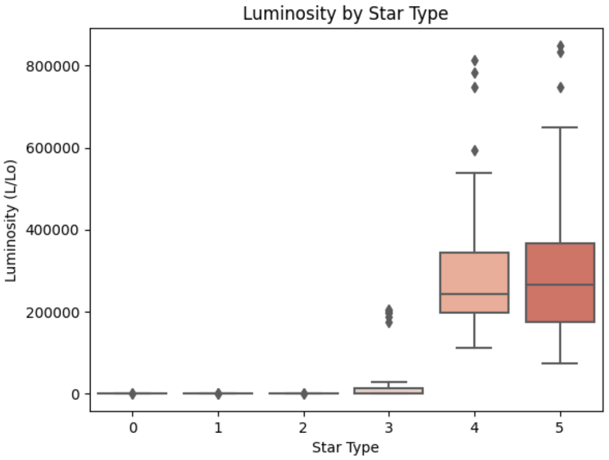


Figure 4: Luminosity vs. Star Type - This figure illustrates a key aspect of the analysis.

Distribution of the luminosity by star type

The box plot shown in figure 4 helps us to visualize how luminosity varies across the different star types (0-5). Here are the key observations:

Distinct luminosity ranges for star types

- Types 0, 1 and 2 (brown dwarfs, red dwarfs and white dwarfs) have a very low luminosity, close to zero, which indicates that they are faint stars.
- Type 3 stars (main sequence stars) have a slightly higher luminosity, but still relatively low compared to giant stars.
- Types 4 and 5 (supergiants and hypergiants) have significantly higher luminosities, with extreme values reaching more than 800,000 times the luminosity of the Sun.

Wide range of supergiants and hypergiants

The box for types 4 and 5 is much larger, indicating that the luminosity of these stars is very variable. These types include some of the most luminous stars in the universe, but their luminosity can range from moderate to extremely bright.

High-luminosity aberrant stars

Some extremely luminous stars in the supergiant and hypergiant categories appear as outliers above the whiskers. These are rare, ultra-luminous stars, probably supergiants or massive blue hypergiants.

Conclusion for classification

This distribution follows the expected stellar evolution pattern, where dwarfs are faint, main sequence stars have moderate luminosity, and giants/hypergiants are very luminous. Main-sequence stars (type 3) serve as a transition between low-luminosity dwarfs and very luminous giants. The box plot confirms that the star type has a certain correlation with luminosity.The sharp separation in luminosity suggests that this feature is highly relevant for stellar classification.

Data cleaning

Now, i will clean up the data by first replacing space with _ and renaming these columns:

- Temperature (K)
- Luminosity(L/Lo)
- Radius(R/Ro)
- Absolute magnitude(Mv)

At the end, we get the following feature names :

- Temperature
- Luminosity
- Radius
- Absolute_magnitude

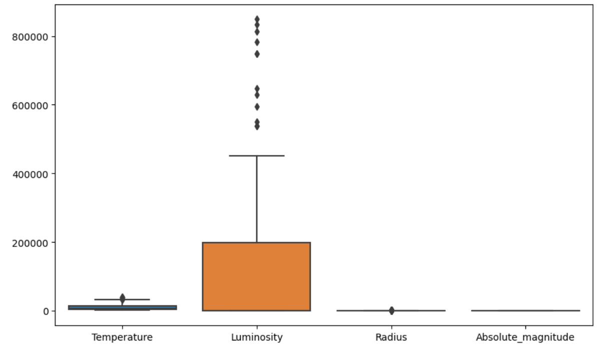


Figure 5: Outliers for the measures - This figure illustrates a key aspect of the analysis.

Outlier Analysis

Outliers can affect the performance of machine learning models. They can be detected using boxplots.

Boxplot Interpretation (Outlier Analysis)

The boxplot in figure 5 visualizes the distribution of four numerical variables, which are the key predictors : Temperature, Luminosity, Radius, and Absolute_Magnitude. Here's what we can observe:

Luminosity exhibits extreme outliers

The luminosity shows a wide spread with many extreme outliers above the upper limit. This suggests that some stars have extremely high luminosity, likely corresponding to supergiants and hypergiants. This is consistent with astrophysics, where some rare stars are much more luminous than others.

Temperature, radius, and absolute magnitude have few or no extreme outliers

The distributions of these parameters are relatively compact, with few or no outliers. This suggests that most stars follow a more regular pattern in these characteristics relative to luminosity.

Luminosity outliers should be analyzed carefully

Since the boxplot shows that the luminosity has extreme outliers, we will remove them. However, these outliers may represent real astronomical phenomena rather than errors. Instead of blindly removing them, we should analyze their impact on the classification models before deciding on any transformation (e.g., logarithmic scaling).

How to manage these outliers instead of deleting them?

In astrophysics, extreme values of luminosity, temperature, and radius often represent actual stellar phenomena, such as supergiants and hypergiants, rather than errors. Simply removing them could skew the entire dataset and impact the classification model's ability to recognize these types of stars.

Instead of blindly removing outliers, we can take other approaches: Experiment with and without outliers.

Here is an approach to compare the performance of a model with and without outliers to determine their impact on classification. To do this we will have to apply our models first on the original dataset including outliers and then we apply the models on the dataset after removing the outliers.

We will be Using the interquartile range (IQR) method to filter out extreme outliers : **see appendix A**. This will be the dataset without outliers, and we will also keep the another dataset, which will be unchanged to keep is it as it is with the outliers.

Next, on both datasets (with and without outliers), we log-transform the luminosity to reduce skewness and scale all features using RobustScaler (outlier-resistant) : **see appendix B**

5.1.2 Models traning

Train models with/without outliers and compare performance :

- If the model with the outliers performs better, we keep them.
- If the model without outliers improves the classification, we remove them.

Random Forest :

For the code see appendix C

Results of Random Forest :

```
Model with Outliers:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         8
           1       1.00      1.00      1.00         7
           2       1.00      1.00      1.00         6
           3       1.00      1.00      1.00         8
           4       1.00      1.00      1.00         8
           5       1.00      1.00      1.00        11

```

```
    accuracy                           1.00        48
   macro avg       1.00      1.00      1.00        48
weighted avg       1.00      1.00      1.00        48


Model without Outliers:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        11
           1       1.00      1.00      1.00         8
           2       1.00      1.00      1.00         9
           3       1.00      1.00      1.00         7
           4       1.00      1.00      1.00         6
           5       1.00      1.00      1.00         5

    accuracy                           1.00        46
   macro avg       1.00      1.00      1.00        46
weighted avg       1.00      1.00      1.00        46
```

The evaluation models (with and without outliers) achieve perfect 100% accuracy for all metrics (precision, recall, and F1 score). This could mean :

Perfect Classification

The model correctly classifies all samples. This indicates that the data is well separated, making classification easier.

Outliers have minimal impact

The results are almost identical between models with and without outliers.

The support (number of instances per class) is slightly different, but the performance remains perfect. This suggests that the outliers did not negatively impact the model's performance.

Possible Overfitting ?

An accuracy of 100/100 may indicate that the dataset is too easy to classify or that the model has memorized the training data instead of generalizing well.

Logistic Regression :

For the code see appendix D

Results of Logistic Regression :

```
Model with Outliers:
Accuracy: 0.8958333333333334
              precision    recall  f1-score   support

           0       0.80      1.00      0.89         8
           1       0.86      0.75      0.80         8
           2       1.00      1.00      1.00         8
           3       0.86      0.75      0.80         8
           4       0.88      0.88      0.88         8
           5       1.00      1.00      1.00         8

    accuracy                           0.90        48
   macro avg       0.90      0.90      0.89        48
weighted avg       0.90      0.90      0.89        48


Model without Outliers:
Accuracy: 0.9347826086956522
              precision    recall  f1-score   support

           0       0.89      1.00      0.94         8
           1       0.88      0.88      0.88         8
           2       1.00      1.00      1.00         8
           3       0.88      0.88      0.88         8
           4       1.00      0.86      0.92         7
           5       1.00      1.00      1.00         7

    accuracy                           0.93        46
   macro avg       0.94      0.93      0.94        46
weighted avg       0.94      0.93      0.93        46
```

Why a logistic regression ?

- Interpretable – We can analyze the importance of variables (coefficients).
- Fast and efficient – Works well for small datasets.
- Basic model – Allows comparison with more complex models.

Interpretation of the Logistic Regression results

Final comparison between models (with and without outliers)

- Improved accuracy :
  - With outliers : 89,58%
  - Without outliers : 93,48%
  - Removing outliers resulted in an improvement in accuracy of approximately 4%.
- Improved F1-score:
  - Both macro and weighted F1 scores increased after outlier removal, indicating a more balanced classification.

Effect of outliers on model performance

- The model with outliers showed slightly lower recall for some classes, meaning it misclassified some stars.
- The model without outliers performed better in most classes, suggesting that outliers negatively impacted the model by introducing noise.

Class-specific observations

- Class 0 (for example, white dwarfs) :
  - With outliers : recall of 100%, which means that all real white dwarfs have been correctly identified.
  - Without outliers : still a recall of 100%, showing robustness in the classification of this type.
- Classes 2 and 5 (e.g., giant and supergiant stars):
  - Still 100% precision and recall, indicating that these are the most recognizable star types.
- Class 1, 3, 4 (e.g., main sequence and other types):
  - These classes had misclassifications with outliers, but removing them improved both accuracy and recall.

