# Astronomical Object Classification

**Shlomi Zecharia, Yosef Desse**

*Ariel University, Ariel, Israel*

Shlomi Zecharia,
Department of Computer Science and Mathematics,
Ariel University, Ariel, Israel
shlomi.ze29@gmail.com

Yosef Desse,
Department of Computer Science and Mathematics,
Ariel University, Ariel, Israel
yosefdassa@gmail.com

**Abstract:** This paper presents a classification model for categorizing astronomical objects into stars, galaxies, and quasars using data from the Sloan Digital Sky Survey DR14. The dataset consists of 10,000 observations with 17 feature columns, including key attributes like redshift, spectral type, and magnitude. The goal is to develop a model that accurately classifies objects based on these features.

**Github:**https://github.com/2shlomi9/Astronomical-Object-Classification.git

## Introduction

Astronomical object classification plays a crucial role in modern astrophysics, as it allows researchers to categorize celestial objects based on their physical and observational properties. With advancements in space observation technology, large-scale surveys like the Sloan Digital Sky Survey DR14 have provided extensive datasets that include detailed information about stars, galaxies, and quasars. These objects can be classified using various attributes such as redshift, spectral type, and magnitude, which reflect their underlying physical properties. The classification process is essential for understanding the distribution, evolution, and properties of astronomical objects. It enables scientists to make predictions about the nature of objects in the universe, identify new phenomena, and explore relationships between different types of celestial bodies. This paper focuses on the development and application of machine learning techniques to classify astronomical objects based on the SDSS DR14 dataset. The goal is to utilize the available features to build a robust and accurate classification model that can reliably distinguish between stars, galaxies, and quasars.

## Dataset Overview

The dataset used in this project is derived from the Sloan Digital Sky Survey DR14, which is one of the most comprehensive astronomical surveys. The data includes information on various celestial objects, including stars, galaxies, and quasars, which will be classified based on their features. The dataset consists of 10,000 observations, each described by 17 feature columns and one class column that assigns each object to one of the following categories: star, galaxy, or quasar.

The key attributes used for classification include redshift, spectral type, and magnitude across multiple wavelength bands. These features are crucial for understanding the characteristics of the objects and differentiating between the various categories. The dataset also includes spatial information (right ascension and declination), which provides insights into the positions of the objects in the sky.

**Key Features:**

- **objid (Object Identifier):** A unique identifier for each observation
  Usage: Primarily used to distinguish between different observations.

- **ra (Right Ascension**): The angular distance along the celestial equator from the Sun at the March equinox to the hour circle of the point above the Earth.
  Usage: Right Ascension, along with Declination, helps pinpoint the position of an object in the sky. In this context, it can provide geographical location data in the celestial coordinate system.

- **dec (Declination):** The angular distance above or below the celestial equator, indicating the location of an object on the celestial sphere.
  Usage: Works in tandem with Right Ascension (RA) to pinpoint the object's location in the sky.

  **u, g, r, i, z (Magnitude):** These represent the brightness of the object observed in the five filters or bands (u, g, r, i, z) of the telescope. These magnitudes are part of the Thuan-Gunn photometric system.

• u: Represents the ultraviolet magnitude.
• g: Represents the green filter.
• r: Represents the red filter.
• i: Represents the near-infrared filter.
• z: Represents the far-infrared filter.
Usage: These magnitudes are essential for determining the color and temperature of the astronomical objects. They help classify objects like stars, galaxies, and quasars based on their brightness across different wavelengths. Hotter objects like stars tend to have specific patterns in these magnitudes, which can aid in classification.

- **redshift:** The shift in the wavelength of light emitted by the object, typically towards the red end of the spectrum. This shift occurs because objects moving away from Earth stretch the light's wavelength.
  Usage: Redshift helps in estimating the distance of an astronomical object and understanding its motion relative to Earth. Higher redshift values typically indicate that the object is farther away, and this information is vital when classifying galaxies and quasars, which can have different redshift values compared to stars.

- **run (Run Number):** A rerun number specifies a reprocessing of the same observation.
  Usage: This is useful in understanding if there were any changes in the data processing methods or if the observation data has been refined or re-analyzed.

- **camcol (Camera Column):** A camera column number identifying the scan line within the observation run.
  Usage: This feature helps identify the specific area within the telescope's field of view during the observation.

- **field (Field Number):** The field number is used to identify a section of the sky observed during the scan.
  Usage: Like other technical features (run, rerun, camcol), it helps pinpoint the location of the observation. While this can be useful in some spatial or time-series analysis.

- **specobjid (Spectral Object Identifier):** A unique identifier for each object in the spectral data view.
  Usage: This serves primarily as a reference for matching spectral data to objects. It's helpful for combining multiple datasets or for linking objects across different views.

- **plate:** The plate number, identifying the specific spectroscopic plate used to capture the object's light.
  Usage: It identifies which plate (in the SDSS's catalog) the object belongs to.

- **mjd (Modified Julian Date):** The date and time the observation was taken, expressed in Modified Julian Date.
  Usage: Knowing the observation date can help in temporal analysis (e.g., to check for trends over time or variations in the data).

- **fiberid:** The identifier for the fiber used to collect light from the object.
  Usage: Each observation is associated with a specific optical fiber, which helps track the light's path through the telescope's optical system.

## Tools and Techniques

**1. SVM (Support Vector Machine)** :
Description: A supervised machine learning model used for classification tasks. SVM aims to find the optimal hyperplane that maximizes the margin between classes (in this case, star, galaxy, and quasar).
Application: SVM will be used to identify the best separating boundary between the three classes based on the provided features (e.g., redshift, magnitude, spectral type).

**2. KNN (K-Nearest Neighbors):**
Description: A non-parametric classification method where the class of a sample is determined by the majority class among its nearest neighbors in the feature space.
Application: KNN will classify objects by analyzing the distance between their feature values (e.g., redshift, magnitude) and their nearest neighbors in the dataset.

**3. Logistic Regression :**
Description: A classification model that predicts the probability of an input belonging to a specific class. It can be extended to multi-class problems using Softmax.
Application: Softmax is used to handle multi-class classification (star, galaxy, or quasar). Cross-entropy loss is applied during training to minimize prediction errors.

**4. Decision Tree Classifier:**
Description: A model that splits the data into subsets based on the most significant features, forming a tree-like structure that makes predictions by following the decision rules.
Application: The decision tree will be used to classify astronomical objects by evaluating the most important features (like spectral type or redshift) and splitting the data accordingly.

**5. Random Forest Classifier:**
Description: A model that combines multiple decision trees, where each tree is trained on a random subset of the data and features. The final prediction is made by aggregating the results of all trees (majority vote for classification).
Application: The random forest will classify astronomical objects by leveraging the collective decision of multiple trees. It provides improved accuracy and robustness compared to a single decision tree and highlights the most important features (e.g., redshift, spectral type) for classification.

## Research Questions

• Can we classify astronomical objects (stars, galaxies, and quasars) based on their features, such as redshift, magnitude?
• Which combination of features provides the most accurate classification for distinguishing stars, galaxies, and quasars?
• Can we achieve a high classification accuracy using machine learning models on this dataset?
• Can PCA (Principal Component Analysis) improve classification accuracy?
• Which of the five machine learning algorithms Logistic Regression, Decision Tree, SVM, KNN and Random Forest performs best for classifying astronomical objects from the SDSS dataset?

## Exploratory Data Analysis (EDA)

### *Data Preprocessing and Cleaning*

First, Let's Examine the First Five Rows of the Data:

| objid | ra | dec | u | g | r | i |
|---|---|---|---|---|---|---|
| 1.237650e+18 | 183.531326 | 0.089693 | 19.47406 | 17.04240 | 15.94699 | 15.50342 |
| 1.237650e+18 | 183.598370 | 0.135285 | 18.66280 | 17.21449 | 16.67637 | 16.48922 |
| 1.237650e+18 | 183.680207 | 0.126185 | 19.38298 | 18.19169 | 17.47428 | 17.08732 |
| 1.237650e+18 | 183.870529 | 0.049911 | 17.76536 | 16.60272 | 16.16116 | 15.98233 |
| 1.237650e+18 | 183.883288 | 0.102557 | 17.55025 | 16.26342 | 16.43869 | 16.55492 |

| z | run | rerun | camcol | field | specobjid | class | redshift | plate |
|---|---|---|---|---|---|---|---|---|
| 15.22531 | 752 | 301 | 4 | 267 | 3.722360e+18 | STAR | -0.000009 | 3306 |
| 16.39150 | 752 | 301 | 4 | 267 | 3.638140e+17 | STAR | -0.000055 | 323 |
| 16.80125 | 752 | 301 | 4 | 268 | 3.232740e+17 | GALAXY | 0.123111 | 287 |
| 15.90438 | 752 | 301 | 4 | 269 | 3.722370e+18 | STAR | -0.000111 | 3306 |
| 16.61326 | 752 | 301 | 4 | 269 | 3.722370e+18 | STAR | 0.000590 | 3306 |

| mjd | fiberid |
|---|---|
| 54922 | 491 |
| 51615 | 541 |
| 52023 | 513 |
| 54922 | 510 |
| 54922 | 512 |

Checking for Missing Values and Numerical Data Types:

```
objid      10000 non-null  float64
ra         10000 non-null  float64
dec        10000 non-null  float64
u          10000 non-null  float64
g          10000 non-null  float64
r          10000 non-null  float64
i          10000 non-null  float64
z          10000 non-null  float64
run        10000 non-null  int64
rerun      10000 non-null  int64
camcol     10000 non-null  int64
field      10000 non-null  int64
specobjid  10000 non-null  float64
class      10000 non-null  object
redshift   10000 non-null  float64
plate      10000 non-null  int64
mjd        10000 non-null  int64
fiberid    10000 non-null  int64
```

There are no missing data values (10,000 non-null entries) and all features are numerical, except for the class (label), we'll deal with that later.

