University of Sussex

Spectral Classification Using Unsupervised Machine Learning

Candidate Number: 277921

Submitted for the degree of Master of Science

University of Sussex

15th August 2024

Abstract

In this thesis, we explore the application of unsupervised machine learning techniques for the spectral classification of stars, galaxies, and QSOs. The primary objectives are to implement and evaluate clustering algorithms for classifying the spectra of these celestial objects and to investigate dimensionality reduction techniques to enhance clustering performance. The methods employed include Principal Component Analysis (PCA) for dimensionality reduction and K-Means and Agglomerative Clustering for classification. The results demonstrate how the combination of PCA and clustering algorithms, with minimal preprocessing, effectively groups similar objects together, highlighting the algorithm's ability to cluster spectral data. This approach provides valuable insights into the underlying properties of these astronomical objects. The conclusions underscore the potential of unsupervised learning in astronomical data analysis and suggest directions for future research aimed at improving classification accuracy and computational efficiency.

Preface

This thesis explores the application of unsupervised machine learning techniques for the spectral classification of celestial objects, specifically stars and galaxies, using clustering algorithms such as K-Means and Agglomerative Clustering, as well as dimensionality reduction techniques like Principal Component Analysis (PCA).

The dataset utilized in this study, comprising 15,000 spectra, was obtained from the Sloan Digital Sky Survey (SDSS) Data Release 18 (DR18). The collection and preprocessing of this data, including the correction for instrumental effects and removal of cosmic rays, were conducted by the SDSS research group. The data normalization and preparation for clustering were performed by myself.

The implementation and evaluation of the PCA, K-Means, and Agglomerative Clustering algorithms were conducted independently by me. The results, including the clustering performance metrics and the visualizations such as the PCA plots and dendrogram plot, represent my original analysis and interpretation.

The literature review and contextual background, as well as the discussion on the significance of the SDSS data and its impact on astronomical research, were synthesized from existing research papers and articles. Proper citations have been provided throughout to acknowledge these sources.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Our comprehension of the universe is largely dependent on the study of astronomical spectra. Based on how different astronomical objects, such as stars, galaxies, and quasars, emit light at different wavelengths, astronomers can identify and describe them using spectral classification. Traditionally, spectral classification has relied on manual techniques and supervised machine learning methods, which require extensive labeled datasets and human expertise. However, the rapid increase in the volume of astronomical data requires the development of automated and scalable methods for spectral analysis.

Unsupervised machine learning offers a promising alternative to traditional approaches by enabling the automatic discovery of patterns and structures in data without the need for labeled examples. Among these techniques, clustering algorithms such as K-Means and Agglomerative Clustering can group similar spectra together, potentially revealing new insights and simplifying the classification process. Coupled with dimensionality reduction methods like Principal Component Analysis (PCA), these algorithms can handle high-dimensional spectral data more efficiently.

The purpose of this thesis is to explore how unsupervised machine learning methods can be used to tackle the spectral classification problem. Our goal is to create an automated technique that can efficiently and accurately categorise astronomical spectra by utilising PCA and clustering methods. This methodology not only overcomes the drawbacks of conventional techniques but also opens the door for the study of large and intricate datasets anticipated from next astronomical surveys.

The motivation for this research is threefold:

1. **Scalability**: The upcoming generation of astronomical surveys, such as the Large Synoptic Survey Telescope (LSST) and the James Webb Space Telescope (JWST), will produce unprecedented amounts of spectral data. Automated methods are essential for processing and analyzing this data at scale.

2. **Novel Discoveries**: Unsupervised learning techniques have the potential to uncover previously unknown patterns and structures in spectral data, leading to new astronomical discoveries and a deeper understanding of the universe.

3. **Efficiency and Automation**: By reducing the reliance on labeled data and manual intervention, unsupervised machine learning can streamline the spectral classification process, making it more efficient and less prone to human error.

This thesis will demonstrate the effectiveness of K-Means and Agglomerative Clustering in classifying astronomical spectra, supported by dimensionality reduction through PCA. We will evaluate the performance of these methods using various metrics. Through this work, we aim to contribute to the advancement of automated spectral classification in astronomy, ultimately aiding in the exploration and understanding of our universe.

1.2 Objectives

The primary aim of this thesis is to develop and assess unsupervised machine learning methods for classifying astronomical spectra. By utilizing clustering algorithms and dimensionality reduction techniques, the goal is to create a scalable approach that efficiently handles large, complex datasets from modern astronomical surveys.

The research objectives include:

1. **Automated Pipeline Development:** Create an automated pipeline for preprocessing, dimensionality reduction, and clustering of astronomical spectra, capable of processing raw spectral data without labeled training data.

2. **Clustering Algorithm Implementation:** Apply and evaluate K-Means and Agglomerative Clustering algorithms to classify spectral data, assessing their effectiveness in identifying distinct celestial object classes.

3. **Dimensionality Reduction Integration:** Utilize Principal Component Analysis (PCA) to reduce spectral data dimensionality, examining how varying principal components impact clustering outcomes.

4. **Clustering Performance Evaluation:** Measure clustering performance using metrics like Silhouette Score, Davies-Bouldin Index, and Adjusted Rand Index (ARI) to determine the most effective method.

5. **Cluster Characteristic Analysis:** Analyze cluster characteristics to identify features that distinguish spectral classes and relate them to astrophysical phenomena.

6. **Visualization and Interpretation:** Produce visualizations, such as 3D PCA plots, to facilitate the interpretation and communication of clustering results.

This thesis aims to advance astronomical data analysis by providing reliable, scalable techniques for the automated classification of spectral data, contributing to the broader effort to explore and understand the universe.

1.3 Spectral Classification

Spectral classification is a cornerstone of modern astronomy, playing a vital role in understanding the universe. By analyzing the light emitted or absorbed by celestial objects, astronomers can infer a wealth of information about their physical and chemical properties. This section discusses the significance of spectral classification in astronomy.

Historical Background

The origins of spectral classification date back to the 19th century, with Angelo Secchi's pioneering work in the 1860s. Secchi classified stars into four categories based on visual inspection of their spectra (Secchi, 1866). His work laid the groundwork for the Harvard Classification Scheme, developed by Edward C. Pickering and his team at the Harvard College Observatory. Annie Jump Cannon, a member of this team, refined the classification system, categorizing stars into spectral types O, B, A, F, G, K, and M, based on their temperatures and spectral lines (Cannon, 1912).

Stellar Spectral Types and Their Significance

Each spectral type is associated with specific physical characteristics, predominantly temperature and the presence of particular absorption lines:

- **O-type**: These are the hottest stars, with surface temperatures exceeding 30,000 K. They show strong ionized helium lines and weak hydrogen lines, indicative of their high energy environments (Walborn, 1971).
- **B-type**: B-type stars have temperatures between 10,000 and 30,000 K. They exhibit strong hydrogen lines and neutral helium lines, reflecting their slightly cooler, but still very hot, nature (Gray and Corbally, 2005).
- **A-type**: With temperatures ranging from 7,500 to 10,000 K, A-type stars are characterized by the strongest hydrogen lines, making them easily identifiable (Cannon, 1912).
- **F-type**: These stars have temperatures between 6,000 and 7,500 K. They show hydrogen lines and ionized metals, such as calcium (Gray and Corbally, 2005).
- **G-type**: G-type stars, including our Sun, have temperatures from 5,200 to 6,000 K. They display strong ionized calcium lines and various metal lines (Gray and Corbally, 2005).
- **K-type**: Cooler than G-type, K-type stars have temperatures from 3,700 to 5,200 K. They exhibit strong molecular bands, particularly from molecules like titanium oxide (Morgan et al., 1978).
- **M-type**: The coolest stars, with temperatures below 3,700 K, are characterized by strong molecular bands and neutral metal lines (Morgan et al., 1978).

Luminosity Classes

In addition to spectral types, stars are also classified by their luminosity, which is indicated by Roman numerals I to V. This classification system, developed by William Wilson Morgan and Philip C. Keenan (Morgan et al., 1943), identifies the star's size and brightness. The luminosity classes are:

- **Ia and Ib**: Supergiants, the largest and most luminous stars.

- **II**: Bright giants.
- **III**: Giants.
- **IV**: Subgiants.
- **V**: Main-sequence (dwarf) stars.

Techniques for Spectral Classification

The methods for spectral classification have evolved significantly since the early days of visual inspection. Modern techniques include:

- **Manual Classification**: Initially, spectral classification involved astronomers manually comparing stellar spectra against standard templates. While effective, this method is labor-intensive and subjective (Cannon, 1912).
- **Automated Classification**: Advances in technology and computational methods have led to the development of automated classification systems. These systems use algorithms to analyze spectral data, improving both accuracy and efficiency (Gray and Corbally, 2005).
- **Photometric Classification**: This method involves using photometric data to infer spectral types. While less precise than spectroscopic methods, it is useful for large-scale surveys where obtaining spectra for every star is impractical (Covey et al., 2007).

Importance of Stellar Spectral Classification

Spectral classification provides critical insights into the nature of stars and their evolutionary paths. Astronomers can determine a star's age, chemical composition, and distance by classifying stars into spectral types and luminosity classes. This information is vital for constructing models of stellar evolution and understanding the life cycles of stars.

For instance, the classification of stars into different spectral types allows astronomers to determine their positions on the Hertzsprung-Russell diagram (DeVorkin, 1984), a key tool for studying stellar evolution. Stars on this diagram follow well-defined evolutionary

paths, from the main sequence through various stages of expansion and contraction, ultimately ending as white dwarfs, neutron stars, or black holes.

Furthermore, peculiar stars that do not neatly fit into standard classifications must be identified using spectral categorisation. Certain stars offer possibilities to study uncommon physical processes and stellar settings, such as chemically odd stars or stars with anomalous spectral properties (Preston, 1974).

Applications in Galactic and Extragalactic Astronomy

Spectral classification is not limited to individual stars but extends to the study of galaxies and other celestial objects. Astronomers can categorise galaxies and determine their stellar populations, star formation rates, and chemical compositions by examining their integrated spectra. This information is crucial for understanding the formation and evolution of galaxies across cosmic time (Kennicutt Jr, 1998).

Spectral classification is useful in extragalactic astronomy for identifying and studying distant phenomena such as active galactic nuclei (AGNs) and quasars. The spectra of these objects often show broad emission lines and other features that provide insights into the extreme physical conditions and energetic processes occurring in their environments (Berk et al., 2001).

Spectral classification remains a fundamental tool in astronomy, enabling the detailed study of stars, galaxies, and other celestial objects. Through ongoing advancements in observational techniques and data analysis, spectral classification continues to play a vital role in expanding our understanding of the universe.

1.4 Machine Learning in Astronomy

The scientific domain of astronomy has seen a significant transformation with the introduction of machine learning (ML). The volume and complexity of astronomical data are growing exponentially, making traditional data analysis techniques insufficient. Machine learning provides effective methods for tackling these problems because of its capacity to recognise patterns and forecast outcomes from massive datasets. This section examines machine learning's importance, uses, and effects in astronomy with the help of current developments and studies.

1.4.1 Significance of Machine Learning in Astronomy

Modern astronomical research relies heavily on machine learning techniques because of the massive volumes of data produced by satellites, telescopes, and other observational tools. Petabytes of data are produced by surveys such as the Large Synoptic Survey Telescope (LSST) and the Sloan Digital Sky Survey (SDSS), which are significantly far beyond the capacity of manual analysis. ML algorithms can efficiently process, analyze, and interpret this data, leading to new discoveries and insights.

Machine learning provides several key advantages:

- **Automation**: Astronomers can concentrate on more complex analysis and interpretation by using machine learning (ML) algorithms to automate repetitive processes like anomaly identification and classification.
- **Scalability**: ML methods can handle large datasets and scale with the increasing volume of data, ensuring that new data can be processed without significant delays (Baron and Poznanski, 2019).
- **Precision and Accuracy**: By learning from large datasets, ML models can achieve high levels of precision and accuracy, often surpassing traditional methods in performance (Li et al., 2020).
- **Discovering Hidden Patterns**: ML can uncover complex, non-linear relationships in data that might be missed by conventional statistical methods, leading to novel discoveries (Ball and Brunner, 2010).

1.4.2 Applications of Machine Learning in Astronomy

Machine learning techniques are applied across various domains in astronomy, including but not limited to:

Star and Galaxy Classification

ML algorithms, particularly supervised learning models, have been extensively used for classifying stars, galaxies, and other celestial objects. For instance, the application of convolutional neural networks (CNNs) has significantly improved the accuracy of galaxy

morphology classification (Lanusse et al., 2023). Similar to this, random forests and support vector machines (SVMs) have been used to categorise stellar spectra and distinguish between various star types (Fiorentin et al., 2007).

Exoplanet Detection

Detecting exoplanets involves identifying the tiny dimming of a star's light as a planet transits in front of it. These faint signals in light curves gathered by missions like Kepler and TESS have been trained to be recognised by machine learning models, including deep learning networks (Shallue and Vanderburg, 2018). By using these models, it has proven effective to find new exoplanets that conventional methods could have missed.

Supernova Classification and Detection

Understanding supernovae is essential to comprehending the universe's expansion. Recurrent neural networks (RNNs), among other machine learning methods, have been used to evaluate time-series data and categorise supernovae according to their light curves (Charnock and Moss, 2017). These techniques increase detection rates and improve the capacity to discriminate between various kinds of supernovae.

Gravitational Wave Detection

Through the discovery of gravitational waves, observatories such as LIGO have provided a fresh perspective on the cosmos. The enormous volumes of data produced are analysed using machine learning models, which separate possible gravitational wave signals from background noise (George and Huerta, 2018). The sensitivity and accuracy of these detection's have increased significantly due to methods like deep learning.

Anomaly Detection

Astronomical data often contains rare and unusual objects that can lead to new scientific discoveries. Large dataset anomalies are found using machine learning models, especially those that use unsupervised learning techniques. To find outliers in spectral data and stellar light curves, for instance, isolation forests and autoencoders have been used (Baron and Poznanski, 2019).

Cosmological Simulations

Machine learning is also applied in cosmology to simulate the formation and evolution of the universe. Realistic simulations of galaxy formation and large-scale structure are produced using generative models, such as generative adversarial networks (GANs) (Mustafa et al., 2019). These models help researchers study the underlying physics and compare theoretical predictions with observational data.

1.4.3 Impact of Machine Learning on Astronomical Research

The integration of machine learning into astronomical research has had a profound impact, enabling more efficient data processing and leading to numerous discoveries. Some notable impacts include:

Accelerated Discoveries

ML algorithms have accelerated the pace of discoveries in astronomy. For instance, the application of deep learning to Kepler data led to the identification of several new exoplanets in a fraction of the time it would have taken using traditional methods (Shallue and Vanderburg, 2018).

Enhanced Precision and Sensitivity

Machine learning has enhanced the precision and sensitivity of instruments and data analysis methods. In gravitational wave astronomy, ML models have improved the ability to detect and localize wave sources, contributing to a better understanding of these phenomena (George and Huerta, 2018).

Resource Optimization

Machine learning enables astronomers to maximise the utilisation of resources, like as computational power and telescope time, by automating data processing and analysis. This efficiency is crucial for managing the massive datasets generated by modern astronomical surveys (Ivezić et al., 2019).

1.4.4 Challenges and Future Directions

Despite its successes, the application of machine learning in astronomy faces several challenges:

- **Data Quality and Bias**: The performance of ML models is highly dependent on the quality of the training data. Biases in the data can lead to inaccurate models and incorrect conclusions (Ball and Brunner, 2010).
- **Interpretability**: Many ML models, particularly deep learning networks, are often seen as "black boxes." Improving the interpretability of these models is essential for gaining scientific insights from their predictions (Rudin, 2019).
- **Integration with Traditional Methods**: It takes careful consideration to combine machine learning with conventional astronomy methods so that the advantages of each are properly utilised(Baron and Poznanski, 2019).

Prospective investigations will probably concentrate on resolving these issues and deepening the integration of machine learning with astronomical research. The use of machine learning (ML) in astronomy will only increase in importance as data volumes grow, leading to new discoveries and expanding our knowledge of the cosmos. The way data is handled, analysed, and interpreted in modern astronomy has completely changed as a result of machine learning.The continued development and application of ML techniques promise to unlock new frontiers in our exploration of the cosmos.

1.5 Structure of the Thesis

This thesis is organized to provide a comprehensive understanding of applying machine learning techniques to classify astronomical spectra. It is divided into several chapters, each addressing a specific aspect of the study.

Chapter 1: Introduction

This chapter introduces the thesis, outlining the motivation, objectives, and significance of machine learning in astronomy. It sets the stage for the study by identifying the main problems and research questions.

Chapter 2: Literature Review

Chapter 2 reviews existing literature on spectral classification and machine learning applications in astronomy. It identifies key advancements, challenges, and gaps that this thesis aims to address, forming the theoretical framework for the research.

Chapter 3: Data Acquisition and Preprocessing

This chapter describes the data collection process and preprocessing techniques such as normalization, smoothing and resampling, providing a foundation for the subsequent analysis.

Chapter 4: Methodology

Chapter 4 outlines the methodological approach, detailing the machine learning algorithms used, including PCA, K-Means, and Agglomerative clustering. It also discusses evaluation metrics and validation methods.

Chapter 5: Results

This chapter presents the experimental results, analyzing the clustering outcomes and model effectiveness.

Chapter 6: Conclusion and Future Work

Chapter 6 summarizes the thesis's findings, assesses the research questions, discusses limitations, and suggests future research directions.

References and Code

The references section lists all cited sources, while the appendices provide the Python code used in this research.

Code

Gives detailed code used in the study. This structure ensures a logical progression from theory to practical applications and findings.

Chapter 2

Literature Review

2.1 Unsupervised Machine Learning Techniques

An important use of unsupervised machine learning methods is the study and classification of astronomical spectra. The ability to find patterns and groupings in big datasets without the need for prior labelling makes these techniques very helpful for exploratory research of astronomical objects. The use of important unsupervised learning techniques in spectral classification is reviewed in this section.

K-Means Clustering

K-Means clustering is a widely used technique in astronomy for classifying spectral data. The algorithm partitions the dataset into K clusters by minimizing the variance within each cluster. MacQueen et al. (1967) introduced the K-Means algorithm, which has since been applied in various astronomical studies. For instance, Ball and Brunner (2010) utilized K-Means to classify galaxy spectra from the Sloan Digital Sky Survey (SDSS), revealing distinct groups of galaxies with similar spectral features. Their findings demonstrated the algorithm's effectiveness in distinguishing between different galaxy types based on their spectral properties.

$$\text{argmin}C \sum i = 1^n \sum_{j=1}^{k} |x_i - \mu_j|^2 \tag{2.1}$$

Where C represents the cluster assignment for each data point , μ_j is the cluster centroid. The choice of K is critical in K-Means clustering. Methods such as the Elbow

method and Silhouette analysis are often employed to determine the optimal number of clusters. Bromová et al. (2014) applied these techniques to identify the appropriate number of clusters for classification of stellar spectra with B and B[e] stars.

Hierarchical Clustering

Hierarchical clustering, which can be either agglomerative or divisive, constructs a tree-like representation of data known as a dendrogram. Johnson (1967) initially proposed this method, and it has since been widely used in astronomy. Agglomerative clustering starts with each data point as a separate cluster and merges them iteratively. Grasha et al. (2017) used hierarchical clustering of young stellar cluster locally and effectively identifying hierarchical structures in the data.

$$d(X, Y) = \min d(x, y) : x \in X, y \in Y \tag{2.2}$$

Where $d(X, Y)$ represents the distance between cluster X and cluster Y. The linkage criterion, such as single, complete, or average linkage, significantly impacts the resulting clusters. According to Ivezić et al. (2020), hierarchical clustering with average linkage works very well for differentiating between star populations.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

DBSCAN is a density-based clustering algorithm that identifies clusters based on the density of data points in a region. Unlike K-Means and hierarchical clustering, DBSCAN can find arbitrarily shaped clusters and is robust to noise. Ester et al. (1996) introduced DBSCAN, and it has been effectively used in various astronomical applications. Strantzalis et al. (2024) applied DBSCAN to identify star clusters in the central region of Small Magellanic Cloud, and it was found to be very successful in recovering actual clusters with high precision.

$$N_\epsilon(p) = q \in D \mid d(p, q) \leq \epsilon \tag{2.3}$$

Where $N_\epsilon(p)$ is the neighborhood of point p with radius ϵ. DBSCAN requires the parameters ϵ (the radius of the neighborhood) and MinPts (the minimum number of points required to form a dense region). Prisinzano et al. (2022) employed DBSCAN on *Gaia* EDR3 data with the goal of locating co-moving and spatially consistent star clusters; this

technique proved to be highly successful for this kind of clustering.

2.2 Dimensionality Reduction Techniques

In order to handle high-dimensional data, lower computing costs, and enhance machine learning algorithm performance, dimensionality reduction approaches are crucial. By converting data into a lower-dimensional space, these techniques maintain the most crucial information. Important dimensionality reduction methods and their uses in spectral classification are reviewed in this section.

Principal Component Analysis(PCA)

Principal Component Analysis (PCA) is a linear dimensionality reduction technique that transforms data into a new coordinate system. The new coordinates, or principal components, are orthogonal and capture the maximum variance in the data. Jolliffe and Cadima (2016) provided a comprehensive introduction to PCA, which has been widely used in astronomy for spectral analysis. $Z = XW$ where X is the data matrix, W is the matrix of eigenvectors, and Z is the transformed data. Singh et al. (1998) used PCA followed by artificial neural networks to analyze the spectra of stars, identifying key features that differentiate various stellar types. Their findings demonstrated the effectiveness of PCA in reducing the dimensionality of spectral data while retaining essential information.

t-Distributed Stochastic Neighbor Embedding (t-SNE)

t-SNE is a non-linear dimensionality reduction technique that visualizes high-dimensional data by embedding it in a lower-dimensional space. It minimizes the divergence between two distributions: one representing pairwise similarities in the high-dimensional space and the other in the lower-dimensional space. Van der Maaten and Hinton (2008) introduced t-SNE, which has become popular for visualizing clusters in data. $\mathrm{KL}(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$ where P is the true joint probability distribution, Q is the reference joint probability distribution, p_{ij} is the probability of the pair (i, j) under P, and q_{ij} is the probability of the pair (i, j) under Q.

By using t-SNE to visualise the spectral data of stars, Traven et al. (2017) was able to identify discrete clusters that corresponded to various star types. Their work demon-

strated how t-SNE can reveal hidden structures in high-dimensional data, offering important new information on how various star classes are classified.

Uniform Manifold Approximation and Projection (UMAP)

UMAP is a non-linear dimensionality reduction technique that optimizes the layout of data in a lower-dimensional space by approximating a manifold. McInnes et al. (2018) introduced UMAP, which has been shown to preserve both local and global structures in the data. $L(M, M') = \sum_{(i,j)\in M} \left(\log \frac{f(d_{ij}^M)}{f(d_{ij}^{M'})} \right)^2$ where M and M' are two different models or matrices, d_{ij}^M and $d_{ij}^{M'}$ are the distances between elements i and j in models M and M', respectively, and $f(\cdot)$ is a function applied to these distances. The expression $L(M, M')$ represents the loss or discrepancy between the two models based on the logarithm of the ratio of the functions of their respective distances.