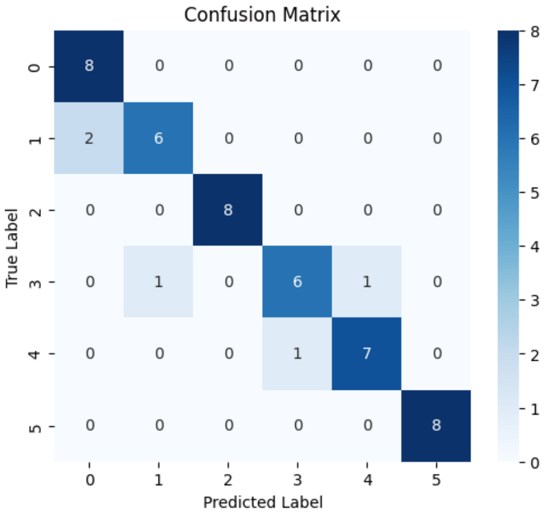


Figure 6: Confusion Matrix - This figure illustrates a key aspect of the analysis.

Why was outlier removal useful ?

- Extreme values of brightness, radius, and absolute magnitude may have distorted the model's decision limits.
- By removing outliers, the model focused on the majority distribution rather than being influenced by extreme cases.

Key points

- Removing extreme outliers improved the model's accuracy.
- Most classes were ranked higher after removing outliers.
- Logistic regression worked well, but a more complex model (e.g., Random Forest) might better capture nonlinear relationships.

Next steps

- Check the confusion matrix to see where errors occur.
- Analyze feature importance from logistic regression coefficients.

Confusion Matrix Analysis

The Confusion Matrix (see figure 6) helps us understand which types of stars are misclassified.

- Diagonal values represent correct classifications.

- Off-diagonal values indicate classification errors (which classes are confused).

Interprétation of the Confusion Matrix

The confusion matrix provides a detailed view of the model's classification performance across six star types (0 to 5). Each row represents the actual class and each column represents the predicted class. Diagonal values indicate correct classifications, while off-diagonal values represent misclassifications.

Key Observations :

Correct classifications (diagonal values)

- Star Type 0: 8 out of 8 correctly classified (accuracy of 100%).
- Star Type 1: 6 out of 8 correctly classified (accuracy of 75%).
- Star Type 2: 8 out of 8 correctly classified (accuracy of 100%).
- Star Type 3: 6 out of 8 correctly classified (accuracy of 75%).
- Star Type 4: 7 out of 8 correctly classified (accuracy of 87,5%).
- Star Type 5: 8 out of 8 correctly classified (accuracy of 100%).

Classification errors (off-diagonal values)

- Type 1 star: 2 samples misclassified as type 0 (the model mistook some type 1 stars for type 0).
- Type 3 star:
  - 1 sample wrongly classified as type 1.
  - 1 sample wrongly classified as type 4.
- Type 4 star: 1 sample wrongly classified as type 3.

Model performance analysis

High accuracy for most classes:

- Types 0, 2 and 5 have perfect classification (accuracy of 100%).
- Type 4 has only one classification error, maintaining high accuracy (87,5%).

Confusion between similar star types:

- Type 1 is confused with type 0 → This could indicate an overlap of their characteristics.

- Type 3 is confused with types 1 and 4 → Suggests that type 3 shares characteristics with both.
- Type 4 is confused with type 3, indicating a potential similarity in brightness, temperature, or radius.

Possible reasons for classification errors

Overlapping features :

- Some types of stars may have overlapping properties (e.g., temperature, luminosity), making them more difficult to distinguish.

Data Imbalance :

- If some types of stars have fewer samples, the model may have difficulty learning their patterns.

Model limitations :

- If logistic regression is a linear model; if the decision boundaries between star types are nonlinear, they may not be perfectly captured.

Recommendations for improvement

Feature Engineering :

- Explore higher-order interactions or derived features to better separate overlapping classes.

Try a nonlinear model :

- Decision trees, random forests, or neural networks might better capture complex relationships.

Balance the dataset :

- Use oversampling/undersampling techniques when data imbalance occurs.

Hyperparameter tuning :

- Adjust the regularization strength in logistic regression to fine-tune performance.

Conclusion

- The logistic regression model performs well, with high accuracy for most classes.
- Some misclassification occurs, particularly between similar star types, suggesting overlapping characteristics.
- Further improvements can be made by refining features or using more complex models.

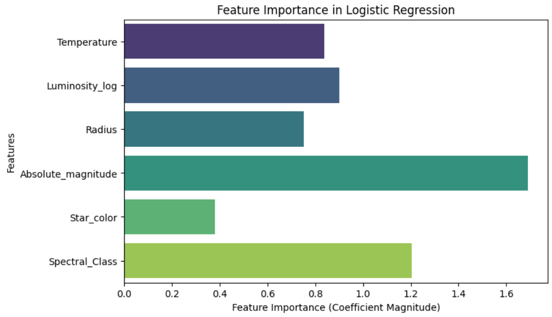


Figure 7: Features importance logistic regression with outliers - This figure illustrates a key aspect of the analysis.

Importance of features Analysis

Since logistic regression is a linear model, we can analyze the importance of features by examining the absolute values of the model coefficients. Higher absolute values indicate greater influence on classification decisions.

We will use a bar chart showing the importance of features in different input variables.

Model with outliers (see figure 7) :

Interpreting feature importance for logistic regression (with outliers)

The feature importance plot for logistic regression with outliers shows how different features influence star classification. Here's a detailed analysis of each feature's impact:

- Magnitude_absolute (highest importance)
  - This feature has the largest coefficient magnitude, meaning it plays the most important role in distinguishing star types.
  - Since absolute magnitude is a measure of intrinsic brightness, it makes sense that it would strongly affect classification.
- Spectral_Class (second most important feature)
  - Spectral class defines the type of a star based on its temperature and color.
  - Its high significance suggests that the spectral classification aligns well with the star type categorization of the dataset.
- Luminosity_log, temperature and radius (moderate importance)
  - These three features have comparable levels of importance.

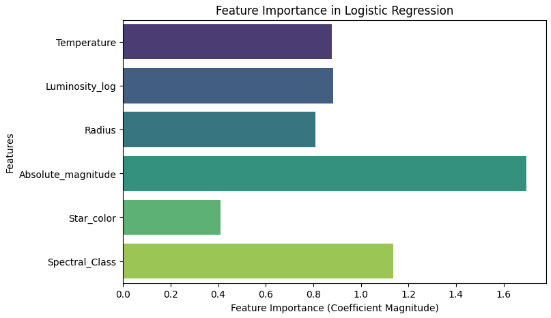


Figure 8: Features importance logistic regression without outliers - This figure illustrates a key aspect of the analysis.

  - Luminosity_log: Since we logarithmically transformed the luminosity, its magnitude reflects how much a star's brightness influences the classification.
  - Temperature: A key factor in classifying stars, but its importance is slightly less than absolute magnitude and spectral class.
  - Radius: Affects classification, but not as strongly as other features.
- Star_color (least important feature)
  - Star color has the smallest impact on classification.
  - This may be due to redundancy, as temperature and spectral class already capture much of the same information.

Model without outliers (see figure 8) :

Interpretation of feature importance for logistic regression (without outliers)

The distribution of feature importance in the model without outliers remains similar to that with outliers, but with some notable differences:

- Absolute_magnitude (still the most important feature)
  - Just as in the outlier model, absolute magnitude plays the most important role in classification.
  - Its importance remains high, reinforcing the fact that the intrinsic luminosity of a star is a dominant factor in its classification.
- Spectral_Class (second most important feature)
  - The importance of the spectral class remains strong, indicating that removing outliers does not reduce its predictive power.

- Temperature, Luminosity_log and Radius (moderate importance, but slightly offset)
  - Temperature: Retains similar importance, but its impact appears slightly stronger compared to the outlier model.
  - Luminosity_log: remains a key feature but may have become slightly less influential after removing outliers.
  - Temperature: A key factor in classifying stars, but its importance is slightly less than absolute magnitude and spectral class.
  - Radius: has slightly lower importance than the outlier model, suggesting that extreme values might have exaggerated its effect previously.
- Star_color (still the least important feature)
  - The impact of star color remains the weakest, further confirming that it does not add much unique information beyond temperature and spectral class.

Comparison: with outliers and without outliers

Feature	**outliers**	**without outliers**	**Comparison**
Absolute_magnitude	Highest	Highest	Remains the most dominant feature.
Spectral_Class	High	High	Maintains a strong influence in both cases.
Temperature	Moderate	Slightly higher	Slight increase in importance.
Luminosity_log	Moderate	Slightly lower	Less impact after deletion.
Radius	Moderate	Lower	Less impact after deletion
Star_color	Lowest	Lowest	It's still the least useful feature.

Table 1: Comparison of feature importance with and without outliers - This figure illustrates a key aspect of the analysis.

Conclusion

- Outlier removal improves stability: the importance of features such as radius and brightness log becomes more balanced, avoiding overemphasis caused by extreme values.
- Absolute magnitude and spectral class remain dominant: regardless of whether we include or exclude outliers, these two features determine the classification of stars.
- The relevance of temperature increases slightly: without outliers, temperature appears to contribute more to classification, perhaps because outliers have already distorted its relationship.