Unique values for each feature:

```
ra: 10000 unique values
dec: 10000 unique values
u: 9730 unique values
g: 9817 unique values
r: 9852 unique values
i: 9890 unique values
z: 9896 unique values
run: 23 unique values
rerun: 1 unique values
camcol: 6 unique values
field: 703 unique values
redshift: 9637 unique values
plate: 487 unique values
mjd: 355 unique values
fiberid: 892 unique values
specobjid: 6349 unique values
objid: 1 unique values
```

It can be observed that rerun and objid contain only a single unique value, making them non-contributory to the dataset. Therefore, we **remove** them from the dataset. Additionally, examining camcol and *run*, it appears that they are likely categorical rather than numerical variables. This is suggested by the fact that camcol has only 6 unique values and *run* has just 23 unique values out of 10,000 samples. We will further analyze this by inspecting their histograms later to determine whether they exhibit characteristics of categorical or numerical data.

Let's see the statistics of the features

| | ra | dec | u | g | r |
|---|---|---|---|---|---|
| count | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 |
| mean | 175.529987 | 14.836148 | 18.619355 | 17.371931 | 16.840963 |
| std | 47.783439 | 25.212207 | 0.828656 | 0.945457 | 1.067764 |
| min | 8.235100 | -5.382632 | 12.988970 | 12.799550 | 12.431600 |
| 25% | 157.370946 | -0.539035 | 18.178035 | 16.815100 | 16.173333 |
| 50% | 180.394514 | 0.404166 | 18.853095 | 17.495135 | 16.858770 |
| 75% | 201.547279 | 35.649397 | 19.259232 | 18.010145 | 17.512675 |
| max | 260.884382 | 68.542265 | 19.599900 | 19.918970 | 24.802040 |

| | i | z | run | camcol | field |
|---|---|---|---|---|---|
| count | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 |
| mean | 16.583579 | 16.422833 | 981.034800 | 3.648700 | 302.380100 |
| std | 1.141805 | 1.203188 | 273.305024 | 1.666183 | 162.577763 |
| min | 11.947210 | 11.610410 | 308.000000 | 1.000000 | 11.000000 |
| 25% | 15.853705 | 15.618285 | 752.000000 | 2.000000 | 184.000000 |
| 50% | 16.554985 | 16.389945 | 756.000000 | 4.000000 | 299.000000 |
| 75% | 17.258550 | 17.141447 | 1331.000000 | 5.000000 | 414.000000 |
| max | 28.179630 | 22.833060 | 1412.000000 | 6.000000 | 768.000000 |

| | specobjid | redshift | plate | mjd | fiberid |
|---|---|---|---|---|---|
| count | 1.000000e+04 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 |
| mean | 1.645022e+18 | 0.143726 | 1460.986400 | 52943.533300 | 353.069400 |
| std | 2.013998e+18 | 0.388774 | 1788.778371 | 1511.150651 | 206.298149 |
| min | 2.995780e+17 | -0.004136 | 266.000000 | 51578.000000 | 1.000000 |
| 25% | 3.389248e+17 | 0.000081 | 301.000000 | 51900.000000 | 186.750000 |
| 50% | 4.966580e+17 | 0.042591 | 441.000000 | 51997.000000 | 351.000000 |
| 75% | 2.881300e+18 | 0.092579 | 2559.000000 | 54468.000000 | 510.000000 |
| max | 9.468830e+18 | 5.353854 | 8410.000000 | 57481.000000 | 1000.000000 |

*Univariate & Bivariate Analysis*

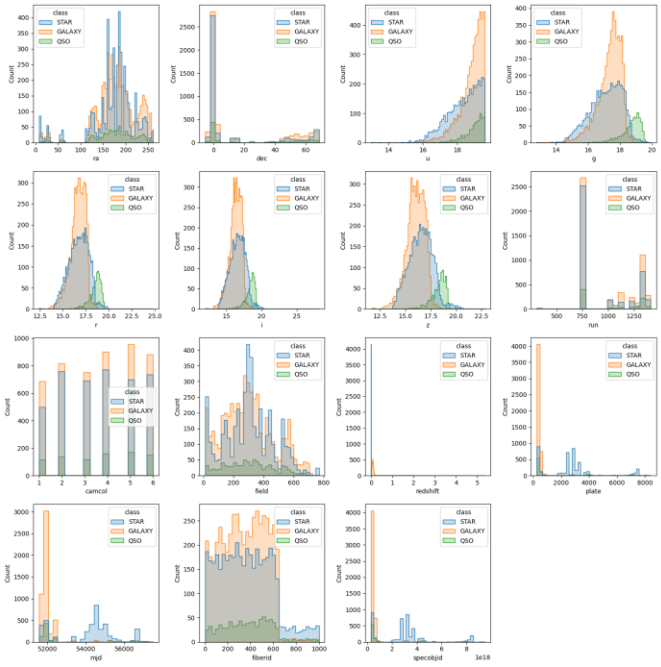



Fig. 1.Histograms - plots the distribution of a numeric variable's values as a series of bars

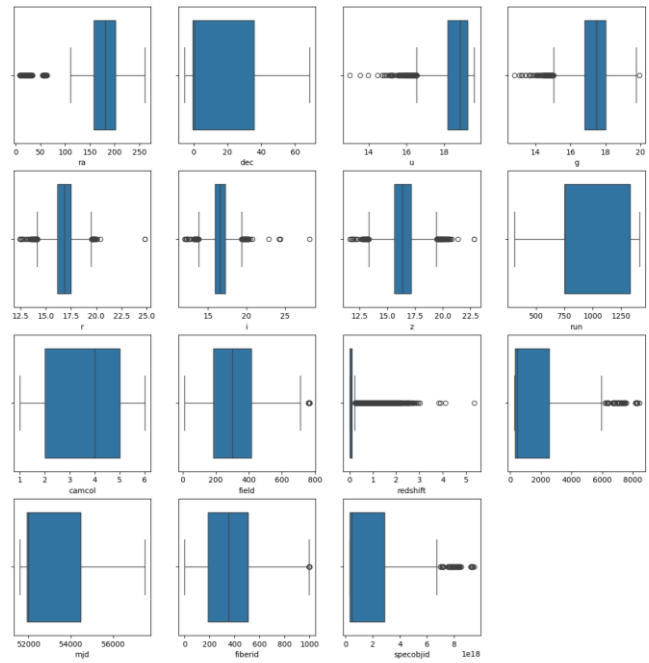



Fig. 2. Box plots - The data shows varying levels of dispersion across features.

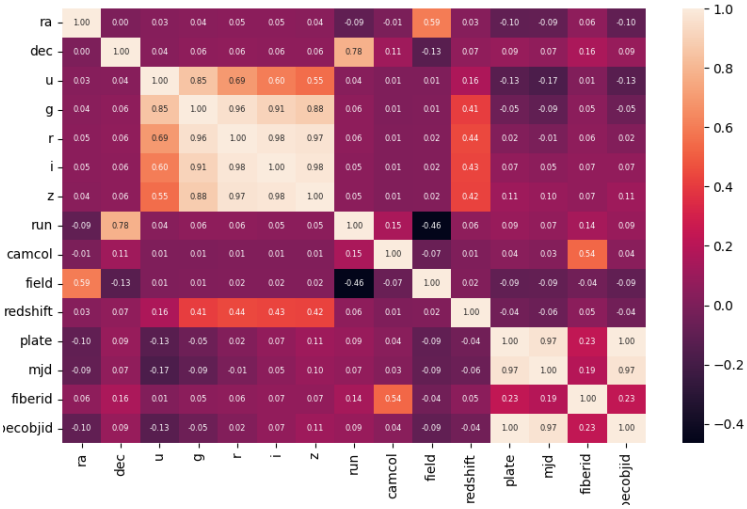



Fig. 3.Heatmap - represents a correlation matrix between various variables. Correlation measures the strength and direction of the linear relationship between two variables.

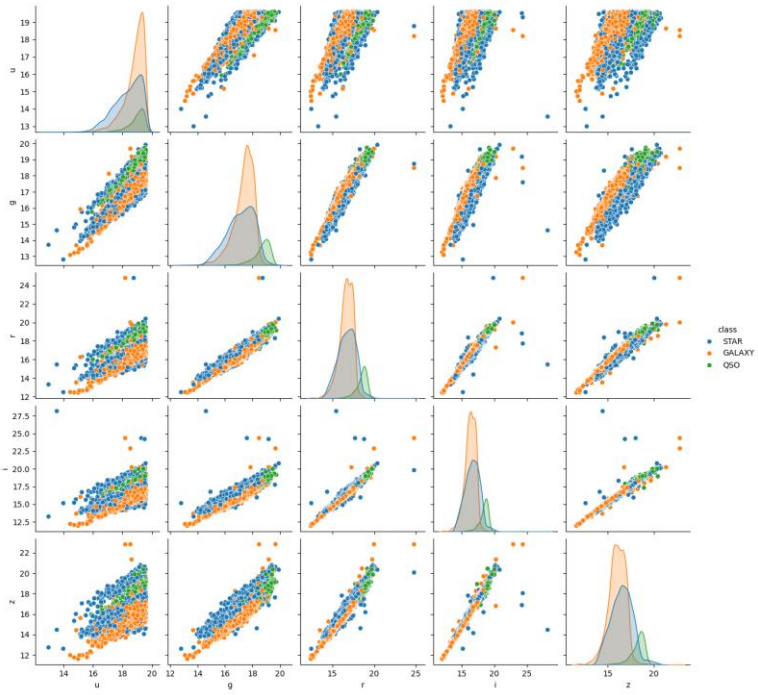



Fig. 4. Scatter Plots - Used to compare pairs of features (u, g, r, i, z).