UMAP can be used in visualizing high-dimensional astronomical data such as in Haggar et al. (2024), who used UMAP to quantify the dynamical state of galaxy clusters, both observational and within simulations. Their findings showed that UMAP could reveal complex structures in the data, aiding in the classification of astronomical objects.

Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) is a supervised dimensionality reduction technique that projects data onto a lower-dimensional space to maximize class separability. Although primarily used for classification, LDA can also be used for dimensionality reduction (Mika et al., 1999). $J(W) = \frac{|W^T S_b W|}{|W^T S_w W|}$ where S_b and S_w are the between-class and within-class scatter matrices, respectively. Zhong-Bao and Li-Peng (2015) applied LDA to classify stellar spectra, achieving high accuracy in distinguishing between different types of stars. Their study demonstrated the effectiveness of LDA in reducing the dimensionality of spectral data while preserving class separability.

Applications in Astronomy

Dimensionality reduction techniques are crucial in the analysis of high-dimensional astronomical data. For instance, Singh et al. (1998) and Haggar et al. (2024) used PCA and UMAP to analyze stellar and galaxy spectra, revealing distinct groups of stars and galaxies with similar properties. These studies have shown that dimensionality reduc-

tion can enhance the interpretability of complex data and improve the performance of clustering algorithms.

Visualizations

Visualizations play a key role in understanding the results of dimensionality reduction techniques. Common visualizations include:

- PCA Projection: A 2D or 3D scatter plot showing the principal components.
- t-SNE Visualization: A 2D or 3D scatter plot revealing clusters in the data.
- UMAP Visualization: A 2D or 3D plot preserving local and global data structures.
- LDA Projection: A plot showing the linear discriminants that separate different classes.

These visualizations help researchers to interpret the results and to identify meaningful patterns in the data. In the above mentioned dimensionality reduction techniques, PCA was chosen for its simplicity, interpretability, and ability to effectively reduce dimensionality while preserving variance, making it suitable for exploratory data analysis and visualization. DBSCAN was not used due to its time-consuming nature and high memory requirements, which would necessitate reducing the dataset size, an impractical solution given the project's scope.

2.3 Evaluation Metrics

Evaluating the performance of clustering algorithms in spectral classification involves several metrics:

Silhouette Score measures how similar a data point is to its own cluster compared to other clusters, with higher values indicating better-defined clusters (Rousseeuw, 1987).

Davies-Bouldin Index (DBI) assesses the average ratio of within-cluster distances to between-cluster distances, where lower values indicate better clustering (Davies and Bouldin, 1979).

Calinski-Harabasz Index (CHI) is the ratio of the sum of between-cluster dispersion to within-cluster dispersion, with higher values indicating better-defined clusters (Caliński and Harabasz, 1974).

Cluster Purity measures the extent to which clusters contain a single class, calculated as the sum of the maximum intersection of each cluster with a class divided by the total number of data points (Decherchi et al., 2009).

Adjusted Rand Index (ARI) measures the similarity between the true labels and the clustering result, adjusted for chance (Hubert and Arabie, 1985).

Normalized Mutual Information (NMI) measures the mutual dependence between the true labels and the clustering result (Strehl and Ghosh, 2002).

Applications in Astronomy

Unsupervised learning techniques have significant applications in astronomy. For example, White and Frenk (1991) and Bromová et al. (2014) used K-Means and hierarchical clustering to classify galaxies and stars based on their spectral and photometric properties. These studies have demonstrated the ability of unsupervised learning algorithms to identify distinct groups of celestial objects, revealing important insights into their physical characteristics and evolutionary history.

2.4 Previous Work on Spectral Classification

Astronomical classification, which offers a methodical approach to classify celestial objects according to their spectra, has been a fundamental aspect of the discipline. The foundation of modern spectral classification was laid in the early 20th century with the development of the Harvard Classification Scheme. This system, pioneered by Pickering (1886), categorized stars based on the strength of hydrogen lines in their spectra. The scheme was further refined by Cannon and Pickering (1912), who introduced the OBAFGKM sequence, organizing stars by temperature and spectral characteristics. Annie J. Cannon expanded this classification system in the Henry Draper Catalogue, classifying over 225,000 stars (Welther, 1993). This monumental work established a comprehensive framework for stellar classification, which has been foundational for subsequent studies. The Morgan-Keenan (MK) system, developed by Morgan et al. (1942), introduced luminosity classes to the spectral classification, distinguishing between dwarf stars, giants, and supergiants. This system provided a more detailed understanding of stellar properties and their evolutionary stages.

Finer spectral information were added by Gray (2021) to their contemporary expansion of the MK method, allowing for more accurate classifications. Their research showed how crucial high-resolution spectroscopy is for identifying minute variations in star spectra. Large-scale astronomical surveys and digital spectroscopy made automated spectral classification essential. Bailer-Jones et al. (1998) utilized neural networks and pioneered automated classification methods, demonstrating their efficacy in handling large datasets from surveys like the Sloan Digital Sky Survey (SDSS).

Wang et al. (2017) applied machine learning algorithms to classify stellar spectra from the SDSS, achieving high accuracy and efficiency. Their research opened the door for more sophisticated methods by demonstrating the potential of machine learning in handling enormous volumes of spectral data. In recent years, machine learning has revolutionized spectral classification. Fabbro et al. (2018) employed convolutional neural networks (CNNs) to classify stellar spectra, demonstrating significant improvements over traditional methods. Their approach leveraged the hierarchical nature of CNNs to capture intricate spectral features, resulting in robust classifications. For spectral classification, Kim et al. (2015) presented a hybrid model that combines principal component analysis (PCA) and support vector machines (SVMs). Their model demonstrated the advantages of mixing various machine learning techniques by effectively reducing the spectral data's dimensionality while retaining classification accuracy. Spectral classification has been widely used not just for stars but also for galaxies. In order to differentiate between star-forming and quiescent galaxies Connolly et al. (1994) developed a technique to classify galaxies based on their spectral energy distributions (SEDs). Their work provided insights into galaxy evolution and star formation processes. Kauffmann et al. (2003) utilized spectral diagnostics to classify galaxies from the SDSS, identifying distinct populations based on their emission lines. Their study highlighted the importance of spectral classification in understanding the physical properties and evolutionary stages of galaxies. Despite significant advancements, spectral classification faces several challenges. Graham et al. (2017) highlighted the issue of spectral contamination, where overlapping spectral features from variable stars or background objects complicate classifications. Addressing these challenges requires improved data preprocessing techniques and more sophisticated models.

Future work in spectral classification is likely to focus on integrating multi-wavelength

data. Schreiber (2016) demonstrated the potential of combining optical and infrared spectra to achieve more comprehensive classifications. Such approaches can leverage the full spectrum of information available from modern telescopes, providing deeper insights into the nature of celestial objects.

Furthermore, more developments in automated categorisation techniques will be required due to the growing amount of data from future surveys such as the Large Synoptic Survey Telescope (LSST). Sen et al. (2022), 4MOST, DESI discussed the role of machine learning in handling these large datasets, emphasizing the need for scalable and efficient algorithms. The field of spectral classification has evolved significantly since its inception, driven by advancements in observational techniques and data analysis methods. From the early work of Cannon and Pickering (1912) and Morgan et al. (1942) to the modern machine learning approaches of Fabbro et al. (2018) and Kim et al. (2015), each development has contributed to a more nuanced understanding of the universe. As new technologies and methodologies emerge, spectral classification will continue to be a vital tool in the astronomer's toolkit, enabling the exploration and characterization of the cosmos.

Chapter 3

Data Acquisition and Preprocessing

3.1 Data Sources

The dataset used in this study comprises of 15,000 spectra obtained from the Sloan Digital Sky Survey (SDSS) Data Release 18 (DR18). SDSS is one of the most extensive astronomical surveys, providing detailed optical spectra, imaging, and redshift information for millions of celestial objects, including stars, galaxies, and quasars (Almeida et al., 2023). The SDSS DR18 dataset continues this tradition, offering high-quality spectral data essential for our spectral classification study.

Sloan Digital Sky Survey (SDSS)

The SDSS, initiated in 2000s, has revolutionized our understanding of the cosmos by providing a wealth of data for both imaging and spectroscopy. Its innovative use of a 2.5-meter telescope at the Apache Point Observatory in New Mexico has enabled the capture of spectra for over three million objects (York et al., 2000). SDSS has released successive data sets, with DR18 being one of the latest and most comprehensive.

SDSS DR18 Dataset

The SDSS DR18 dataset includes spectra with a wavelength range of approximately 3800 to 9200 Å, covering the ultraviolet to the near-infrared regions of the electromagnetic spectrum. Each spectrum is recorded with a resolution power of around 2000, ensuring detailed spectral features can be accurately identified (Blanton et al., 2017).

The spectra are provided in the form of FITS (Flexible Image Transport System) files, each containing multiple extensions. The primary extensions used in this study include:

- The flux array: which records the intensity of light at each wavelength.
- The log-wavelength array: which provides the logarithm of the wavelength values corresponding to the flux array.
- Metadata: including the classification and redshift of the object.

The spectra were preprocessed to correct for instrumental effects and to remove cosmic rays. Sample spectra are plotted in 3.1

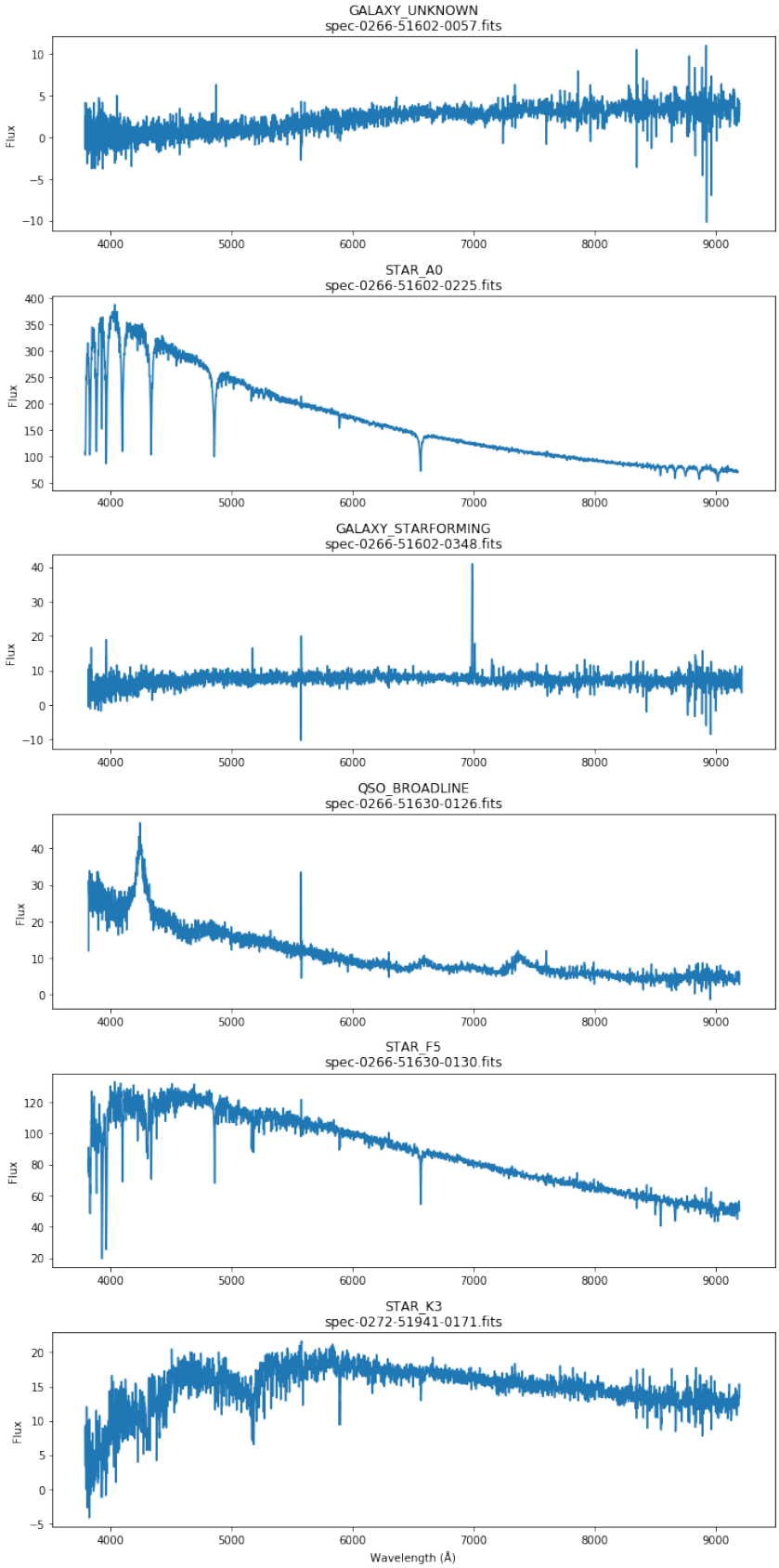

Figure 3.1: Sample spectra from the SDSS DR18 dataset. The spectra exhibit characteristic features such as emission and absorption lines, which are crucial for spectral classification.

Visualization and Interpretation

To illustrate the diversity and quality of the spectra in the dataset, Figure 3.1 shows sample spectra from different types of celestial objects. This visual representation highlights the variety of spectral features present and underscores the importance of spectral classification in astronomy.

This data was obtained by executing a query on the official SDSS website to obtain 15,000 randomly selected spectra from a vast collection comprising a variety of objects.

Class Distribution

The SDSS DR18 dataset encompasses a diverse range of celestial objects, categorized broadly into galaxies, stars, and quasars. Understanding the distribution of these classes is fundamental for evaluating the dataset's comprehensiveness and the balance among different object types.

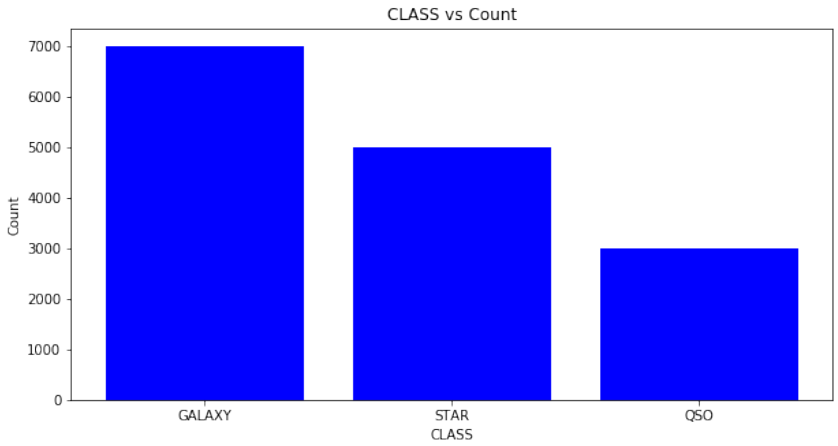

Figure 3.2: Number of objects by CLASS. The dataset is predominantly composed of galaxies, with a smaller number of stars and quasars.

As shown in Figure 3.2, the majority of the objects in the dataset are galaxies, with stars and quasars forming smaller but significant portions. This distribution is reflective of the selection criteria of the SDSS, aimed at obtaining a comprehensive sample of the universe's diverse constituents. The dataset consists of around 7000 galaxies, 5000 stars and 3000 QSO's. This was chosen so as to preserve the comprehension of the dataset while avoiding significantly distorting it with a single class. In the SDSS dataset, for example, the majority of objects are galaxies than stars and fewer QSOs.

Subclass Distribution

Further breakdown of the dataset into subclasses provides a more granular view of the types of objects included. This detailed categorization is essential for specific classification tasks and for understanding the dataset's richness.

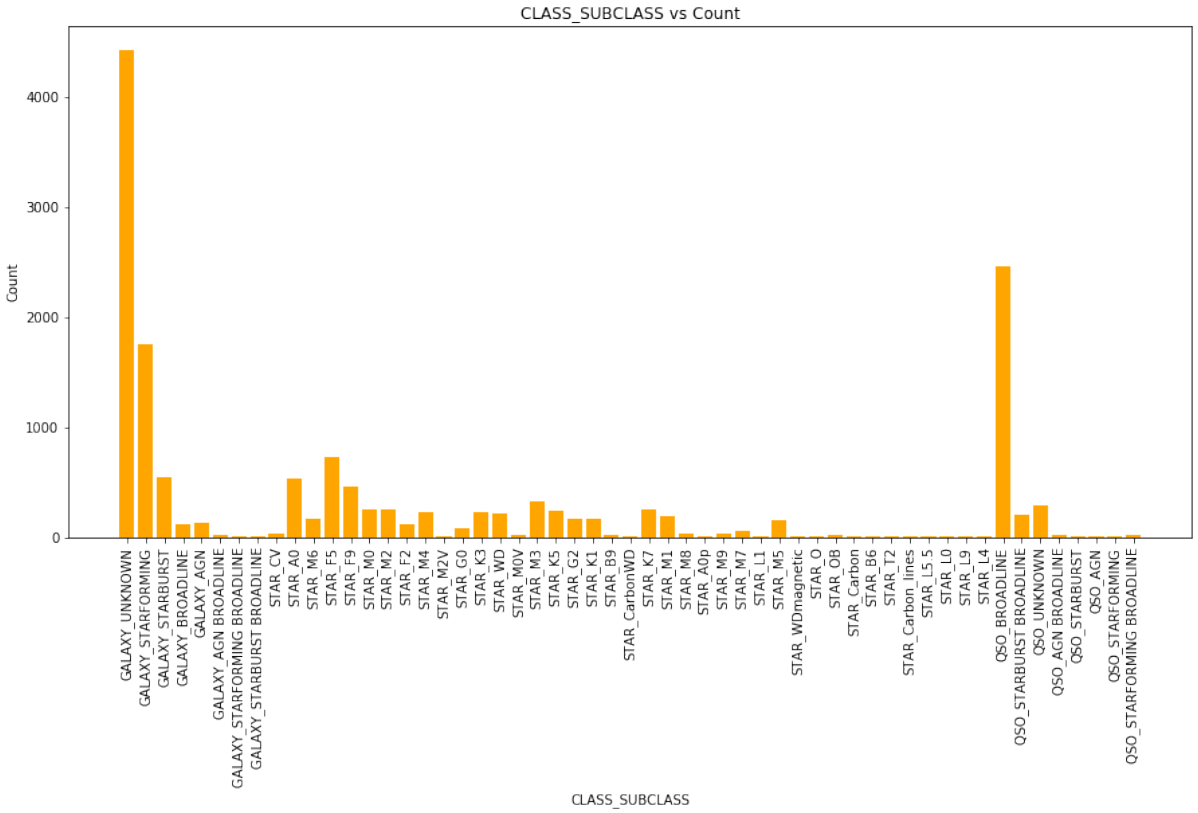

Figure 3.3: Number of objects by CLASS_SUBCLASS. The dataset includes a wide variety of subclasses, highlighting the diversity of objects surveyed by SDSS.

Figure 3.3 illustrates the number of objects within each subclass. Notably, the "GALAXY_UNKNOWN" category has the highest count, indicating a significant portion of galaxies that require further analysis for precise classification. The presence of various star subclasses and quasar types showcases the dataset's capability to support detailed astrophysical studies.

Significance of the Data

The SDSS DR18 dataset is invaluable for studying various astronomical phenomena. Robust machine learning models and extensive statistical analysis are made possible by its vast sample size and excellent spectra. The comprehensive spectral information in this collection makes it possible to classify celestial objects in detail according to their

spectral characteristics, which is essential for comprehending their physical characteristics and evolutionary stages.

For instance, the spectra exhibit prominent features such as hydrogen Balmer lines, metal absorption lines, and molecular bands, which are key indicators of the temperature, composition, and other physical characteristics of the objects. These features are used to classify objects into categories such as main-sequence stars, white dwarfs, red giants, and various types of galaxies and quasars.

Future Data Releases

Future data releases from SDSS and other astronomical surveys like DESI and 4MOST will continue to expand the available spectral data, offering new opportunities for research and discovery. More improvements in the amount and quality of spectral data will be made possible by the continuous developments in telescope technology and data processing methods, allowing for more precise and in-depth classifications.

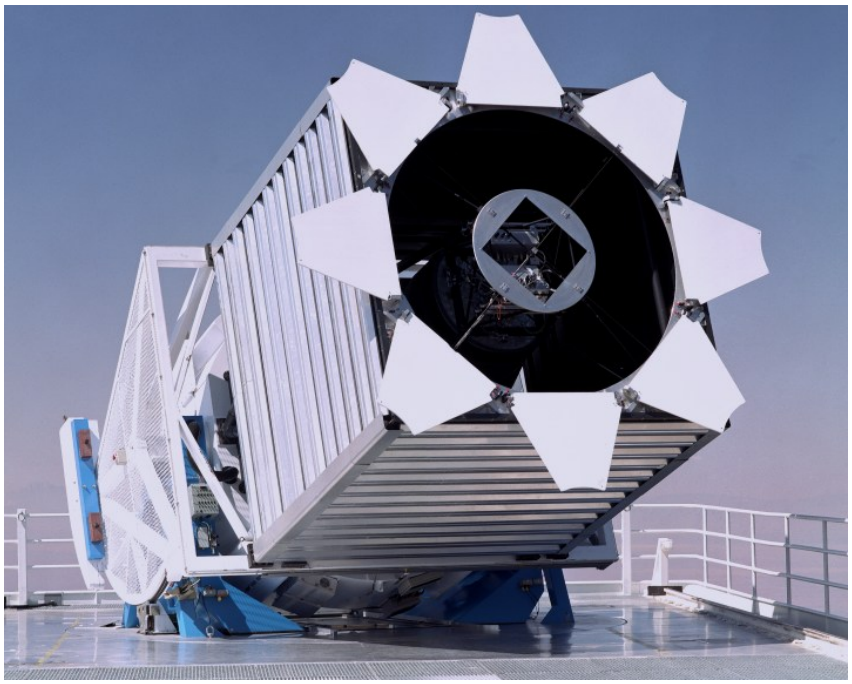

Figure 3.4: The 2.5-meter SDSS telescope at the Apache Point Observatory, used for data acquisition in the SDSS project. Credit: https://www.sdss.org/instruments

3.2 Data Preprocessing

Data preprocessing is a crucial step in any machine learning project, especially in spectral classification, where the quality and consistency of the input data significantly impact the performance of the model. We describe here the procedures used to preprocess the spectral data that were taken from the SDSS DR18 dataset. This involves normalisation, smoothing, resampling and the extraction of relevant columns.

3.2.1 Data Extraction

The raw data from the SDSS DR18 dataset are stored in FITS (Flexible Image Transport System) files, which contain several extensions. The primary extensions used in this study include the flux array, log-wavelength array, and metadata. Table 3.1 provides a description of the columns used in the project.