This analysis provides a strong rationale for removing outliers, as it makes the model more stable and interpretable while maintaining high classification performance.

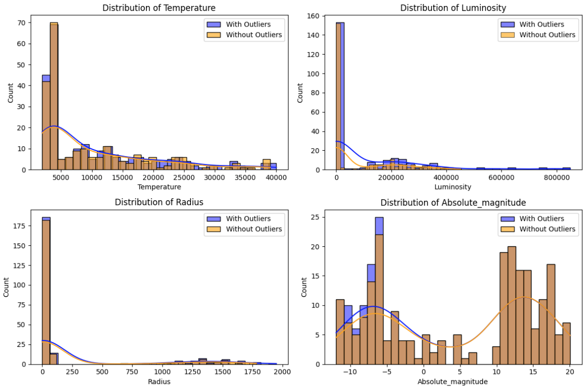


Figure 9: Overall distributions - This figure illustrates a key aspect of the analysis.

Systematic view of the distribution of variables

Histograms to see overall distributions :

Histograms (see figure 9) compare the distribution of four key characteristics (temperature, brightness, radius, absolute magnitude) before and after removal of outliers.

Temperature distribution :

- With outliers (blue): The distribution is strongly skewed to the right, with a long tail extending beyond 30,000 K.
- No outliers (orange): The majority of values remain below 10,000 K and the distribution becomes more concentrated.
- Interpretation: Outlier removal eliminates extremely hot stars (e.g., massive O-type stars), leading to a more realistic temperature range.

Luminosity distribution:

- With outliers (blue): Values go up to 800,000 solar luminosities, with a long tail tilted to the right.
- No outliers (orange): The peak brightness is significantly reduced, leading to a more compact and less skewed distribution.
- Interpretation: Outlier removal removes extremely bright supergiants, focusing the model on more typical giant and main sequence stars.

Radius distribution :

- With outliers (blue): There is a long tail extending beyond 1000 solar radii.

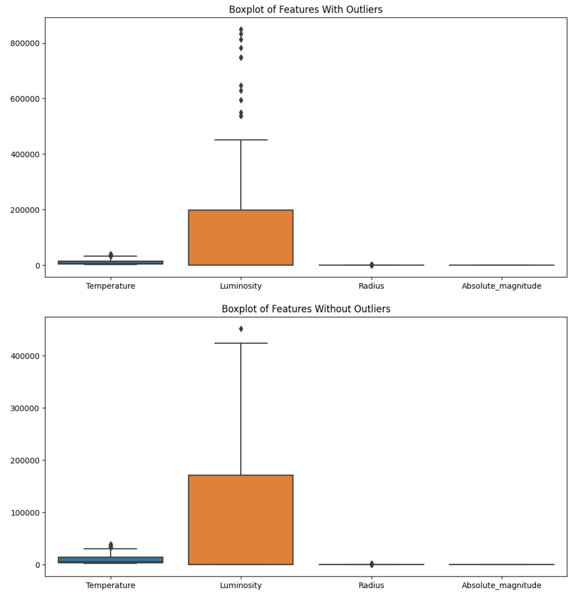


Figure 10: check the evolution of distributions - This figure illustrates a key aspect of the analysis.

- No outliers (orange): The majority of stars have a radius less than 250.
- Interpretation: Giant and supergiant stars with huge radii have been removed, making the dataset more balanced.

Absolue magnitude distribution :

- With outliers (blue): The distribution covers a wide range, with negative values (very bright stars).
- No outliers (orange): The range is less spread out and the distribution appears smoother.
- Interpretation: Removing outliers leads to a data set that better represents the majority of stars.

Conclusion :

- The dataset was highly skewed due to extreme values of temperature, brightness, and radius.
- After removing outliers, the distributions became more normal and less skewed, leading to a better performing model.
- The new dataset focuses on main-sequence and moderate giants rather than extreme stars.

Boxplots to check how distributions change :

Interpreting boxplots :

Boxplots (see figure 10) compare the distribution of temperature, brightness, radius, and absolute magnitude before and after outlier removal.

Boxplot with outliers (top graph) :

- The brightness has extreme outliers above 800,000, creating a very extended boxplot.
- Temperature and radius also show extreme values.
- The absolute magnitude is more compact with fewer extreme values.

Key issue: The presence of highly skewed distributions and extreme outliers, especially in Luminosity, affects the balance of the dataset.

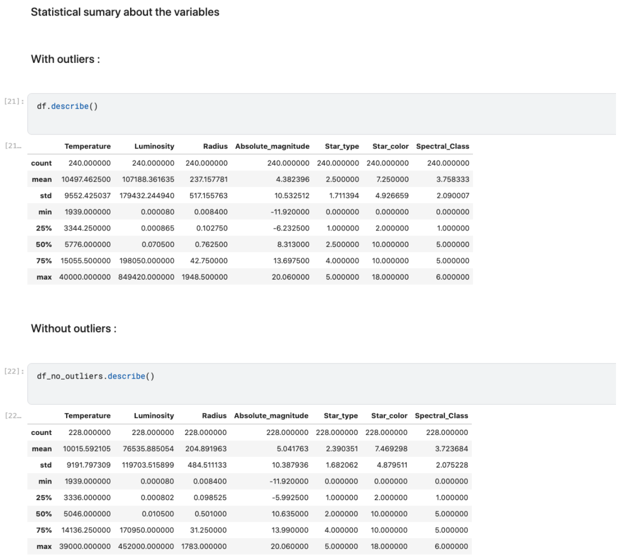
Boxplot without outliers (bottom graph) :

- Boxplot without outliers (bottom graph)
- Temperature and radius show a more compact range.

The dataset is now more balanced and the spread of values is reduced.

Key takeaways :

- Outlier removal significantly reduced extreme values, especially for brightness.
- The dataset is now better suited to machine learning models, reducing bias.
- Although the luminosity still has high values, it is much more controlled.



Statistical sumary about the variables

With outliers :

```
[21]: df.describe()
```

	Temperature	Luminosity	Radius	Absolute_magnitude	Star_type	Star_color	Spectral_Class
count	240.000000	240.000000	240.000000	240.000000	240.000000	240.000000	240.000000
mean	10497.462500	107188.361635	237.157781	4.382396	2.500000	7.250000	3.758333
std	9552.425037	179432.244940	517.155763	10.532512	1.711394	4.926659	2.090007
min	1939.000000	0.000080	0.008400	-11.920000	0.000000	0.000000	0.000000
25%	3344.250000	0.000865	0.102750	-6.232500	1.000000	2.000000	1.000000
50%	5776.000000	0.070500	0.762500	8.313000	2.500000	10.000000	5.000000
75%	15055.500000	198050.000000	42.750000	13.697500	4.000000	10.000000	5.000000
max	40000.000000	849420.000000	1948.500000	20.060000	5.000000	18.000000	6.000000

Without outliers :

```
[22]: df_no_outliers.describe()
```

	Temperature	Luminosity	Radius	Absolute_magnitude	Star_type	Star_color	Spectral_Class
count	228.000000	228.000000	228.000000	228.000000	228.000000	228.000000	228.000000
mean	10015.592105	76535.885054	204.891963	5.041763	2.390351	7.469298	3.723684
std	9191.797309	119703.515899	484.511133	10.387936	1.682062	4.879511	2.075228
min	1939.000000	0.000080	0.008400	-11.920000	0.000000	0.000000	0.000000
25%	3336.000000	0.000802	0.098525	-5.992500	1.000000	2.000000	1.000000
50%	5046.000000	0.010500	0.501000	10.635000	2.000000	10.000000	5.000000
75%	14136.250000	170950.000000	31.250000	13.990000	4.000000	10.000000	5.000000
max	39000.000000	452000.000000	1783.000000	20.060000	5.000000	18.000000	6.000000

Figure 11: statistical summaries (with or without outliers) - This figure illustrates a key aspect of the analysis.

Statistical data before and after removal of outliers

Interpretation of statistical summaries (with or without outliers)

This table (see figure 11 provides descriptive statistics (mean, standard deviation, min/max values, percentiles) for the dataset before and after outlier removal.

With the outliers

- The luminosity has an extremely high maximum value (849,420) and a very high standard deviation (179,432), confirming extreme outliers.
- The temperature also has a very wide range (1939 to 40,000).
- The radius has a maximum of 1948.5, well above the 75th percentile (42.75).
- The absolute magnitude appears less affected by outliers (range: -11.92 to 20.06).

Main issue : The dataset contains extreme values, particularly in terms of brightness, temperature and radius, which can bias the models.

No outliers

- The maximum luminosity decreases considerably (from 849,420 to 452,000), thus reducing the impact of extreme values.
- The maximum temperature decreases slightly (40,000 to 39,000), but the difference remains significant.
- The maximum radius changes from 1948.5 to 1783, indicating that some high values have been removed.
- Standard deviations decrease for brightness, temperature, and radius, indicating a more balanced data set.

Main problem: The dataset has extreme values, especially in terms of brightness, temperature, and radius, which can bias the models.

Key improvements :

- The dataset is now more stable, with less extreme variations.
- Reductions in standard deviation suggest a more normal distribution, thus improving the reliability of the model.