*Conclusions of the analysis*

Fig .1 : Most numerical variables (u, g, r, i, z) exhibit a normal-like distribution, indicating that their values are centered around a mean. This is particularly useful when preprocessing, as features with normal distributions can benefit more from standardization or scaling. The other variables, like plate, fiberid, and mjd, show relatively uniform distributions, suggesting that these features might not have as much predictive power on their own. Additionally, the histograms reveal that *camcol* and *run* are categorical variables. This is evident from the clear gaps between the bars in their histograms, indicating a lack of continuity and confirming that they represent distinct categories rather than continuous numerical values. A similar pattern can be observed for plate, mjd, and specobjid.
For instance, specobjid contains close to 6,500 unique values. However, its range spans from approximately 2.995780e+17 to 9.468830e+18, which is an extremely large interval given the limited number of unique values. Despite this wide range, its histogram exhibits characteristics of a categorical variable rather than a continuous one. This suggests that specobjid may not be useful for learning, as categorical features with a high number of unique values (65% of the dataset in this case) provide little to no meaningful information for modeling. In fact, they can negatively impact learning by introducing unnecessary patterns, as each category would have an average of only 1.54 samples, making it ineffective for generalization. Therefore, specobjid is likely to be removed from the dataset.

Fig .2 : Features dispersion:
u, g, r, i, z: These exhibit relatively low dispersion. Their small IQR (Interquartile Range) indicates clustered data and low variability.

m, dec, run, field, plate, mjd, fiberid: These show higher dispersion. Their wider boxes reflect a larger range of values and greater variability.

redshift, specobjid: These contain numerous outliers. Points outside the box plot whiskers denote extreme values, which may require further scrutiny.

Fig .3: Correlations between features

Strong Positive Correlation:
Range: 0.7 to 1
High values of one variable are strongly associated with high values of the other. When one increases, the other tends to increase as well.

Moderate Positive Correlation:
Range: 0.3 to 0.7
There is a noticeable relationship between the variables, but it is not particularly strong. An increase in one variable is somewhat associated with an increase in the other.

Weak Positive Correlation:
Range: 0 to 0.3
There is a slight positive relationship, but it is weak and not consistent.

No Correlation:
Range: -0.3 to 0.3
There is no significant linear relationship between the variables. Changes in one variable do not consistently affect the other.

Weak Negative Correlation:
Range: -0.3 to 0
There is a slight inverse relationship, but it is weak.

Moderate Negative Correlation:
Range: -0.7 to -0.3
An increase in one variable is somewhat associated with a decrease in the other.

Strong Negative Correlation:

Range: -1 to -0.7
High values of one variable are strongly associated with low values of the other, and vice versa.

Strong positive correlations:

- There is a very strong correlation between (u, g), (g, r), (g, i), (g, z), (i, r), (i, z), and (r, z), indicating a strong linear relationship between these wavelengths.
- A strong correlation exists between plate and mjd, as well as between plate and specobjid, which is logical due to their technical interconnection. Our analysis indicates that mjd, plate, and specobjid are categorical rather than numerical variables. When categorical features exhibit a high correlation, such as the perfect correlation (1.0) between plate, and specobjid, it suggests that one is a deterministic function of the other, meaning that knowing the value of one fully determines the value of the other. This redundancy indicates that both features encode the same underlying information, providing little additional predictive power. Keeping both may introduce unnecessary complexity and multicollinearity, which can negatively impact the model. Therefore, one of these features should likely be removed, so we removed specobjid.

Strong Negative correlations:

- There are no strongly negative correlations between the features.

Weak or no correlations:

- Most other correlations are weak or non-existent, meaning there is no significant linear relationship between these variables.

The strong positive correlations suggest that these variables might carry redundant information. Techniques like PCA or feature selection could help reduce dimensionality while retaining the most relevant information.

Fig .4 : Analysis of Scatter Plots for u, g, r, i, z features:

As observed in the heatmap (Fig. 3), a strong positive correlation exists among the u, g, r, i, and z features. To further investigate these relationships, we generated scatter plots for each pair of these features.

The scatter plots reveal varying degrees of correlation strength and differences in data spread. For instance, the i vs. r plot demonstrates a high degree of correlation, as evidenced by tightly clustered points forming a clear linear pattern. This suggests that i and r values change together consistently, with relatively low dispersion around the trend line.

Conversely, the u vs. z plot exhibits a weaker correlation. The points are more widely dispersed, indicating that while there might be a general positive trend, the relationship is less pronounced and more variable.

By examining the scatter plots, we gain a deeper understanding of the relationships between the u, g, r, i, and z features. The strong correlations suggest that some of these features may carry redundant information, potentially leading to high dimensionality without adding significant discriminatory power. Applying dimensionality reduction techniques like PCA could help simplify the model while retaining most of the relevant information.

## *Prepare Data & Feature Engineering*

**Normalization**: Data normalization is an essential preprocessing step in machine learning that aims to standardize the range of independent variables or features of the dataset. In many cases, raw data may come in varying scales, and features with larger numeric ranges could dominate the learning process, leading to biased results or slower convergence during model training.

We will use the Standardization (Standard Scaling) method, this technique rescales the data to have a mean of 0 and a standard deviation of 1 by subtracting the mean and dividing by the standard deviation of each feature.

Let's look at the number of samples from each class:

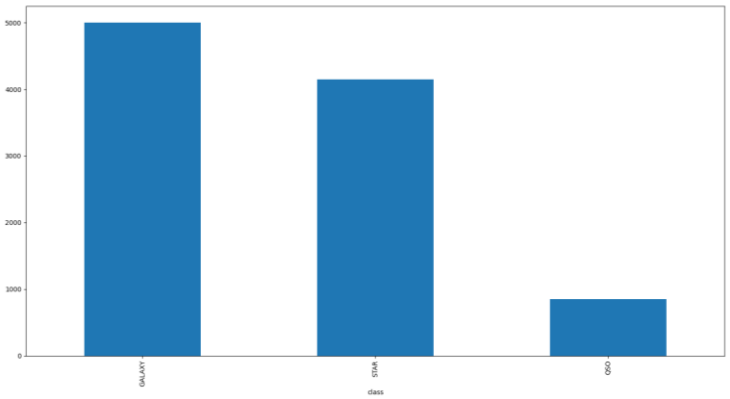



Fig. 5. Bar Plot of Class Distribution - This graph visually represents the number of samples in each class (**Galaxy**, **Star**, and **QSO**) using a bar chart.

Fig. 5: The dataset demonstrates a significant imbalance between the classes. Specifically, we have nearly 5,000 samples for the **Galaxy** class, slightly over 4,000 samples for the **Star** class, and fewer than 1,000 samples for the **QSO** class. We will first attempt to classify the data using the original distribution as is. If the classification results for the **QSO** class are unsatisfactory, we will consider applying one of the following techniques to address the class imbalance: Oversampling, Data Augmentation or Weighted Loss Function.

Split data to train, validation, test:
We will divide the data into training, validation and test so that the data is balanced.

We will convert the class values to numeric values:
STAR:0 ,GALAXY:1 ,QSO:2

## Model Training

Before we remove features with strong correlations and separate categorical and quantitative features, let's see the results of the models before all the changes.
For the models Logistic Regression, SVM, KNN we used **grid search**:
Grid search is used for hyperparameter tuning by exhaustively searching through a specified parameter grid to find the optimal combination that maximizes model performance.
For the models Decision Tree, Random Forest we used **bayes search**:
Bayes Search is a more efficient method for hyperparameter tuning, particularly when dealing with a large hyperparameter space. Instead of exhaustively searching all possibilities, Bayes Search uses probabilistic models to predict which combinations of hyperparameters are most likely to yield good results. This process reduces the number of iterations required and typically leads to better performance in less time.

We split the data into **60% training**, **20% validation** and **20% testing**, and during training we use cross validation into **five folds**.

### *SVM (Support Vector Machine)*

Best model from grid search :

Model Kernel – Linear:

The linear kernel is used when the data is approximately linearly separable. It finds a straight hyperplane that best separates the classes, making it computationally efficient and interpretable.

Regularization Parameter – C = 10:
The regularization parameter **C** controls the trade-off between maximizing the margin and minimizing classification errors. A higher value (C=10) prioritizes correct classification over a wider margin, making the model more sensitive to training data but potentially increasing the risk of overfitting.

Result :

```
Train Result:
Accuracy: 0.988125

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      1.00      0.99      2667
      GALAXY       0.99      0.98      0.99      3181
         QSO       0.98      0.97      0.97       552

    accuracy                           0.99      6400
   macro avg       0.99      0.98      0.98      6400
weighted avg       0.99      0.99      0.99      6400


Confusion Matrix
 [[2667    0    0]
 [  47 3123   11]
 [   1   17  534]]
Validation Result:
Accuracy: 0.99

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      1.00      0.99       655
      GALAXY       0.99      0.99      0.99       817
         QSO       0.98      0.95      0.96       128

    accuracy                           0.99      1600
   macro avg       0.99      0.98      0.98      1600
weighted avg       0.99      0.99      0.99      1600


Confusion Matrix
 [[655   0   0]
 [  7 808   2]
 [  0   7 121]]
```

```
Test Result:
Accuracy: 0.9855

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      1.00      0.99       830
      GALAXY       0.99      0.98      0.99      1000
         QSO       0.96      0.97      0.97       170

    accuracy                           0.99      2000
   macro avg       0.98      0.98      0.98      2000
weighted avg       0.99      0.99      0.99      2000


Confusion Matrix
 [[830   0   0]
 [ 18 976   6]
 [  0   5 165]]
```

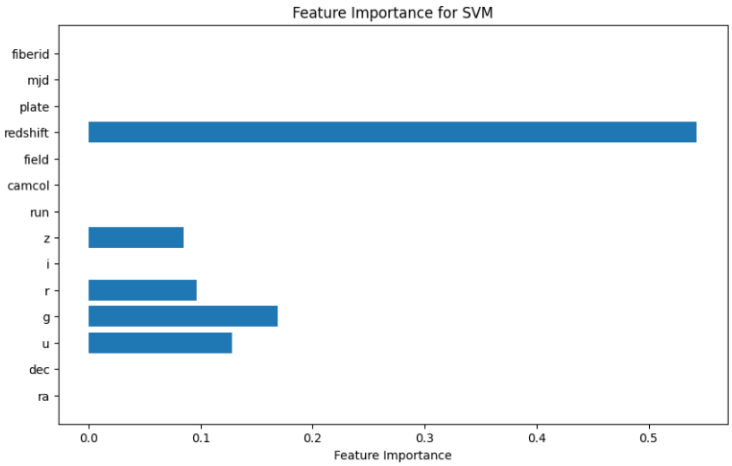



Fig. 6. The importance features in SVM classification

SVM Model Performance Analysis:

The SVM model delivers outstanding classification results with high accuracy across all datasets.