Table 3.1: Description of Columns Used from FITS File

Column Name	**Description**
`flux`	Array of flux values (intensity of light) at each wavelength
`loglam`	Logarithm of the wavelength values corresponding to the flux array
`z`	Redshift of the object
`CLASS`	Spectral classification of the object (e.g., Star, Galaxy, QSO)
`SUBCLASS`	Sub-classification of the object (e.g., Galaxy Starforming, Star A0, Star K3, QSO-Broadline)
`plate`	Plate number of the observation
`mjd`	Modified Julian Date of the observation
`fiberID`	Fiber identification number

3.2.2 Normalization

Normalization is the process of scaling the data to a common range, which is essential for ensuring that the features contribute equally to the model's learning process. For spectral data, normalization typically involves scaling the flux values to a range between 0 and 1.

The normalization of a flux value f_i can be performed using the following formula:

$$f_i' = \frac{f_i - f_{\min}}{f_{\max} - f_{\min}}$$

where f_i' is the normalized flux value, f_i is the original flux value, $f_{\min}$ is the minimum flux value in the spectrum, and $f_{\max}$ is the maximum flux value in the spectrum.

Figure 3.5 shows a raw spectrum before normalization. The flux values in the units erg s^{-1} cm^{-2} $Å^{-1}$ vary significantly, making it challenging for the machine learning model to learn effectively. After normalization, as shown in Figure 3.6, the flux values are scaled to a uniform range, facilitating better model performance.

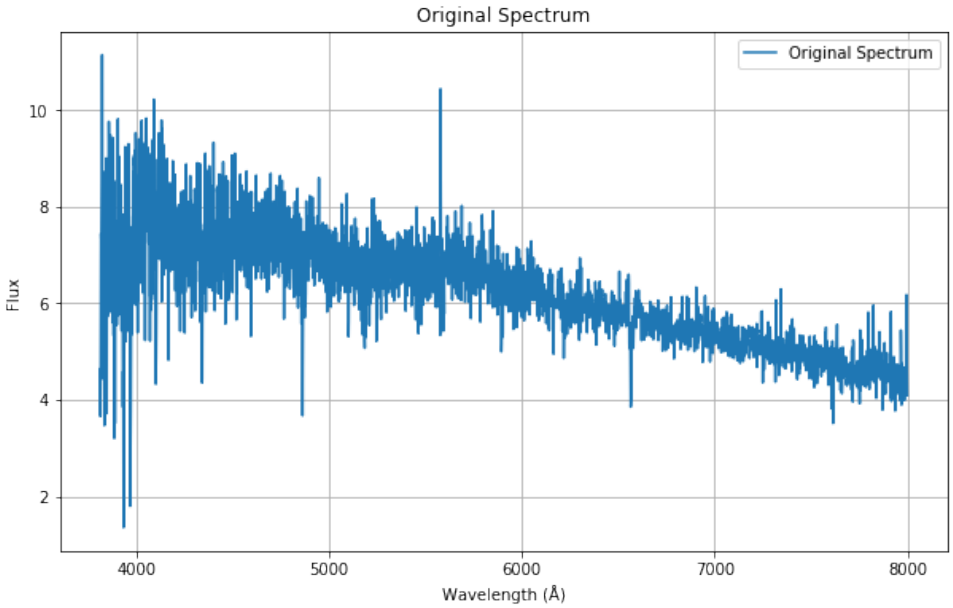

Figure 3.5: Raw Spectra of an Object.

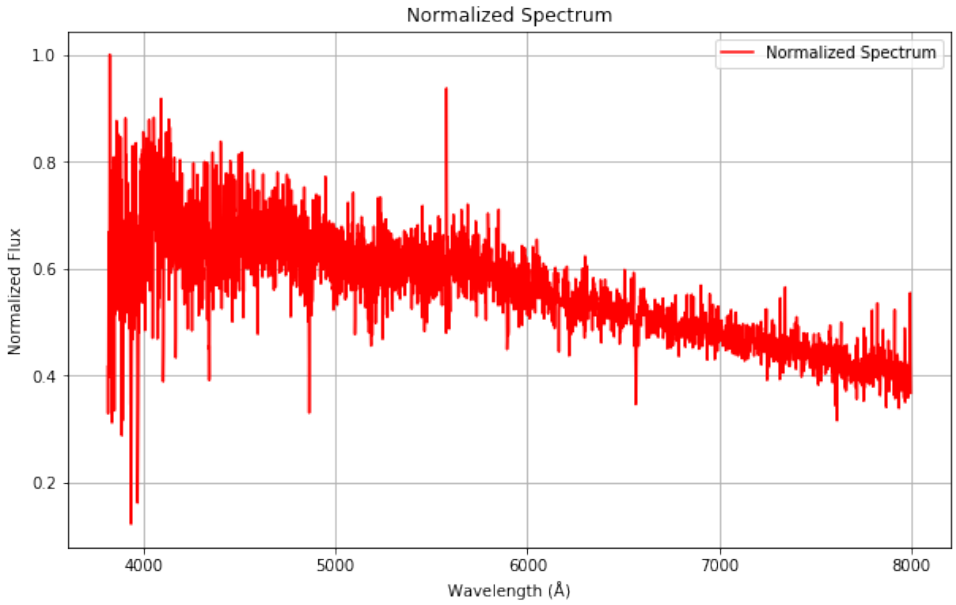

Figure 3.6: Spectra after normalization.

3.2.3 Smoothing

Smoothing is applied to reduce noise and enhance significant features in the spectral data. A common smoothing technique is the Savitzky-Golay filter, which fits successive polynomials to the data to preserve high-frequency features while reducing noise.

The Savitzky-Golay filter works by performing a local polynomial regression on a series of values to determine the smoothed value for each point. The formula for the Savitzky-Golay filter can be expressed as:

$$y_i' = \sum_{j=-m}^{m} c_j y_{i+j}$$

where y_i' is the smoothed value at point i, y_{i+j} are the original data points, c_j are the filter coefficients, and m is the window size.
Figure 3.5 displays a spectrum before smoothing, highlighting the presence of noise. After applying the Savitzky-Golay filter, the resulting spectrum in Figure 3.7 shows a smoother curve with reduced noise, allowing for more accurate classification of spectral features.

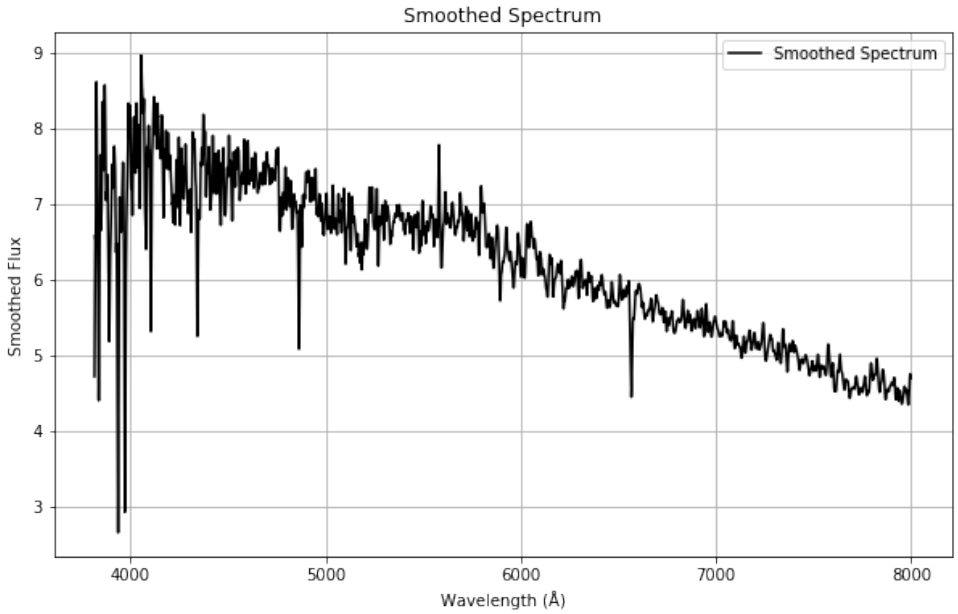

Figure 3.7: Spectra after smoothing using the Savitzky-Golay filter.

3.2.4 Resampling

In spectral analysis, resampling is an essential pre-processing step that attempts to standardise the wavelength intervals across all of the dataset's spectra. By doing this, it is made sure that every spectrum has an equal amount of data points, which makes comparison and other machine learning jobs easier. We go into great length on the resampling procedure in this part, along with the underlying mathematical formulas and how they affect the spectral data.

Importance of Resampling

Variations in the observational setup and conditions result in variable wavelength intervals in the spectra acquired from the SDSS DR18 dataset. This unpredictability can render the processes of analysis and classification more difficult. In order to ensure consistency throughout the dataset, resampling interpolates the spectra onto a similar wavelength grid in order to address this problem.

Resampling Process

The resampling process involves the following steps:

1. **Define a Common Wavelength Grid**: A common wavelength grid is defined, spanning the entire wavelength range covered by the dataset. The grid is typically chosen

to have equally spaced intervals (e.g., 1 Å).

2. **Interpolation**: Each spectrum is interpolated onto the common wavelength grid. Interpolation techniques such as linear interpolation, spline interpolation, or cubic interpolation can be used. In this study, linear interpolation is employed due to its simplicity and efficiency.

The mathematical formulation for linear interpolation is given by:

$$f(\lambda) = f(\lambda_i) + \frac{f(\lambda_{i+1}) - f(\lambda_i)}{\lambda_{i+1} - \lambda_i}(\lambda - \lambda_i) \tag{3.1}$$

where $f(\lambda)$ is the interpolated flux at wavelength λ, λ_i and λ_{i+1} are the neighboring wavelengths in the original spectrum, and $f(\lambda_i)$ and $f(\lambda_{i+1})$ are the corresponding flux values.

3. **Resampling**: Using the interpolation formula, the flux values are calculated at each point of the common wavelength grid. This results in resampled spectra with uniform wavelength intervals.

Impact of Resampling on Spectra

Resampling has several significant impacts on the spectral dataset:

- **Uniform Data Points**: Each resampled spectrum has the same number of data points, making it easier to apply machine learning algorithms that require input data of consistent dimensions.
- **Enhanced Comparability**: Resampling makes it easier to compare different spectra directly by matching them onto a common wavelength grid, which helps to identify common patterns and characteristics.
- **Improved Data Quality**: Interpolation helps to fill in gaps and smooth out irregularities in the original data, leading to cleaner and more reliable spectra.

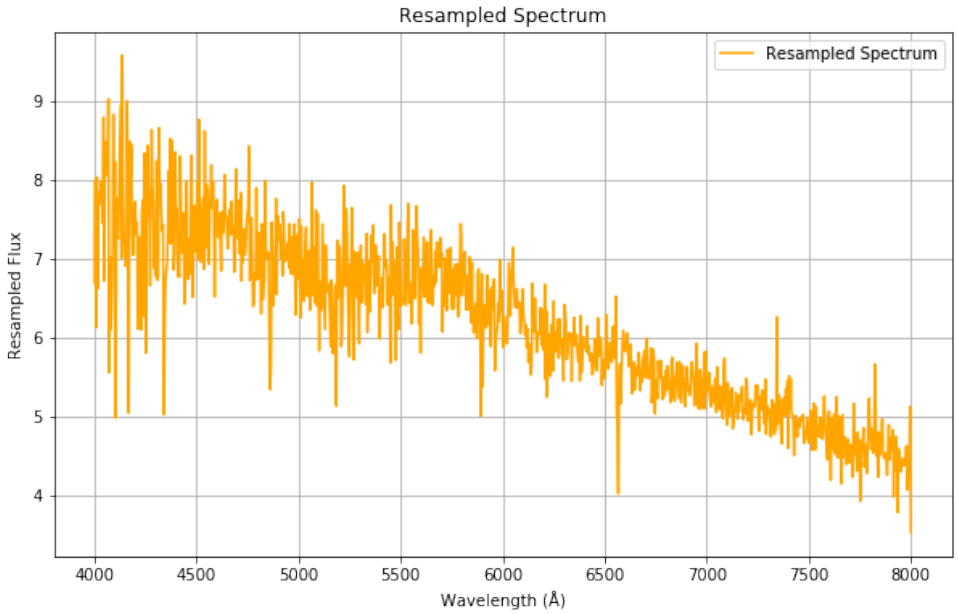

Figure 3.8: Sample spectrum after resampling onto a common wavelength grid.

3.2.5 Final Preprocessed Data

After applying normalization, smoothing and resampling, the spectra are ready for further analysis and classification. In order to ensure that the machine learning models can effectively learn from the data, the pre-processed spectra are free of baseline trends and noise while maintaining their significant features.

Figure 3.9 when compared to original Figure 3.5 shows how a sample spectrum after all preprocessing steps have been applied. The figure demonstrates how the combined preprocessing techniques improve the clarity and consistency of the spectral data.

During the preprocessing of the SDSS DR18 dataset, several challenges were encountered, including dealing with noisy data and varying spectral resolutions. Each of these challenges required careful consideration and the application of appropriate preprocessing techniques to ensure the integrity and usability of the data.

- **Noisy Data**: The Savitzky-Golay filter was chosen for its ability to preserve high-frequency features while effectively reducing noise.

- **Varying Spectral Resolutions**: Normalization ensured that all spectra were scaled to a common range, making them comparable despite resolution differences.

By addressing these challenges, the preprocessing pipeline ensured that the resulting

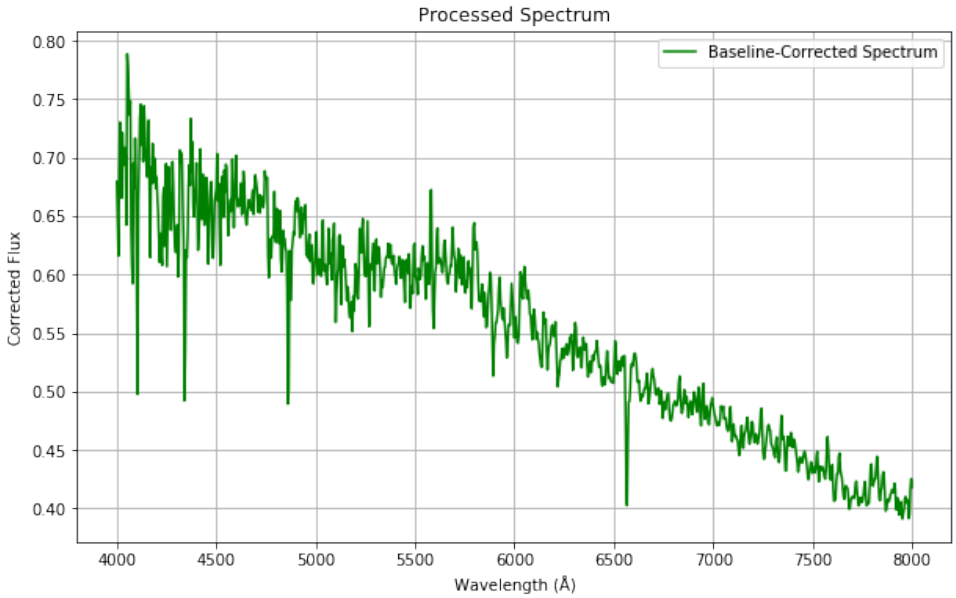

Figure 3.9: Sample spectrum after normalization, smoothing, baseline correction, and resampling.

data were of high quality and suitable for subsequent machine learning tasks.

Effective data preprocessing is essential for the success of any machine learning project, particularly in spectral classification. To improve the quality and consistency of the SDSS DR18 spectral data, the preprocessing procedures—normalization, smoothing, baseline correction, and resampling—described in this section were thoughtfully planned and carried out.

The resulting preprocessed data, devoid of systematic patterns and noise, offer a strong basis for machine learning model evaluation and training, which eventually results in spectral classifications that are more precise and trustworthy.

Chapter 4

Methodology

4.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a fundamental dimensionality reduction technique widely used in data analysis and machine learning. In this project, PCA is critical for reducing the high-dimensional spectral data to a lower-dimensional space while retaining essential features. This section details the PCA process, including its mathematical formulation and application to the preprocessed spectral dataset.

Introduction to PCA

PCA is an unsupervised learning algorithm that transforms original data into a new coordinate system, where the first coordinate (first principal component) has the greatest variance, the second has the second greatest variance, and so forth. This transformation reduces data dimensionality, simplifying analysis and visualization.

Mathematical Formulation

The PCA process involves the following steps:

Standardization :

The data is first standardized to have a mean of zero and a standard deviation of one, as PCA is sensitive to the variances of the original variables. For a dataset X with n samples and p features, the standardized data matrix Z is computed as:

$$Z = \frac{X - \mu}{\sigma} \tag{4.1}$$

where μ is the mean and σ is the standard deviation of X.

Covariance Matrix Calculation :

Next, the covariance matrix Σ is calculated to understand how variables correlate. It is computed as:

$$\Sigma = \frac{1}{n-1} Z^T Z \tag{4.2}$$

Eigenvalue Decomposition :

The covariance matrix is decomposed into eigenvalues and eigenvectors, where eigenvalues represent the variance explained by each principal component, and eigenvectors determine the component directions:

$$\Sigma v_i = \lambda_i v_i \tag{4.3}$$

Here, λ_i are the eigenvalues, and v_i are the eigenvectors.

Selecting Principal Components :

Eigenvectors are sorted based on eigenvalues in descending order. The top k eigenvectors (principal components) are selected to form a new feature space. The number k is chosen based on cumulative explained variance, ensuring that selected components capture significant data variance.

Transforming the Data :

The original data is projected onto the new feature space defined by the selected principal components, resulting in a transformed data matrix Y:

$$Y = ZV_k \tag{4.4}$$

where V_k is the matrix of the top k eigenvectors.

Application to Spectral Data

In this project, PCA is applied to the preprocessed spectral data for dimensionality reduction prior to clustering. The following outlines its application:

Standardization : The preprocessed spectral data, after normalization, smoothing, and resampling, is standardized to ensure a mean of zero and a standard deviation of one, making it suitable for PCA.

Covariance Matrix and Eigenvalue Decomposition : The covariance matrix is computed from the standardized data, followed by eigenvalue decomposition. The eigenvalues and eigenvectors are obtained, representing the variances and directions of the principal components.

Selecting Principal Components : The eigenvalues are sorted, and the top k components are selected. In this study, three principal components are chosen, capturing a significant portion of the variance in the spectral data.

Data Transformation : The standardized spectral data is projected onto the new feature space, resulting in a lower-dimensional representation that facilitates visualization and clustering.

Impact of PCA on Spectral Data

PCA reduces the dimensionality of the spectral dataset while retaining key features, producing a more manageable and interpretable dataset. This reduction simplifies the application of clustering algorithms like K-Means and Agglomerative Clustering, enhancing the accuracy and efficiency of spectral classification.

By reducing data complexity, PCA enables more effective clustering and classification, ultimately improving spectral data analysis. The PCA process and mathematical formulations ensure that essential information is preserved for accurate spectral analysis.

In conclusion, PCA is an essential dimensionality reduction method that preserves key features while transforming high-dimensional spectral data into a lower-dimensional space. Applying PCA to preprocessed spectral data produces a more manageable and comprehensible dataset, leading to more successful clustering and classification.

4.2 K-Means Clustering

K-Means Clustering is a widely utilized unsupervised learning technique for partitioning datasets into distinct, non-overlapping clusters. In this project, spectral data undergoes dimensionality reduction via Principal Component Analysis (PCA) before being grouped into clusters using K-Means. This section details the K-Means procedure, including its mathematical formulation, algorithmic steps, and application to the spectral dataset.

Introduction to K-Means Clustering

K-Means Clustering seeks to partition n observations into k clusters, where each observation belongs to the cluster with the nearest mean, representing the cluster's prototype. The primary objective is to minimize the within-cluster sum of squares (WCSS), also known as inertia, which measures the variance within each cluster.

Mathematical Formulation

Given a dataset $\{x_1, x_2, \ldots, x_n\}$ in $\mathbb{R}^d$, K-Means partitions the data into k clusters $\{C_1, C_2, \ldots, C_k\}$ by minimizing the following cost function:

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2 \tag{4.5}$$

where μ_i is the centroid of cluster C_i, defined as:

$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x \tag{4.6}$$

Algorithmic Steps

The K-Means algorithm follows these iterative steps:

1. **Initialization:** Randomly select k initial centroids from the dataset.
2. **Assignment Step:** Assign each data point to the nearest centroid using the Euclidean distance:

$$d(x, \mu_i) = \|x - \mu_i\| \tag{4.7}$$

3. **Update Step:** Recalculate the centroids by averaging the data points within each cluster:

$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x \tag{4.8}$$

4. **Convergence Check:** Repeat the assignment and update steps until centroids stabilize or a maximum number of iterations is reached.

Application to Spectral Data

In this study, K-Means Clustering is applied to spectral data post-PCA. The process is outlined below:

Initialization: The reduced-dimensionality spectral data serves as input to the K-Means algorithm. The number of clusters k is determined based on domain knowledge or exploratory data analysis.

Assignment and Update: The algorithm assigns each spectrum to the nearest centroid, updating centroids iteratively.

Optimal Number of Clusters: Determining the optimal k is crucial. Several methods can be employed:

- **Elbow Method:** Plot WCSS against k and identify the "elbow point" where the decrease in WCSS slows.
- **Silhouette Score:** Evaluate clustering quality by comparing intra-cluster cohesion to inter-cluster separation.
- **Davies-Bouldin Index:** Assess cluster similarity relative to their most similar counterparts.

Impact on Spectral Data

K-Means Clustering effectively partitions spectral data into meaningful groups, potentially representing distinct astronomical objects or conditions. The clustering results provide insights into the underlying structure of the spectral data, facilitating further analysis and classification tasks.

Challenges and Considerations

- **Scalability:** K-Means can be computationally demanding for large datasets. Mini-Batch K-Means can be employed to improve scalability.
- **Initialization Sensitivity:** The choice of initial centroids significantly impacts clustering results. Multiple runs with different initializations (e.g., K-Means++) help mitigate this issue.
- **Cluster Shape Assumption:** K-Means assumes spherical, similarly-sized clusters, which may not always hold in real-world data. For complex cluster shapes, alternative algorithms like DBSCAN or Gaussian Mixture Models may be more appropriate.

In conclusion, K-Means Clustering is integral to this project's methodology, enabling the grouping of spectral data into meaningful clusters. By minimizing within-cluster variance, K-Means aids in pattern recognition, laying the foundation for further analysis and interpretation. The combination of PCA and K-Means Clustering provides a robust approach for managing high-dimensional spectral data, enhancing the accuracy and efficiency of spectral classification applications.

4.3 Agglomerative Clustering

Agglomerative Clustering, or hierarchical clustering, is a powerful unsupervised learning algorithm used to uncover nested clusters in data. Agglomerative Clustering creates a hierarchy of clusters visualized in a dendrogram. This section explains the Agglomerative Clustering process, including its mathematical formulation, algorithmic steps, and application to spectral data.

Introduction to Agglomerative Clustering

Agglomerative Clustering is a bottom-up approach where each data point starts as its own cluster, and clusters are merged based on a distance metric, which measures dissimilarity between them.