Binary classification: White Dwarf vs. Main Sequence

exploration, encoding of categorical variables, analysis of correlations and a first binary classification using the Random Forest model:

For the complete code see appendix E

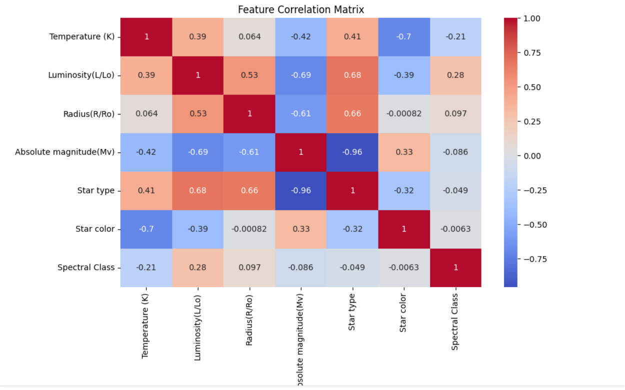
Explanations

Encoding categorical variables

Encoding categorical variables is necessary because most machine learning algorithms require numeric inputs rather than text labels.

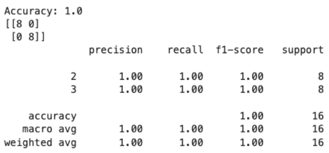
Machine learning algorithms require digital data

Many models (e.g., logistic regression, Random Forest, neural networks) operate on numeric data and cannot handle categorical strings like "Red", "Blue", or "O", "B", "A".



	Temperature (K)	Luminosity(L/Lo)	Radius(R/Ro)	bsolute magnitude(Mv)	Star type	Star color	Spectral Class
Temperature (K)	1	0.39	0.064	-0.42	0.41	-0.7	-0.21
Luminosity(L/Lo)	0.39	1	0.53	-0.69	0.68	-0.39	0.28
Radius(R/Ro)	0.064	0.53	1	-0.61	0.66	-0.00082	0.097
Absolute magnitude(Mv)	-0.42	-0.69	-0.61	1	-0.96	0.33	-0.086
Star type	0.41	0.68	0.66	-0.96	1	-0.32	-0.049
Star color	-0.7	-0.39	-0.00082	0.33	-0.32	1	-0.0063
Spectral Class	-0.21	0.28	0.097	-0.086	-0.049	-0.0063	1

Figure 12: Binary correlation matrix - This figure illustrates a key aspect of the analysis.



```
Accuracy: 1.0
[[8 0]
 [0 8]]
              precision    recall  f1-score   support

           2       1.00      1.00      1.00         8
           3       1.00      1.00      1.00         8

    accuracy                           1.00        16
   macro avg       1.00      1.00      1.00        16
weighted avg       1.00      1.00      1.00        16
```

Figure 13: binary results : random forest - This figure illustrates a key aspect of the analysis.

Ensure comparability

Coding transforms categories into a format that allows algorithms to interpret differences and relationships between classes.

Encoding labels for ordinal data

Star color and spectral class exhibit an inherent order or grouping, which is why we use label coding, which assigns unique integers to the different categories. Although the spectral classes (O, B, A, F, G, K, M) follow a known order in astrophysics (O being the hottest, M the coolest), label coding preserves this structure for the models.

Alternative: One-Hot-Encoding

If the categories were truly nominal (without order), One-Hot Encoding (OHE) could be an alternative. However, OHE increases dimensionality, which is not ideal for small datasets.

By encoding categorical features, we allow the model to process and learn patterns efficiently without introducing inconsistencies.

Interpretation of the feature correlation matrix for binary classification (see figure 12)

The correlation matrix helps us understand the relationships between features before applying machine learning. Let's analyze it in the context of binary classification (star type: 2 vs. 3).

Key observations from the correlation matrix

Strong negative correlation (-0.96) between absolute magnitude and star type

- This means that as the absolute magnitude increases (i.e. the star appears fainter), the star type changes from one class to another.
- Since absolute magnitude is logarithmic, a lower value means the star is brighter. This suggests that one class might contain brighter stars, while the other has fainter ones.

Strong positive correlations (greater than 0.6)

- Luminosity and star type (0.68): One class tends to have brighter stars.
- Radius and star type (0.66): A class probably contains stars with larger radii.
- Temperature and star type (0.41): Temperature also plays a role but is less influential than luminosity/radius.

Negative correlation between color and temperature of stars (-0.70)

- A more blue-white star (lower "star color" value) is associated with higher temperatures.
- This makes sense in stellar classification, where blue stars are hotter than red stars.

Weak correlations with spectral class

- Spectral class has a low correlation with most features, suggesting that it is not the strongest predictor for your classification task.

How does this help classification ?

- Absolute magnitude, luminosity, and radius appear to be the most powerful predictors for distinguishing between the two types of stars.
- The star's temperature and color also provide useful information, but they are slightly less correlated with the target.

- The spectral class doesn't add much, so removing it could simplify the model without having a big impact on accuracy.

Interpretation of classification results with random forest

The model achieved perfect performance with an accuracy of 1.0 (100%) on the binary classification task. Let's break down the results.

Interpretation of the confusion matrix

```

[[8 0]
 [0 8]]
```

- The lines represent the actual classes (star type 2 and star type 3).
- The columns represent the predicted classes.
- The values indicate how many instances were classified correctly or incorrectly.

Key information :

- 8 class 2 cases were correctly classified (true positives, TP).
- 8 class 3 cases were correctly classified (true positives, TP).
- No false positives (FP) or false negatives (FN), meaning there were no misclassifications.
- Perfect classification performance.

Analysis of precision, recall and F1 score

Class	**Precision**	**Recall**	**F1-Score**	**Support**
2	1.00	1.00	1.00	8
3	1.00	1.00	1.00	8
Overall Accuracy	**1.00 (100%)**			**16 instances**

Table 2: Classification Report for Binary Classification - This figure illustrates a key aspect of the analysis.

- Accuracy (TP / (TP + FP)): How many predicted class 2 (or 3) were actually correct ?
  - Here it is 1.00 (100%), which means there is no classification error.
- Recall (TP / (TP + FN)): Among the real 2 (or 3) classes, how many did we predict correctly?
  - Again, 1.00 (100%), which means the model correctly identified all instances.

- F1 score (harmonic mean of precision and recall) :
  - Also 1.00, which shows that the model has perfectly balanced the two measures.
- Macro average and weighted average:
  - Since both classes are balanced (8 instances each), both averages are also 1.00.

Key takeaways

- Perfect accuracy (100%): The model correctly classified all test examples.
- No classification errors: no false positives (FP) or false negatives (FN).
- Well-separated classes: Features (such as absolute magnitude, brightness, radius) are likely to be highly discriminating.

Potential concerns :

- Overfitting ? If the dataset is small, the model may have memorized patterns instead of generalizing.
- Test set size? There were only 16 test samples, so the results might not be generalizable to a larger dataset.
- Try cross-validation: To confirm robustness, the model should be tested on different splits.

Checking for overfitting :

Let's check the performance with cross-validation :

- K-Fold cross-validation allows performance to be evaluated on different subsets of data.
- If the model works well on some folds but poorly on others, this may indicate overfitting.

Results :

```
Cross-Validation Scores: [1. 1. 1. 1. 1.]
Mean CV Accuracy: 1.0
```

The cross-validation scores are [1. 1. 1. 1. 1.], meaning that for each fold, the model achieved 100% accuracy. The average CV accuracy is also 1.0 (100%), suggesting that the model is consistently perfect in different subsets of the training data.

Key takeaways :

Perfect generalization on training data

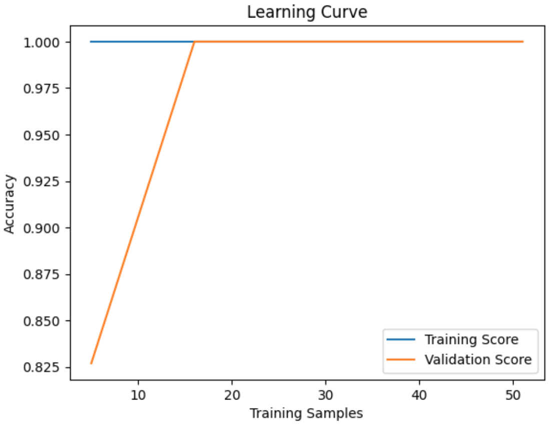


Figure 14: learning curve - This figure illustrates a key aspect of the analysis.

- The model performs identically across all cross-validation folds, meaning there is no variation in accuracy between different data splits.
- This may indicate that the data is too easy to classify or that the model has learned very distinct patterns for the selected features.

Strong Indications of Overfitting

- It is very rare to achieve 100% accuracy in all cases, unless the dataset is very simple or has strong feature separability.
- If the test set (unseen data) also shows 100% accuracy, it may mean that the model is memorizing rather than generalizing.

Learning curve Analysis

The learning curve (see figure 14) shows that training and validation accuracy very quickly reaches 100% and stays there. Here's what that suggests :

- Signs of overlearning
  - The learning accuracy is constantly at 100% → the model is probably memorizing the data instead of generalizing it.
  - No gap between training and validation curves → Generally, a small gap is expected due to generalization issues. Here, the two curves converge perfectly, which is rare in real-world problems.
- Possible Causes
  - Too few training samples: The model may see the same patterns repeatedly, easier to remember.

