Universidad Autónoma de Querétaro
división de investigación y posgrado

Faculty of Engineering
MCIA

Activity 1

Describing a DataSet

Student:
Juan Manuel
Aviña Muñoz

Professor:
Dr. Marco
Aceves Fernandez

August 25th, 2023

Contents

1 Introduction **2**

2 Theoretical Foundation **4**
2.1 General concepts 4

3 Methods and Materials **6**
3.1 DataSet 6
3.2 Resources 6
3.3 Libraries 6

4 Results **8**

5 Discussion **17**

6 Conclusions **19**

References **20**

Appendix **21**

1 Introduction

Data science is an interdisciplinary field that combines various techniques, algorithms, processes, and systems to extract knowledge and insights from structured and unstructured data. It involves using both programming skills and domain knowledge to uncover patterns, trends, and valuable information that can drive decision-making and solve complex problems.

Key components of data science include:

- **Data Collection and Preparation:** Data science starts with the acquisition of relevant data from various sources, such as databases, API's, web scraping, sensors, and more. The data often requires cleaning, preprocessing, and transformation to make it suitable for analysis.
- **Exploratory Data Analysis (EDA):** This phase involves understanding the data by visualizing and summarizing its key characteristics. EDA helps identify missing values, outliers, correlations, and potential patterns within the data.
- **Feature Engineering:** Involves selecting, transforming, and creating new features from the raw data to improve the performance of machine learning models. Effective feature engineering can significantly impact the quality of predictions and insights.
- **Machine Learning:** Is a core component of data science. It involves training models to learn patterns from data and make predictions or decisions. This includes various types of algorithms such as regression, classification, clustering, and more.
- **Model Evaluation and Selection:** Once machine learning models are trained, they need to be evaluated using appropriate metrics to measure their performance. The best-performing models are selected for deployment or further refinement.
- **Data Visualization:** Visualizations help communicate insights and findings effectively to both technical and non-technical audiences. Tools like matplotlib, Seaborn, and Plotly in Python, and libraries in R, are commonly used for this purpose.
- **Statistical Analysis:** Statistical techniques are used to understand the significance of findings, test hypotheses, and make inferences about the data. These techniques help validate assumptions and conclusions drawn from the data.
- **Big Data Technologies:** When dealing with massive datasets, data scientists may employ technologies such as distributed computing, cloud services, and big data frameworks (e.g., Hadoop, Spark) to handle and analyze data efficiently.
- **Domain Expertise:** Understanding the specific domain or industry you're working in is crucial. Domain knowledge helps in asking the right questions, identifying relevant data, and interpreting the results in a meaningful context.
- **Ethics and Privacy:** Data scientists must also consider ethical considerations and privacy concerns when handling sensitive or personal data. Ensuring data security and adhering to legal and ethical standards is essential.

Data science has applications in various fields, including business, healthcare, finance, marketing, social sciences, and more. It has led to advancements in predictive analytics, recommendation systems, fraud detection, image and speech recognition, natural language processing, and autonomous systems.

To work in data science, individuals often need a combination of skills in programming (Python, R), statistics, machine learning, domain expertise, and effective communication. Data scientists use tools like Jupyter notebooks, Python libraries (NumPy, pandas, scikit-learn), R, and data visualization tools (matplotlib, Seaborn, Tableau) to perform their tasks [1].

The importance of selecting an appropriate database for data science lies in it's potential to significantly impact the overall success and efficiency of your data analysis and machine learning endeavors. The performance of a chosen database directly influences data storage and retrieval speeds, which in turn affect the responsiveness of your data processing tasks. A well-designed database optimized for efficient read and write operations can lead to faster data access and analysis.

As the data volumes grow over time, a scalable database becomes crucial to ensure that data science projects can handle increasing data demands without sacrificing performance. Scalability ensures that the analytical capabilities remain effective even as data needs expand. Maintaining data integrity and consistency is vital for data-driven insights. A solid database enforces data accuracy, validity, and consistency, ensuring that the insights derived from data analysis are trustworthy and reliable.

Different databases offer distinct data models, such as relational, document-oriented, graph, and time series databases. Selecting the appropriate database model aligned with the data's structure and relationships simplifies data manipulation and querying. Powerful querying and analytical capabilities are essential for data science projects. Databases optimized for data analysis provide advanced querying options, indexing, and aggregation functions that expedite data retrieval for generating valuable insights.

In specific applications, real-time data processing becomes indispensable. Databases designed for real-time analytics and streaming data can support tasks like monitoring, anomaly detection, and immediate decision-making.

Security considerations are of utmost importance, especially with data privacy regulations such as GDPR. A capable database should offer features for access control, encryption, and audit trails to ensure data protection.

Cost considerations encompass licensing models and infrastructure requirements, directly impacting the total cost of ownership over time. Understanding the cost implications aids in making informed decisions. Aligning with evolving technologies and trends is crucial to avoid migration challenges down the road.

In essence, the choice of a database for data science is a strategic decision that influences data processing, analysis, and decision-making. Factors such as performance, scalability, data modeling, querying capabilities, integration, security, cost, and long-term compatibility must be weighed to ensure successful data science outcomes.

2 Theoretical Foundation

In the context of databases, the concepts of attributes, instances, and metadata play crucial roles in organizing, understanding, and effectively utilizing data. Let's delve into each of these concepts:

Attributes: are the individual pieces of information that describe specific aspects of the entities or objects stored in a database. In a relational database, attributes correspond to columns in a table. Each attribute represents a distinct characteristic or property associated with the objects being observed. For instance, in a database containing information about customers, attributes could include "Name," "Age," "Address," and "Email." Attributes define the properties or characteristics that allow us to differentiate and categorize the data.

Instances: refer to the individual occurrences or records in a database. They represent the specific entries or data points that collectively form the dataset. In a relational database, instances correspond to rows in a table. Each instance contains values for each attribute, providing a comprehensive snapshot of an entity or object. For example, in a customer database, each row could represent a single customer with their corresponding attributes filled in.