Mathematical Formulation

Given a dataset $\{x_1, x_2, \ldots, x_n\}$ in $\mathbb{R}^d$, the algorithm begins with n clusters, each containing one data point. The algorithm iteratively merges the two closest clusters according to a linkage criterion until all data points are grouped into a single cluster.

Linkage Criteria: Various linkage criteria determine the distance between clusters:

- **Single Linkage:** The minimum distance between any two points in the clusters:

$$d_{\text{single}}(C_i, C_j) = \min_{x \in C_i, y \in C_j} \|x - y\| \tag{4.9}$$

- **Complete Linkage:** The maximum distance between any two points in the clusters:

$$d_{\text{complete}}(C_i, C_j) = \max_{x \in C_i, y \in C_j} \|x - y\| \tag{4.10}$$

- **Average Linkage:** The average distance between all pairs of points, one from each cluster:

$$d_{\text{average}}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i} \sum_{y \in C_j} \|x - y\| \tag{4.11}$$

- **Ward's Linkage:** Minimizes total within-cluster variance. The distance is defined as the increase in variance when clusters are merged:

$$d_{\text{Ward}}(C_i, C_j) = \sum_{x \in C_i \cup C_j} \|x - \mu_{C_i \cup C_j}\|^2 - \sum_{x \in C_i} \|x - \mu_{C_i}\|^2 - \sum_{x \in C_j} \|x - \mu_{C_j}\|^2 \tag{4.12}$$

Algorithm Steps: The Agglomerative Clustering algorithm proceeds as follows:

1. **Initialization:** Start with n clusters, each containing a single data point.
2. **Distance Calculation:** Compute distances between all pairs of clusters using the chosen linkage criterion.
3. **Cluster Merging:** Merge the two clusters with the smallest distance.
4. **Update:** Recompute distances between the new cluster and remaining clusters.
5. **Repeat:** Continue merging until all points are in a single cluster.

Application to Spectral Data

In this project, Agglomerative Clustering is applied to spectral data after PCA-based dimensionality reduction. The steps include:

Initialization: The lower-dimensional spectral data is used as input, with each spectrum starting as an individual cluster.

Distance Calculation and Merging: The algorithm calculates pairwise distances between all clusters and merges the closest pairs iteratively.

Determining the Number of Clusters: To extract meaningful clusters, the dendrogram is "cut" at a specific height, corresponding to a distance threshold. This threshold can be selected based on domain knowledge or cluster validation indices like the silhouette score.

Impact on Spectral Data

Agglomerative Clustering reveals hierarchical relationships within spectral data, aiding in the identification of nested clusters. These clusters might represent distinct astronomical objects or varying conditions within similar object types. The method complements K-Means Clustering, offering a deeper understanding of the data's structure.

Challenges and Considerations

- **Computational Complexity:** The algorithm's $O(n^3)$ time complexity can make it computationally expensive for large datasets.
- **Linkage Criterion Sensitivity:** The choice of linkage criterion significantly impacts results, necessitating careful selection based on domain knowledge and exploratory analysis.
- **Dendrogram Interpretation:** Dendrograms can be complex, particularly for large datasets, and selecting the correct threshold for cutting is crucial for meaningful clustering.

Agglomerative Clustering is essential in this project's methodology, enabling the exploration of hierarchical relationships in spectral data. Combined with PCA, it provides a powerful tool for managing high-dimensional data, enhancing spectral classification accuracy and efficiency.

4.4 Evaluation Metrics

Evaluation metrics are essential for assessing the performance of clustering algorithms. They provide quantitative measures to compare different models and determine the quality of the clustering results. This section discusses various evaluation metrics used in this project, including Silhouette Score, Davies-Bouldin Index, Calinski-Harabasz Index, Cluster Purity, Adjusted Rand Index (ARI), and Normalized Mutual Information (NMI).

4.4.1 Silhouette Score

The Silhouette Score measures how similar an object is to its own cluster compared to other clusters. It ranges from -1 to 1, with higher values indicating better clustering.

For a data point i, let $a(i)$ be the average distance to all other points in the same cluster and $b(i)$ be the minimum average distance to points in a different cluster. The silhouette coefficient $s(i)$ for point i is defined as:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \tag{4.13}$$

The overall Silhouette Score is the mean silhouette coefficient of all data points:

$$S = \frac{1}{n} \sum_{i=1}^{n} s(i) \tag{4.14}$$

In this project, the Silhouette Score helps evaluate the cohesion and separation of clusters formed by K-Means and Agglomerative Clustering.

4.4.2 Davies-Bouldin Index

The Davies-Bouldin Index (DBI) evaluates the average similarity ratio of each cluster with its most similar cluster. Lower DBI values indicate better clustering.

For each cluster C_i, let σ_i be the average distance between each point in C_i and the centroid of C_i. The similarity between two clusters C_i and C_j is defined as:

$$R_{ij} = \frac{\sigma_i + \sigma_j}{d(C_i, C_j)} \tag{4.15}$$

where $d(C_i, C_j)$ is the distance between the centroids of C_i and C_j.

The Davies-Bouldin Index is the average of the maximum R_{ij} values for each cluster:

$$DBI = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} R_{ij} \tag{4.16}$$

In this project, DBI is used to assess the compactness and separation of clusters.

4.4.3 Calinski-Harabasz Index

The Calinski-Harabasz Index (CHI), also known as the Variance Ratio Criterion, evaluates the ratio of the sum of between-cluster dispersion and within-cluster dispersion. Higher CHI values indicate better-defined clusters.

Let B_k be the between-cluster dispersion matrix and W_k be the within-cluster dispersion matrix. The CHI is defined as:

$$CHI = \frac{\text{trace}(B_k)}{\text{trace}(W_k)} \cdot \frac{N - k}{k - 1} \tag{4.17}$$

where N is the number of data points and k is the number of clusters.

In this project, CHI provides insight into the separation and cohesion of clusters formed by different algorithms.

4.4.4 Cluster Purity

Cluster Purity measures the extent to which clusters contain a single class. Higher purity indicates better clustering performance.

For a cluster C_i, let n_i be the number of data points in C_i and m_i be the number of data points in the most common class that is determined by the most repetative label in

the data within C_i. Cluster Purity is defined as:

$$\text{Purity} = \frac{1}{N} \sum_{i=1}^{k} m_i \tag{4.18}$$

In this project, Cluster Purity helps evaluate the accuracy of clustering algorithms in assigning spectral data to the correct classes.

4.4.5 Adjusted Rand Index (ARI)

The Adjusted Rand Index (ARI) measures the similarity between the true labels and the predicted clusters, adjusted for chance. It ranges from -1 to 1, with higher values indicating better clustering.

Let a be the number of pairs of data points that are in the same cluster in both the true labels and the predicted clusters. Let b be the number of pairs of data points that are in different clusters in both the true labels and the predicted clusters. The ARI is defined as:

$$ARI = \frac{2(a+b)}{n(n-1)} \tag{4.19}$$

In this project, ARI is used to compare the clustering results with the true labels of spectral data.

4.4.6 Normalized Mutual Information (NMI)

Normalized Mutual Information (NMI) measures the mutual dependence between the true labels and the predicted clusters. It ranges from 0 to 1, with higher values indicating better clustering.

Let $H(Y)$ and $H(C)$ be the entropies of the true labels and the predicted clusters, respectively. Let $I(Y;C)$ be the mutual information between the true labels and the predicted clusters. NMI is defined as:

$$NMI = \frac{2I(Y;C)}{H(Y) + H(C)} \tag{4.20}$$

In this project, NMI provides a measure of how well the clustering algorithms capture the true structure of the spectral data.

4.4.7 Cluster Analysis

Cluster analysis is a fundamental technique in unsupervised machine learning used to group similar data points based on their features, allowing for the discovery of patterns without relying on predefined labels. In this project, cluster analysis was applied to categorize spectral data of astronomical objects, aiming to identify natural groupings that correspond to different types of stars and galaxies.

The process begins with feature extraction, where key characteristics of each spectrum are identified and used to represent the data in a reduced-dimensional space. Principal Component Analysis (PCA) is employed to simplify the data while retaining its most significant variance. This transformed data serves as input for clustering algorithms, such as K-Means or Agglomerative Clustering, which group spectra based on their similarities.

Once clustering is complete, the analysis involves examining the composition of each cluster by comparing the distribution of true labels within them. This helps determine the most common class in each cluster, providing insight into how well the clustering corresponds to known classifications. The analysis allows for:

- Assessing the homogeneity of clusters concerning the true classes.
- Identifying the predominant class within each cluster for interpretation.
- Evaluating the presence of mixed classes, indicating overlaps in spectral features.

In this project, cluster analysis is crucial for interpreting unsupervised learning results, offering a deeper understanding of the relationships between clustered data and the original spectral classes of astronomical objects.

4.4.8 Visual Evaluation

In addition to quantitative metrics, visual evaluation of clusters is crucial for understanding the clustering results. Plots such as the PCA scatter plot,K-Means and Agglormerative scatter plot and dendrogram help visualize the clustering structure and validate the results.

Evaluation metrics are essential for evaluating how well clustering algorithms perform. This project ensures a thorough assessment of the clustering outcomes by combining quantitative measures with visual evaluation. These metrics help determine the optimal

clustering strategy for spectral data by offering information about the quality, coherence, and separation of clusters.

Chapter 5

Results

5.1 PCA Results

Principal Component Analysis (PCA) is a powerful dimensionality reduction technique used to transform high-dimensional data into a lower-dimensional form while retaining most of the variance present in the original data. In this project, PCA was applied to the preprocessed spectral data to reduce its dimensionality and facilitate the clustering process.

5.1.1 3D PCA Plot

The 3D PCA plot (see Figure 5.1) represents the spectral data transformed into three principal components. Each point in the plot corresponds to a spectrum from the dataset, and the color coding represents different classes of astronomical objects, such as various types of galaxies, stars, and quasars. The legend on the plot provides a clear distinction between these classes.

5.1.2 Interpretation of PCA Results

The 3D PCA plot reveals several key insights into the structure of the spectral data:

- **Cluster Separation**: Different classes of astronomical objects form distinct clusters in the 3D PCA space. For instance, classes like GALAXY_UNKNOWN(Bright Orange) and QSO_AGN(Light Orange) are clustered in two different spaces. Stars in green and gray are clustered seperately as seen in figure 5.1. So there is clearly a distinct

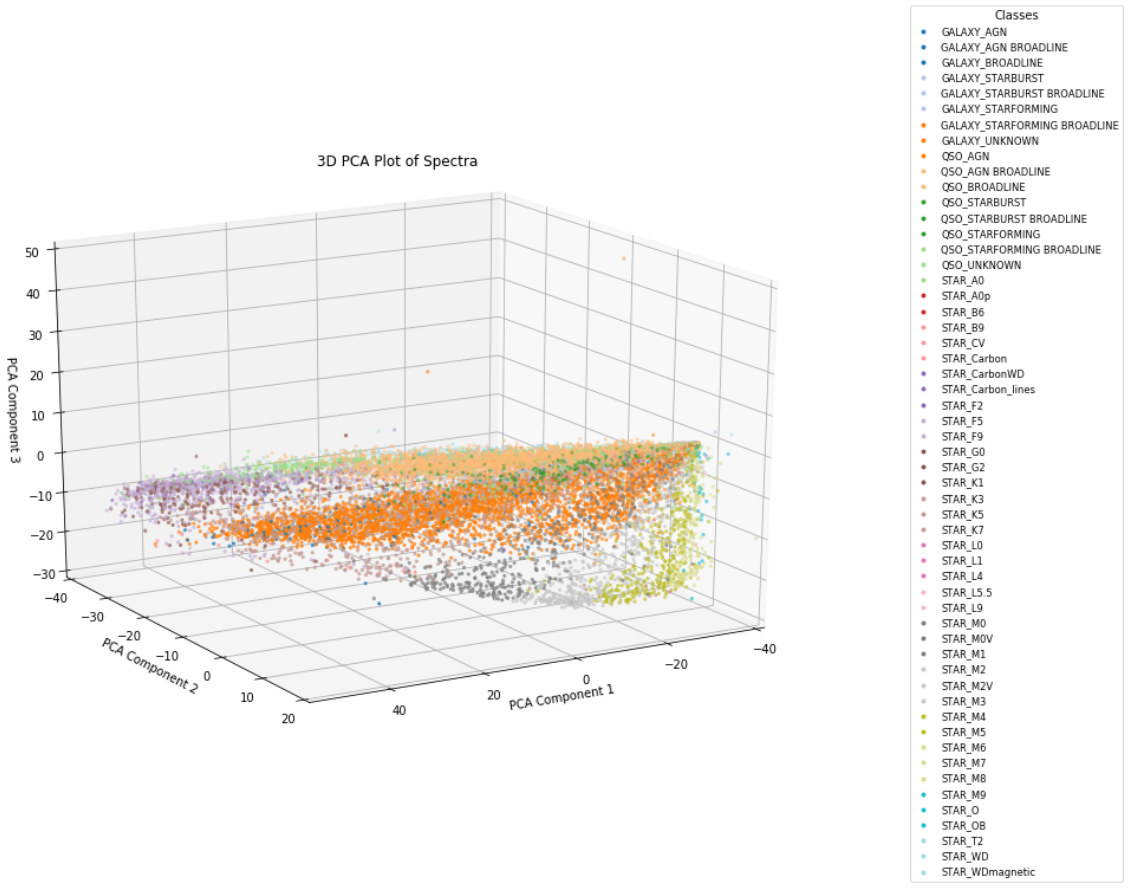

Figure 5.1: 3D PCA Plot of Spectral Data. Different colors represent different classes of astronomical objects.

seperation between these even with so many classes and subclasses of objects present in the dataset.

- **Overlapping Regions**: Some overlapping regions are observed, particularly among similar classes, such as different types of galaxies and stars. This overlap suggests that while PCA reduces dimensionality and highlights variance, some spectral similarities are inherent across these classes.

The PCA is plotted at an angle to see these distinction between these clusters more openly and to identify their respective labels.

5.1.3 Variance Explanation

The first three principal components explain a significant portion of the variance in the data. The exact percentages of variance explained by each component can be quantified by the corresponding eigenvalues. The explained variance for each Principal Component is calculated as follows:

$$\text{Explained Variance Ratio} = \frac{\lambda_i}{\sum_{j=1}^{p} \lambda_j} \tag{5.1}$$

where λ_i is the eigenvalue associated with the i-th principal component, and $\sum_{j=1}^{p} \lambda_j$ is the sum of all eigenvalues for the p components. The eigenvalues are derived from the covariance matrix of the original data and represent the variance captured by each principal component.

As illustrated in Figure 5.2,

- The first principal component explains the largest portion of the variance, as evident from the steep rise in the graph after the first component.
- The second component adds to the cumulative explained variance, capturing additional variability not accounted for by the first component.
- By the third component, the cumulative explained variance plateaus, indicating that these three components together capture over 95% of the total variance in the data.

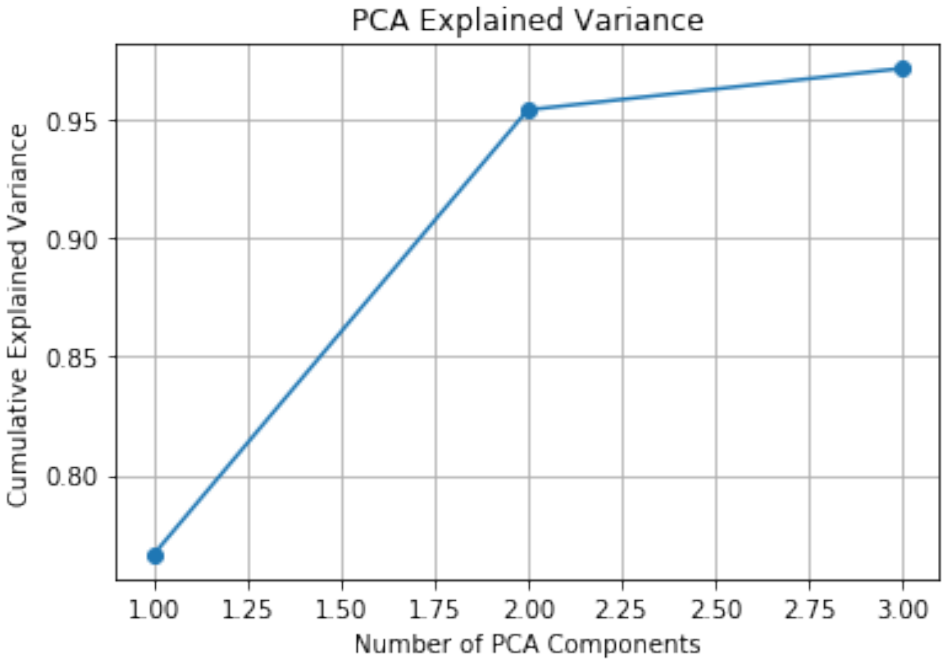

Figure 5.2: The cumulative explained variance by the first three PCA components.

5.1.4 PCA Implementation Overview

The PCA implementation in the project follows these key steps using specific variables in the code:

1. **Standardization**: The spectral data is standardized using the 'StandardScaler' from scikit-learn, which is applied to the preprocessed dataset stored in the variable 'scaled_fluxes'.

2. **PCA Transformation**: PCA is performed using the 'PCA' class from scikit-learn, with the number of components set to 3. The PCA object is fit to the standardized data stored in 'scaled_fluxes', and the transformation is applied to obtain the principal components, resulting in the variable 'pca_components'.

3. **Variance Explanation**: The explained variance ratio for each of the three principal components is obtained from the 'explained_variance_ratio_' attribute of the PCA object. This information is used to quantify the amount of variance captured by the principal components.

4. **Visualization**: The transformed data in 'pca_components' is plotted using a 3D scatter plot, with different colors representing different classes of astronomical objects. The variable 'encoded_labels' is used to color-code the points according to their respective classes.

The application of PCA in this project effectively reduced the dimensionality of the spectral data, facilitating better visualization and clustering. The 3D PCA plot provides a clear representation of the data's structure, highlighting the separation and overlap among different classes of astronomical objects. This transformation plays a crucial role in the subsequent clustering analysis, as it preserves the significant variance in the data while simplifying the complexity inherent in high-dimensional spectral features.

5.2 Clustering Results

5.2.1 K-Means Clustering

K-Means clustering is a widely used unsupervised machine learning technique for partitioning a dataset into k distinct, non-overlapping subsets (or clusters). In this project, the K-Means clustering algorithm was applied to the spectral data after it was reduced

to three principal components using PCA. The algorithm was configured to generate 22 clusters, chosen to match the diversity of spectral classes within the dataset. The results of the clustering are visualized and analyzed to understand the distribution and characteristics of the spectral classes.

Cluster Distribution

The distribution of cluster sizes obtained from the K-Means algorithm is shown in Figure 5.3. This histogram represents the number of spectra assigned to each cluster. The x-axis represents the cluster labels, while the y-axis shows the size of each cluster.

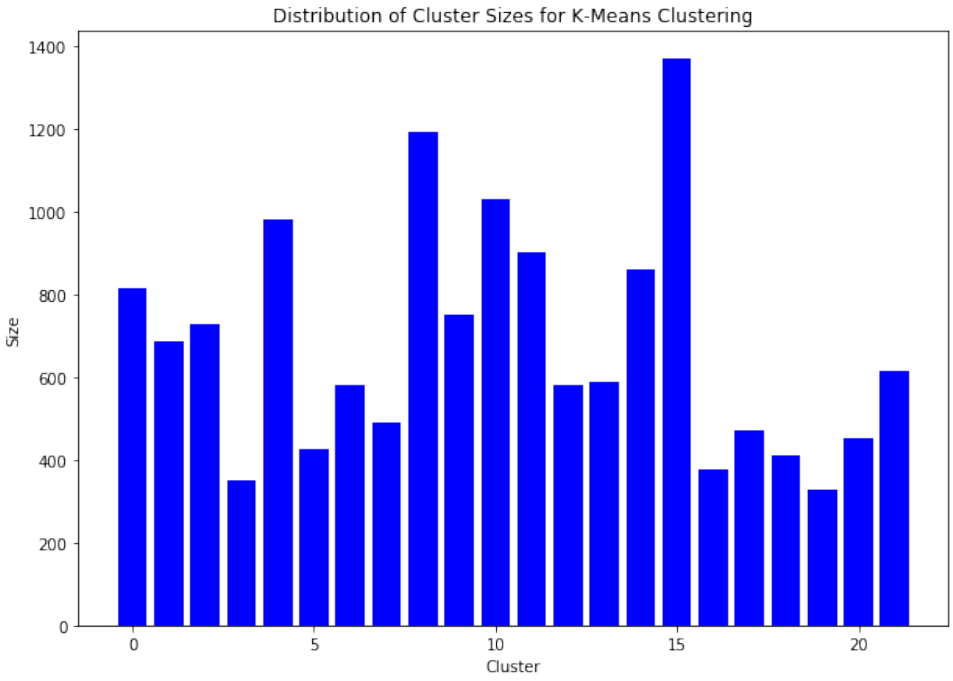

Figure 5.3: Distribution of Cluster Sizes for K-Means Clustering

The histogram reveals that the cluster sizes vary significantly, with some clusters containing a large number of spectra and others containing relatively few. This uneven distribution is expected in real-world datasets, where certain classes or types of objects are more prevalent than others. For instance, the most populated clusters likely correspond to common types of stars or galaxies, while smaller clusters may represent more rare or unusual objects.

3D PCA Plot with K-Means Clusters

To further analyze the clustering results, the spectra data were visualized in a 3D PCA plot with the K-Means cluster labels. This visualization helps to understand how well the PCA components capture the variance in the data and how effectively the K-Means algorithm groups similar spectra together. Figure 5.4 shows the 3D PCA plot with each point colored according to its assigned K-Means cluster.