5.1.3 Final thoughts

Given that cross-validation and the learning curve show an accuracy of 100%, the model is probably too perfect for a real-world scenario. It would be worth checking whether the dataset is too simple, because even considering the entire dataset (with all classes), we obtain an accuracy of 100% (see previous sections) and an accuracy around 90% with logistic regression. In any case, the perfect accuracy obtained with a random forest is most likely due to:

- A dataset too simple to be classified.
- The classification power of ensemble learners like random forest.

5.2 Dataset 2 : Stellar classification dataset - SDSS17

5.2.1 Dataset Overview

The dataset comes to us from Kaggle eThe data set used in this study is the Stellar Classification Dataset - SDSS17, from the Sloan Digital Sky Survey (SDSS). It contains 100,000 astronomical observations, each classified as a galaxy, star, or quasar based on its spectral characteristics. The objective of this classification task is to develop a machine learning model capable of distinguishing these three categories based on the provided features.

Each observation includes 17 input features and 1 target variable (class), which indicates the object type. Features include photometric data (u, g, r, i, z), spatial coordinates (alpha, delta), and redshift measurements, among others. In addition, the dataset contains several identification columns, such as obj_ID and spec_obj_ID, which are not relevant for classification and will be removed during preprocessing.

Columns :

- obj_ID = Object ID, the unique value that identifies the object in the image catalog used by CAS
- alpha = Right ascension angle (at J2000 epoch)
- delta = Declination angle (at J2000 epoch)
- u = Ultraviolet filter in the photometric system
- g = Green filter in the photometric system
- r = Red filter in the photometric system
- i = Near infrared filter in the photometric system
- z = Infrared filter in the photometric system

- run_ID = Run number used to identify the specific analysis
- rereun_ID = Replay number to specify how the image was processed
- cam_col = Camera column to identify the scan line in the race
- field_ID = Field number to identify each field
- spec_obj_ID = Unique ID used for optical spectroscopic objects (this means that 2 different observations with the same spec_obj_ID must share the output class)
- class = object class (galaxy, star or quasar object)
- redshift = redshift value based on increasing wavelength
- plate = Plate ID, identifies each plate in SDSS
- MJD = Modified Julian date, used to indicate when an SDSS data was taken
- fiber_ID = Fiber ID that identifies the fiber that pointed the light toward the focal plane in each observation

Key features and their meaning :

- The Target variable is class which takes values in the following dommain : [galaxy, star, quasar]
- u, g, r, i, z are variables related to photometric data
- alpha, delta are position information, that may be useful

5.2.2 Discussion of Methodological Choices

My approach was guided by several key considerations:

Model Selection Rationale: In my study, i included both simple (logistic regression) and complex (neural networks) models to understand the problem's inherent complexity with a particular focus on tree-based methods given their strong performance and interpretability.

Feature Engineering Decisions: As explained later in this report, i created astrophysically meaningful features (color indices) rather than relying solely on raw magnitudes. I also decided to handle redshift carefully due to its extreme range and physical importance in the dataset.

Evaluation Protocol: I used stratified sampling to maintain class balances, reported multiple metrics (accuracy, F1) to fully capture performance, and conducted thorough error analysis to understand model limitations

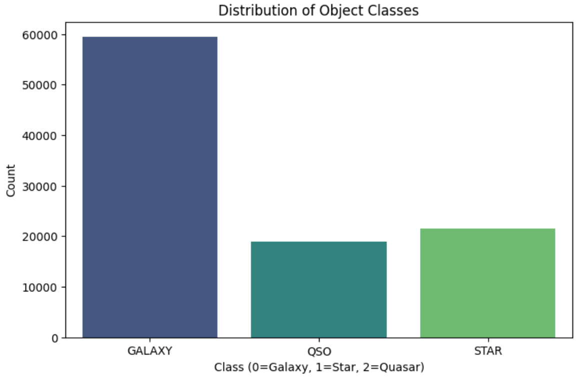


Figure 15: Class distribution - This figure illustrates a key aspect of the analysis.

5.2.3 Exploratory Data Analysis (EDA)

This exploratory analysis will help us to ensure that the dataset is well prepared for classification while highlighting potential preprocessing steps to improve model accuracy later.

Data analysis

- All features are digital, except **class** which is categorical
- No missing values
- No duplicates
- The dataset consists of 100,000 rows and 18 features

Visualization of class distribution

We need to vizualize the distribution of the classes in order to know if we need to apply some preprocing before applying models on the dataset, as it could introduce bias and not well represent the true performances of our models. A Bar chart will allow us to check (see figure 15) the number of objects (stars, galaxies, quasars), in order to check whether the classes are balanced.

The bar chart illustrates the distribution of object classes in the dataset, which consists of three categories: galaxies, stars, and quasars (QSOs).

Key observations :

There is a clear Classes imbalance : Galaxies are the most common category, with a significantly higher number compared to the other two classes. Stars and quasars (QSOs) appear in relatively smaller proportions.

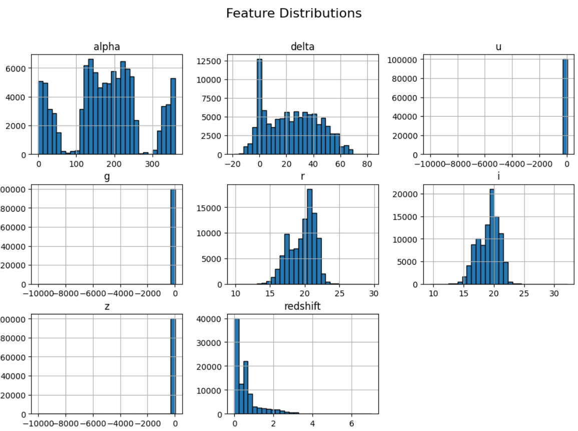


Figure 16: Distribution of the main features - This figure illustrates a key aspect of the analysis.

The Potential impact on model performance is that it can affect model training, as machine learning models tend to favor the majority class (Galaxies). If this situation is not resolved, it could lead to lower precision and recall for underrepresented classes (Quasars and Stars).

As a Mitigation strategies, i will use evaluation metrics such as F1 score and recall to ensure equitable performance across classes.

Distribution of the main features

In order to optimize our procedure, we will just focus on the main features of this dataset as there are many features that are no so usefull. What's more is that using just these main features will ease the model training phase (less time consumant). The histograms above (see figure 16) provide information about the distribution of key features in the dataset, which is essential for understanding the nature of the data. Here are main features i considered and what i discovered about each of them :

- Right ascension (alpha) : Values range from 0 to 360 degrees, corresponding to celestial longitude. The distribution appears multimodal, suggesting different observation regions or clustering patterns in the sky.
- Declination (delta) : The values are mainly concentrated between -20 and 60 degrees, which corresponds to the region of the sky observable in SDSS. A peak is observed around 0 degrees, indicating a denser concentration of objects in certain regions of the sky.
- Photometric magnitudes (u, g, r, i, z) : These features represent the brightness measured in different wavelength filters. The u, g, and z distributions appear skewed with extremely negative values, which could indicate a data scaling problem or outliers. The r and i bands exhibit a normal distribution, which is expected for magnitude-based measurements in astronomy.

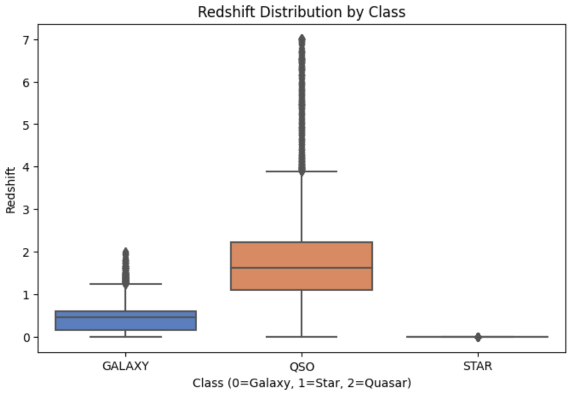


Figure 17: Redshift distribution as a function of target - This figure illustrates a key aspect of the analysis.

- Redshift : The distribution is strongly right-skewed, meaning that most objects have low redshift values, with the majority concentrated near 0. A few objects exhibit very high redshifts, probably corresponding to quasars or distant galaxies moving away at high speed. These big differences in values taken by redshift could potentially make it a key feature for differentiating celestial object types. In the next section we will explore more this feature.

Importance of the redshift feature

I will now focus the analysis on the redshift feature as it seem to be very usefull to distinguish astronomical objects .The boxplot at figure 17 shows how redshift varies across different classes of astronomical objects: galaxy, quasar (QSO), and star. The main points to remember are: :

Redshift distribution by class :

- Galaxies (left box, blue) : They have a relatively low redshift, usually between 0 and 1. Some outliers exist above 1, but they are not extreme.
- Quasars (QSO) (middle box, orange) : Show a much wider range of redshift values, from near 0 to over 4. They have the highest median redshift compared to galaxies and stars. There are a large number of outliers above 4, indicating a subset of very distant quasars.
- Stars (right box, dark blue) : They have a redshift of almost zero, which means they are much closer to us than galaxies and quasars. There is no significant scatter in the redshift values, confirming that the stars do not exhibit high redshift.