Metadata: encompasses the additional information that provides context and details about the data itself. It describes the structure, organization, and properties of the data stored in the database. Metadata might include information such as the data source, the format of the data, the meaning of each attribute, units of measurement, creation date, and any constraints on the data. Metadata enhances the understanding and usability of the data by offering insights into its origin, quality, and interpretation.

In a database, the interaction of these concepts is integral to managing and extracting value from data. Attributes define what type of information is stored, instances capture the individual data points, and metadata contextualizes and clarifies the data's significance. Together, they provide the foundation for effective data storage, retrieval, analysis, and decision-making within various domains, including business, research, healthcare, and more.

2.1 General concepts

While databases provide a structured and organized storage solution for data, it's through the lens of statistical methods and techniques that we can extract meaningful insights, transform raw data into actionable knowledge, and make informed decisions. Here's an expansion on your point:

A database, as a repository, holds a wealth of information gathered over time. This information represents not just isolated data points, but a narrative of activities, interactions, transactions, and behaviors. However, this raw data is akin to a puzzle with scattered pieces, waiting to be assembled into a coherent picture.

Mean

Also known as the average, is calculated by summing up all the values in the dataset and then dividing by the total number of values. It's the most commonly used measure of central tendency and is sensitive to extreme values (outliers). The mean is suitable for both

continuous and discrete data. To calculate the mean, we use Equation 1:

$$\mu = \frac{\sum_{i=1}^{n} x_i}{n} \tag{1}$$

Where x_i represents each individual value in the dataset and n is the total number of instances. The mean provides a measure of the central tendency or the "*average*" value of the dataset.

Mode

It is the value that appears most frequently in a dataset, the element that occurs with the highest frequency. In a dataset, there can be one mode (*unimodal*), more than one mode (*multimodal*), or no mode at all if all values have the same frequency. The mode is particularly useful for categorical or discrete data, but it can also be applied to continuous data.

Median

It is the middle value when a dataset is ordered from least to greatest. In other words, it's the value that divides the dataset into two equal halves. If there's an odd number of observations, the median is the middle value. If there's an even number of observations, the median is the average of the two middle values.

Standard deviation

The standard deviation is a statistical measure that quantifies the amount of dispersion or spread of a set of values around the mean (average). In other words, it indicates how much the individual values in a dataset deviate from the mean value. A larger standard deviation implies greater variability, while a smaller standard deviation indicates less variability. The formula for calculating the sample standard deviation (s) is as follows:

$$s = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}} \tag{2}$$

Where x_i represents each individual value in the dataset, $\bar{x}$ represents the sample mean and n is the total number of instances.

3 Methods and Materials

3.1 DataSet

We have a set of 240 Stars with 5 Types:

- Red Dwarf - 0
- Brown Dwarf - 1
- White Dwarf - 2
- Main Sequence - 3
- Super Giants - 4
- Hyper Giants - 5

Temperature: Temperature in Kelvin.
R: Radius of star relative to the sun.
L: Luminosity of the star relative to the sun.
Absolute Magnitude: Magnitude of the star relative to the sun.
Color: Color of the star.
Spectral Class: An asteroid spectral type is assigned to asteroids based on their emission spectrum, color, and sometimes albedo. These types are thought to correspond to an asteroid's surface composition.

Database obtained from the Kaggle platform [2] sourced from NASA observations.

3.2 Resources

Intel i7-9750H @ 2.60 GHz processor, 16 GB DDR4 @ 2666 MHz, 1 Tb HDD + 1 Tb SSD, GeForce GTX 1660 Ti 6 GB VRAM GDDR6.

3.3 Libraries

The following are the libraries used for this activity:

- **NumPy (Numerical Python)** is a fundamental package for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a variety of mathematical functions to operate on these arrays. NumPy is a core library in the data science ecosystem and is used for tasks like numerical computations, linear algebra operations, random number generation, and more.
- **Pandas** is a powerful library for data manipulation and analysis in Python. It provides data structures (primarily Series and DataFrame) to efficiently handle and manipulate structured data. Pandas enables tasks such as loading data from various file formats (CSV, Excel, SQL databases), cleaning and preprocessing data, filtering, merging, transforming, and summarizing data. It's widely used for data wrangling and exploration.

- **Seaborn** is a statistical data visualization library built on top of Matplotlib. It provides a high-level interface for creating attractive and informative statistical graphics. Seaborn simplifies the process of creating complex visualizations like scatter plots, bar plots, histograms, box plots, and more. It also supports color palettes, themes, and additional statistical plotting capabilities.

- **Matplotlib** is a versatile 2D plotting library in Python. It provides a wide range of functions to create static, interactive, and animated visualizations. While it can be used to create basic plots, it can also be customized to create complex visualizations. Matplotlib is the foundation for many other visualization libraries and tools.

- **Plotly Express** is a high-level library for creating interactive visualizations. It's built on top of Plotly, which is a more general-purpose interactive visualization library. Simplifies the creation of various visualizations, including scatter plots, line plots, bar plots, area plots, and more. These visualizations are interactive and can be embedded in web applications or shared online.

4 Results

Figure 1 shows the statistical data obtained from the pandas library.

	Temperature	L	R	A_M	Color	Spectral_Class	Type
0	3068	0.002400	0.1700	16.12	Red	M	0
1	3042	0.000500	0.1542	16.60	Red	M	0
2	2600	0.000300	0.1020	18.70	Red	M	0
3	2800	0.000200	0.1600	16.65	Red	M	0
4	1939	0.000138	0.1030	20.06	Red	M	0
..	...	...	...	...	...	...	...
235	38940	374830.000000	1356.0000	-9.93	Blue	O	5
236	30839	834042.000000	1194.0000	-10.63	Blue	O	5
237	8829	537493.000000	1423.0000	-10.73	White	A	5
238	9235	404940.000000	1112.0000	-11.23	White	A	5
239	37882	294903.000000	1783.0000	-7.80	Blue	O	5

[240 rows x 7 columns]

Figure 1: *Statistical Information about features*

Checking for missing values is a fundamental and critical step in data analysis and preprocessing, as shown in Figure 2.