Key Observations:

- STAR class achieves perfect recall, meaning all STAR instances were correctly identified across all datasets.
- GALAXY and QSO maintain strong classification performance, with QSO achieving a 96% precision in the test set, indicating well-separated feature distributions.
- Misclassifications are minimal, primarily between GALAXY and QSO, as shown in the confusion matrices.
- Consistent accuracy across training, validation, and test sets suggests minimal overfitting, indicating that the model generalizes well to unseen data.
- Validation performance aligns closely with training and test results, further reinforcing the model's robustness.

Overall, the SVM model effectively distinguishes between classes, demonstrating clear decision boundaries in the feature space.

*KNN (K-Nearest Neighbors)*

Best model from grid search :

Number of nearest neighbors = 3:
This parameter specifies how many of the closest data points (neighbors) are considered when making predictions. A lower value (e.g. 3) means that the

classifier will be more sensitive to local patterns in the data, while higher values might smooth out decisions but could lose local detail.

P_ Distance = 1:
The p parameter defines the distance metric used to calculate the proximity between points. When p = 1, the model uses the **Manhattan distance** (also known as L1 distance), which sums the absolute differences between the coordinates of the points.

Result :

```
Train Result:
Accuracy: 0.9525

Classification Report:
              precision    recall  f1-score   support

        STAR       0.96      0.94      0.95      2667
      GALAXY       0.94      0.97      0.96      3181
         QSO       0.98      0.90      0.94       552

    accuracy                           0.95      6400
   macro avg       0.96      0.94      0.95      6400
weighted avg       0.95      0.95      0.95      6400


Confusion Matrix
 [[2517  148    2]
 [  91 3084    6]
 [  23   34  495]]
```

```
Validation Result:
Accuracy: 0.91

Classification Report:
              precision    recall  f1-score   support

        STAR       0.92      0.88      0.90       655
      GALAXY       0.89      0.95      0.92       817
         QSO       0.96      0.82      0.89       128

    accuracy                           0.91      1600
   macro avg       0.93      0.88      0.90      1600
weighted avg       0.91      0.91      0.91      1600


Confusion Matrix
 [[574  79   2]
 [ 38 777   2]
 [  9  14 105]]
```

```
Test Result:
Accuracy: 0.9085

Classification Report:
              precision    recall  f1-score   support

        STAR       0.92      0.88      0.90       830
      GALAXY       0.89      0.95      0.92      1000
         QSO       0.95      0.83      0.88       170

    accuracy                           0.91      2000
   macro avg       0.92      0.88      0.90      2000
weighted avg       0.91      0.91      0.91      2000


Confusion Matrix
 [[729  97   4]
 [ 49 947   4]
 [ 13  16 141]]
```

KNN Model Performance Analysis:

The KNN model shows solid performance but exhibits signs of overfitting.

Key Observations:

- STAR class recall is lower (87% in validation, 88% in test), suggesting that some STAR instances are being misclassified.
- QSO class exhibits the most variance, with recall dropping from 90% (train) to 82% (validation & test), indicating instability in classification.
- Confusion matrices show misclassifications primarily between STAR and GALAXY.
- A notable drop in accuracy from training to validation/test indicates overfitting, likely caused by an overly specific model fitting the training set too closely.

Potential Overfitting Explanation:

- The KNN model was optimized using Grid Search, which selected K=3 as the best parameter.
- With only 3 neighbors, the decision boundary is highly sensitive to local variations and noise in the training data.
- This can lead to overfitting, as the model may classify based on very few data points, making it less generalizable to unseen samples.
- A higher K value (e.g., 5 or 7) could help reduce sensitivity to noise and improve generalization.

Overall, while KNN performs well, the choice of K=3 suggests excessive reliance on individual data points, leading to reduced stability in validation and test performance. Therefore, grid search will not always give us the best results because it looks for the best results for the training data and therefore it can cause overfitting.

Let's change to K=7 to try reduce sensitivity to noise:

```
Train Result:
Accuracy: 0.929375

Classification Report:
              precision    recall  f1-score   support

        STAR       0.94      0.91      0.92      2667
      GALAXY       0.91      0.96      0.94      3181
         QSO       0.98      0.85      0.91       552

    accuracy                           0.93      6400
   macro avg       0.95      0.91      0.92      6400
weighted avg       0.93      0.93      0.93      6400


Confusion Matrix
 [[2416  249    2]
 [ 110 3065    6]
 [  33   52  467]]
Validation Result:
Accuracy: 0.900625

Classification Report:
              precision    recall  f1-score   support

        STAR       0.92      0.86      0.89       655
      GALAXY       0.87      0.95      0.91       817
         QSO       0.98      0.80      0.88       128

    accuracy                           0.90      1600
   macro avg       0.93      0.87      0.89      1600
weighted avg       0.90      0.90      0.90      1600


Confusion Matrix
 [[563  92   0]
 [ 39 776   2]
 [  7  19 102]]
```

```
Test Result:
Accuracy: 0.9055

Classification Report:
              precision    recall  f1-score   support

        STAR       0.93      0.86      0.90       830
      GALAXY       0.88      0.96      0.92      1000
         QSO       0.95      0.80      0.87       170

    accuracy                           0.91      2000
   macro avg       0.92      0.87      0.89      2000
weighted avg       0.91      0.91      0.90      2000


Confusion Matrix
 [[717 110   3]
 [ 38 958   4]
 [ 14  20 136]]
```

Key Observations:

- The KNN model with K=7 shows strong performance in training, achieving an accuracy of **92.94%**.
  The **STAR** class has a precision of **0.94** and recall of **0.91**, which are fairly good, but there's still a slight drop in recall compared to other classes.
  **QSO** class has a high precision of **0.98** but a lower recall of **0.85**, which indicates that while the model is good at identifying QSO instances when it does, it misses some.

- The model experiences a slight drop in performance on the validation set, with an accuracy of **90.06%**.
  The **STAR** class has the largest recall drop, from **91%** on training to **86%** on validation, meaning some **STAR** instances are being misclassified.
  The **QSO** class also shows a drop in recall (from **85%** to **80%**), which indicates that the model is struggling a bit more to generalize for this class.

- The model performance on the test set is very similar to the validation set, with an accuracy of **90.55%**.
  **STAR** continues to have a slightly lower recall, suggesting that the model is still struggling to correctly classify all instances of this class.
  The **QSO** class still exhibits a notable drop in recall when compared to the training set, confirming that the model is less stable for this class.

Potential Overfitting Explanation:

Despite a decrease in performance from training to validation/test sets, the drop is less severe than when K=3, which suggests that the choice of K=7 helped reduce sensitivity to noise.

However, there is still **mild overfitting** present. The recall for the **QSO** class drops from **85%** in training to **80%** in both validation and test, which shows the model is still a little too specific to the training data.

**STAR** class recall also shows a slight drop (91% to 86%), indicating that the model could still benefit from further generalization.

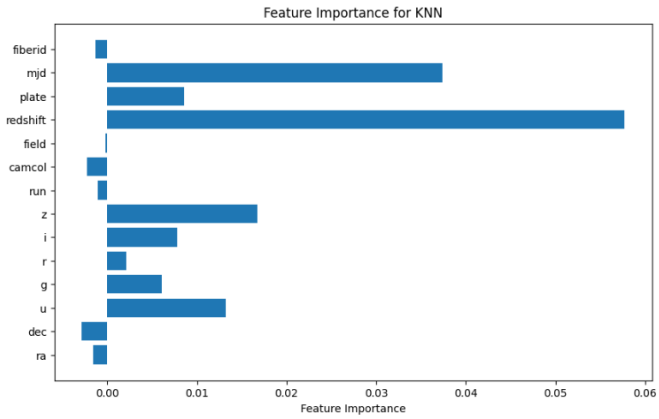



Fig. 7. The importance features in KNN classification

## *Logistic Regression*

Best model from grid search :

Regularization Type – Lasso:
Lasso (L1 Regularization) adds a penalty based on the absolute values of the coefficients. It can set some coefficients to zero, effectively removing those features from the model. This helps with feature selection and prevents overfitting by making the model simpler.

Regularization Parameter – C = 100:
C in Lasso regularization controls the strength of the penalty. A larger C means less regularization (the model allows larger coefficients), while a smaller C increases the penalty, encouraging more coefficients to shrunk to zero, leading to a simpler model with fewer features. (C = 100 means coefficient = 1/100)

Result :

```
Train Result:
Accuracy: 0.98453125

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      1.00      0.99      2667
      GALAXY       0.98      0.99      0.98      3181
         QSO       0.97      0.92      0.95       552

    accuracy                           0.98      6400
   macro avg       0.98      0.97      0.97      6400
weighted avg       0.98      0.98      0.98      6400


Confusion Matrix
 [[2655   12    0]
 [  29 3137   15]
 [   1   42  509]]
```

```
Validation Result:
Accuracy: 0.984375

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      0.99      0.99       655
      GALAXY       0.98      0.99      0.98       817
         QSO       0.97      0.91      0.94       128

    accuracy                           0.98      1600
   macro avg       0.98      0.97      0.97      1600
weighted avg       0.98      0.98      0.98      1600


Confusion Matrix
 [[649   6   0]
 [  4 809   4]
 [  0  11 117]]

Test Result:
Accuracy: 0.9845

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      1.00      0.99       830
      GALAXY       0.99      0.98      0.98      1000
         QSO       0.96      0.94      0.95       170

    accuracy                           0.98      2000
   macro avg       0.98      0.97      0.98      2000
weighted avg       0.98      0.98      0.98      2000


Confusion Matrix
 [[827   3   0]
 [ 12 982   6]
 [  0  10 160]]
```

Logistic Regression Model Performance Analysis:

The Logistic Regression model also demonstrates strong classification ability.