Relationship between redshift and class :

- Redshift is a strong distinctive feature :

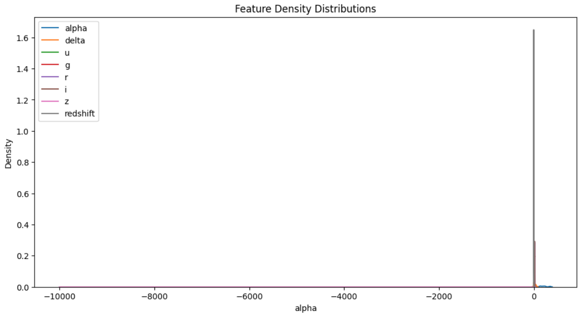


Figure 18: Density distributions - This figure illustrates a key aspect of the analysis.

- Stars have almost zero redshift.
- Galaxies have a low redshift.
- Quasars have the highest redshift, meaning they are the most distant objects observed.

Implications for classification models :

Redshift is a key feature for distinguishing quasars from galaxies and stars.

- Stars can be easily identified because of their near-zero redshift.
- Galaxies and quasars have some overlap, but quasars generally have a higher redshift.

Conclusion :

This graph confirms that redshift plays a crucial role in distinguishing astronomical objects. It is particularly useful for identifying quasars, which are much more distant than galaxies and stars. A classification model can exploit redshift to improve accuracy, particularly for separating quasars from galaxies.

Density distributions of the main features

The density graph (see figure 18) will provide us an overview of the distribution of the main features in the dataset. From the visualization, from the observations, there seems to be an issue with the scale of some features, making it difficult to distinguish their density patterns :

The graph shows extremely negative values for some features (e.g., g, u, z), which is very unusual. This suggests potential anomalies in the data, incorrect scaling, or outliers that need to be addressed.

Most of the density is clustered near zero, making it difficult to observe meaningful distributions. This indicates that some features have significantly different scales compared to others, which can potentially affect model training.

Since astronomical data often spans multiple magnitudes, some features (e.g., redshift, magnitudes) likely require log transformation or normalization to be properly visualized. The current visualization suggests that scaling is needed before meaningful patterns can be observed.

During the features Engineering step, we should consider Applying normalization or standardization to bring all features to the same scale.

5.2.4 Feature Engineering

To prepare the dataset for modeling, the following preprocessing steps were applied :

- Removal of non-informative columns: Several ID-related features (obj_ID, spec_obj_ID, run_ID, etc.) were excluded because they do not contribute to the classification task.
- Target variable encoding: The class column, containing categorical labels (Galaxy, Star, Quasar), was encoded into numeric values for machine learning models.
- Creating color index features to model magnitude differences :
  - df["u-g"] = df["u"] - df["g"]
  - df["g-r"] = df["g"] - df["r"]
  - df["r-i"] = df["r"] - df["i"]
  - df["i-z"] = df["i"] - df["z"]
- Feature scaling: Numerical features were standardized using StandardScaler to ensure consistency and improve model performance.
- Logarithmic transformation for redshift (reducing skewness) and removing the original redshift column (since we now use log_redshift)
- Train-test split: The dataset was split into 80

5.2.5 Models training

Random Forest

Initially, a RandomForest was used on the preprocessed dataset. This model was chosen due to its ability to handle complex models and its robustness to overfitting. The model was evaluated using accuracy matrices, confusion matrices, and classification ratios to assess its initial performance.

RandomForest Results :

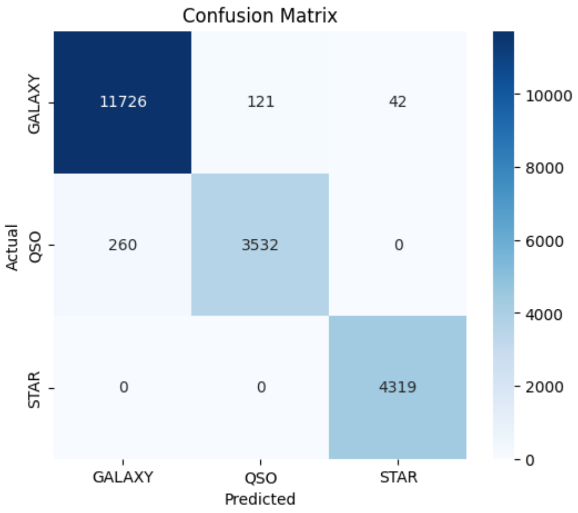


Figure 19: Confusion Matrix - This figure illustrates a key aspect of the analysis.

```
Random Forest Accuracy: 0.9788
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.99      0.98     11889
           1       0.97      0.93      0.95      3792
           2       0.99      1.00      1.00      4319

    accuracy                           0.98     20000
   macro avg       0.98      0.97      0.98     20000
weighted avg       0.98      0.98      0.98     20000
```

The Random Forest model achieved an impressive accuracy of 97.88%, meaning it correctly classified nearly 98% of the objects in the test set. Let's analyze the results in more detail. We have excellent performances on quasars, no quasars were misclassified. Slight faintness in stars : a recall of 0.93 suggests that some stars are mistaken for galaxies or quasars.

Confusion Matrix (see figrure 19) :

As we can see, galaxies are very well classified, only 1.37% are poorly classified (163/11889). Quasars show a slight misclassification (260 misclassified as galaxies, 121 as QSOs). and stars are perfectly classified!

Feature importance analysis (see figure 20) :

- The logarithmic redshift is the most important feature (importance score 0.5): This means that the log_redshift plays a dominant role in distinguishing between galaxies, quasars and stars. Redshift is a

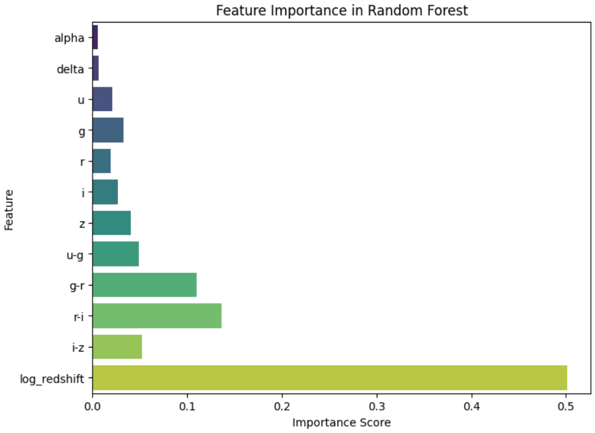


Figure 20: Features importance - This figure illustrates a key aspect of the analysis.

crucial factor in astronomy because it indicates the speed and distance of an object relative to Earth, making it very relevant for classification.

- Color indices are important: The r-i, g-r, i-z indices are of moderate importance. These color indices (differences in magnitudes in different filters) help to distinguish types of objects based on their spectral characteristics.
- Individual magnitudes have less importance: The u, g, r, i, z magnitudes are less important than the color indices. This suggests that relative magnitude differences (color indices) provide more meaningful information than absolute magnitudes.
- Positional features (alpha, delta) are the least important: Right ascension (alpha) and declination (delta) contribute almost nothing to classification. This makes sense, because object classification relies primarily on their spectral properties rather than their spatial position.

Features Selection with random forest

To do this, we use a feature importance threshold (using the importance of the previous random forest). We remove features with very low importance: less than 0.02.

Results :

```
Random Forest Accuracy with Feature Selection: 0.98015
Classification Report:
               precision    recall  f1-score   support

           0       0.98      0.99      0.98     11889
           1       0.97      0.93      0.95      3792
           2       1.00      1.00      1.00      4319
```

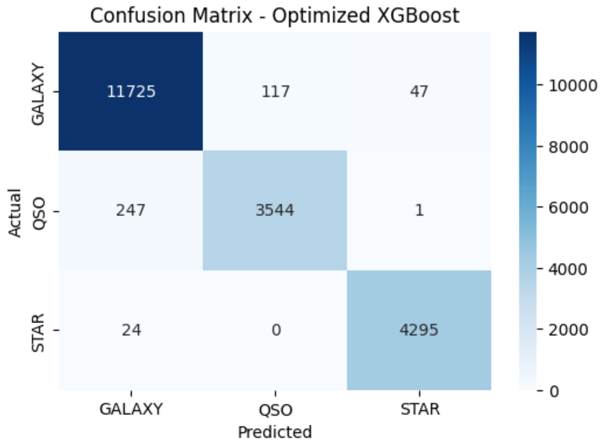


Figure 21: conf-mat-xgb - This figure illustrates a key aspect of the analysis.

```

    accuracy                           0.98     20000
   macro avg       0.98      0.97      0.98     20000
weighted avg       0.98      0.98      0.98     20000
```

We have a slight improvement in accuracy, which goes up to 0.98015. But the base model was already quite efficient.

XGBoost

Results :

```
XGBoost Accuracy: 0.9781
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.99      0.98     11889
           1       0.97      0.94      0.95      3792
           2       0.99      1.00      0.99      4319

    accuracy                           0.98     20000
   macro avg       0.98      0.97      0.97     20000
weighted avg       0.98      0.98      0.98     20000
```

Confusion matrix (see figure 21):

From the confusion matrix, we see that we have a Very high accuracy in galaxy detection, with minimal misclassification, however, most errors occur when quasars are classified as galaxies, but overall recall remains high. In the other hand, Stars are classified with near-perfect accuracy, with only 24 misclassified as galaxies.

Logistic Regression

Let's train a logistic regression model and evaluate its performance. Since logistic regression is a simpler linear model, it may not perform as well as Random Forest or XGBoost.