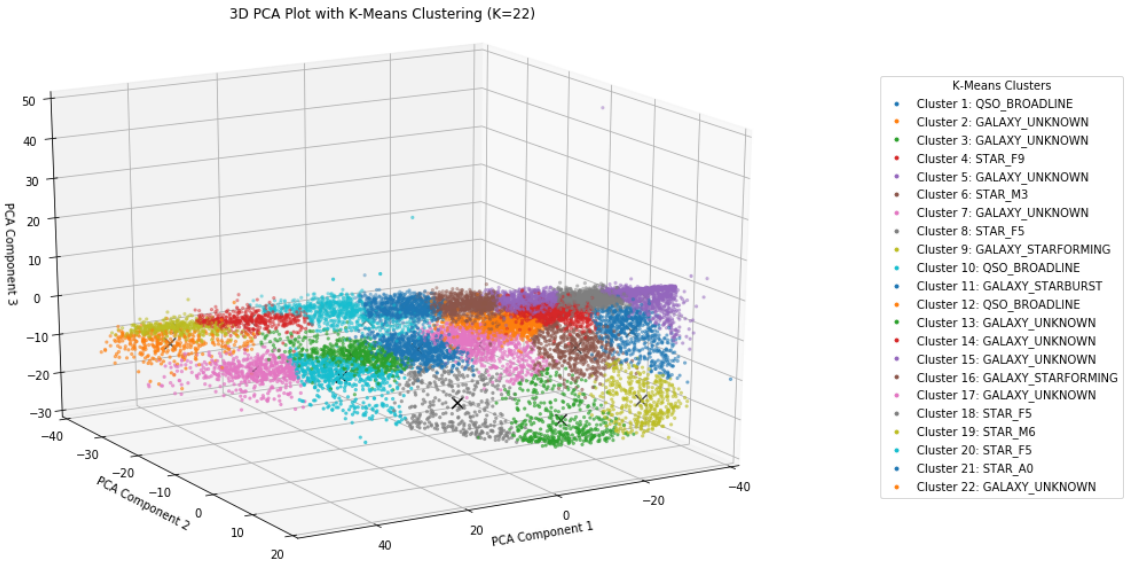

Figure 5.4: 3D PCA Plot with K-Means Clusters

2D Projections of PCA Components

In addition to the 3D visualization, 2D projections of the PCA components provide further insight into the cluster formations. The following figures illustrate how the clusters are distributed across different pairs of PCA components:

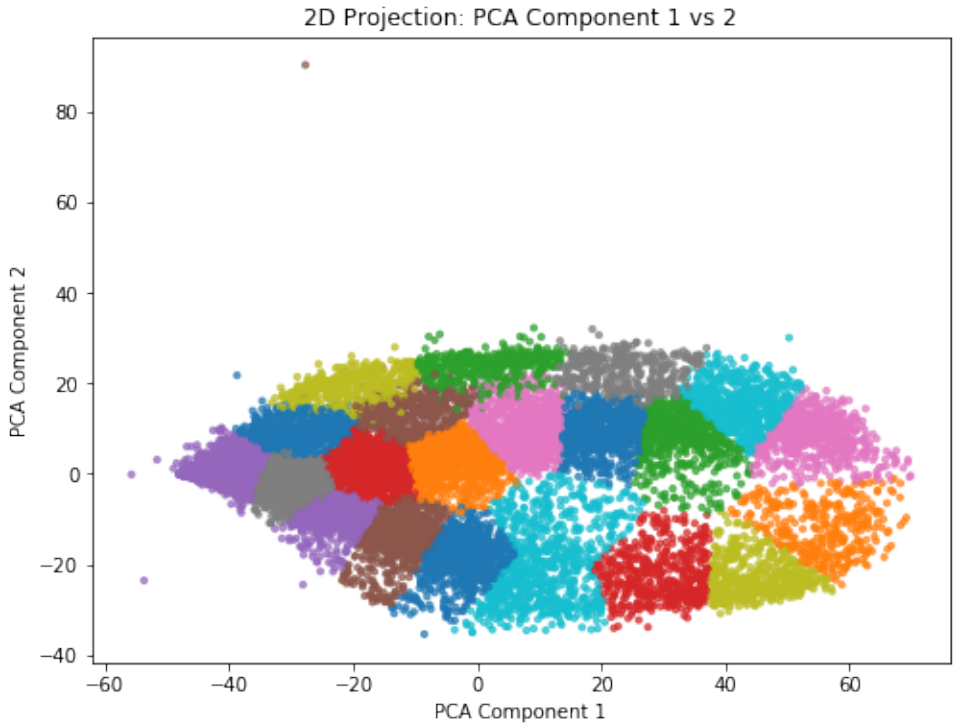

Figure 5.5: 2D Projection of PCA Component 1 vs 2

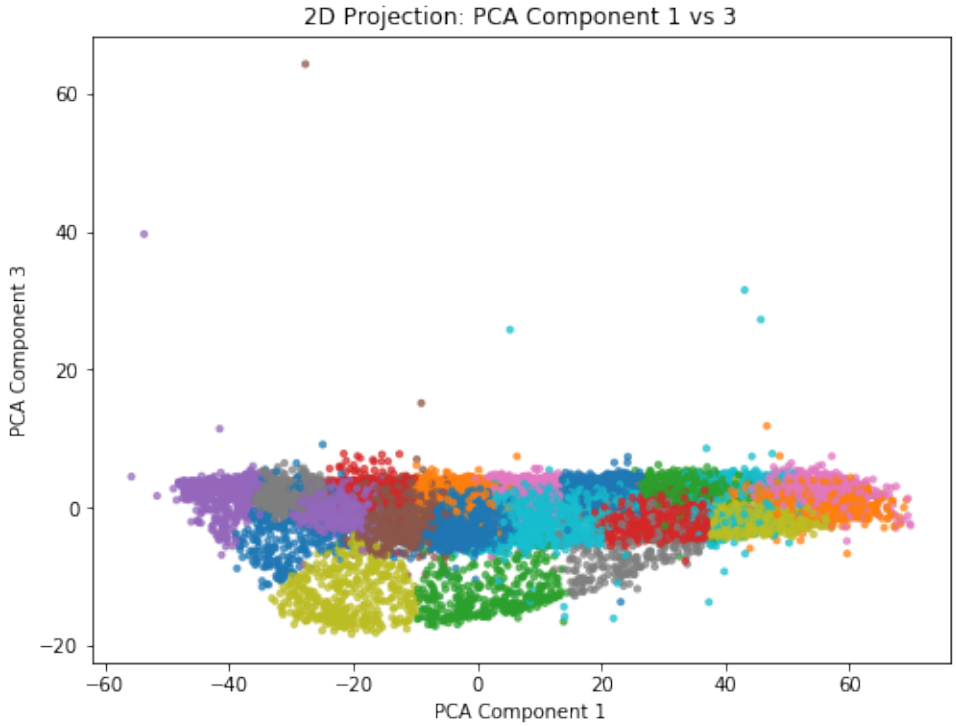

Figure 5.6: 2D Projection of PCA Component 1 vs 3

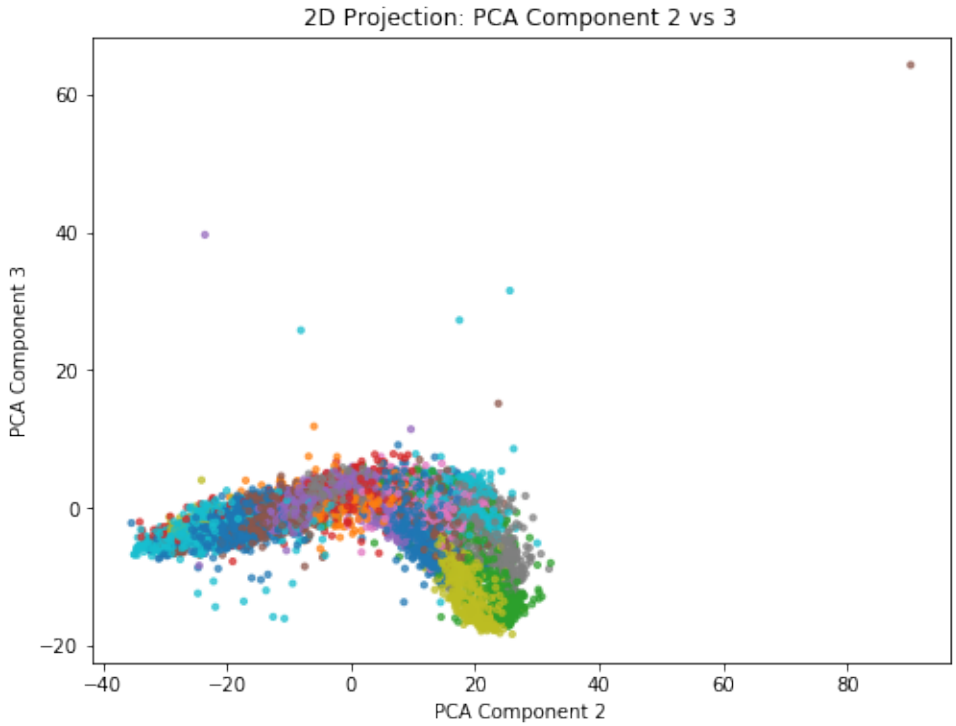

Figure 5.7: 2D Projection of PCA Component 2 vs 3

These projections illustrate how the clusters separate in two-dimensional planes, revealing additional details about their relationships and overlaps in different principal component spaces.

Cluster Interpretation

Each cluster in the K-Means algorithm can be interpreted by examining the common characteristics of the spectra within it. For example, clusters containing spectra with similar absorption or emission features likely correspond to specific types of stars or galaxies. In the provided 3D PCA plot (Figure 5.4), the clusters are color-coded according to their labels. Here's a brief interpretation of some of the clusters:

- **Cluster 2, Cluster 3, Cluster 5, Cluster 7, Cluster 9, Cluster 13, Cluster 14, Cluster 15, Cluster 17, Cluster 22**: The most frequent label occurrences in these clusters is GALAXY_UNKNOWN, indicating that the spectra in these clusters are thought to belong to the unlabelled galaxy class.

- **Cluster 4, Cluster 6, Cluster 8, Cluster 18, Cluster 19, Cluster 20, Cluster 21**: These clusters are more concentrated with different types of star subclasses, with STAR_F5 dominating in most of these clusters

- **Cluster 1, Cluster 10, Cluster 12**: These clusters might represent objects with more unique or rare spectral characteristics. The distinct object in these clusters is QSO's especially the QSO_BROADLINE.

The various groupings may be clearly distinguished from one another due to the color coding, and additional statistical analysis can be done to determine the specific characteristics of each cluster. For instance, one could calculate the average spectrum for each cluster and compare these averages to known spectral types.

Insights from Clustering

The clustering results provide several insights:

- **Data Structure**: The clustering reveals the inherent structure of the spectral data, showing which spectra are similar and form natural groups.
- **Dimensionality Reduction Effectiveness**: The clear separation of clusters in the 3D PCA plot confirms that the PCA effectively reduced the dataset's dimensionality while retaining the important features necessary for distinguishing between different types of objects.
- **Spectral Classification**: By examining the spectra within each cluster, one can identify and classify various types of astronomical objects. This is particularly useful in large datasets where manual classification is impractical.

In general, K-Means clustering in conjunction with PCA provides a strong tool for assessing and analysing large spectrum datasets. Astronomers can discover patterns and groupings that may not be immediately apparent thanks to its assistance in the classification of a wide variety of celestial objects. The section on Cluster Analysis does a more thorough study.

Further work may involve exploring new, previously unclassified types of objects or refining the number of clusters (k) to better match the actual types of objects present. Additionally, integrating more sophisticated clustering algorithms or combining different clustering techniques could yield even more accurate and insightful results.

In conclusion, the K-Means clustering results, as visualized through the PCA components, demonstrate the effectiveness of unsupervised learning techniques in handling

and analyzing complex astronomical data, paving the way for automated classification and discovery in large-scale astronomical surveys.

5.2.2 Clustering Results: Agglomerative Clustering

Agglomerative clustering is a hierarchical clustering method that builds nested clusters by successively merging or splitting them. This technique is well-suited for the analysis of large datasets, such as the 15,000 spectra from the Sloan Digital Sky Survey (SDSS) DR18 used in this study. The primary goal of agglomerative clustering in this context is to classify spectra based on their characteristics, aiding in the identification of various astronomical objects.

Dendrogram Analysis

The dendrogram in Figure 5.8 provides a visual representation of the hierarchical clustering process with $p = 6$. Each merge is represented by a horizontal line, with the y-axis showing the distance or dissimilarity between clusters. The height of each merge indicates the distance between the clusters being merged.

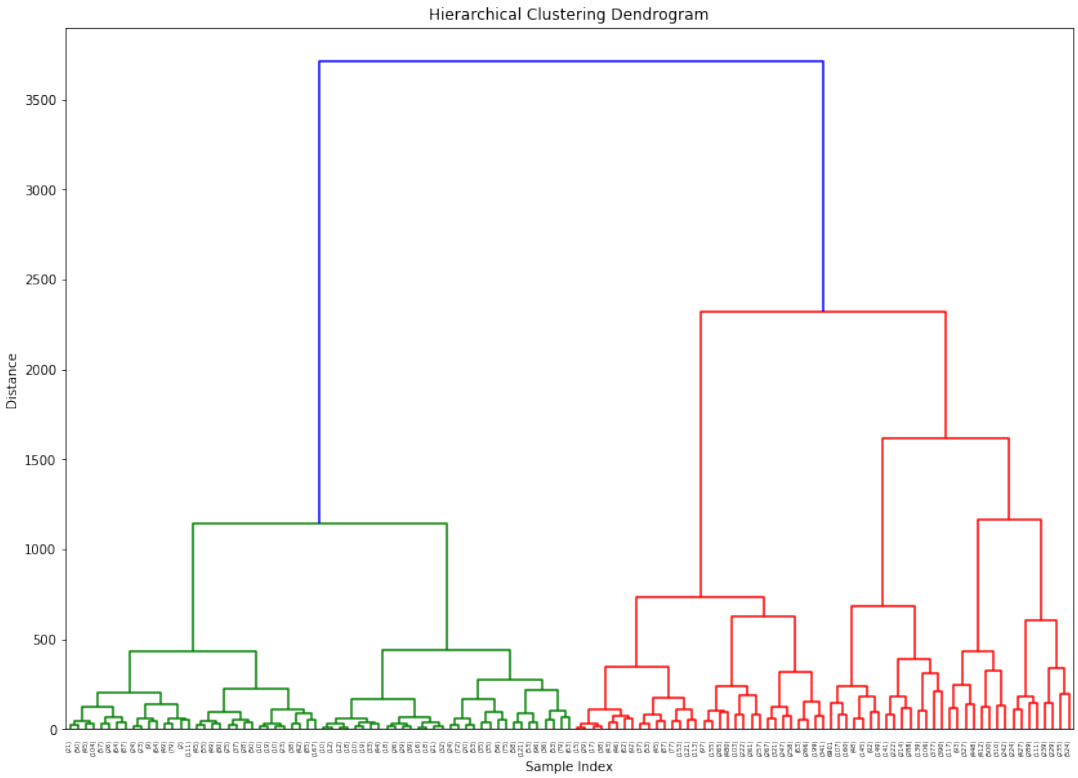

Figure 5.8: Dendrogram for Agglomerative Clustering with $p = 6$

From the dendrogram, we can observe how clusters are formed at various levels of dissimilarity. The large gaps between some merges indicate significant differences between clusters, suggesting distinct spectral classes. The detailed hierarchical structure aids in understanding the relationships and relative dissimilarities among the spectra. A cutoff distance of 240 was chosen to yield 22 clusters, providing a balance between granularity and interpretability.

3D PCA Plot with Dendrogram-Based Clusters

Figure 5.9 presents a 3D Principal Component Analysis (PCA) plot colored by the clusters obtained from the dendrogram with a cutoff distance of 240. This visualization helps to understand the clustering structure in the context of reduced dimensionality. The PCA reduces the complex spectral data into three principal components, which can then be plotted in a 3D space.

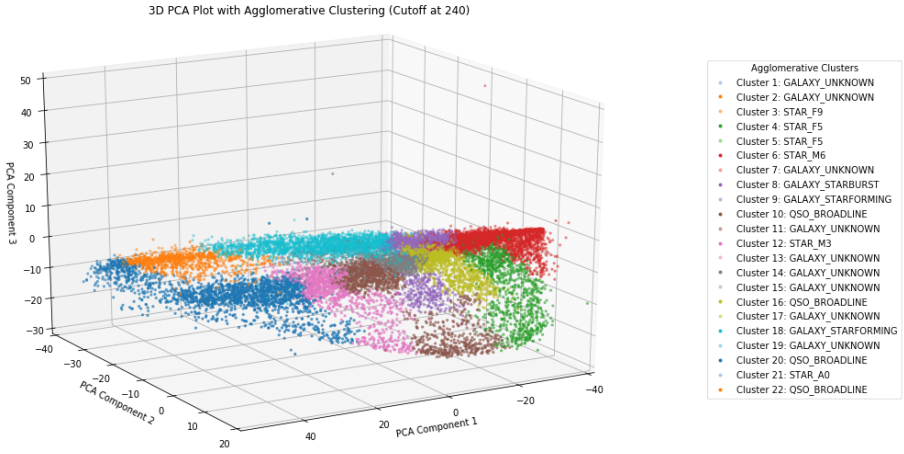

Figure 5.9: 3D PCA Plot with Clusters Based on Dendrogram Cutoff Distance = 240

In the PCA plot, each point represents a spectrum, and the color indicates the assigned cluster. The labels on the legend denote the different types of astronomical objects identified by the clustering algorithm.

Cluster Size Distribution

The first image (Figure 5.10) illustrates the distribution of cluster sizes obtained from agglomerative clustering. The histogram displays the number of objects within each cluster, providing an overview of the cluster sizes and their distribution.

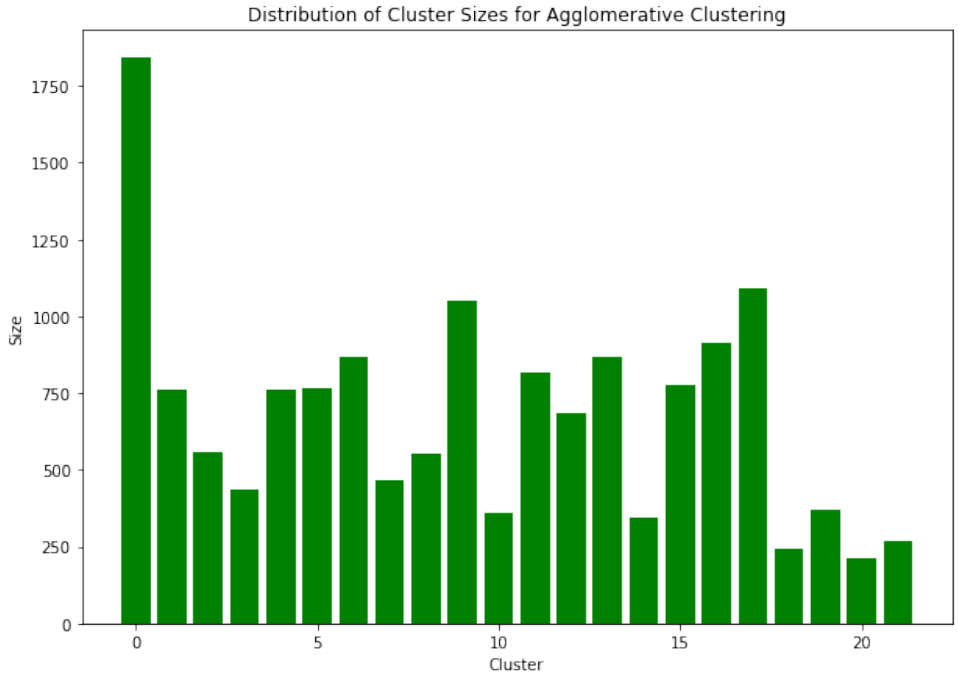

Figure 5.10: Distribution of Cluster Sizes for Agglomerative Clustering

From the histogram, it is evident that the sizes of the clusters vary significantly. Some clusters contain over a thousand spectra, while others are much smaller, consisting of only a few hundred spectra. The presence of both large and small clusters suggests that agglomerative clustering effectively captures the diversity within the dataset, identifying both dominant and less prevalent spectral classes.

3D PCA Plot with Agglomerative Clusters

The second image (Figure 5.11) presents a 3D Principal Component Analysis (PCA) plot with agglomerative clustering results based on the optimal number of clusters ($k = 22$). This visualization helps to understand the clustering structure in the context of reduced dimensionality. The PCA reduces the complex spectral data into three principal components, which can then be plotted in a 3D space.

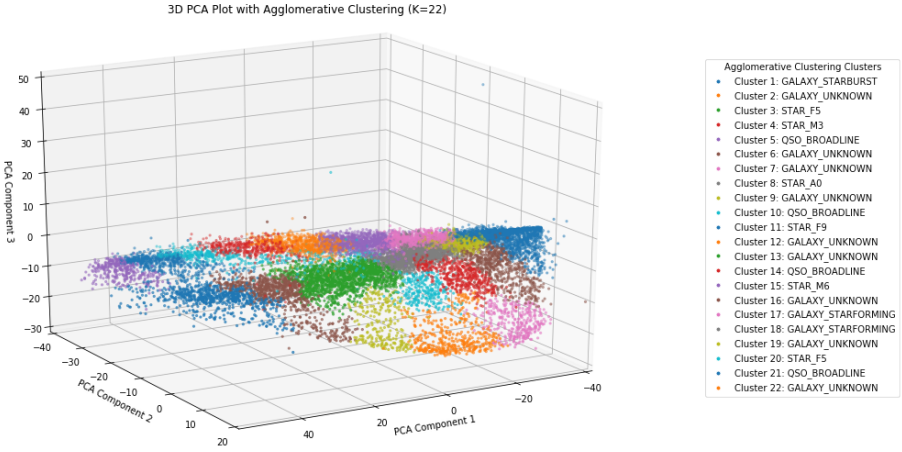

Figure 5.11: 3D PCA Plot with Agglomerative Clustering Clusters

2D PCA Plots

To provide a more detailed view of the clustering results, 2D PCA plots are shown for the different pairs of principal components:

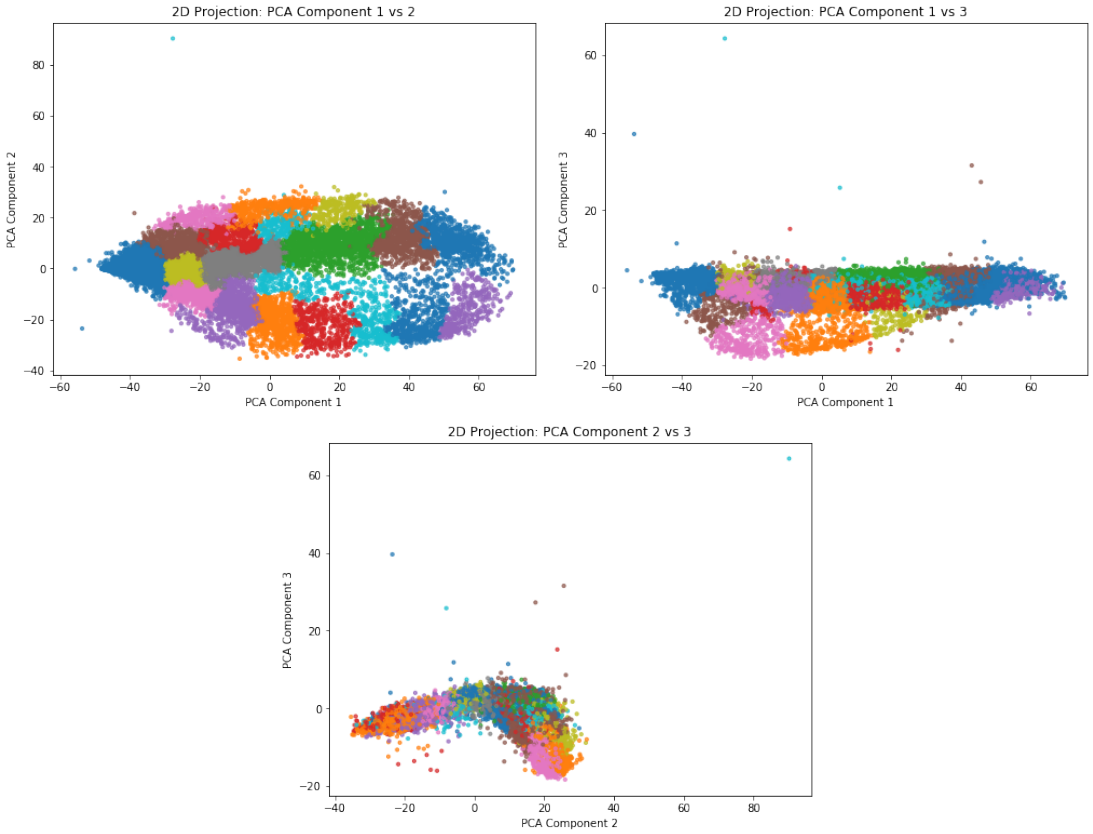

Figure 5.12: 2D PCA Plots: (a) Principal Component 1 vs 2, (b) Principal Component 1 vs 3, (c) Principal Component 2 vs 3

In these PCA plots, each point represents a spectrum, and the color indicates the assigned cluster. This further helps to understand the separation and distribution of clusters in the reduced dimensional space.