Key Observations:

- All classes achieve high precision, recall, and F1-scores, suggesting strong decision boundaries.
- QSO recall is slightly lower (92% in train, 91% in validation, 94% in test), indicating some difficulty distinguishing QSO samples from other classes.
- The confusion matrices show misclassifications mainly between GALAXY and QSO.
- Validation results closely match training and test results, suggesting excellent generalization.

Overall, the Logistic Regression model provides a strong balance between performance and generalization.

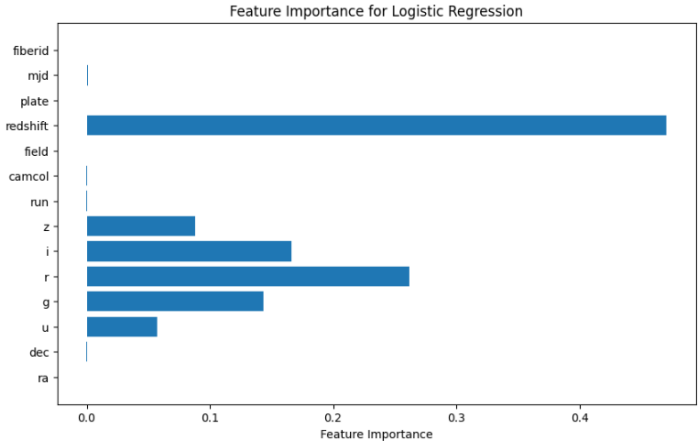



Fig. 8. The importance features in Logistic Regression

*Decision Tree*

Best model from bayes search :

Criterion – Entropy:
In a Decision Tree algorithm, the criterion defines the method for splitting nodes. Using entropy as the criterion means the tree will select splits based on information gain, aiming to reduce disorder in the dataset. A split that maximizes entropy reduction leads to more pure groups, improving the accuracy of the model by ensuring that the data within each branch is as homogeneous as possible.

Max Depth – 4:
The max depth parameter controls how deep the tree can grow. A depth of 4 prevents the tree from becoming too complex and overfitting the data. Limiting the depth ensures the tree generalizes well by capturing essential patterns without fitting noise. This makes the model simpler and more interpretable while avoiding overfitting to the training data.

Min Samples Leaf – 6:
The min samples leaf parameter ensures that each leaf node in the tree contains at least 6 samples. This requirement helps the model avoid creating overly specific branches that may fit only a small portion of the training data, thus reducing the risk of overfitting. It forces the tree to make splits that are meaningful across a larger portion of the data, leading to better generalization.

Result :

```
Train Result:
Accuracy: 0.99125

Classification Report:
              precision    recall  f1-score   support

        STAR       1.00      1.00      1.00      2667
      GALAXY       0.99      1.00      0.99      3181
         QSO       0.98      0.93      0.95       552

    accuracy                           0.99      6400
   macro avg       0.99      0.97      0.98      6400
weighted avg       0.99      0.99      0.99      6400


Confusion Matrix
 [[2664    3    0]
 [   3 3168   10]
 [   1   39  512]]
Validation Result:
Accuracy: 0.99125

Classification Report:
              precision    recall  f1-score   support

        STAR       1.00      1.00      1.00       655
      GALAXY       0.99      1.00      0.99       817
         QSO       0.97      0.91      0.94       128

    accuracy                           0.99      1600
   macro avg       0.99      0.97      0.98      1600
weighted avg       0.99      0.99      0.99      1600


Confusion Matrix
 [[655   0   0]
 [  0 814   3]
 [  0  11 117]]

Test Result:
Accuracy: 0.9855

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      1.00      0.99       830
      GALAXY       0.99      0.98      0.99      1000
         QSO       0.96      0.92      0.94       170

    accuracy                           0.99      2000
   macro avg       0.98      0.97      0.97      2000
weighted avg       0.99      0.99      0.99      2000


Confusion Matrix
 [[830   0   0]
 [  9 984   7]
 [  0  13 157]]
```

Key Observations:

- High accuracy across all sets (Train: 99.1%, Validation: 99.1%, Test: 98.55%), indicating strong classification performance.
- Perfect or near-perfect precision for STAR and GALAXY, with minor misclassifications in QSO, especially in the test set where QSO recall drops to 92%.
- QSO misclassification primarily occurs with GALAXY, as seen in the confusion matrices, suggesting some overlap in feature space.
- Minimal overfitting, despite slightly lower recall for QSO in validation and test sets compared to training.
- Model generalizes well, maintaining consistency across different datasets.

Overall, the Decision Tree classifier performs well with strong accuracy and minimal overfitting. However, it struggles slightly with distinguishing QSO from GALAXY, leading to a small drop in recall for QSO. While effective, it may benefit from additional feature engineering or ensemble methods to further improve generalization.

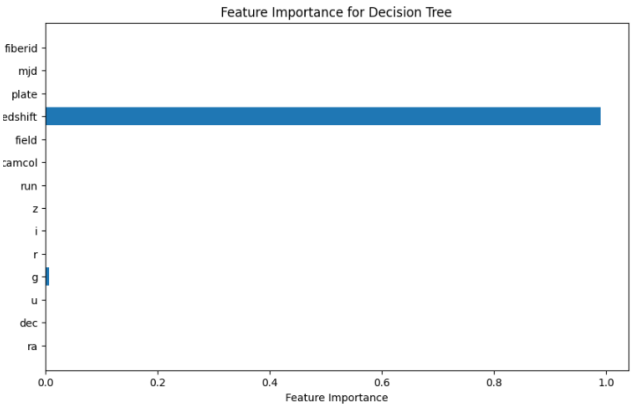



Fig. 9. The importance features in Decision Tree

*Random Forest*

Best model from bayes search :

Criterion – Entropy:
Like we explain in Decision Tree.

Max Depth – 14:
Like we explain in Decision Tree, limits the depth of the trees to prevent excessive complexity and overfitting while ensuring the model captures key patterns in the data.

Number of Estimators – 200:
Uses 200 decision trees in the ensemble, enhancing stability and predictive performance by reducing variance and improving generalization.

Result:

```
Train Result:
Accuracy: 0.99890625

Classification Report:
              precision    recall  f1-score   support

        STAR       1.00      1.00      1.00      2667
      GALAXY       1.00      1.00      1.00      3181
         QSO       1.00      0.99      0.99       552

    accuracy                           1.00      6400
   macro avg       1.00      1.00      1.00      6400
weighted avg       1.00      1.00      1.00      6400


Confusion Matrix
 [[2667    0    0]
 [   0 3181    0]
 [   0    7  545]]
Validation Result:
Accuracy: 0.99

Classification Report:
              precision    recall  f1-score   support

        STAR       1.00      1.00      1.00       655
      GALAXY       0.99      0.99      0.99       817
         QSO       0.98      0.92      0.95       128

    accuracy                           0.99      1600
   macro avg       0.99      0.97      0.98      1600
weighted avg       0.99      0.99      0.99      1600


Confusion Matrix
 [[655   0   0]
 [  3 811   3]
 [  0  10 118]]

Test Result:
Accuracy: 0.9875

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      1.00      0.99       830
      GALAXY       0.99      0.98      0.99      1000
         QSO       0.95      0.96      0.95       170

    accuracy                           0.99      2000
   macro avg       0.98      0.98      0.98      2000
weighted avg       0.99      0.99      0.99      2000


Confusion Matrix
 [[830   0   0]
 [  9 982   9]
 [  0   7 163]]
```

Key Observations:

- Exceptional training accuracy (99.89%) suggests the model almost perfectly learns the training data.
- QSO recall in training is slightly below 100%, indicating a small number of misclassifications.
- Validation and test accuracy remain consistently high (~99%), showing strong generalization.
- QSO misclassification persists in validation and test sets, similar to the Decision Tree model, but with slightly better performance.
- Slight overfitting risk, as training accuracy is extremely high compared to validation/test performance.

Overall, Random Forest delivers outstanding accuracy and robustness across all datasets. The model handles STAR and GALAXY classes exceptionally well but has minor difficulties with QSO. While the risk of overfitting is low, the slight discrepancy between training and test performance suggests that further tuning or feature selection could improve QSO classification.

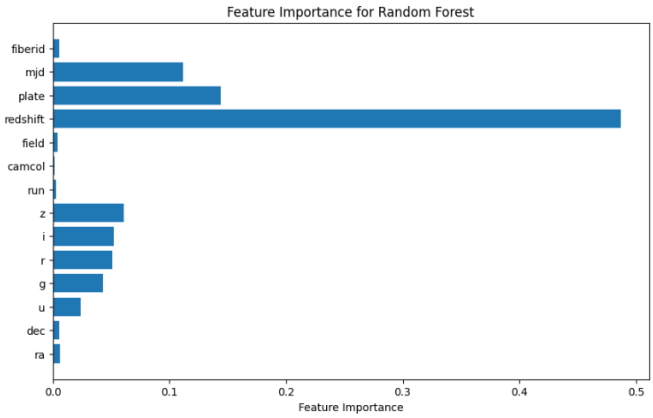



Fig. 10. The importance features in Random Forest

## *Conclusions from the results*

Fig .6 : Only **redshift, z, r, g, u** are important, as removing any of them reduces accuracy. Since the SVM with kernel=linear effectively separates classes with minimal errors, these features likely provide the most discriminative information. All other features neither improve nor degrade accuracy and do not contribute to the model.

Fig .7 : The features **redshift, z, r, g, u, i, mjd, plate** are essential, as removing any of them decreases accuracy. All other features negatively impact performance—

removing them actually improves accuracy, indicating they introduce noise rather than valuable information.