Temperature	0
L	0
R	0
A_M	0
Color	0
Spectral_Class	0
Type	0
dtype: int64	

Figure 2: *Null data from dataset*

In order to better understand our data we need to use some of the Equations described in Theoretical Foundation, some of our libraries can help with the process. as shown in Figure 3

	Temperature	L	R	A_M	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

Figure 3: *Description from dataset*

Figure 4 shows the correlation between features.

	Temperature	L	R	A_M	Type
Temperature	1.000000	0.393404	0.064216	-0.420261	0.411129
L	0.393404	1.000000	0.526516	-0.692619	0.676845
R	0.064216	0.526516	1.000000	-0.608728	0.660975
A_M	-0.420261	-0.692619	-0.608728	1.000000	-0.955276
Type	0.411129	0.676845	0.660975	-0.955276	1.000000

Figure 4: *Correlation between attributes*

Figure 5 shows the correlation between columns in a heatmap.

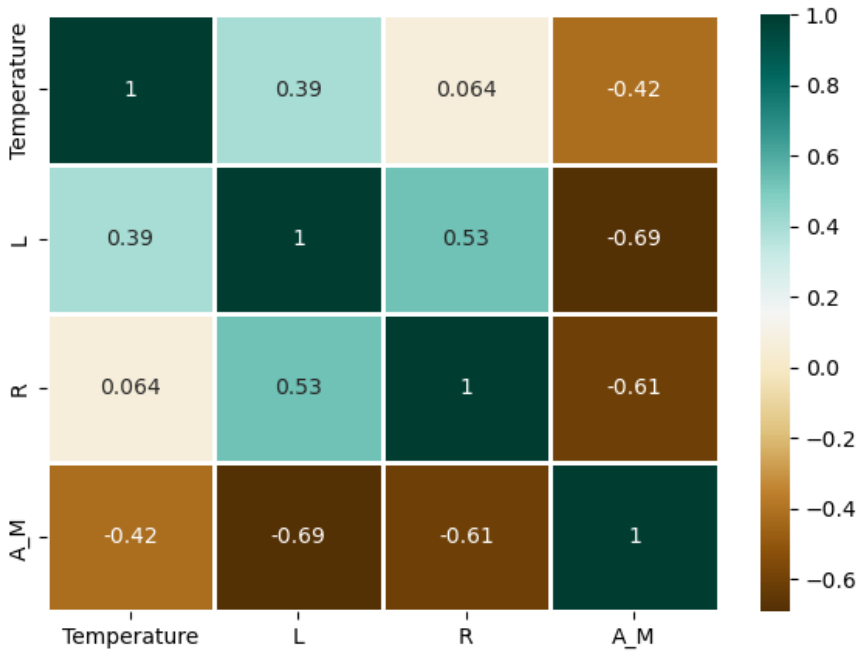

Figure 5: *Correlation Heatmap*

Now we need to know if there is any variation between our attributes in the color section.

```
array(['Red', 'Blue White', 'White', 'Yellowish White', 'Blue white',
       'Pale yellow orange', 'Blue', 'Blue-white', 'Whitish',
       'yellow-white', 'Orange', 'White-Yellow', 'white', 'yellowish',
       'Yellowish', 'Orange-Red', 'Blue-White'], dtype=object)
```

Figure 6: *Unique data from Color attribute*

```
Red                   112
Blue                   56
Blue-white             26
Blue White             10
yellow-white            8
White                   7
Blue white              4
white                   3
Yellowish White         3
yellowish               2
Whitish                 2
Orange                  2
White-Yellow            1
Pale yellow orange      1
Yellowish               1
Orange-Red              1
Blue-White              1
Name: Color, dtype: int64
```

Figure 7: *Unique data values from Color attribute*

Figures 6 and 7 show that there are not only five attributes in this section, so we need to unify this data as shown in Figure 8.

```
Red              112
Blue              56
Blue-white        41
yellow-white      15
white             12
Orange             4
Name: Color, dtype: int64
```

Figure 8: *Data from color attribute after unifying the color attributes*

Now we show a column chart for our unified color data, Figure 9.

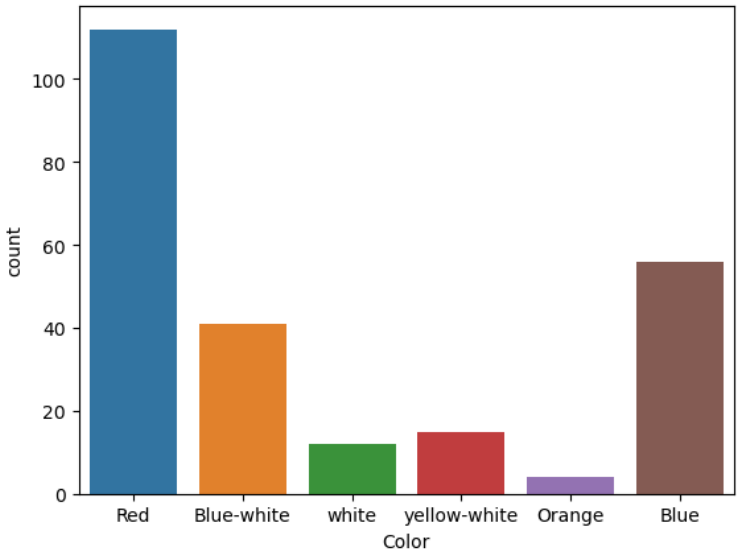

Figure 9: *Data from color attribute after unifying some attributes*

Now, for the distribution of our values in our dataset, we use several histograms, seen in Figure 10.