Interpretation of the PCA Plot

The PCA plot reveals several key insights about the clustering results:

1. **Cluster Separation:** The clusters are separated in the 3D PCA space, indicating that the agglomerative clustering algorithm effectively distinguishes between different spectral classes. For instance, the GALAXY_UNKNOWN (orange) and STAR_F5 (green) STAR_M3 (red) are in seperate clusters indicating that the clusters are seperated

2. **Homogeneity within Clusters:** Each cluster's points are densely packed together, indicating that the clusters are homogeneous. This suggests that spectra in the same cluster share similar properties, which is essential for precise classification. Like the clusters which have GALAXY_UNKNOWN as the most common label, the algorithm has its difficulty differentiating between different types of galaxies, same with the stars and qso's

3. **Overlap Between Clusters:** Some clusters exhibit overlap, indicating that certain spectral classes have similarities. For example, the GALAXY_UNKNOWN and GALAXY_STARFORMING, STAR_F9 and STAR_F5 clusters show some degree of overlap, reflecting their similar spectral characteristics.

Significance of Agglomerative Clustering Results

The results of agglomerative clustering offer significant fresh insight into how astronomical objects are classified spectrally. Through the process of clustering similar spectra, the technique aids in the identification and classification of various galaxy, quasar, and star types. This classification can assist astronomers in understanding the composition and evolution of the universe.

- **Astronomical Object Identification:** The clustering results facilitate the identification of various astronomical objects, such as starburst galaxies, star-forming

galaxies, and different types of stars. This information is crucial for studying the lifecycle of stars and the formation of galaxies.

- **Anomaly Detection:** The presence of outliers and small clusters highlights potential anomalies or rare objects in the dataset. These anomalies can lead to new discoveries and a deeper understanding of the universe's diversity.

- **Data Reduction and Analysis:** By reducing the dimensionality of the dataset through PCA and clustering, the study makes it easier to analyze and interpret the vast amount of spectral data. This approach streamlines the process of identifying significant patterns and trends in the dataset.

The results of agglomerative clustering show how well hierarchical clustering works for spectral categorization of celestial objects. A thorough overview of the clustering structure is given by the cluster size distribution, dendrogram, and 3D PCA plot, which emphasize both common and uncommon spectral classes. This study opens the door for more astronomical research and discovery in addition to helping with the identification and classification of celestial objects.

5.3 Evaluation of Clustering Performance

Evaluating clustering algorithms is vital for understanding the quality and reliability of the clusters formed. This section assesses the clustering results of both K-Means and Agglomerative Clustering using metrics like Silhouette Score, Davies-Bouldin Index, Calinski-Harabasz Index, Adjusted Rand Index (ARI), Normalized Mutual Information (NMI), and Cluster Purity. These metrics provide a comprehensive understanding of clustering effectiveness and how well the clusters reflect the inherent structure of the data. Additionally, the Elbow Method, Silhouette Score, Calinski-Harabasz Index, and Davies-Bouldin Index plots were used to determine the optimal number of clusters for K-Means, selecting 22 clusters as optimal (Figure 5.13).

The use of unsupervised machine learning in this project was to evaluate which algorithm delivers better results when compared to others. However, these methods are not yet ready for real-world application without further refinement and validation. Significant

work is needed to make these clustering techniques reliable and practical for real-world use.

5.3.1 K-Means Clustering Evaluation

K-Means clustering performance was evaluated using 22 clusters, identified as optimal. The metrics are:

- **Silhouette Score:** 0.3351
- **Davies-Bouldin Index:** 0.8743
- **Calinski-Harabasz Index:** 16686.4289
- **Adjusted Rand Index (ARI):** 0.1286
- **Normalized Mutual Information (NMI):** 0.3678
- **Cluster Purity:** 0.5248

Key Metrics

The Silhouette Score of 0.3351 suggests moderate cluster separation and compactness, while the Davies-Bouldin Index of 0.8743 indicates reasonably well-separated clusters. A high Calinski-Harabasz Index of 16686.4289 confirms well-defined clusters. The ARI of 0.1286 shows low to moderate agreement with true class labels, and an NMI of 0.3678 suggests significant overlap with actual classes. A Cluster Purity of 0.5248 indicates that over half of the objects in each cluster belong to the dominant class, showing moderate homogeneity.

Optimal Number of Clusters

The optimal number of clusters was determined using various methods:

- **Elbow Method:** Indicates a point of inflection at 22 clusters.
- **Silhouette Score:** Peaks at 22 clusters, suggesting good cluster separation.
- **Calinski-Harabasz Index:** Significant increase at 22 clusters, indicating well-defined clusters.

- **Davies-Bouldin Index:** Minimum value at 22 clusters, indicating better cluster separation.

These methods are illustrated in Figure 5.13.

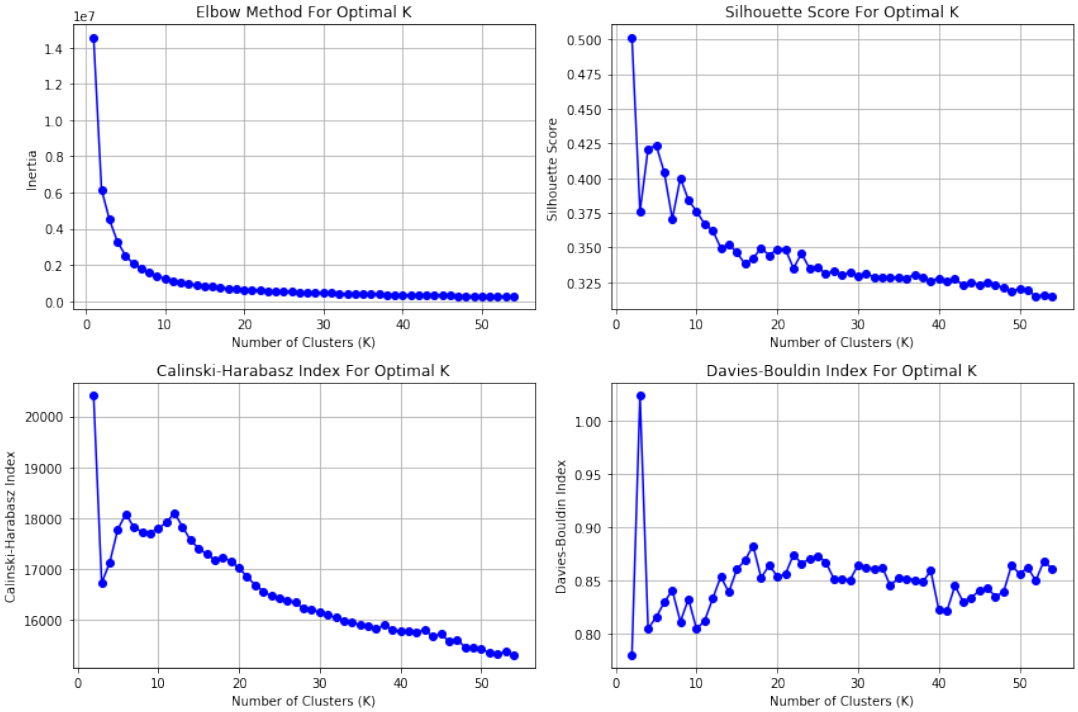

Figure 5.13: Evaluation metrics for determining the optimal number of clusters for K-Means: (a) Elbow Method, (b) Silhouette Score, (c) Calinski-Harabasz Index, (d) Davies-Bouldin Index.

5.3.2 Agglomerative Clustering Evaluation

Agglomerative Clustering was also evaluated using 22 clusters, with the following metrics:

- **Silhouette Score:** 0.3139
- **Davies-Bouldin Index:** 0.8804
- **Calinski-Harabasz Index:** 14703.6313
- **Adjusted Rand Index (ARI):** 0.1616
- **Normalized Mutual Information (NMI):** 0.3311
- **Cluster Purity:** 0.4529

Key Metrics

The Silhouette Score of 0.3139 suggests moderate compactness and separation, slightly lower than K-Means. The Davies-Bouldin Index of 0.8804 reflects reasonable separation, comparable to K-Means. A Calinski-Harabasz Index of 14703.6313 indicates well-defined but less distinct clusters compared to K-Means. The ARI of 0.1616 shows a moderate agreement with true labels, while the NMI of 0.3311 suggests significant but imperfect overlap with true classes. The Cluster Purity of 0.4529 reflects moderate homogeneity, lower than K-Means.

Optimal Number of Clusters

For Agglomerative Clustering, the optimal number of clusters was similarly determined:

- **Silhouette Score:** Peaks at 22 clusters, indicating good separation.
- **Calinski-Harabasz Index:** Significant increase at 22 clusters, indicating well-defined clusters.
- **Davies-Bouldin Index:** Minimum value at 22 clusters, indicating better separation.

These methods are visualized in Figure 5.14.

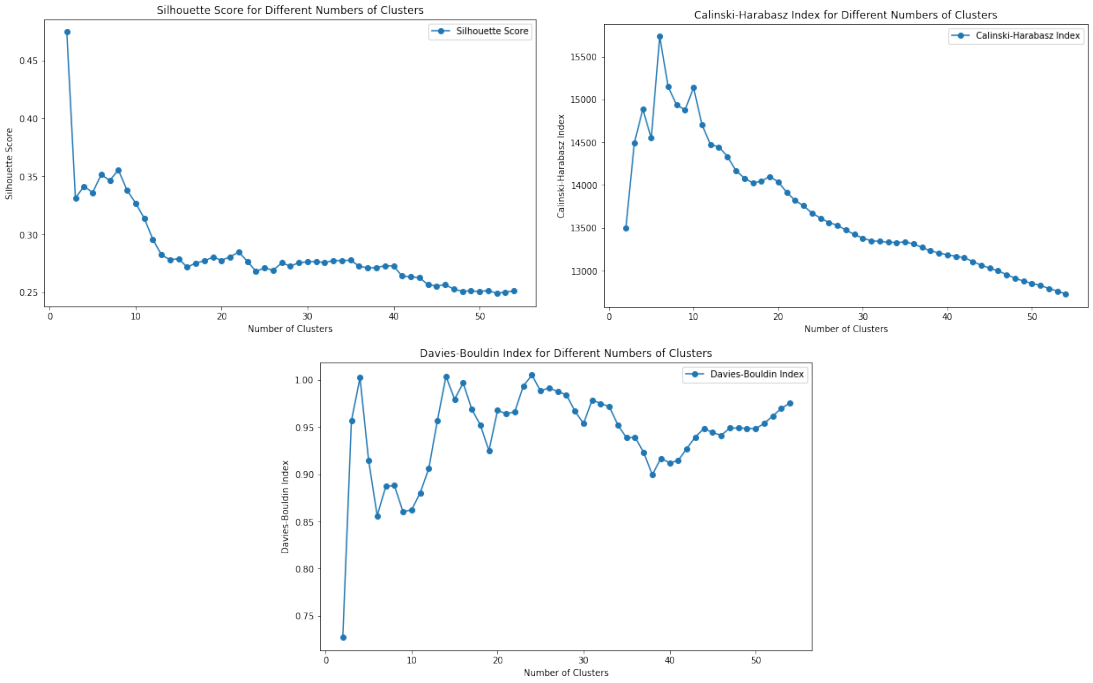

Figure 5.14: Evaluation metrics for determining the optimal number of clusters for Agglomerative Clustering: (a) Silhouette Score, (b) Calinski-Harabasz Index, (c) Davies-Bouldin Index.

5.3.3 Comparative Analysis

Comparing both methods, K-Means generally performs better across most metrics, forming more compact and well-separated clusters that better align with the actual class structure. However, Agglomerative Clustering remains a valuable method, particularly for cases requiring hierarchical structures and nested clusters. Despite K-Means slightly outperforming Agglomerative Clustering in metrics like Silhouette Score and Cluster Purity, both methods provide reasonable clustering results.

The clustering results highlight the effectiveness of unsupervised learning in classifying astronomical spectra. These evaluations not only validate the clustering quality but also provide insights into the data's structure, aiding in the identification and analysis of various astronomical objects.

5.4 Cluster Analysis

5.4.1 K-Means Clustering Analysis

K-means clustering was employed to uncover distinct groupings within the dataset, based on the spectral classifications of various celestial objects. This analysis led to the identification of clusters with varying compositions, dominated by different types of stars, galaxies, and quasars. The key findings for each cluster are summarized below:

- **Cluster 0** predominantly consists of *QSO_BROADLINE* objects, making up about 67% of the cluster. This indicates a significant presence of broadline quasars, alongside other star types such as *STAR_A0* and *STAR_WD*.

- **Cluster 1** is mainly characterized by *GALAXY_UNKNOWN* objects, comprising nearly 49% of the cluster. Additionally, this cluster includes a variety of M-type stars, suggesting a mix of unidentified galaxies and late-type stars.

- **Cluster 2** is overwhelmingly dominated by *GALAXY_UNKNOWN* objects, which represent approximately 75% of the cluster. The presence of star-forming galaxies and K-type stars further characterizes this cluster.

- **Cluster 3** is primarily composed of *STAR_F9* objects, accounting for almost 50% of the cluster. This indicates a significant grouping of F9-type stars, with related star types like *STAR_K1* and *STAR_F5* also present.

- **Cluster 4** also has a high proportion of *GALAXY_UNKNOWN* objects (around 41%) but includes a notable fraction of *GALAXY_STARFORMING* galaxies, indicating a mixture of unknown and star-forming galaxies.

- **Cluster 5** is rich in M-type stars, with *STAR_M3* being the most common class, constituting 34% of the cluster. This cluster effectively groups various late-type stars.

- **Cluster 6** also shows a dominance of *GALAXY_UNKNOWN* objects, comprising over 70% of the cluster. A smaller proportion of star-forming galaxies and K-type stars are also present.

- **Cluster 7** is characterized by a majority of *STAR_F5* objects, representing nearly 48% of the cluster. This suggests a focus on F5-type stars, with other F-type and A-type stars also included.

- **Cluster 8** has a balanced distribution, with *GALAXY_STARFORMING* being the most common class at 50%. This cluster also includes unknown galaxies and a mix of quasar types, indicating a focus on active star-forming regions.

- **Cluster 9** is primarily composed of *QSO_BROADLINE* objects, making up around 66% of the cluster. This composition is similar to Cluster 0 but with a slightly different distribution of other classes.

- **Cluster 10** is notable for its *GALAXY_STARBURST* objects, which constitute about 24% of the cluster. This cluster has a diverse composition, including other galaxy types and various M-type stars.

- **Cluster 11** has a significant proportion of *QSO_BROADLINE* objects (around 71%), indicating a predominant grouping of broadline quasars, along with some white dwarfs and other star types.

- **Cluster 12** is another cluster dominated by *GALAXY_UNKNOWN* objects (approximately 73%), with some M-type stars, indicating a focus on unidentified galaxies and late-type stars.

- **Cluster 13** also shows a high percentage of *GALAXY_UNKNOWN* objects (about 63%) and includes K-type stars and broadline galaxies, suggesting a diverse mix of galaxy types.

- **Cluster 14** is heavily dominated by *GALAXY_UNKNOWN* objects (74%), with a notable presence of star-forming galaxies and K-type stars, indicating a focus on galaxy classifications.

- **Cluster 15** features a majority of *GALAXY_STARFORMING* objects (around 36%), and includes a mix of broadline quasars and starburst galaxies, suggesting active galactic environments.

- **Cluster 16** shows a dominance of *GALAXY_UNKNOWN* objects (approximately 33%), but also has a significant proportion of M-type stars, indicating a mix of galaxies and late-type stars.
- **Cluster 17** is characterized by *STAR_F5* objects, which make up 46% of the cluster. This cluster groups F-type stars, with a smaller presence of G-type stars and other stellar types.
- **Cluster 18** has a mixture of different M-type stars like *STAR_M6*, *STAR_M5*, *STAR_M4*, and *STAR_M3*, making up 80% of the cluster.
- **Cluster 19** has a majority of *STAR_F5* objects (approximately 23%), along with several other types of stars and some quasars.
- **Cluster 20** shows around 50% of objects are *STAR_A0*, with the rest of the cluster composed of other star types and quasars.
- **Cluster 21** is once again dominated by *GALAXY_UNKNOWN* objects, which make up 72% of the cluster.

K-means clustering approach has effectively identified meaningful groupings within the dataset, highlighting the dominant spectral classes within each cluster. This analysis provides valuable insights into the underlying structure and distribution of spectral types in the astronomical data, revealing how certain clusters are focused on specific types of stars or galaxies, while others exhibit a more diverse composition.

5.4.2 Agglomerative Clustering

Agglomerative Clustering is a type of hierarchical clustering that builds nested clusters by successively merging or splitting them based on distance metrics. Here's how to interpret the results for each cluster from the given solution:

Cluster 0:

Most Common Class: GALAXY_STARBURST.

Class Distribution:

The cluster predominantly contains objects classified as GALAXY_STARBURST (23.59%),

followed by GALAXY_UNKNOWN (20%), and QSO_BROADLINE (18.04%).
This cluster shows a significant diversity, indicating it may include a variety of objects with some degree of similarity.

Cluster 1:

Most Common Class: GALAXY_UNKNOWN.
Class Distribution:
The cluster is heavily dominated by GALAXY_UNKNOWN (62.65%).
Other significant classes include STAR_K5, STAR_K3, and STAR_K7, each representing different types of stars.
This cluster may consist primarily of unclassified or unknown galaxies with some presence of specific star types.

Cluster 2:

Most Common Class: STAR_F5.
Class Distribution:
The majority class STAR_F5 (37.92%) is followed by STAR_F9 (22%) and STAR_G2 (9.66%).
This cluster is likely representing stars within a similar spectral range, focusing mainly on the F-type stars.

Cluster 3:

Most Common Class: STAR_M3.
Class Distribution:
Dominated by M-type stars, with STAR_M3 (35.02%) and STAR_M2 (27.42%) being the most prevalent.
This cluster is clearly representing late-type M stars.

Cluster 4:

Most Common Class: QSO_BROADLINE.
Class Distribution:

A strong presence of QSO_BROADLINE (56.52%) indicates this cluster is largely composed of quasars with broad emission lines.
It also contains some stellar types like STAR_A0 and STAR_F5.

Cluster 5:

Most Common Class: GALAXY_UNKNOWN.
Class Distribution:
Predominantly GALAXY_UNKNOWN (73.79%), with some contribution from GALAXY_STARFORMI
(9.52%) and various star types.
This cluster might group unclassified galaxies with a minor presence of star-forming galaxies.

Cluster 6:

Most Common Class: GALAXY_UNKNOWN.
Class Distribution:
Very similar to Cluster 5, this cluster is dominated by GALAXY_UNKNOWN (74.68%) and GALAXY_STARFORMING (13.06%).
It could represent another grouping of galaxies that are mostly unclassified but include some active star-forming galaxies.

Cluster 7:

Most Common Class: STAR_A0.
Class Distribution:
This cluster is characterized by STAR_A0 (34.40%) and has a significant presence of quasars like QSO_BROADLINE (22.86%).
It suggests a mix of A-type stars and quasars.

Cluster 8:

Most Common Class: GALAXY_UNKNOWN.
Class Distribution:
Primarily GALAXY_UNKNOWN (71.56%) with a range of M-type stars.
This cluster may group galaxies and some late-type stars.

Cluster 9:

Most Common Class: QSO_BROADLINE.

Class Distribution:

This cluster is dominated by quasars (QSO_BROADLINE 66.51%), with a variety of stars including STAR_WD.

Likely represents a strong grouping of quasars.

Cluster 10:

Most Common Class: STAR_F9.

Class Distribution:

Focused on F-type stars (STAR_F9 39.78% and STAR_F5 19.88%).

This cluster likely groups F-type stars with some G-type stars.

Cluster 11:

Most Common Class: GALAXY_UNKNOWN.

Class Distribution:

Another cluster dominated by GALAXY_UNKNOWN (67.73%).

Contains a variety of star types, suggesting a mixed group of galaxies and stars.

Cluster 12:

Most Common Class: GALAXY_UNKNOWN.

Class Distribution:

GALAXY_UNKNOWN (49.42%) with a strong presence of M-type stars like STAR_M3.

This cluster may represent a mix of unclassified galaxies and M-type stars.

Cluster 13:

Most Common Class: QSO_BROADLINE.

Class Distribution:

Strong presence of quasars (QSO_BROADLINE 73.65%).

This cluster likely represents a large group of quasars.

Cluster 14:

Most Common Class: STAR_M6.

Class Distribution:

This cluster is focused on late M-type stars, particularly STAR_M6 (27.19%).

Represents a specific grouping of late-type stars.

Cluster 15:

Most Common Class: GALAXY_UNKNOWN.

Class Distribution:

Mixed cluster with GALAXY_UNKNOWN (47.10%) and GALAXY_STARFORMING (34.19%).

Suggests a mix of unclassified and star-forming galaxies.

Cluster 16:

Most Common Class: GALAXY_STARFORMING.

Class Distribution:

Primarily GALAXY_STARFORMING (46.93%) with a substantial presence of GALAXY_UNKNOWN (27.19%).

Likely groups star-forming galaxies with some unclassified ones.

Cluster 17:

Most Common Class: GALAXY_STARFORMING.

Class Distribution:

Strongly GALAXY_STARFORMING (56.72%) with some GALAXY_UNKNOWN (12.53%).

Represents a concentrated cluster of star-forming galaxies.

Cluster 18:

Most Common Class: GALAXY_UNKNOWN.

Class Distribution:

Mostly GALAXY_UNKNOWN (28.57%) with a mix of M-type stars.

This cluster might be grouping galaxies with specific M-type stars.

Cluster 19:

Most Common Class: STAR_F5.

Class Distribution:

This cluster is focused on STAR_F5 (41.35%) with some presence of A-type and G-type stars.

Represents a group of F-type stars with some other stellar types.

Cluster 20:

Most Common Class: QSO_BROADLINE.

Class Distribution:

This cluster is focused on QSO_BROADLINE (49.75%) with some presence of STAR_F9 spectra (15.1%).