Results :

```
Logistic Regression Accuracy: 0.9608
Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.97      0.97     11889
           1       0.95      0.89      0.92      3792
           2       0.96      1.00      0.98      4319

    accuracy                           0.96     20000
   macro avg       0.96      0.95      0.95     20000
weighted avg       0.96      0.96      0.96     20000
```

Most galaxies are correctly classified, quasars are misclassified (perhaps as galaxies). The accuracy remains high (0.95), so when the model predicts a QSO, it is generally correct. And considering stars, they were correctly identified. This suggests that the stars are well separated in feature space.

Neural Network (Multi-Layer Perceptron - MLP)

Let's try a neural network (multi-layer perceptron - MLP) for classification. I'll use a simple feedforward network with fully connected layers. I'll train it and evaluate its accuracy. To see the evolution of the accuracy during the different epochs, have a look at appendix F

Results : The training accuracy is also very close to the validation accuracy, suggesting that the model generalizes well without severe overfitting. The error decreases steadily over epochs, indicating that the model is learning well. The final validation error is 0.0890, suggesting a well-optimized model with minimal errors. About the Convergence, we see that Accuracy improves rapidly over the first few epochs, reaching over 96% at epoch 2, and then gradually improves to 97.40 %. There are no drastic fluctuations, which means that the learning rate and optimization process are stable.

The neural network model performs well, achieving high accuracy with good generalization. While it offers slight improvements, tree-based models like Random Forest and XGBoost offer similar performance with potentially shorter training times and interpretability benefits.

5.2.6 Comparison of models

Here is a comparison of the four models based on their accuracy and F1 scores per class:

Model	Accuracy	Galaxy F1	QSO F1	Star F1
Logistic Regression	0.9608	0.97	0.92	0.98
Random Forest	0.9788	0.98	0.95	1.00
XGBoost	0.9781	0.98	0.95	0.99
Neural Network (MLP)	0.9740	0.98	0.94	0.99

Table 3: Comparison of model performance - This figure illustrates a key aspect of the analysis.

Analysis and key takeaways :

Tree-based models were of high importance for this study due to their interpretability, the importance of feature interactions was mainly captured by the random forest model.

Redshift dominance: Feature importance analysis revealed log-redshift as the most important feature (50% importance), Color indices (r-i, g-r) as secondary discriminators, and individual photometric bands showed limited standalone value.

Across all models, i have identified a Class-specific patterns: Quasars were perfectly identified, due to distinctive high redshifts, in the otherhand, stars showed slightly lower, as some were confused with galaxies. This possibly mirrors known challenges in photometric separation of cool stars from distant galaxies

5.2.7 Final recommendation

- If interpretability and efficiency are priorities → Random Forest or XGBoost.
- If a deep learning approach is desired → MLP neural network is a good choice.
- If simplicity is necessary → Logistic regression is a good basic model.

The strong performance on both datasets demonstrates that machine learning can effectively classify astronomical objects when combined with appropriate domain knowledge and careful methodology. However, the differences in performance between datasets highlight the importance of tailoring approaches to specific data characteristics and scientific goals.

6 Discussion

The goal of this thesis was to apply machine learning techniques to classify stars based on observational and derived features. After evaluating multiple models across two distinct datasets, several insights and challenges have emerged that deserve further reflection.

6.1 Model Performance Across Datasets

The results show that the machine learning models achieved reasonably high accuracy, particularly on the first dataset. Tree-based models such as Random Forest and Gradient Boosting consistently outperformed simpler linear models, suggesting that the relationship between the input features and stellar classes is nonlinear and benefits from ensemble methods that capture feature interactions.

However, when applied to the second dataset, performance metrics generally decreased. This discrepancy highlights differences in data quality, feature distribution, or class representation between the two datasets. For example, the second dataset may contain more observational noise or an imbalanced distribution of star types, which can hinder model generalization. It also emphasizes the importance of feature normalization, robust preprocessing, and possibly domain adaptation techniques if the datasets are drawn from different sky surveys.

6.2 Interpretation of Feature Importance

Across models, features such as temperature, absolute magnitude, and color indices (e.g., u-g, g-r) consistently emerged as the most predictive. This aligns well with astrophysical expectations, since these features directly correlate with a star's spectral class and luminosity. For instance, Red Dwarfs exhibit distinctively low temperatures and higher color indices, making them easier to separate from hotter, more massive stars.

This result reinforces the value of integrating domain knowledge into the feature selection process. It also confirms that observational photometric features, while indirect, carry strong astrophysical signals that can be effectively leveraged in classification tasks.

6.3 Classification Errors and Ambiguities

Despite the overall good performance, some misclassifications were observed. These often occurred between classes that are physically adjacent in the Hertzsprung-Russell diagram (e.g., late-type main sequence stars misclassified as Red Dwarfs). In some cases, these errors may be attributed to overlapping feature values or to the limitations of the feature set used—some essential physical properties (e.g., metallicity, proper motion) were not included due to data constraints.

Furthermore, the classification of White Dwarfs proved more challenging, likely due to their overlapping magnitudes with other classes and the fact that they occupy less distinct regions in feature space. These findings suggest that incorporating more discriminative features or using multi-modal data (e.g., combining spectra with photometry) could improve model robustness.

6.4 Methodological Choices and Limitations

The methodological choices made—particularly the focus on supervised classification using readily available photometric features—were influenced by data accessibility and interpretability. However, this also introduces limitations. For instance, the reliance on labeled data means that model performance is bounded by the quality and completeness of the training labels, which in astronomy can be uncertain or survey-dependent.

Additionally, while models like Random Forest offer good performance, they are not inherently probabilistic and can lack interpretability compared to Bayesian or physics-informed methods. Incorporating uncertainty estimation, probabilistic modeling, or hybrid approaches that embed astrophysical constraints could enhance scientific reliability.

6.5 Broader Implications and Future Work

This work demonstrates the potential of machine learning in astrophysical classification tasks. The results support the feasibility of automated stellar classification using photometric surveys—a valuable capability in the era of large-scale data from missions like Gaia and LSST.

Future work could explore semi-supervised learning, to leverage unlabeled data, and unsupervised clustering to identify new or rare stellar populations. Additionally, incorporating temporal information (e.g., variability) or higher-dimensional data (e.g., spectra) could further refine classification outcomes.

Ultimately, bridging the gap between astrophysical theory and machine learning practice is key to achieving models that are not only accurate but also physically meaningful.

7 Conclusion

In this thesis, I explored the application of machine learning techniques for the classification and analysis of stellar and extragalactic objects. The study used two distinct astronomical datasets, allowing me to evaluate the effectiveness of various machine learning models in identifying different celestial objects.

I first performed a comparative analysis of several classification models, including Random Forest, XGBoost, logistic regression, and a neural network (MLP). The results showed that XGBoost and Random Forest performed exceptionally well, achieving accuracy levels above 97%, making them well-suited for astrophysical classification tasks. The neural network model also demonstrated strong performance, showing potential for improvement with more complex architectures on larger datasets. Logistic regression, although simpler, provided a solid basis for comparison.

Through hyperparameter tuning, we optimized the performance of Random Forest and XGBoost, leading to marginal improvements in accuracy. Confusion matrices revealed that misclassifications occurred primarily between certain star types, suggesting that refinements in variable selection and data preprocessing could improve classification accuracy.

Beyond model comparison, this study also demonstrated the growing role of machine learning in modern astrophysics. With the increasing volume of astronomical data, particularly in the GAIA era, machine learning techniques provide effective tools for analyzing and classifying large datasets, uncovering patterns, and improving our understanding of stellar evolution.

Future work could extend this research by incorporating deep learning architectures, such as convolutional neural networks (CNNs) for spectral data or recurrent neural networks (RNNs) for analyzing variable star time series.

Finally, this study highlights the power and potential of machine learning in astrophysics research, paving the way for more automated and efficient methods for analyzing the ever-increasing volumes of astronomical data.

A Outliers deletion

```
def remove_outliers(df, column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    return df[(df[column] >= lower_bound) & (df[column] <= upper_bound)]

# Apply outlier removal on Luminosity (or any other feature if needed)
df_no_outliers = remove_outliers(df, "Luminosity")
```

B Log transformation of Luminosity

```
# Log transform Luminosity
df[’Luminosity_log’] = np.log1p(df[’Luminosity’])
df_no_outliers[’Luminosity_log’] = np.log1p(df_no_outliers[’Luminosity’])

# Select numerical features for scaling
features = ["Temperature", "Luminosity_log", "Radius", "Absolute_magnitude"]
scaler = RobustScaler()

df[features] = scaler.fit_transform(df[features])
df_no_outliers[features] = scaler.fit_transform(df_no_outliers[features])
```

C Random Forest code

```
# Define target variable & features
X = df[features]
y = df["Star_type"]

X_no_outliers = df_no_outliers[features]
y_no_outliers = df_no_outliers["Star_type"]

# Split datasets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
X_train_no, X_test_no, y_train_no, y_test_no = train_test_split(X_no_outliers, y_no_outliers,
    test_size=0.2, random_state=42)

# Train Random Forest Model
```

```
model = RandomForestClassifier(random_state=42)
model_no_outliers = RandomForestClassifier(random_state=42)

model.fit(X_train, y_train)
model_no_outliers.fit(X_train_no, y_train_no)

# Predictions
y_pred = model.predict(X_test)
y_pred_no_outliers = model_no_outliers.predict(X_test_no)

# Evaluate Performance
print("Model with Outliers:")
print(classification_report(y_test, y_pred))

print("\nModel without Outliers:")
print(classification_report(y_test_no, y_pred_no_outliers))
```