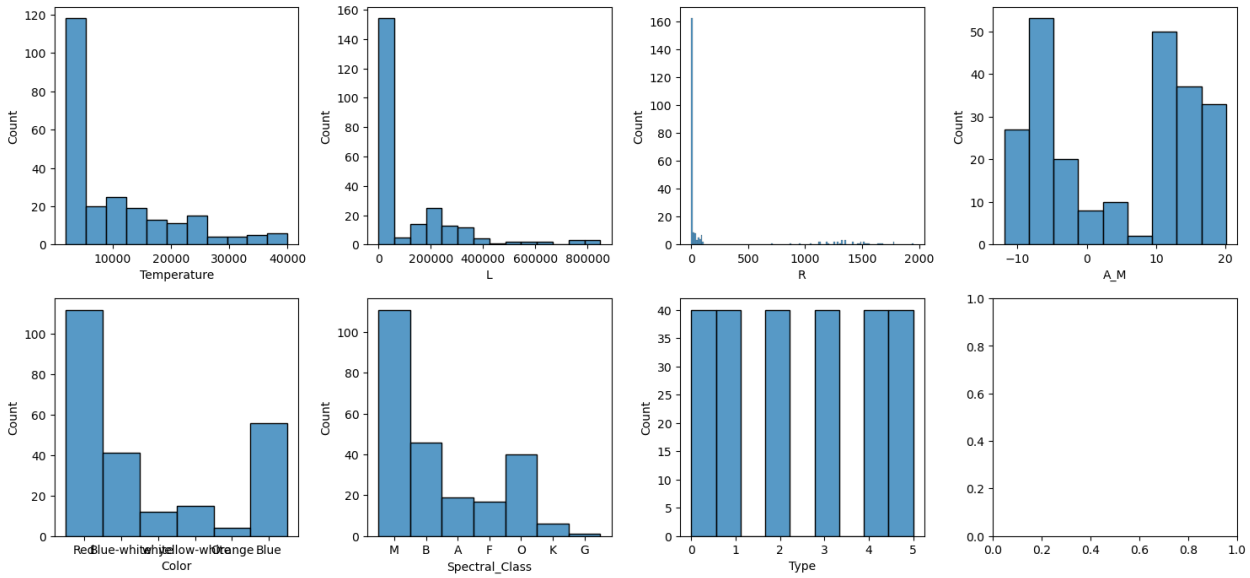

Figure 10: *Attributes Histogram*

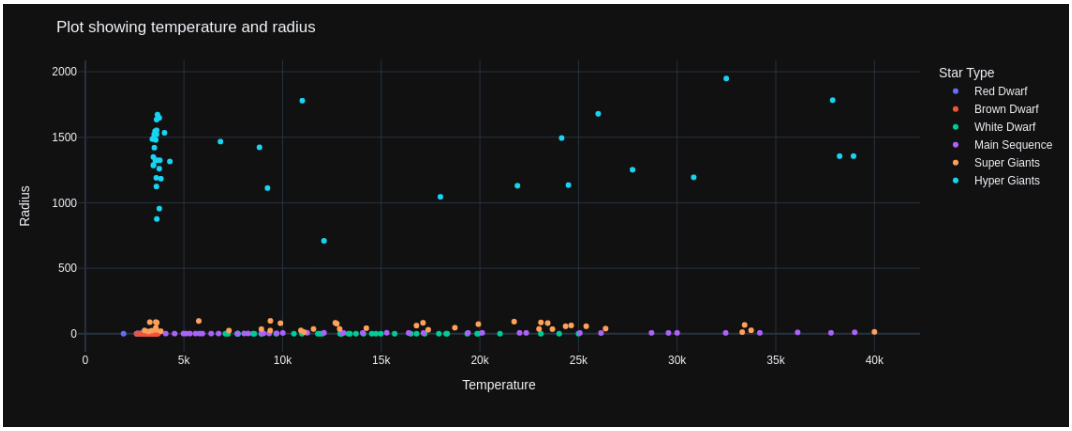

Figure 11: *Correlation between Temperature and Radius*

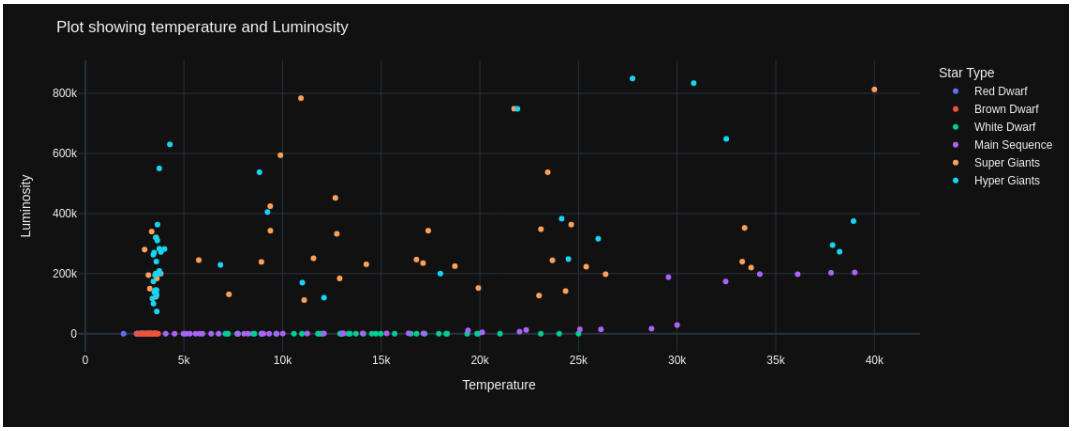

Figure 12: *Correlation between Temperature and Luminosity*

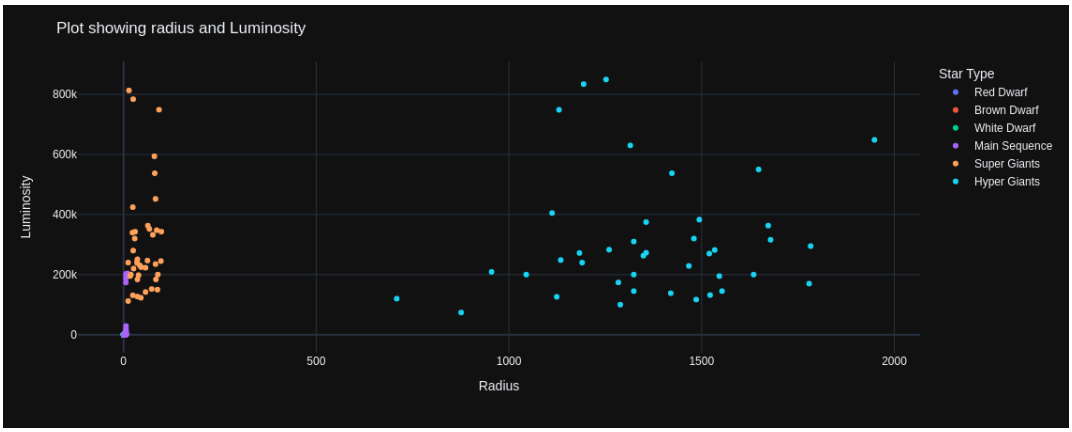

Figure 13: *Correlation between Radius and Luminosity*