This cluster likely represents a large group of quasars.

Cluster 21:

Most Common Class: GALAXY_UNKNOWN.

Class Distribution:

This cluster mostly consists of GALAXY_UNKNOWN class (73.5%).

Represents another cluster with massive unknown galaxies clustering.

Each cluster from the Agglomerative Clustering analysis, groups similar types of astronomical objects together, based on their spectral characteristics and other features. The clusters have distinct dominant classes, with some clusters focusing on galaxies (either known or unknown types), while others emphasize specific types of stars or quasars. Like K-Means, Agglomerative Clustering provides insights on the distribution of spectral types and their clustering tendencies seen above.

Chapter 6

Conclusion and Future Work

6.1 Summary of Findings

This thesis has explored the application of Principal Component Analysis (PCA) and clustering techniques—specifically K-Means and Agglomerative Clustering—on astronomical spectral data. The objective was to classify different kinds of stars and galaxies according to their spectral characteristics, using clustering and dimensionality reduction techniques to improve comprehension and classification effectiveness.

Principal Component Analysis (PCA)

PCA was employed as the initial step to reduce the dimensionality of the high-dimensional spectral data. The resulting 3D PCA plot provided a visual representation of the data, enabling a preliminary understanding of the distribution and separation of different spectral classes. The PCA components effectively captured the variance in the data, facilitating the subsequent clustering process. The visualization illustrated distinct groupings for different galaxy and star types, indicating that PCA was successful in preserving the intrinsic structure of the data despite the reduction in dimensionality.

K-Means Clustering Results

The application of K-Means clustering on the PCA-transformed data revealed several key insights:

- The 3D PCA plot with K-Means clusters demonstrated that the algorithm could

effectively group similar spectral data points together. However, some overlap between clusters was observed, particularly among galaxy classes.

- The distribution of cluster sizes indicated a varied clustering pattern, with some clusters having significantly more data points than others. This imbalance suggests that certain spectral types are more prevalent in the dataset.

- Evaluation metrics for K-Means clustering showed a Silhouette Score of 0.3351, a Davies-Bouldin Index of 0.8743, and a Calinski-Harabasz Index of 16686.429. These metrics reflect a moderate level of cohesion and separation in the clustering results. The Adjusted Rand Index (ARI) of 0.1286 and Normalized Mutual Information (NMI) of 0.3678 indicated a reasonable agreement with the true labels.

- The cluster analysis revealed that while K-Means effectively clustered certain classes like GALAXY_STARFORMING and GALAXY_UNKNOWN, there were notable misclassifications, particularly between similar galaxy and star types. This highlights the challenge of differentiating between spectrally similar classes.

Agglomerative Clustering Results

Agglomerative Clustering provided a comparative approach to evaluate clustering performance:

- The 3D PCA plot with Agglomerative Clustering clusters exhibited similar patterns to K-Means, with distinct groupings for different classes but also some overlap.

- The distribution of cluster sizes for Agglomerative Clustering also showed variability, with certain clusters being significantly larger, indicating a similar prevalence of certain spectral types.

- The dendrogram provided insights on the hierarchical clustering process and the gaps between some merges which indicated differences in said clusters.

- Evaluation metrics for Agglomerative Clustering included a Silhouette Score of 0.3139, a Davies-Bouldin Index of 0.8804, and a Calinski-Harabasz Index of 14703.6313. These metrics suggest a slightly lower clustering quality compared to K-Means. The ARI of 0.1616 and NMI of 0.3311 indicated reasonable, though slightly lower, agreement with the true labels.

- The cluster analysis for Agglomerative Clustering highlighted similar misclassification patterns as K-Means, with significant misclassifications among galaxy types like GALAXY_STARBURST and GALAXY_STARFORMING. Despite this, certain classes like GALAXY_UNKNOWN were effectively clustered.

Comparison of Clustering Techniques

The comparison of K-Means and Agglomerative Clustering revealed several important points:

- **Clustering Quality**: K-Means generally outperformed Agglomerative Clustering in terms of evaluation metrics, indicating better cluster cohesion and separation. This was evidenced by higher Silhouette and Calinski-Harabasz scores and a lower Davies-Bouldin Index.
- **Classification Accuracy**: Both algorithms showed reasonable performance in clustering major classes but struggled with similar spectral types. K-Means had a slightly higher ARI and NMI, suggesting better alignment with true labels.
- **Cluster Purity**: K-Means achieved slightly higher cluster purity, reflecting more accurate classification within clusters. However, both algorithms showed potential for improvement in reducing misclassifications.

Implications and Future Work

The findings from this study have several implications and highlight areas for future work:

- **Enhanced Feature Engineering**: To improve clustering accuracy, future work could focus on incorporating additional spectral features or leveraging advanced feature extraction techniques. This might help in better distinguishing between similar classes.
- **Algorithm Optimization**: Further optimization of clustering algorithms, including parameter tuning and exploring alternative methods such as density-based clustering, could enhance performance. Techniques like DBSCAN or ensemble clustering could be investigated.

- **Advanced Evaluation Metrics**: Employing more sophisticated evaluation metrics and validation techniques could provide deeper insights into clustering performance and guide improvements.
- **Domain-Specific Knowledge**: Integrating domain-specific knowledge, such as astrophysical properties and contextual information, could refine clustering approaches and enhance classification accuracy.

6.2 Future Work

The findings and insights from this study provide a solid foundation for further exploration and development in the field of astronomical data analysis using clustering techniques. There are several promising directions for future work that could enhance the effectiveness and accuracy of clustering algorithms when applied to astronomical spectra.

Advanced Feature Engineering

Improving feature engineering methods is a major focus of future research. Investigating alternative dimensionality reduction techniques like t-Distributed Stochastic Neighbour Embedding (t-SNE) or Uniform Manifold Approximation and Projection (UMAP) could offer more nuanced insights into the data structure, even though PCA was successful in reducing dimensionality and maintaining data structure. Furthermore, by adding more discriminative characteristics, domain-specific spectral properties such line intensities, redshift data, and other astrophysical factors could enhance the clustering efficiency.

Incorporating Deep Learning Techniques

Astronomy is one of the many fields in which deep learning has demonstrated significant promise. By incorporating deep learning models for feature extraction and clustering, like autoencoders, the quality of the clustering could be greatly improved. Autoencoders provide the ability to acquire elaborate, non-linear feature representations, potentially capturing more nuanced aspects of the spectral data. Using these representations in conjunction with conventional clustering techniques may provide clusters that are more precise and significant.

Exploring Alternative Clustering Algorithms

Investigating alternative clustering methods could enhance performance even more, even though K-Means and Agglomerative Clustering offered insightful information. One such approach is Density-Based Spatial Clustering of Applications with Noise (DBSCAN), which is able to efficiently handle noise while identifying clusters of different sizes and forms. Furthermore, more reliable and consistent clustering outcomes might be obtained by using ensemble clustering techniques, which integrate several clustering algorithms to create a consensus solution.

Algorithm Optimization and Scalability

Clustering algorithm optimisation represents a significant avenue for future research. Clustering performance could be improved by creating new optimisation techniques and optimising the parameters of current algorithms. Furthermore, it's critical to make sure these algorithms can scale to handle big astronomical datasets. Large-scale spectral data processing could be made possible by utilising high-performance computing resources and parallel computing approaches to solve scalability concerns.

Enhanced Evaluation Metrics

Employing more sophisticated evaluation metrics and validation techniques could provide deeper insights into clustering performance and guide improvements. Metrics such as the Adjusted Mutual Information (AMI), Variation of Information (VI), and others could offer complementary perspectives on clustering quality. Additionally, using bootstrapping and cross-validation techniques could provide more robust assessments of clustering stability and reliability.

Integration with Astrophysical Models

More thorough insights may be obtained by combining clustering results with astrophysical models and simulations. To verify the results, one may, for instance, compare the clustering results with theoretical models of galaxy formation and evolution. New patterns and anomalies that could result in new astrophysical discoveries may also be found with the aid of this integration.

Interactive Visualization Tools

The creation of interactive visualisation tools to investigate clustering results has the potential to greatly improve the findings' interpretability and usefulness. For researchers and astronomers, tools that enable interaction with 3D PCA plots, investigation of alternative clustering solutions, and diverse visualization of the spectral data could potentially enhance the analysis's comprehensibility and usefulness.

Application to Other Astronomical Surveys

Another interesting avenue is to apply the discovered clustering approaches to other astronomical datasets and surveys. The generalizability and robustness of the methods can be evaluated by verifying them on various datasets. This may also result in other surveys finding new astronomical objects and phenomena.

Open Source Implementation

Making the developed clustering algorithms and tools available as open-source software could benefit the wider research community. Providing comprehensive documentation, user guides, and example datasets could encourage adoption and collaboration, leading to further advancements and refinements.

In conclusion, there are lots of directions that future research could take in order to expand on the results of this study. The clustering of astronomical spectra can be significantly advanced, leading to new discoveries and a deeper understanding of the universe, by investigating advanced feature engineering, incorporating deep learning, optimising algorithms, improving evaluation metrics, integrating with astrophysical models, developing interactive tools, applying to other surveys, collaborating with the community, and offering open-source implementations.

Acknowledgements

I extend my sincere gratitude to Dr. Jonathon Loveday for his invaluable guidance and support on this project. Dr. Loveday's expertise in spectral classification and machine learning has been instrumental in shaping my understanding and approach.

I am also grateful to the faculty and staff of the Astronomy Department at the University of Sussex for providing the necessary resources and a conducive environment for this study. Their collective efforts have enriched this project and expanded my horizons in astrophysics and data science. I am grateful to my family and friends for their encouragement and support during this endeavour, which has been a source of strength and motivation.

Lastly, I acknowledge the Sloan Digital Sky Survey Data Release 18 (SDSS DR18) team for providing access to the spectral data that enabled this research to take place. The resources from SDSS DR18 have significantly contributed to advancing our understanding universe. This project would not have been possible without the support and contributions of these individuals and organizations. I am truly thankful for their impact on my academic and professional journey.

Bibliography

Almeida, A., Anderson, S. F., Argudo-Fernández, M., Badenes, C., Barger, K., Barrera-Ballesteros, J. K., Bender, C. F., Benitez, E., Besser, F., Bird, J. C., et al. (2023). The eighteenth data release of the sloan digital sky surveys: Targeting and first spectra from sdss-v. *The Astrophysical Journal Supplement Series*, 267(2):44.

Bailer-Jones, C. A., Irwin, M., and Von Hippel, T. (1998). Automated classification of stellar spectra—ii. two-dimensional classification with neural networks and principal components analysis. *Monthly Notices of the Royal Astronomical Society*, 298(2):361–377.

Ball, N. M. and Brunner, R. J. (2010). Data mining and machine learning in astronomy. *International Journal of Modern Physics D*, 19(07):1049–1106.

Baron, D. and Poznanski, D. (2019). Machine learning in astronomical spectra classification. *Nature Astronomy*, 3:93–98.

Berk, D. E. V., Richards, G. T., Bauer, A., Strauss, M. A., Schneider, D. P., Heckman, T. M., York, D. G., Hall, P. B., Fan, X., Knapp, G., et al. (2001). Composite quasar spectra from the sloan digital sky survey. *The Astronomical Journal*, 122(2):549.

Blanton, M. R., Bershady, M. A., Abolfathi, B., Albareti, F. D., Prieto, C. A., Almeida, A., Alonso-García, J., Anders, F., Anderson, S. F., Andrews, B., et al. (2017). Sloan digital sky survey iv: Mapping the milky way, nearby galaxies, and the distant universe. *The Astronomical Journal*, 154(1):28.

Bromová, P., Škoda, P., and Vážnỳ, J. (2014). Classification of spectra of emission line stars using machine learning techniques. *International Journal of Automation and Computing*, 11(3):265–273.

Caliński, T. and Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27.

Cannon, A. J. (1912). Preliminary general catalogue of 6,188 stars for the epoch 1900. *Annals of Harvard College Observatory*, 56:1–24.

Cannon, A. J. and Pickering, E. C. (1912). Classification of 1,477 stars by means of their photographic spectra. *Annals of the Astronomical Observatory of Harvard College; v. 56, no. 4, Cambridge, Mass.: The Observatory, 1912., p. 65-114; 30 cm.*, 56:65–114.

Charnock, T. and Moss, A. (2017). Deep recurrent neural networks for supernovae classification. *The Astrophysical Journal Letters*, 837(2):L28.

Connolly, A. J., Szalay, A., Bershady, M., Kinney, A., and Calzetti, D. (1994). Spectral classification of galaxies: an orthogonal approach. *arXiv preprint astro-ph/9411044*.

Covey, K. R., Ivezić, Ž., Schlegel, D., Finkbeiner, D., Padmanabhan, N., Lupton, R. H., Agüeros, M. A., Bochanski, J. J., Hawley, S. L., West, A. A., et al. (2007). Stellar seds from 0.3 to 2.5 μm: Tracing the stellar locus and searching for color outliers in the sdss and 2mass. *The Astronomical Journal*, 134(6):2398.

Davies, D. L. and Bouldin, D. W. (1979). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, PAMI-1(2):224–227.

Decherchi, S., Gastaldo, P., Redi, J., and Zunino, R. (2009). A text clustering framework for information retrieval. *Journal of information Assurance and Security*, 4:174–182.

DeVorkin, D. (1984). Stellar evolution and the origin of the hertzsprung-russell diagram. *The General history of astronomy*, 4.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231.

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

Fiorentin, P. R., Bailer-Jones, C., Lee, Y. S., Beers, T. C., Sivarani, T., Wilhelm, R., Prieto, C. A., and Norris, J. (2007). Estimation of stellar atmospheric parameters from sdss/segue spectra. *Astronomy & Astrophysics*, 467(3):1373–1387.

George, D. and Huerta, E. A. (2018). Deep learning for real-time gravitational wave detection and parameter estimation: Results with advanced ligo data. *Physics Letters B*, 778:64–70.

Graham, M., Drake, A., Djorgovski, S., Mahabal, A., and Donalek, C. (2017). Challenges in the automated classification of variable stars in large databases. In *EPJ Web of Conferences*, volume 152, page 03001. EDP Sciences.

Grasha, K., Calzetti, D., Adamo, A., Kim, H., Elmegreen, B. G., Gouliermis, D., Dale, D. A., Fumagalli, M., Grebel, E. K., Johnson, K. E., et al. (2017). The hierarchical distribution of the young stellar clusters in six local star-forming galaxies. *The Astrophysical Journal*, 840(2):113.

Gray, R. O. (2021). Stellar spectral classification. *Princeton University Press*.

Gray, R. O. and Corbally, C. J. (2005). *Stellar Spectral Classification*. Princeton University Press.

Haggar, R., De Luca, F., De Petris, M., Sazonova, E., Taylor, J. E., Knebe, A., Gray, M. E., Pearce, F. R., Contreras-Santos, A., Cui, W., et al. (2024). Reconsidering the dynamical states of galaxy clusters using pca and umap. *Monthly Notices of the Royal Astronomical Society*, 532(1):1031–1048.

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of classification*, 2:193–218.

Ivezić, Ž., Connolly, A. J., VanderPlas, J. T., and Gray, A. (2020). *Statistics, data mining, and machine learning in astronomy: a practical Python guide for the analysis of survey data*, volume 8. Princeton University Press.

Ivezić, Ž., Kahn, S. M., Tyson, J. A., Abel, B., Acosta, E., Allsman, R., Alonso, D., AlSayyad, Y., Anderson, S. F., Andrew, J., et al. (2019). Lsst: from science drivers to reference design and anticipated data products. *The Astrophysical Journal*, 873(2):111.

Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254.

Jolliffe, I. T. and Cadima, J. (2016). Principal component analysis: a review and recent developments. *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202.

Kauffmann, G., Heckman, T. M., Tremonti, C., Brinchmann, J., Charlot, S., White, S. D., Ridgway, S. E., Brinkmann, J., Fukugita, M., Hall, P. B., et al. (2003). The host galaxies of active galactic nuclei. *Monthly Notices of the Royal Astronomical Society*, 346(4):1055–1077.

Kennicutt Jr, R. C. (1998). Star formation in galaxies along the hubble sequence. *Annual Review of Astronomy and Astrophysics*, 36(1):189–231.

Kim, E. J., Brunner, R. J., and Carrasco Kind, M. (2015). A hybrid ensemble learning approach to star–galaxy classification. *Monthly Notices of the Royal Astronomical Society*, 453(1):507–521.

Lanusse, F. et al. (2023). The dawes review 10: The impact of deep learning for the analysis of galaxy surveys. *Publications of the Astronomical Society of Australia*, 40:e001.

Li, X., Zheng, Y., Wang, X., and Wang, L. (2020). Predicting solar flares using a novel deep convolutional neural network. *The Astrophysical Journal*, 891(1):10.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

McInnes, L., Healy, J., and Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Mika, S., Ratsch, G., Weston, J., Scholkopf, B., and Mullers, K.-R. (1999). Fisher discriminant analysis with kernels. In *Neural networks for signal processing IX: Proceedings of the 1999 IEEE signal processing society workshop (cat. no. 98th8468)*, pages 41–48. Ieee.

Morgan, W., Keenan, P. C., and Kellman, E. (1942). An atlas of stellar spectra. *University of Chicago*.

Morgan, W. W., Abt, H. A., and Tapscott, J. (1978). *Revised MK spectral atlas for stars earlier than the sun*. Yerkes Observatory, University of Chicago.

Morgan, W. W., Keenan, P. C., and Kellman, E. (1943). An atlas of stellar spectra.

Mustafa, M., Bard, D., Bhimji, W., Lukić, Z., Al-Rfou, R., and Kratochvil, J. M. (2019). Cosmogan: creating high-fidelity weak lensing convergence maps using generative adversarial networks. *Computational Astrophysics and Cosmology*, 6:1–13.

Pickering, E. C. (1886). Photographic study of stellar spectra. *Science*, 7(164):278–278.

Preston, G. W. (1974). The chemically peculiar stars of the upper main sequence. *In: Annual review of astronomy and astrophysics. Volume 12.(A75-13476 03-90) Palo Alto, Calif., Annual Reviews, Inc., 1974, p. 257-277.*, 12:257–277.

Prisinzano, L., Damiani, F., Sciortino, S., Flaccomio, E., Guarcello, M., Micela, G., Tognelli, E., Jeffries, R., and Alcalá, J. (2022). Low-mass young stars in the milky way unveiled by dbscan and gaia edr3: Mapping the star forming regions within 1.5 kpc. *Astronomy & Astrophysics*, 664:A175.

Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215.

Schreiber, C. (2016). *A statistical and multi-wavelength study of star formation in galaxies*. Springer.

Secchi, A. (1866). Classificazione delle stelle. *Memorie della Societa degli Spettroscopisti Italiani*, 1:47–53.

Sen, S., Agarwal, S., Chakraborty, P., and Singh, K. P. (2022). Astronomical big data processing using machine learning: A comprehensive review. *Experimental Astronomy*, 53(1):1–43.

Shallue, C. J. and Vanderburg, A. (2018). Identifying exoplanets with deep learning: A five-planet resonant chain around kepler-80 and an eighth planet around kepler-90. *The Astronomical Journal*, 155(2):94.

Singh, H. P., Gulati, R. K., and Gupta, R. (1998). Stellar spectral classification using principal component analysis and artificial neural networks. *Monthly Notices of the Royal Astronomical Society*, 295(2):312–318.

Strantzalis, A., Lazarou, D., Hatzidimitriou, D., Zezas, A., Antoniou, V., and Reskos, N. (2024). A robust automated machine-learning method for the identification of star clusters in the central region of the small magellanic cloud. *Astronomy & Astrophysics*, 681:A24.

Strehl, A. and Ghosh, J. (2002). Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec):583–617.

Traven, G., Matijevič, G., Zwitter, T., Žerjal, M., Kos, J., Asplund, M., Bland-Hawthorn, J., Casey, A. R., De Silva, G., Freeman, K., et al. (2017). The galah survey: classification and diagnostics with t-sne reduction of spectral information. *The Astrophysical Journal Supplement Series*, 228(2):24.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Walborn, N. R. (1971). Spectroscopic studies of the o stars. *Astrophysical Journal Supplement Series*, 23:257–285.

Wang, K., Guo, P., and Luo, A.-L. (2017). A new automated spectral feature extraction method and its application in spectral classification and defective spectra recovery. *Monthly Notices of the Royal Astronomical Society*, 465(4):4311–4324.

Welther, B. (1993). Annie jump cannon:"life after the henry draper catalogue.". In *American Astronomical Society, 182nd AAS Meeting, id. 81.03; Bulletin of the American Astronomical Society, Vol. 25, p. 933*, volume 25, page 933.

White, S. D. and Frenk, C. S. (1991). Galaxy formation through hierarchical clustering. *Astrophysical Journal, Part 1 (ISSN 0004-637X), vol. 379, Sept. 20, 1991, p. 52-79. Research supported by NASA, NSF, and SERC.*, 379:52–79.

York, D. G., Adelman, J., Anderson Jr, J. E., Anderson, S. F., Annis, J., Bahcall, N. A., Bakken, J., Barkhouser, R., Bastian, S., Berman, E., et al. (2000). The sloan digital sky survey: Technical summary. *The Astronomical Journal*, 120(3):1579.

Zhong-Bao, L. and Li-Peng, S. (2015). Stellar spectral subclasses classification based on fisher criterion and manifold learning. *Publications of the Astronomical Society of the Pacific*, 127(954):789–794.