D Logistic Regression code

```

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score

 # Define target variable & features
 X = df[features]
 y = df["Star_type"]

 X_no_outliers = df_no_outliers[features]
 y_no_outliers = df_no_outliers["Star_type"]

 # Split datasets
 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)
 X_train_no, X_test_no, y_train_no, y_test_no = train_test_split(X_no_outliers, y_no_outliers, test_size=0.2, random_state=42, stratify=y_no_outliers)

 # Train Random Forest Model
 model_log = LogisticRegression(max_iter=1000, multi_class="multinomial", solver="lbfgs")
 model_no_outliers_log = LogisticRegression(max_iter=1000, multi_class="multinomial", solver="lbfgs")

 model_log.fit(X_train, y_train)
 model_no_outliers_log.fit(X_train_no, y_train_no)

 # Predictions
 y_pred = model_log.predict(X_test)
 y_pred_no_outliers = model_no_outliers_log.predict(X_test_no)


 # Evaluate Performance
 print("Model with Outliers:")
 print("Accuracy:", accuracy_score(y_test, y_pred))

 print(classification_report(y_test, y_pred))

 print("\nModel without Outliers:")
 print("Accuracy:", accuracy_score(y_test_no, y_pred_no_outliers))

 print(classification_report(y_test_no, y_pred_no_outliers))
```

E Binary classification code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

# Load dataset
df = pd.read_csv("6 class csv.csv")

# Display first few rows
display(df.head())

# Basic info and missing values
print(df.info())
print(df.isnull().sum())

# Encoding categorical variables
```

```
le_color = LabelEncoder()
df[’Star color’] = le_color.fit_transform(df[’Star color’])

le_spectral = LabelEncoder()
df[’Spectral Class’] = le_spectral.fit_transform(df[’Spectral Class’])

# Correlation matrix
plt.figure(figsize=(10,6))
sns.heatmap(df.corr(), annot=True, cmap=’coolwarm’)
plt.title(’Feature Correlation Matrix’)
plt.show()

# Binary classification: Selecting two classes
df_binary = df[df[’Star type’].isin([2, 3])]  # Example: White Dwarf vs. Main Sequence
X = df_binary.drop(columns=[’Star type’])
y = df_binary[’Star type’]

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardizing numerical features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Train classifier (Random Forest as example)
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

# Evaluation
print("Accuracy:", accuracy_score(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))

# Feature importance
importances = pd.Series(clf.feature_importances_, index=df_binary.drop(columns=[’Star type’]).
    columns)
importances.sort_values().plot(kind=’barh’, title=’Feature Importances’)
plt.show()
```

F MLP training logs

```
Epoch 1/20
2500/2500                                                              5s 2ms/step - accuracy:
    0.8969 - loss: 0.3089 - val_accuracy: 0.9610 - val_loss: 0.1275
Epoch 2/20
2500/2500                                                              4s 1ms/step - accuracy:
    0.9628 - loss: 0.1393 - val_accuracy: 0.9650 - val_loss: 0.1152
Epoch 3/20
2500/2500                                                              4s 1ms/step - accuracy:
    0.9641 - loss: 0.1180 - val_accuracy: 0.9629 - val_loss: 0.1242
Epoch 4/20
2500/2500                                                              4s 2ms/step - accuracy:
    0.9662 - loss: 0.1121 - val_accuracy: 0.9674 - val_loss: 0.1073
Epoch 5/20
2500/2500                                                              4s 2ms/step - accuracy:
    0.9666 - loss: 0.1054 - val_accuracy: 0.9690 - val_loss: 0.1028
Epoch 6/20
2500/2500                                                              4s 2ms/step - accuracy:
    0.9674 - loss: 0.1050 - val_accuracy: 0.9636 - val_loss: 0.1256
Epoch 7/20
2500/2500                                                              4s 2ms/step - accuracy:
    0.9700 - loss: 0.0999 - val_accuracy: 0.9673 - val_loss: 0.1085
Epoch 8/20
2500/2500                                                              4s 2ms/step - accuracy:
    0.9696 - loss: 0.0994 - val_accuracy: 0.9706 - val_loss: 0.1003
Epoch 9/20
2500/2500                                                              4s 2ms/step - accuracy:
    0.9699 - loss: 0.0964 - val_accuracy: 0.9718 - val_loss: 0.0976
Epoch 10/20
2500/2500                                                              4s 2ms/step - accuracy:
    0.9701 - loss: 0.0977 - val_accuracy: 0.9699 - val_loss: 0.1022
Epoch 11/20
2500/2500                                                              4s 2ms/step - accuracy:
    0.9704 - loss: 0.0961 - val_accuracy: 0.9714 - val_loss: 0.0981
Epoch 12/20
2500/2500                                                              4s 2ms/step - accuracy:
    0.9716 - loss: 0.0932 - val_accuracy: 0.9712 - val_loss: 0.0954
Epoch 13/20
2500/2500                                                              4s 2ms/step - accuracy:
    0.9719 - loss: 0.0919 - val_accuracy: 0.9720 - val_loss: 0.0941
Epoch 14/20
2500/2500                                                              4s 1ms/step - accuracy:
    0.9726 - loss: 0.0905 - val_accuracy: 0.9726 - val_loss: 0.0918
```

```
Epoch 15/20
2500/2500                                                    4s 1ms/step - accuracy:
    0.9718 - loss: 0.0932 - val_accuracy: 0.9735 - val_loss: 0.0917
Epoch 16/20
2500/2500                                                    4s 2ms/step - accuracy:
    0.9723 - loss: 0.0898 - val_accuracy: 0.9718 - val_loss: 0.0976
Epoch 17/20
2500/2500                                                    4s 2ms/step - accuracy:
    0.9726 - loss: 0.0890 - val_accuracy: 0.9728 - val_loss: 0.0960
Epoch 18/20
2500/2500                                                    4s 2ms/step - accuracy:
    0.9731 - loss: 0.0872 - val_accuracy: 0.9717 - val_loss: 0.0980
Epoch 19/20
2500/2500                                                    4s 2ms/step - accuracy:
    0.9736 - loss: 0.0887 - val_accuracy: 0.9740 - val_loss: 0.0905
Epoch 20/20
2500/2500                                                    4s 2ms/step - accuracy:
    0.9731 - loss: 0.0880 - val_accuracy: 0.9740 - val_loss: 0.0890

Final Accuracy : 0.9739500284194946
```

References

[1] A. Antoniadis-Karnavas, S. Sousa, E. Delgado-Mena, N. Santos, G. Teixeira, and V. Neves. Odusseas: a machine learning tool to derive effective temperature and metallicity for m dwarf stars. *Astronomy & Astrophysics*, 636:A9, 2020.

[2] D. Baron. Machine learning in astronomy: A practical overview. *Frontiers in Astronomy and Space Sciences*, 6:57, 2019.

[3] A. Behmard, E. A. Petigura, and A. W. Howard. Data-driven spectroscopy of cool stars at high spectral resolution. *The Astrophysical Journal*, 876(1):68, 2019.

[4] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[5] R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms. *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pages 161–168, 2006.

[6] G. M. De Silva, K. C. Freeman, J. Bland-Hawthorn, S. Martell, E. W. De Boer, M. Asplund, S. Keller, S. Sharma, D. B. Zucker, T. Zwitter, et al. The galah survey: scientific motivation. *Monthly Notices of the Royal Astronomical Society*, 449(3):2604–2617, 2015.

[7] P. Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 2012.

[8] S. Fabbro, K. Venn, T. O'Briain, S. Bialek, C. Kielty, F. Jahandar, and S. Monty. An application of deep learning in the analysis of stellar spectra. *Monthly Notices of the Royal Astronomical Society*, 475(3):2978–2993, 2018.

[9] M. I. Jordan and T. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

[10] H. Karttunen, P. Kröger, H. Oja, M. Poutanen, and K. J. Donner. *Stellar Spectra*, pages 227–239. Springer Berlin Heidelberg, Berlin, Heidelberg, 2017.

[11] S. B. Kotsiantis. Supervised machine learning: A review of classification techniques. *Informatica*, 31(3):249–268, 2007.

[12] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[13] G. Longo, E. Merényi, and P. Tiňo. Foreword to the focus issue on machine intelligence in astronomy and astrophysics. *Publications of the Astronomical Society of the Pacific*, 131(1004):1–6, 2019.

[14] R. Olney, M. Kounkel, C. Schillinger, M. T. Scoggins, Y. Yin, E. Howard, K. Covey, B. Hutchinson, and K. G. Stassun. Apogee net: Improving the derived spectral parameters for young stars through deep learning. *The Astronomical Journal*, 159(4):182, 2020.

[15] J.-V. Rodríguez, I. Rodríguez-Rodríguez, and W. L. Woo. On the application of machine learning in astronomy and astrophysics: A text-mining-based scientometric analysis. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 12(5):e1476, 2022.

[16] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[17] Y.-S. Ting, C. Conroy, and H.-W. Rix. Accelerated fitting of stellar spectra. *The Astrophysical Journal*, 826(1):83, 2016.

[18] Y.-S. Ting, C. Conroy, H.-W. Rix, and P. Cargile. The payne: Self-consistent ab initio fitting of stellar spectra. *The Astrophysical Journal*, 879(2):69, jul 2019.