To know the values distribution in our Temperature attribute, we use a boxplot and a distribution plot, Figures 14 and 15.

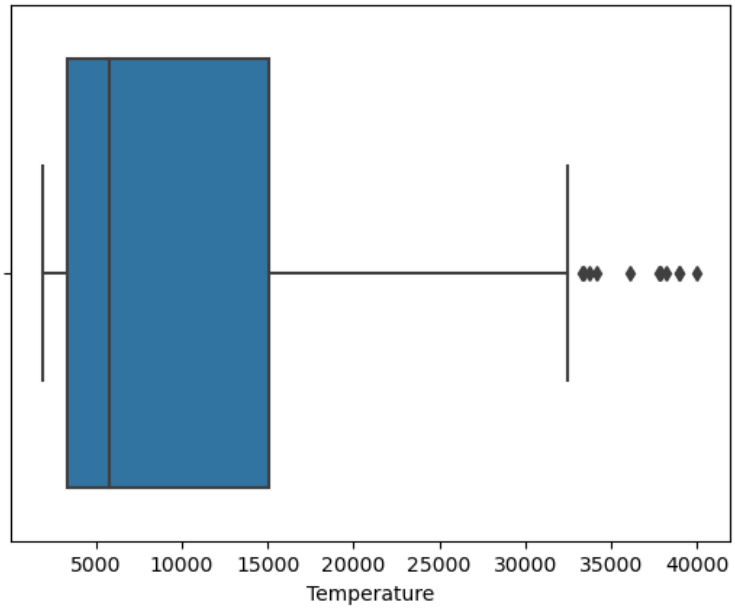

Figure 14: *Temperature Boxplot*

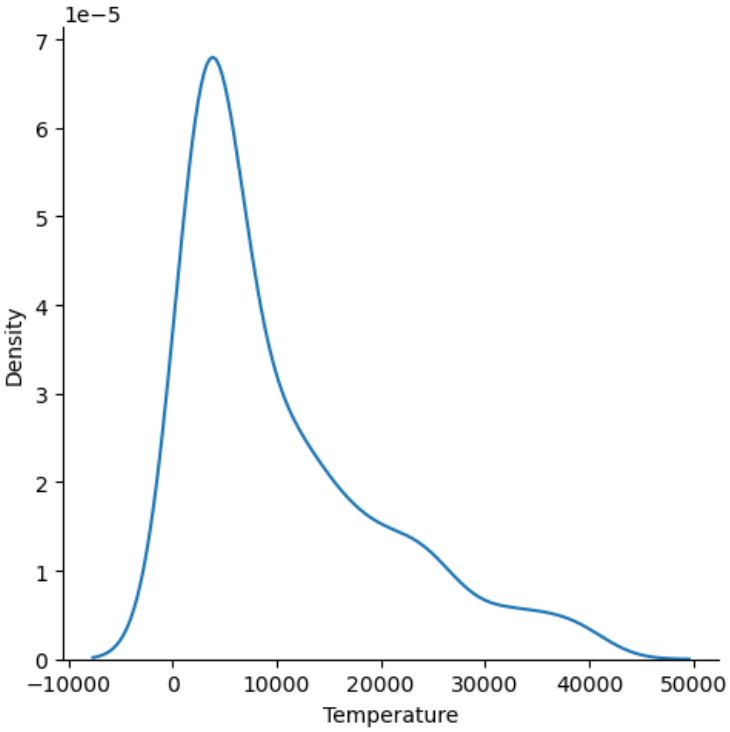

Figure 15: *Temperature Distplot*

To know the values distribution in our Absolute Magnitude attribute, we use a boxplot and a distribution plot, Figures 16 and 17.