Appendix A

Code

```
import os
import random
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn.decomposition import PCA
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.cluster import KMeans, AgglomerativeClustering
from sklearn.metrics import confusion_matrix
import seaborn as sns
from astropy.io import fits
from scipy.interpolate import interp1d
from scipy.ndimage import gaussian_filter1d
from scipy.stats import mode
import matplotlib.cm as cm

# Load spectrum function
def load_spectrum(file_path, cutoff_wavelength=8000):
    try:
        with fits.open(file_path) as hdul:
            if len(hdul) < 3:
                raise ValueError("Expected at least 3 HDUs in the FITS file")

            data = hdul[1].data
            if data is None:
```

```
                raise ValueError("No data found in HDU 1")

            loglam = data['loglam']
            flux = data['flux']
            wavelength = 10**loglam

            mask = wavelength <= cutoff_wavelength
            wavelength = wavelength[mask]
            flux = flux[mask]

            hdu2_data = hdul[2].data
            if hdu2_data is None:
                raise ValueError("No data found in HDU 2")

            class_label = hdu2_data['CLASS'][0].strip()

            subclass_label = 'UNKNOWN'
            if 'SUBCLASS' in hdu2_data.columns.names:
                subclass_value = hdu2_data['SUBCLASS'][0]
                if isinstance(subclass_value, str) and subclass_value.
    strip():
                    subclass_label = subclass_value.strip()

        return wavelength, flux, class_label, subclass_label
    except Exception as e:
        print(f"Error loading spectrum from {file_path}: {e}")
        return None, None, None, None

# Normalization, smoothing, and baseline correction
def normalize_flux(flux):
    return flux / np.max(flux)

def smooth_flux(flux, sigma=2):
    return gaussian_filter1d(flux, sigma=sigma)


def pre_process_spectrum(wavelength, flux):
    flux = normalize_flux(flux)
    flux = smooth_flux(flux)
```

```
64      return wavelength, flux
65
66  def resample_spectrum(wavelength, flux, common_wavelength):
67      interp_flux = interp1d(wavelength, flux, kind='linear',
        bounds_error=False, fill_value="extrapolate")
68      return interp_flux(common_wavelength)
69
70  # Load spectra and process
71  spectra_folder = 'SPECTRA'
72  spectra_files = os.listdir(spectra_folder)
73
74  num_spectra_to_load = 15000
75  random_files = random.sample(spectra_files, num_spectra_to_load)
76
77  all_fluxes = []
78  all_combined_labels = []
79
80  # Define a common wavelength grid
81  common_wavelength = np.linspace(4000, 8000, 1000)
82
83  for file_name in random_files:
84      file_path = os.path.join(spectra_folder, file_name)
85      wavelength, flux, class_label, subclass_label = load_spectrum(
        file_path)
86
87      if wavelength is None or flux is None:
88          continue
89
90      # Preprocess and resample spectrum
91      wavelength, flux = pre_process_spectrum(wavelength, flux)
92      flux = resample_spectrum(wavelength, flux, common_wavelength)
93
94      all_fluxes.append(flux)
95      all_combined_labels.append(f"{class_label}_{subclass_label}")
96
97  # Stack all fluxes for PCA
98  all_fluxes = np.array(all_fluxes)
99  all_combined_labels = np.array(all_combined_labels)
100
```

```
print(f"Number of spectra after loading: {len(all_fluxes)}")

# Encode the labels to numeric values
label_encoder = LabelEncoder()
encoded_labels = label_encoder.fit_transform(all_combined_labels)

scaler = StandardScaler()
scaled_fluxes = scaler.fit_transform(all_fluxes)

# Perform PCA
pca = PCA(n_components=3)
pca_components = pca.fit_transform(scaled_fluxes)
# Perform PCA Plotting in 3D
fig = plt.figure(figsize=(14, 10))
ax = fig.add_subplot(111, projection='3d')

# Use a larger colormap with many distinct colors
cmap = cm.get_cmap('tab20', num_clusters)

# Plot the PCA components
scatter = ax.scatter(pca_components[:, 0], pca_components[:, 1],
    pca_components[:, 2],
                     c=encoded_labels, cmap=cmap, s=5)

# Set axis limits to zoom in on the dense region
ax.set_xlim(-40, 55)
ax.set_ylim(-40, 20)
ax.set_zlim(-30, 50)

# Set axis labels
ax.set_xlabel('PCA Component 1')
ax.set_ylabel('PCA Component 2')
ax.set_zlabel('PCA Component 3')

plt.title('3D PCA Plot of Spectra')

# Adjust the viewing angle to better see the layers
ax.view_init(elev=20, azim=60)

```

```
# Directly use the encoded labels for the legend
unique_labels = np.unique(encoded_labels)
handles = [plt.Line2D([0], [0], marker='o', color='w', markerfacecolor=
    cmap(i), markersize=5) for i in unique_labels]
legend_labels = label_encoder.inverse_transform(unique_labels)

# Place the legend outside the plot
legend = ax.legend(handles, legend_labels, loc="center left",
    bbox_to_anchor=(1.05, 0.5), fontsize='small', title="Classes",
    title_fontsize='medium')

plt.show()
num_clusters = len(np.unique(all_combined_labels))

# Explained Variance Plot
plt.figure(figsize=(6, 4))
plt.plot(np.arange(1, len(pca.explained_variance_ratio_) + 1), np.
    cumsum(pca.explained_variance_ratio_), marker='o')
plt.xlabel('Number of PCA Components')
plt.ylabel('Cumulative Explained Variance')
plt.title('PCA Explained Variance')
plt.grid(True)
plt.show()

# Define range of K values to test
K = range(1, 55)  # Test cluster sizes from 1 to 14

# Initialize lists to store the metrics
inertia = []
silhouette_scores = []
calinski_harabasz_scores = []
davies_bouldin_scores = []

# Perform K-Means clustering for each K and calculate the metrics
for k in K:
    kmeans = KMeans(n_clusters=k, random_state=42)
    cluster_labels = kmeans.fit_predict(pca_components)

    # Calculate metrics
```

```
174     inertia.append(kmeans.inertia_)
175
176     if k > 1:  # Silhouette score is undefined for k=1
177         silhouette_scores.append(silhouette_score(pca_components,
    cluster_labels))
178         calinski_harabasz_scores.append(calinski_harabasz_score(
    pca_components, cluster_labels))
179         davies_bouldin_scores.append(davies_bouldin_score(
    pca_components, cluster_labels))
180
181 # Plot Elbow Method (Inertia)
182 plt.figure(figsize=(12, 8))
183 plt.subplot(2, 2, 1)
184 plt.plot(K, inertia, 'bo-')
185 plt.title('Elbow Method For Optimal K')
186 plt.xlabel('Number of Clusters (K)')
187 plt.ylabel('Inertia')
188 plt.grid(True)
189
190 # Plot Silhouette Scores
191 plt.subplot(2, 2, 2)
192 plt.plot(K[1:], silhouette_scores, 'bo-')
193 plt.title('Silhouette Score For Optimal K')
194 plt.xlabel('Number of Clusters (K)')
195 plt.ylabel('Silhouette Score')
196 plt.grid(True)
197
198 # Plot Calinski-Harabasz Index
199 plt.subplot(2, 2, 3)
200 plt.plot(K[1:], calinski_harabasz_scores, 'bo-')
201 plt.title('Calinski-Harabasz Index For Optimal K')
202 plt.xlabel('Number of Clusters (K)')
203 plt.ylabel('Calinski-Harabasz Index')
204 plt.grid(True)
205
206 # Plot Davies-Bouldin Index
207 plt.subplot(2, 2, 4)
208 plt.plot(K[1:], davies_bouldin_scores, 'bo-')
209 plt.title('Davies-Bouldin Index For Optimal K')
```

```
plt.xlabel('Number of Clusters (K)')
plt.ylabel('Davies-Bouldin Index')
plt.grid(True)

plt.tight_layout()
plt.show()
# Function to match clusters to original labels
def match_labels_to_clusters(original_labels, clusters):
    matched_labels = np.zeros_like(clusters)
    for cluster in np.unique(clusters):
        mask = clusters == cluster
        matched_labels[mask] = mode(original_labels[mask])[0]
    return matched_labels

import matplotlib.cm as cm  # Ensure you have imported the colormap module
from scipy.stats import mode
# Perform K-Means Clustering with the chosen number of clusters
optimal_k = 22
kmeans = KMeans(n_clusters=optimal_k, random_state=42)
kmeans_clusters = kmeans.fit_predict(pca_components)

# Create a dictionary to map each cluster to the most frequent original label
cluster_labels = {}
for cluster in np.unique(kmeans_clusters):
    mask = kmeans_clusters == cluster
    most_frequent_label = mode(encoded_labels[mask])[0][0]
    cluster_labels[cluster] = most_frequent_label

# 3D PCA Plot with Centroids
fig = plt.figure(figsize=(14, 10))
ax = fig.add_subplot(111, projection='3d')

# Plot the PCA components colored by their cluster assignment
scatter = ax.scatter(pca_components[:, 0], pca_components[:, 1], pca_components[:, 2],
                     c=kmeans_clusters, cmap='tab10', s=5)

```

```
# Plot the centroids
centroids = kmeans.cluster_centers_
ax.scatter(centroids[:, 0], centroids[:, 1], centroids[:, 2], c='black', s=100, marker='x', label='Centroids')

# Set axis limits to zoom in on the dense region
ax.set_xlim(-40, 55)
ax.set_ylim(-40, 20)
ax.set_zlim(-30, 50)

# Set axis labels
ax.set_xlabel('PCA Component 1')
ax.set_ylabel('PCA Component 2')
ax.set_zlabel('PCA Component 3')
ax.set_title(f'3D PCA Plot with K-Means Clustering (K={optimal_k})')

# Generate unique labels for each cluster in the legend
handles = []
legend_labels = []
for cluster in np.unique(kmeans_clusters):
    cluster_color = cm.tab10(cluster % 10)  # Use the colormap from matplotlib.cm
    handles.append(plt.Line2D([0], [0], marker='o', color='w', markerfacecolor=cluster_color, markersize=5))
    legend_labels.append(f"Cluster {cluster + 1}: {label_encoder.inverse_transform([cluster_labels[cluster]])[0]}")

# Place the legend outside the plot
legend = ax.legend(handles, legend_labels, title="K-Means Clusters", loc="center left", bbox_to_anchor=(1.05, 0.5))

ax.view_init(elev=20, azim=60)

plt.show()
# 2D PCA Projection: PCA Component 1 vs Component 2
fig, ax = plt.subplots(figsize=(8, 6))

# Scatter plot
```

```
ax.scatter(pca_components[:, 0], pca_components[:, 1], c=
    kmeans_clusters, cmap='tab10', s=10, alpha=0.7)

# Set axis labels and title
ax.set_xlabel('PCA Component 1')
ax.set_ylabel('PCA Component 2')
ax.set_title('2D Projection: PCA Component 1 vs 2')

# Show the plot
plt.show()
# 2D PCA Projection: PCA Component 1 vs Component 3
fig, ax = plt.subplots(figsize=(8, 6))

# Scatter plot
ax.scatter(pca_components[:, 0], pca_components[:, 2], c=
    kmeans_clusters, cmap='tab10', s=10, alpha=0.7)

# Set axis labels and title
ax.set_xlabel('PCA Component 1')
ax.set_ylabel('PCA Component 3')
ax.set_title('2D Projection: PCA Component 1 vs 3')

# Show the plot
plt.show()
# 2D PCA Projection: PCA Component 1 vs Component 3
fig, ax = plt.subplots(figsize=(8, 6))

# Scatter plot
ax.scatter(pca_components[:, 1], pca_components[:, 2], c=
    kmeans_clusters, cmap='tab10', s=10, alpha=0.7)

# Set axis labels and title
ax.set_xlabel('PCA Component 2')
ax.set_ylabel('PCA Component 3')
ax.set_title('2D Projection: PCA Component 2 vs 3')

# Show the plot
plt.show()

```

```
import pandas as pd
# Create a DataFrame for analysis
cluster_analysis_df = pd.DataFrame({'True_Label': encoded_labels, '
    Predicted_Cluster': kmeans_clusters})

# Group by predicted clusters and analyze the distribution of true
    labels
for cluster_id in range(optimal_k):
    cluster_data = cluster_analysis_df[cluster_analysis_df['
    Predicted_Cluster'] == cluster_id]
    most_common_label, count = mode(cluster_data['True_Label'])
    print(f"Cluster {cluster_id}: Most common label is {
    most_common_label[0]} with {count[0]} occurrences.")
    print(f"Label distribution in this cluster:\n{cluster_data['
    True_Label'].value_counts(normalize=True)}\n")
    # Distribution of cluster sizes for K-Means
kmeans_cluster_sizes = np.bincount(kmeans_clusters)

# Plotting the distribution
plt.figure(figsize=(10, 7))
plt.bar(range(len(kmeans_cluster_sizes)), kmeans_cluster_sizes, color='
    blue')
plt.title('Distribution of Cluster Sizes for K-Means Clustering')
plt.xlabel('Cluster')
plt.ylabel('Size')
plt.show()

from sklearn.metrics import adjusted_rand_score,
    normalized_mutual_info_score
# Statistical quantities
kmeans_silhouette = silhouette_score(pca_components, kmeans_clusters)
kmeans_davies_bouldin = davies_bouldin_score(pca_components,
    kmeans_clusters)
kmeans_calinski_harabasz = calinski_harabasz_score(pca_components,
    kmeans_clusters)
kmeans_ari = adjusted_rand_score(encoded_labels, kmeans_clusters)

# Calculate the Normalized Mutual Information (NMI)
```

```
kmeans_nmi = normalized_mutual_info_score(encoded_labels,
    kmeans_clusters)
# Cluster purity
def cluster_purity(true_labels, cluster_labels):
    contingency_matrix = confusion_matrix(true_labels, cluster_labels)
    return np.sum(np.amax(contingency_matrix, axis=0)) / np.sum(
    contingency_matrix)

kmeans_purity = cluster_purity(encoded_labels, kmeans_clusters)
# Print statistical quantities
print("K-Means Clustering Statistics:")
print(f"Silhouette Score: {kmeans_silhouette}")
print(f"Davies-Bouldin Index: {kmeans_davies_bouldin}")
print(f"Calinski-Harabasz Index: {kmeans_calinski_harabasz}")
print(f"Adjusted Rand Index (ARI): {kmeans_ari}")
print(f"Normalized Mutual Information (NMI): {kmeans_nmi}")
print(f"Cluster Purity: {kmeans_purity}")

from scipy.cluster.hierarchy import linkage, dendrogram, fcluster

# Generate the linkage matrix using 'ward' method (commonly used in
    hierarchical clustering)
Z = linkage(pca_components, method='ward')

# Plot the dendrogram
plt.figure(figsize=(14, 10))
dendro = dendrogram(Z, truncate_mode='level', p=6)  # p=6 means show
    only the last 6 levels
plt.title('Hierarchical Clustering Dendrogram')
plt.xlabel('Sample Index')
plt.ylabel('Distance')
plt.show()
from scipy.stats import mode
import matplotlib.cm as cm
# cmap = cm.get_cmap('tab20', num_clusters)
cutoff_distance = 240

# Form clusters by cutting the dendrogram at the chosen distance
```

```
agg_clusters_from_dendrogram = fcluster(Z, cutoff_distance, criterion='distance')

# Number of clusters formed
num_clusters_formed = len(np.unique(agg_clusters_from_dendrogram))
print(f'Number of clusters formed: {num_clusters_formed}')
cmap = cm.get_cmap('tab20', num_clusters_formed)
# Assuming you have the `agg_cluster_labels` that map clusters to CLASS_SUBCLASS names
unique_labels = np.unique(agg_cluster_labels)
num_labels = len(unique_labels)

fig = plt.figure(figsize=(14, 10))
ax = fig.add_subplot(111, projection='3d')

# Plot the PCA components colored by their cluster assignment from the dendrogram
scatter = ax.scatter(pca_components[:, 0], pca_components[:, 1], pca_components[:, 2],
                     c=agg_clusters_from_dendrogram, cmap='tab10', s=5)

# Set axis limits to zoom in on the dense region
ax.set_xlim(-40, 55)
ax.set_ylim(-40, 20)
ax.set_zlim(-30, 50)

# Set axis labels
ax.set_xlabel('PCA Component 1')
ax.set_ylabel('PCA Component 2')
ax.set_zlabel('PCA Component 3')
plt.title(f'3D PCA Plot with Agglomerative Clustering (Cutoff at {cutoff_distance})')
# # Create a dictionary to map each cluster to the most frequent CLASS_SUBCLASS label
cluster_to_class_subclass = {}

# For each cluster, find the most frequent CLASS_SUBCLASS label
for cluster in np.unique(agg_clusters_from_dendrogram):
```

```
    mask = agg_clusters_from_dendrogram == cluster  # Select data points in the current cluster
    most_frequent_label = mode(encoded_labels[mask])[0][0]  # Find the most frequent label in this cluster
    cluster_to_class_subclass[cluster] = most_frequent_label

# Decode the labels to get the actual CLASS_SUBCLASS names
cluster_to_class_subclass_names = {cluster: label_encoder.inverse_transform([label])[0]
                                   for cluster, label in cluster_to_class_subclass.items()}
agg_cluster_labels = np.array([
    cluster_to_class_subclass_names[cluster]
    for cluster in agg_clusters_from_dendrogram
])

# # Print out the mapping
for cluster, class_subclass in cluster_to_class_subclass_names.items():
  print(f"Cluster {cluster}: {class_subclass}")

agg_cluster_labels = np.array([cluster_to_class_subclass_names[cluster]
                               for cluster in agg_clusters_from_dendrogram])
 ax.view_init(elev=20, azim=60)

plt.show()
plt.show()

# Generate the legend using cluster numbers and CLASS_SUBCLASS names
handles = []
legend_labels = []
for cluster in np.unique(agg_clusters_from_dendrogram):
    cluster_color = plt.cm.tab20(cluster % 20)  # Use tab20 colormap with cycling colors
    handles.append(plt.Line2D([0], [0], marker='o', color='w', markerfacecolor=cluster_color, markersize=5))
    legend_labels.append(f"Cluster {cluster}: {cluster_to_class_subclass_names[cluster]}")

```

```
# Place the legend outside the plot
legend = ax.legend(handles, legend_labels, title="Agglomerative Clusters", loc="center left", bbox_to_anchor=(1.05, 0.5))

# Adjust the viewing angle to better see the clusters
ax.view_init(elev=20, azim=60)

plt.show()
from scipy.stats import mode
import matplotlib.cm as cm
# Perform Agglomerative Clustering with the chosen number of clusters
optimal_k = 22
agg_clustering = AgglomerativeClustering(n_clusters=optimal_k)
agg_clusters = agg_clustering.fit_predict(pca_components)

# Create a dictionary to map each cluster to the most frequent original label
cluster_labels = {}
for cluster in np.unique(agg_clusters):
    mask = agg_clusters == cluster
    most_frequent_label = mode(encoded_labels[mask])[0][0]
    cluster_labels[cluster] = most_frequent_label

# 3D PCA Plot with Centroids
fig = plt.figure(figsize=(14, 10))
ax = fig.add_subplot(111, projection='3d')

# Plot the PCA components colored by their cluster assignment
scatter = ax.scatter(pca_components[:, 0], pca_components[:, 1], pca_components[:, 2],
                     c=agg_clusters, cmap='tab10', s=5)

# Set axis limits to zoom in on the dense region
ax.set_xlim(-40, 55)
ax.set_ylim(-40, 20)
ax.set_zlim(-30, 50)

# Set axis labels
ax.set_xlabel('PCA Component 1')
```

```
ax.set_ylabel('PCA Component 2')
ax.set_zlabel('PCA Component 3')
ax.set_title(f'3D PCA Plot with Agglomerative Clustering (K={optimal_k})')

# Generate unique labels for each cluster in the legend
handles = []
legend_labels = []
for cluster in np.unique(agg_clusters):
    cluster_color = cm.tab10(cluster % 10)  # Use the colormap from matplotlib.cm
    handles.append(plt.Line2D([0], [0], marker='o', color='w', markerfacecolor=cluster_color, markersize=5))
    legend_labels.append(f"Cluster {cluster + 1}: {label_encoder.inverse_transform([cluster_labels[cluster]])[0]}")

# Place the legend outside the plot
legend = ax.legend(handles, legend_labels, title="Agglomerative Clustering Clusters", loc="center left", bbox_to_anchor=(1.05, 0.5))

ax.view_init(elev=20, azim=60)

plt.show()

# 2D PCA Projection: PCA Component 1 vs Component 2
fig, ax = plt.subplots(figsize=(8, 6))

# Scatter plot
ax.scatter(pca_components[:, 0], pca_components[:, 1], c=agg_clusters, cmap='tab10', s=10, alpha=0.7)

# Set axis labels and title
ax.set_xlabel('PCA Component 1')
ax.set_ylabel('PCA Component 2')
ax.set_title('2D Projection: PCA Component 1 vs 2')

# Show the plot
plt.show()
# 2D PCA Projection: PCA Component 1 vs Component 3
```

```
fig, ax = plt.subplots(figsize=(8, 6))

# Scatter plot
ax.scatter(pca_components[:, 0], pca_components[:, 2], c=agg_clusters,
    cmap='tab10', s=10, alpha=0.7)

# Set axis labels and title
ax.set_xlabel('PCA Component 1')
ax.set_ylabel('PCA Component 3')
ax.set_title('2D Projection: PCA Component 1 vs 3')

# Show the plot
plt.show()
# 2D PCA Projection: PCA Component 1 vs Component 3
fig, ax = plt.subplots(figsize=(8, 6))

# Scatter plot
ax.scatter(pca_components[:, 1], pca_components[:, 2], c=agg_clusters,
    cmap='tab10', s=10, alpha=0.7)

# Set axis labels and title
ax.set_xlabel('PCA Component 2')
ax.set_ylabel('PCA Component 3')
ax.set_title('2D Projection: PCA Component 2 vs 3')

# Show the plot
plt.show()

from sklearn.metrics import adjusted_rand_score,
    normalized_mutual_info_score

agg_silhouette = silhouette_score(pca_components, agg_clusters)
agg_davies_bouldin = davies_bouldin_score(pca_components, agg_clusters)
agg_calinski_harabasz = calinski_harabasz_score(pca_components,
    agg_clusters)
agg_ari = adjusted_rand_score(encoded_labels, agg_clusters)
agg_nmi = normalized_mutual_info_score(encoded_labels, agg_clusters)
# Cluster purity
def cluster_purity(true_labels, cluster_labels):
```

```
    contingency_matrix = confusion_matrix(true_labels, cluster_labels)
    return np.sum(np.amax(contingency_matrix, axis=0)) / np.sum(
    contingency_matrix)
agg_purity = cluster_purity(encoded_labels, agg_clusters)
print("\nAgglomerative Clustering Statistics:")
print(f"Silhouette Score: {agg_silhouette}")
print(f"Davies-Bouldin Index: {agg_davies_bouldin}")
print(f"Calinski-Harabasz Index: {agg_calinski_harabasz}")
print(f"Adjusted Rand Index (ARI): {agg_ari}")
print(f"Normalized Mutual Information (NMI): {agg_nmi}")
print(f"Cluster Purity: {agg_purity}")
# Distribution of cluster sizes for Agglomerative Clustering
agg_cluster_sizes = np.bincount(agg_clusters)
plt.figure(figsize=(10, 7))
plt.bar(range(len(agg_cluster_sizes)), agg_cluster_sizes, color='green'
    )
plt.title('Distribution of Cluster Sizes for Agglomerative Clustering')
plt.xlabel('Cluster')
plt.ylabel('Size')
plt.show()
import pandas as pd

original_class_names = label_encoder.inverse_transform(encoded_labels)

# Create a DataFrame for analysis
cluster_analysis_df = pd.DataFrame({
    'True_Label': encoded_labels,            # Encoded labels
    'True_Class_Name': original_class_names, # Decoded class names
    'Predicted_Cluster': agg_clusters        # Clusters from
    agglomerative clustering
})

# Group by predicted clusters and analyze the distribution of true
    labels
for cluster_id in range(optimal_k):
    cluster_data = cluster_analysis_df[cluster_analysis_df['
    Predicted_Cluster'] == cluster_id]
    most_common_label_encoded, count = mode(cluster_data['True_Label'])
```

```
578     most_common_class_name = label_encoder.inverse_transform([
        most_common_label_encoded[0]])[0]
579
580     print(f"Cluster {cluster_id}: Most common class is '{
        most_common_class_name}' with {count[0]} occurrences.")
581     print(f"Class distribution in this cluster:\n{cluster_data['
        True_Class_Name'].value_counts(normalize=True)}\n")
```