Fig .8 : The features **redshift, z, r, g, u, i** significantly contribute to accuracy. Other features have either a negligible effect or no contribution at all. (LASSO was used to enhance feature selection.)

Fig .9 : Only **redshift** is strongly important, with a very minor contribution from **g**. Since the tree splits based on entropy reduction, it likely prioritizes **redshift** as the most informative feature, while the rest provide little to no added value.

Fig .10 : All features contribute to the model, but **mjd, plate, i, u, z, r, g, redshift** are the most influential, as removing them leads to a notable drop in accuracy. The remaining features still contribute, but only marginally. The ensemble nature of Random Forest allows it to capture complex patterns, utilizing all available features to improve overall performance.

## *PCA - Principal Component Analysis*

PCA is a dimensionality reduction technique used to transform high dimensional data into a lower dimensional space while preserving as much variance as possible. The main steps of PCA are:

**Standardization**: The data is centered and scaled to have a mean of zero and unit variance.
**Covariance Matrix Computation**: The relationships between features are captured in a covariance matrix.
**Eigen Decomposition**: The eigenvalues and eigenvectors of the covariance matrix are computed. Eigenvalues represent the amount of variance explained by each eigenvector, and the eigenvectors define new feature directions.
**Selection of Principal Components**: The eigenvectors corresponding to the highest eigenvalues are selected as principal components. These components capture the most significant variance in the dataset.
**Projection**: The original data is projected onto the new feature space defined by these principal components.

For this analysis, **9** principal components were selected, capturing 98.69% of the total variance:
**Explained Variance Ratio:** [0.3313, 0.1707, 0.1308, 0.1115, 0.0878, 0.0629, 0.0323, 0.0310, 0.0288]
**Final Variance Ratio:** 0.9869

## *SVM – With PCA :*

Result :

```
Train Result With PCA:
Accuracy: 0.9834375

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      1.00      0.99      2667
      GALAXY       0.99      0.98      0.98      3181
         QSO       0.97      0.93      0.95       552

    accuracy                           0.98      6400
   macro avg       0.98      0.97      0.98      6400
weighted avg       0.98      0.98      0.98      6400


Confusion Matrix
 [[2659    8    0]
 [  47 3120   14]
 [   1   36  515]]
Validation Result With PCA:
Accuracy: 0.985625

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      1.00      0.99       655
      GALAXY       0.99      0.99      0.99       817
         QSO       0.98      0.92      0.95       128

    accuracy                           0.99      1600
   macro avg       0.98      0.97      0.98      1600
weighted avg       0.99      0.99      0.99      1600


Confusion Matrix
 [[654   1   0]
 [  9 805   3]
 [  0  10 118]]


Test Result With PCA:
Accuracy: 0.9815

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      1.00      0.99       830
      GALAXY       0.99      0.97      0.98      1000
         QSO       0.95      0.95      0.95       170

    accuracy                           0.98      2000
   macro avg       0.97      0.97      0.97      2000
weighted avg       0.98      0.98      0.98      2000


Confusion Matrix
 [[828   2   0]
 [ 18 974   8]
 [  0   9 161]]
```

Key Observations:
SVM remains robust across all datasets, indicating that PCA successfully retained class-separating information. Minimal misclassifications suggest that PCA did not negatively affect SVM performance.

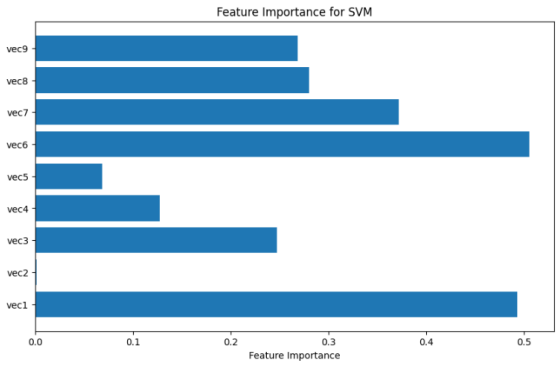



Fig.11. Important eigenvectors in SVM

*KNN – With PCA :*

Result:

```
Train Result With PCA:
Accuracy: 0.943125

Classification Report:
              precision    recall  f1-score   support

        STAR       0.95      0.93      0.94      2667
      GALAXY       0.93      0.96      0.95      3181
         QSO       0.99      0.90      0.94       552

    accuracy                           0.94      6400
   macro avg       0.96      0.93      0.94      6400
weighted avg       0.94      0.94      0.94      6400


Confusion Matrix
 [[2479  187    1]
 [ 115 3061    5]
 [  27   29  496]]
Validation Result With PCA:
Accuracy: 0.8925

Classification Report:
              precision    recall  f1-score   support

        STAR       0.90      0.85      0.87       655
      GALAXY       0.87      0.93      0.90       817
         QSO       0.97      0.86      0.91       128

    accuracy                           0.89      1600
   macro avg       0.92      0.88      0.90      1600
weighted avg       0.89      0.89      0.89      1600


Confusion Matrix
 [[555  98   2]
 [ 53 763   1]
 [  6  12 110]]
```

```
Test Result With PCA:
Accuracy: 0.8935

Classification Report:
              precision    recall  f1-score   support

        STAR       0.90      0.85      0.88       830
      GALAXY       0.88      0.93      0.91      1000
         QSO       0.95      0.86      0.90       170

    accuracy                           0.89      2000
   macro avg       0.91      0.88      0.90      2000
weighted avg       0.89      0.89      0.89      2000


Confusion Matrix
 [[706 118   6]
 [ 64 934   2]
 [ 11  12 147]]
```

Key Observations:
PCA negatively impacted KNN, reducing accuracy. Since KNN relies on distance metrics, the transformed feature space can distort its effectiveness, and we see clearly overfitting (94.3% vs. 89.3%)

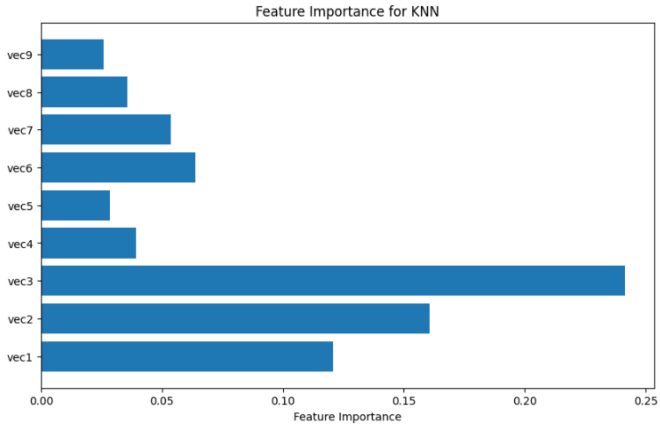



Fig.12. Important eigenvectors in KNN

*Logistic Regression – With PCA :*

Result:

```
Train Result With PCA:
Accuracy: 0.9809375

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      0.99      0.99      2667
      GALAXY       0.98      0.98      0.98      3181
         QSO       0.96      0.93      0.95       552

    accuracy                           0.98      6400
   macro avg       0.98      0.97      0.97      6400
weighted avg       0.98      0.98      0.98      6400


Confusion Matrix
 [[2650   17    0]
 [  47 3113   21]
 [   1   36  515]]
```

```
Validation Result With PCA:
Accuracy: 0.981875

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      0.99      0.99       655
      GALAXY       0.98      0.99      0.98       817
         QSO       0.94      0.91      0.92       128

    accuracy                           0.98      1600
   macro avg       0.97      0.96      0.97      1600
weighted avg       0.98      0.98      0.98      1600


Confusion Matrix
 [[650   4   1]
 [  5 805   7]
 [  0  12 116]]
Test Result With PCA:
Accuracy: 0.98

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      1.00      0.99       830
      GALAXY       0.99      0.97      0.98      1000
         QSO       0.94      0.94      0.94       170

    accuracy                           0.98      2000
   macro avg       0.97      0.97      0.97      2000
weighted avg       0.98      0.98      0.98      2000


Confusion Matrix
 [[828   2   0]
 [ 17 972  11]
 [  0  10 160]]
```

Key Observations:
Logistic Regression remained highly effective after PCA, suggesting that PCA preserved meaningful variance for this model. Although, like in SVM, the accuracy decreased by less than a percent, the model is much simpler, with fewer dimensions, and still maintains high accuracy.

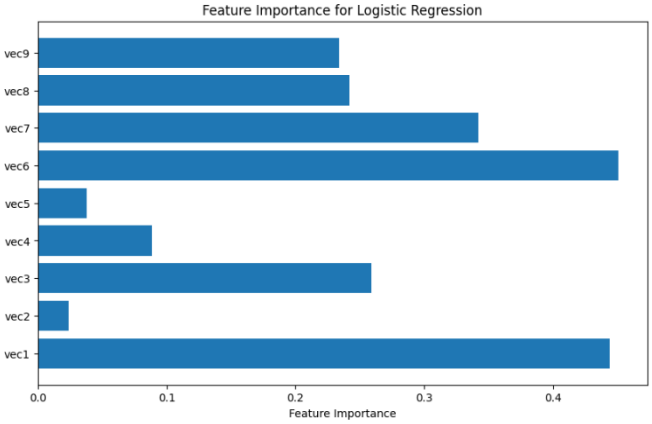



Fig.13. Important eigenvectors in Logistic Regression

*Decision Tree – With PCA*

Result :

```
Train Result With PCA:
Accuracy: 0.93078125

Classification Report:
              precision    recall  f1-score   support

        STAR       0.93      0.91      0.92      2667
      GALAXY       0.92      0.96      0.94      3181
         QSO       0.99      0.86      0.92       552

    accuracy                           0.93      6400
   macro avg       0.95      0.91      0.93      6400
weighted avg       0.93      0.93      0.93      6400


Confusion Matrix
 [[2438  227    2]
 [ 137 3043    1]
 [  35   41  476]]
Validation Result With PCA:
Accuracy: 0.8875

Classification Report:
              precision    recall  f1-score   support

        STAR       0.89      0.86      0.87       655
      GALAXY       0.87      0.93      0.90       817
         QSO       0.98      0.80      0.88       128

    accuracy                           0.89      1600
   macro avg       0.92      0.86      0.88      1600
weighted avg       0.89      0.89      0.89      1600


Confusion Matrix
 [[562  93   0]
 [ 59 756   2]
 [ 10  16 102]]

Test Result With PCA:
Accuracy: 0.8865

Classification Report:
              precision    recall  f1-score   support

        STAR       0.89      0.86      0.87       830
      GALAXY       0.88      0.92      0.90      1000
         QSO       0.97      0.82      0.89       170

    accuracy                           0.89      2000
   macro avg       0.91      0.87      0.89      2000
weighted avg       0.89      0.89      0.89      2000


Confusion Matrix
 [[713 116   1]
 [ 76 920   4]
 [ 15  15 140]]
```

Key Observations:
The gap between training and test accuracy (93.0% vs. 88.5%) suggests mild overfitting. The model generalizes fairly well but slightly memorizes training patterns.
The model accuracy dropped from 99% to 93, so PCA is not good for this model.