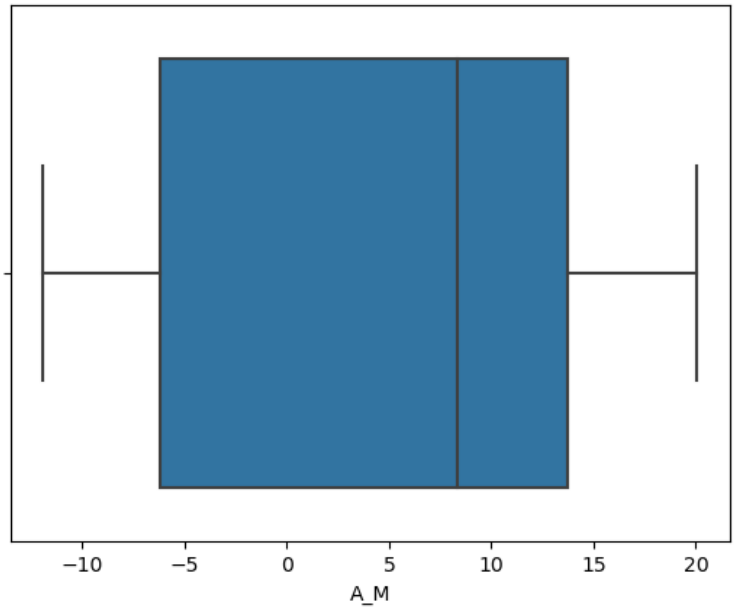

Figure 16: *Absolute Magnitude Boxplot*

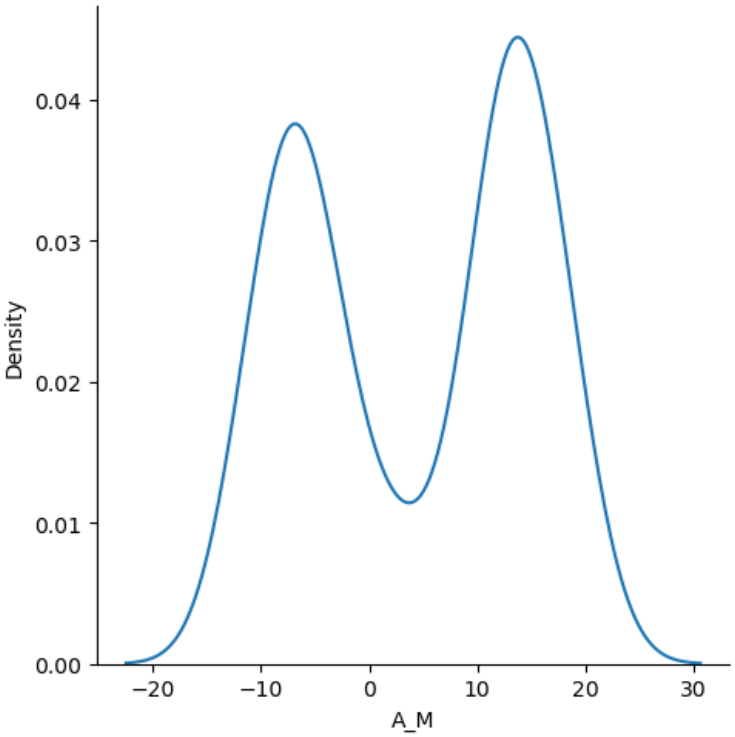

Figure 17: *Absolute Magnitude Distplot*

To know the values distribution in our Luminosity attribute, we use a boxplot and a distribution plot, Figures 18 and 19.

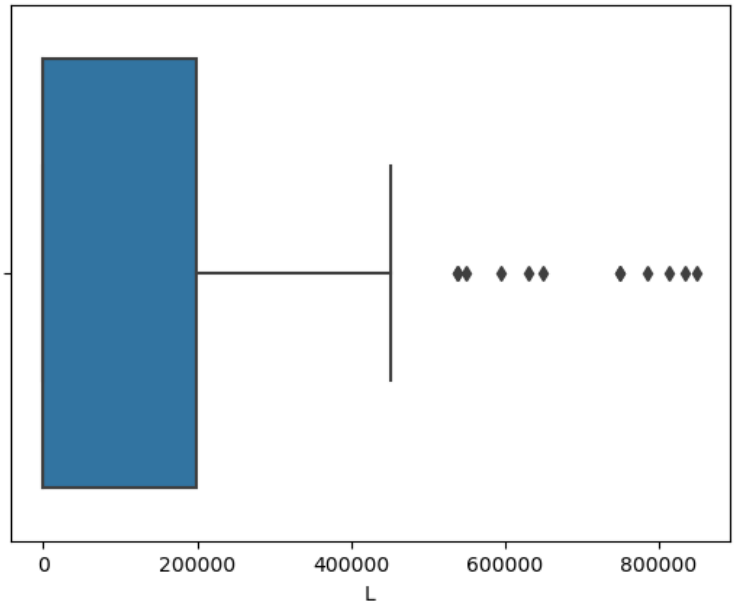

Figure 18: *Luminosity Boxplot*

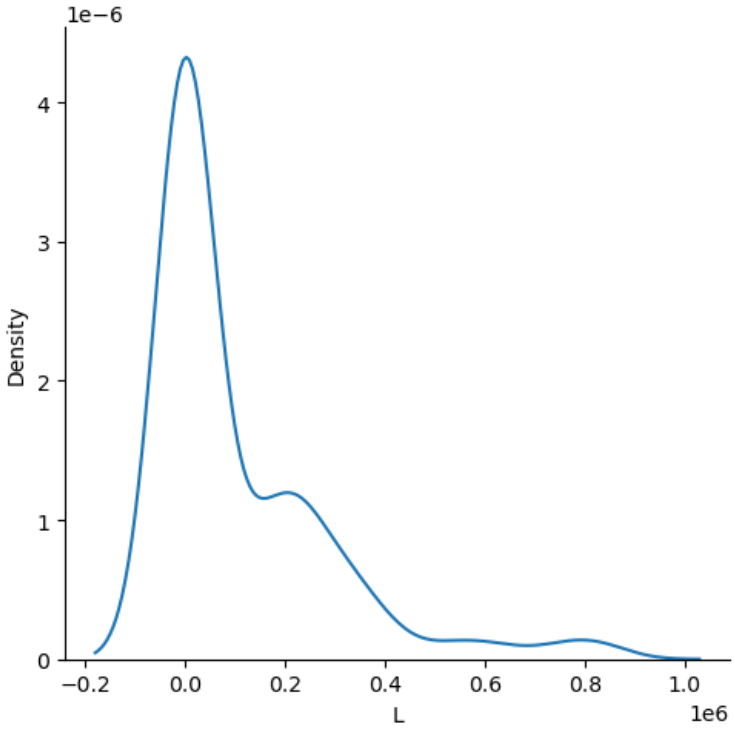

Figure 19: *Luminosity Distplot*

To know the values distribution in our Radius attribute, we use a boxplot and a distribution plot, Figures 20 and 21.