Of course, these results are not surprising at all, Decision Trees struggled with PCA due to the transformation reducing interpretability of features crucial for tree splits.

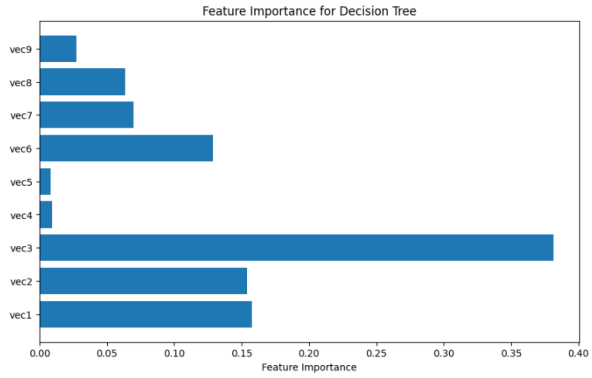



Fig.14. Important eigenvectors in Decision Tree

*Random Forest – With PCA*

Result :

```
Train Result With PCA:
Accuracy: 0.99640625

Classification Report:
              precision    recall  f1-score   support

        STAR       1.00      0.99      1.00      2667
      GALAXY       0.99      1.00      1.00      3181
         QSO       1.00      0.99      0.99       552

    accuracy                           1.00      6400
   macro avg       1.00      0.99      1.00      6400
weighted avg       1.00      1.00      1.00      6400


Confusion Matrix
 [[2651   16    0]
 [   0 3181    0]
 [   0    7  545]]
Validation Result With PCA:
Accuracy: 0.918125

Classification Report:
              precision    recall  f1-score   support

        STAR       0.92      0.90      0.91       655
      GALAXY       0.91      0.94      0.93       817
         QSO       0.97      0.88      0.92       128

    accuracy                           0.92      1600
   macro avg       0.93      0.90      0.92      1600
weighted avg       0.92      0.92      0.92      1600


Confusion Matrix
 [[588  66   1]
 [ 46 769   2]
 [  7   9 112]]
```

```
Test Result With PCA:
Accuracy: 0.914

Classification Report:
              precision    recall  f1-score   support

        STAR       0.91      0.89      0.90       830
      GALAXY       0.91      0.93      0.92      1000
         QSO       0.98      0.91      0.95       170

    accuracy                           0.91      2000
   macro avg       0.93      0.91      0.92      2000
weighted avg       0.91      0.91      0.91      2000


Confusion Matrix
 [[742  88   0]
 [ 66 931   3]
 [  8   7 155]]
```

Key Observations:
The extremely high training accuracy (99.64%) compared to test accuracy (91.4%) suggests clear overfitting. The model performs exceptionally on training data but struggles slightly on unseen data, particularly with QSO classification.
PCA removes original feature interpretability, affecting tree-based models like Random Forest, which rely on raw feature importance for decision-making.

Again, like we explain before on decision tree, these results are not surprising at all, Random Forest build from decision trees, so this model struggled with PCA due to the transformation reducing interpretability of features crucial for tree splits.

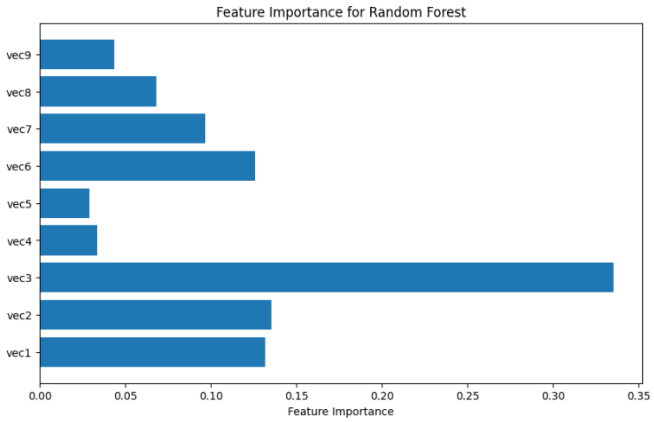
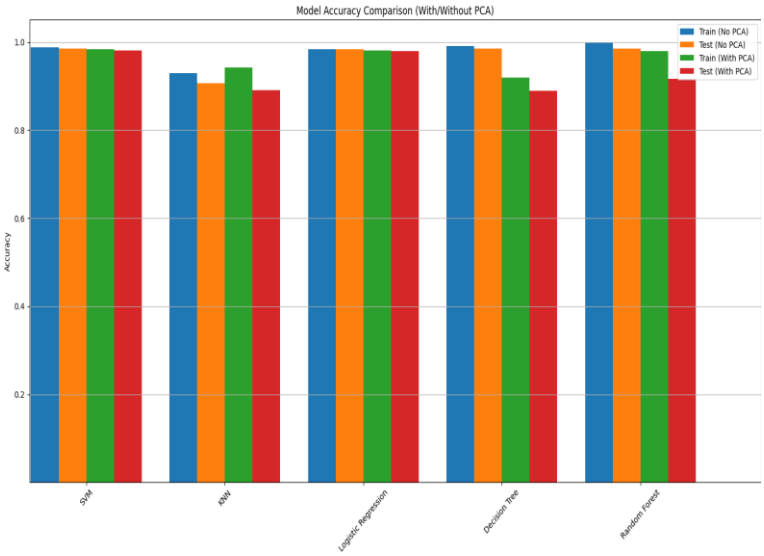



Fig.15. Important eigenvectors in Random Forest

## *Conclusions from the results*

SVM and Logistic Regression maintained high accuracy across train, validation, and test sets.
Random Forest had the highest training accuracy but suffered a test accuracy drop, suggesting some memorization of the training data.
PCA significantly improved the performance of SVM and Logistic Regression while negatively impacting Decision Trees, Random Forests, and KNN. This highlights the importance of selecting dimensionality reduction techniques based on model characteristics. For tree-based models, feature selection techniques may be preferable over PCA.
Like we see in fig.11 to fig.15, all the eigenvectors important, unless eigenvector – vec2 in SVM model.




**Best Model: Random Forest (with or without PCA)**
The best performing model is Random Forest based on its high accuracy and strong generalization.

- **Train Accuracy**: 99.89%
- **Validation Accuracy**: 99.00%
- **Test Accuracy**: 98.75%

Random Forest outperforms other models by maintaining high accuracy across all sets with minimal overfitting.
The classification metrics show high precision and recall across all classes, making Random Forest the most reliable model for this task.

**Best Model with PCA**
After applying PCA, the best model is SVM with PCA, which reduces the feature space from 14 to 9 while maintaining high accuracy.

- **Train Accuracy**: 98.34%
- **Validation Accuracy**: 98.56%
- **Test Accuracy**: 98.15%

SVM performs well with fewer dimensions, keeping classification performance high while improving efficiency.
While Random Forest without PCA remains the best overall, SVM with PCA provides an excellent balance between accuracy and dimensionality reduction.

## *Classification using only the important features*

Before using PCA, we saw that the main features that receive the most importance are - redshift, i, z, u, g, r.
We saw that mjd, plate also receive high computation in KNN for example and Random Forest, but in KNN we got that we have a weak model that suffers from overfitting and has a not so good generalization ability, and also the importance of each feature is measured by the accuracy of the model when we remove that feature and not by a combination of several features that it is included in.
In Random Forest, each tree is trained on a different sample of the data, so certain features may seem important in specific decision trees but not be really useful for the model as a whole. In addition, features such as mjd and plate may be highly important because they help the model separate examples in the given dataset, but they are not necessarily general to new examples.
**We will try to perform classification only using the features: redshift, i, z, u, g, r, because we saw that they are the only ones with high importance in all models.**

## *SVM*

Best model from grid search :

- Model Kernel – Linear
- Regularization Parameter – C = 10

Result:

```
Train Result:
Accuracy: 0.9875

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      1.00      0.99      2667
      GALAXY       0.99      0.98      0.99      3181
         QSO       0.98      0.97      0.97       552

    accuracy                           0.99      6400
   macro avg       0.98      0.98      0.98      6400
weighted avg       0.99      0.99      0.99      6400


Confusion Matrix
 [[2667    0    0]
 [  49 3120   12]
 [   1   18  533]]
Validation Result:
Accuracy: 0.99

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      1.00      0.99       655
      GALAXY       0.99      0.99      0.99       817
         QSO       0.98      0.95      0.97       128

    accuracy                           0.99      1600
   macro avg       0.99      0.98      0.98      1600
weighted avg       0.99      0.99      0.99      1600


Confusion Matrix
 [[655   0   0]
 [  8 807   2]
 [  0   6 122]]
```

```
Test Result:
Accuracy: 0.9845

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      1.00      0.99       830
      GALAXY       0.99      0.97      0.98      1000
         QSO       0.96      0.97      0.96       170

    accuracy                           0.98      2000
   macro avg       0.98      0.98      0.98      2000
weighted avg       0.98      0.98      0.98      2000


Confusion Matrix
 [[830   0   0]
 [ 19 974   7]
 [  0   5 165]]
```

Key Observations:

- All classes achieve high precision, recall, and F1-scores, indicating strong classification performance.
- QSO recall is slightly lower (97% in train, 95% in validation, 97% in test), showing minor difficulty in distinguishing QSO samples.
- The confusion matrices show some misclassification between GALAXY and QSO, but overall, the model maintains high accuracy.
- Validation and test results are very close to training results, demonstrating excellent generalization without overfitting.