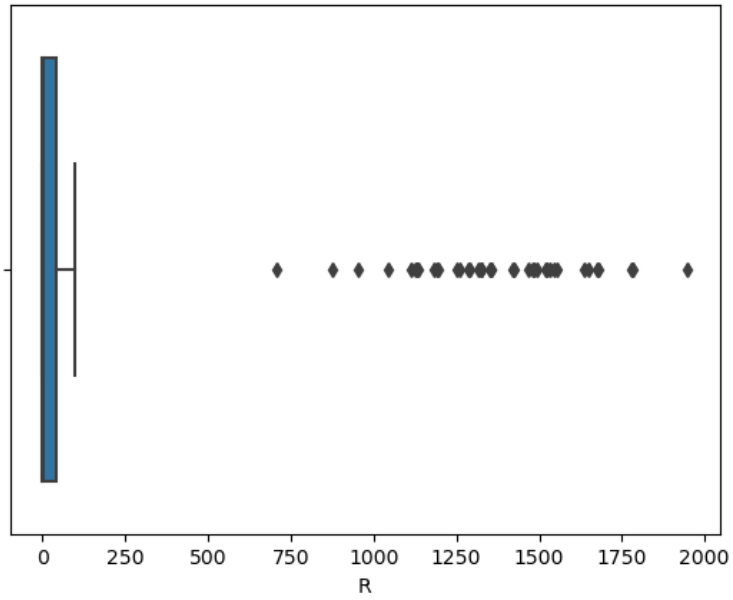

Figure 20: *Radius Boxplot*

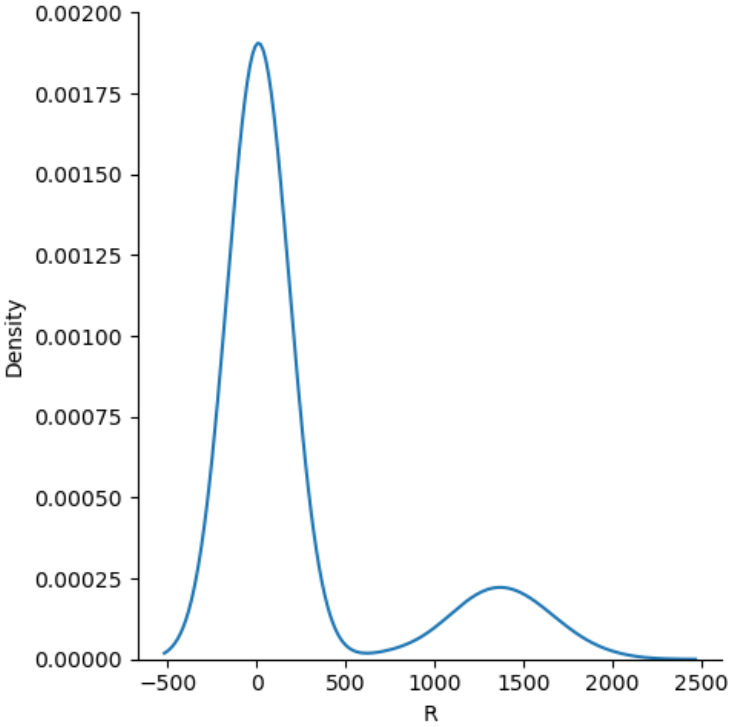

Figure 21: *Radius Distplot*

5 Discussion

Missing values

Null values, are data entries that lack information or have no assigned value. Dealing with null values is a critical aspect of data analysis and data preprocessing. Null values can arise due to various reasons, such as incomplete data collection, data entry errors, or the absence of certain attributes for specific instances.

Addressing this data cal help us make a more accurate analysis, ensure the models can be trained and tested effectively, reduce *bias* an *variance* in our data, improve our quality of our data, resulting in increased credibility and usability.

Unique data count

Knowing the unique data within a dataset refers to identifying and understanding the distinct values or elements present in that dataset. Uniqueness is crucial when working with data because it allows us to grasp the diversity, variety, and range of the information being analyzed.

Here's why understanding the count of unique data in an attribute is important:

- A high count of unique values in an attribute might suggest that the data is of high quality, indicating a rich variety of distinct information. On the other hand, a low count could hint at potential data quality issues, such as missing or incorrectly recorded values.

- The count of unique data helps understand the uniqueness or commonality of values within an attribute. This insight is particularly important when dealing with categorical or nominal attributes.

- In machine learning and statistical analysis, attributes with low unique value counts might not contribute significantly to modeling. Identifying such attributes aids in efficient feature selection and dimensionality reduction.

- Unique value counts can guide preprocessing steps, such as deciding whether to treat an attribute as categorical or continuous, or whether to apply certain transformation techniques.

- Unusually high or low unique value counts might indicate anomalies or errors in the data that warrant further investigation.

- Visual representations of unique data counts, such as bar plots or histograms, provide a clear overview of attribute distributions and help in conveying insights to non-technical stakeholders.

- In tasks like one-hot encoding or creating indicator variables, unique value counts help decide the appropriate approach for converting categorical attributes into numerical formats.

- Consistency in unique value counts ensures data integrity and helps identify potential data entry errors or duplication issues.

Boxplots

These tools provide a concise summary of the distribution of data. They display the median, quartiles, potential outliers, and overall spread of a dataset.

Distribution plots

It is a powerful tool that displays the distribution of a univariate dataset. It combines a histogram, which displays the frequency of data values within specified bins, with a kernel density estimation (KDE), which provides a smoothed estimate of the probability density function of the data.

Outliers

Boxplots and distribution plots are powerful visualization tools commonly used in data analysis to gain insights into the distribution, spread, and central tendencies of a dataset. Distplots help us understand the underlying distribution of your data and make informed decisions about analysis techniques or assumptions.

We notice there are a lot of outliers in our dataset, which can significantly influence analysis, decision-making, and model performance. Identifying and managing outliers are integral steps in maintaining data integrity and obtaining accurate insights from your data.

6 Conclusions

The process of selecting a suitable database cannot be overstated in today's data-driven landscape. The efficiency, performance, and overall success of any data-intensive project heavily hinge upon this crucial decision. Through the utilization of Python libraries, we are empowered to not only evaluate the attributes, instances, and metadata of a database but also to streamline the entire selection process.

Equally vital is the exploration of metadata, which provides crucial information about the database itself. This encompasses details such as schema, data types, relationships, and constraints. Armed with this knowledge, we can ensure that the chosen database aligns seamlessly with our application's requirements, reducing potential compatibility issues and enhancing overall system reliability.

In this context, Python libraries, with their rich ecosystem, provide a powerful toolkit for working with databases. Libraries such as SQLAlchemy, pandas, and Dask facilitate seamless interactions with databases, simplifying tasks like querying, aggregation, and transformation. Furthermore, by utilizing libraries for database exploration and visualization, such as Matplotlib and Seaborn, one can gain valuable insights into the dataset's structure and characteristics, aiding in the development of meaningful data-driven narratives.

In a world where data plays a pivotal role in shaping business strategies, research endeavors, and technological innovations, the significance of selecting the right database and comprehending its attributes, instances, and metadata cannot be overstated. Armed with the knowledge and tools that Python libraries provide, professionals are better equipped to navigate the complexities of databases, unlocking the true potential of the information they contain. As the data landscape continues to evolve, the synergy between robust database selection, meticulous understanding, and proficient Python library usage will undoubtedly remain a linchpin for success in countless applications across diverse domains.

References

[1] L. Pierson, *Data Science for Dummies.* Wiley, 2015. [Online]. Available: https://books.google.com.mx/books?id=Jx$_J BgAAQBAJ$

[2] B. Dincer. (2021) Star Type Classification/NASA. [Online]. Available: https://www.kaggle.com/datasets/brsdincer/star-type-classification

[3] M. Aceves, *Inteligencia Artificial Para Programadores Con Prisa.* Universo de Letras, 2021.

Appendix

Introduction

In astronomy, stellar classification is the classification of stars based on their spectral characteristics. The classification scheme of galaxies, quasars, and stars is one of the most fundamental in astronomy. The early cataloguing of stars and their distribution in the sky has led to the understanding that they make up our own galaxy and, following the distinction that Andromeda was a separate galaxy to our own, numerous galaxies began to be surveyed as more powerful telescopes were built. This dataset aims to classify stars, galaxies, and quasars based on their spectral characteristics.

The database helps astronomers classify stars based on their properties, aiding in understanding star formation and evolution.

By tracking different star types across their life cycles, researchers gain insights into how stars change over time.

The database aids in comprehending star distributions in galaxies and star clusters, shedding light on their formation and history.

Stars provide information about galaxy structure and evolution, contributing to broader cosmological investigations.

The properties of host stars influence exoplanet habitability assessments, and the database assists in identifying suitable candidates.

Large-scale surveys generate vast data, and the database helps efficiently manage and analyze this information.

The database serves as an educational tool, teaching about star diversity and processes to students and the public.

Astrophysicists use the data to create models predicting star behavior under different conditions.

The database aids in identifying variable stars and transient events, essential for understanding stellar behaviors.

Justification

In essence, a database that classifies stars, galaxies, and quasars based on their spectral characteristics plays a crucial role in advancing our understanding of the universe, aiding various research areas, supporting education, and fostering collaboration within the astronomical community.

Objectives

- Secure robustness and scalability.
- Secure repeatability.
- Settle foundations for data quality.

Database Management

Obtaining information for stars classification involves a combination of data collection methods, utilizing existing resources, and potentially collaborating with experts in the field.

- Collaborate with astronomers, researchers, and institutions in the field. They might have access to specialized data or insights that can enhance the quality of your database.
- Access established astronomical databases such as SIMBAD, VizieR, NASA's Astrophysics Data System (ADS), and the Exoplanet Archive, as well as data releases from space agencies, observatories and research institutions.. These platforms provide a wealth of well-curated data that to extract and integrate into the database.
- Choose a suitable relational database management system (MySQL) to store and manage the information.

Data audits will be conducted regularly to maintain the integrity of the database.

Structure

Designing a database structure for stars classification involves planning how to organize and store the data efficiently. Here's a more detailed breakdown of the database structure:

- Set a unique identifier for each star.
- Common or scientific name of the star.
- Information about spectral type.
- Information about brightness of the star.
- Surface temperature of the star.
- Distance from the Sun.
- Information about it's current evolution state (color).
- Mass of the star.
- Radius of the star.
- If the star i part of a binary or multiple star system, add information.

Documentation and Maintenance

Scheduled maintenance and regular backups are essential practices to ensure the integrity, reliability, and security of the database.

Scheduled monthly maintenance

- Documenting the database structure involves creating a detailed reference that outlines the organization of tables, attributes, relationships, and constraints. This documentation helps developers, administrators, and future users understand the database's design.
- Regular data updates involve reviewing existing records and information in the database. Update any outdated or incorrect data with accurate and verified information. This ensures that the data remains current and relevant.
- As new observations, surveys, or research results become available, incorporate the relevant data into the database. This expansion ensures that the database continues to reflect the latest knowledge in the field of astronomy.
- During maintenance, perform validation checks on incoming data. Ensure that new data adheres to the established data structure and integrity constraints. Clean the data by resolving any inconsistencies or errors.
- After updating the data, conduct testing to ensure that the database remains functional and accurate. Test various queries, data retrieval methods, and user interactions to confirm that everything is working as expected.

Scheduled monthly maintenance

- Create a full backup of the database at regular intervals. A backup is a snapshot of the entire database's current state, including all data, schema, and configurations.
- Backups serve as a safety net in case of data corruption, accidental deletion, hardware failure, or other unforeseen issues. If any of these incidents occur, the backup can be used to restore the database to a previous state.
- Periodically test the backup restoration process to ensure that backups are functional and can be successfully restored if needed. Regular testing validates the backup's reliability.
- Backups will be stored on a separate Hard Disc Drive in order to safeguard the backups from the same risks that could affect our primary database.

Meta data and Citations

It's necessary to include metadata indicating the source of the data and any associated references or citations in order to add transparency and credibility to database.