Overall, the SVM model performs exceptionally well, achieving 98.75% train accuracy, 99% validation accuracy, and 98.45% test accuracy. After removing unimportant features, the model remains highly effective, suggesting that only the selected features are necessary for strong classification. SVM is a robust choice, providing high accuracy and generalization with a well-balanced decision boundary.

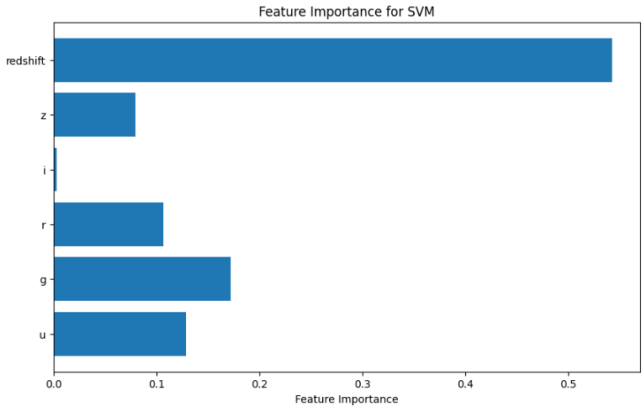



Fig.16. The importance features in SVM classification

*KNN*

Best model from grid search :

- Number of nearest neighbors = 3
- P_ Distance = 1

Result :

```
Train Result:
Accuracy: 0.9765625

Classification Report:
              precision    recall  f1-score   support

        STAR       0.96      0.99      0.98      2667
      GALAXY       0.99      0.96      0.98      3181
         QSO       0.99      0.96      0.97       552

    accuracy                           0.98      6400
   macro avg       0.98      0.97      0.98      6400
weighted avg       0.98      0.98      0.98      6400


Confusion Matrix
 [[2652   15    0]
 [ 107 3068    6]
 [   2   20  530]]
Validation Result:
Accuracy: 0.961875

Classification Report:
              precision    recall  f1-score   support

        STAR       0.94      0.99      0.96       655
      GALAXY       0.98      0.94      0.96       817
         QSO       0.98      0.92      0.95       128

    accuracy                           0.96      1600
   macro avg       0.97      0.95      0.96      1600
weighted avg       0.96      0.96      0.96      1600


Confusion Matrix
 [[650   5   0]
 [ 44 771   2]
 [  0  10 118]]

 Test Result:
 Accuracy: 0.96

 Classification Report:
               precision    recall  f1-score   support

         STAR       0.94      0.99      0.96       830
       GALAXY       0.98      0.94      0.96      1000
          QSO       0.96      0.94      0.95       170

     accuracy                           0.96      2000
    macro avg       0.96      0.96      0.96      2000
 weighted avg       0.96      0.96      0.96      2000


 Confusion Matrix
  [[818  11   1]
  [ 52 942   6]
  [  3   7 160]]
```

Key Observations:

- Overfitting is no longer an issue, as validation and test accuracy (96.18% and 96%) are now much closer to train accuracy (97.65%).
- Significant improvement over the previous KNN model, which suffered from overfitting due to unnecessary features.
- Most misclassifications occur between GALAXY and QSO, as seen in the confusion matrices.
- Precision, recall, and F1-scores remain high for all classes, showing that the model effectively differentiates between them.

Overall, KNN now generalizes much better than before, thanks to feature selection that removed non essential attributes. The model achieves strong classification performance without severe overfitting, making it a valid alternative when interpretability is prioritized over computational efficiency.

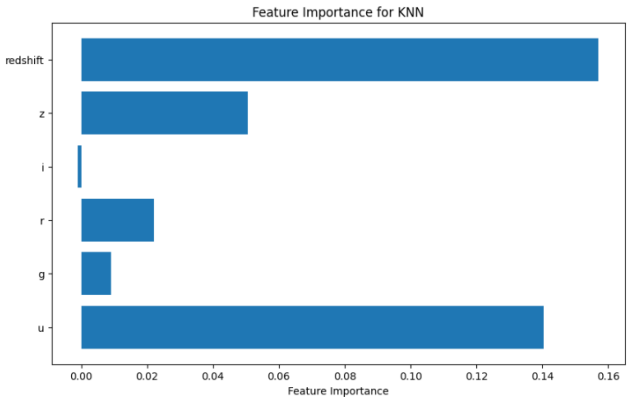



Fig.17. The importance features in KNN classification

Logistic Regression

Result :

```
Train Result:
Accuracy: 0.98109375

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      0.99      0.99      2667
      GALAXY       0.98      0.98      0.98      3181
         QSO       0.96      0.90      0.93       552

    accuracy                           0.98      6400
   macro avg       0.97      0.96      0.97      6400
weighted avg       0.98      0.98      0.98      6400


Confusion Matrix
 [[2653   14    0]
 [  29 3130   22]
 [   1   55  496]]
```

```
Validation Result:
Accuracy: 0.979375

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      0.99      0.99       655
      GALAXY       0.97      0.99      0.98       817
         QSO       0.96      0.88      0.92       128

    accuracy                           0.98      1600
   macro avg       0.97      0.95      0.96      1600
weighted avg       0.98      0.98      0.98      1600


Confusion Matrix
 [[646   9   0]
 [  4 808   5]
 [  0  15 113]]
Test Result:
Accuracy: 0.983

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      0.99      0.99       830
      GALAXY       0.98      0.98      0.98      1000
         QSO       0.96      0.94      0.95       170

    accuracy                           0.98      2000
   macro avg       0.98      0.97      0.97      2000
weighted avg       0.98      0.98      0.98      2000


Confusion Matrix
 [[825   5   0]
 [ 11 982   7]
 [  0  11 159]]
```

Key Observations:

- The model achieves high accuracy across all datasets (98.1% train, 97.9% validation, 98.3% test).
- QSO recall is lower than other classes (90% train, 88% validation, 94% test), meaning some QSO samples are misclassified.
- The confusion matrices show most misclassifications occur between GALAXY and QSO, similar to SVM.
- Good generalization—performance is consistent across train, validation, and test sets, indicating no overfitting.

Overall, Logistic Regression provides a strong baseline model with high accuracy and generalization. While slightly weaker than SVM and Random Forest, it still performs well, particularly for STAR and GALAXY classifications. The lower recall for QSO suggests some difficulty distinguishing QSOs from GALAXY, likely due to overlapping feature distributions. However, it remains a fast and interpretable model with reliable performance.

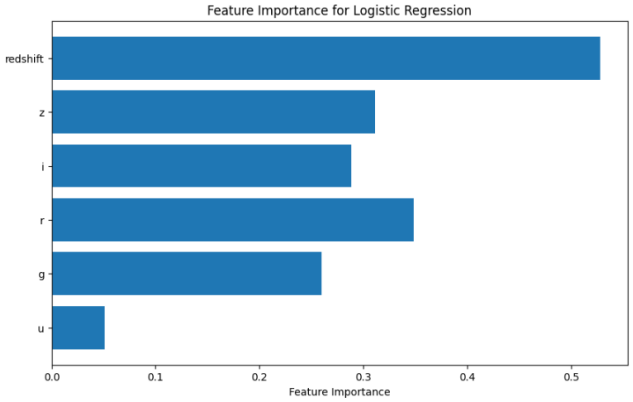



Fig.18. The importance features in Logistic Regression

*Decision Tree*

Result :

```
Train Result:
Accuracy: 0.996875

Classification Report:
              precision    recall  f1-score   support

        STAR       1.00      1.00      1.00      2667
      GALAXY       1.00      1.00      1.00      3181
         QSO       1.00      0.97      0.98       552

    accuracy                           1.00      6400
   macro avg       1.00      0.99      0.99      6400
weighted avg       1.00      1.00      1.00      6400


Confusion Matrix
 [[2667    0    0]
 [   3 3176    2]
 [   1   14  537]]
Validation Result:
Accuracy: 0.988125

Classification Report:
              precision    recall  f1-score   support

        STAR       1.00      1.00      1.00       655
      GALAXY       0.98      1.00      0.99       817
         QSO       0.97      0.88      0.92       128

    accuracy                           0.99      1600
   macro avg       0.98      0.96      0.97      1600
weighted avg       0.99      0.99      0.99      1600


Confusion Matrix
 [[655   0   0]
 [  0 813   4]
 [  0  15 113]]
```

```
Test Result:
Accuracy: 0.985

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      1.00      0.99       830
      GALAXY       0.99      0.98      0.98      1000
         QSO       0.95      0.94      0.94       170

    accuracy                           0.98      2000
   macro avg       0.97      0.97      0.97      2000
weighted avg       0.98      0.98      0.98      2000


Confusion Matrix
 [[830   0   0]
 [ 10 981   9]
 [  0  11 159]]
```

Key Observations:

- High accuracy across all datasets (99.7% train, 98.8% validation, 98.5% test), indicating strong classification performance.
- Slight drop in accuracy from training to validation and test, but not significant enough to suggest severe overfitting.
- QSO recall is lower in validation (88%), suggesting some difficulty in distinguishing quasars, though performance improves in the test set (94%).
- Confusion matrices show minor misclassifications between GALAXY and QSO, while STAR classification is nearly perfect.

Overall , Decision Trees perform well but shows that there is a small overfitting, as indicated by near perfect training performance, while in validation, test it does not show a perfect classification but still high.

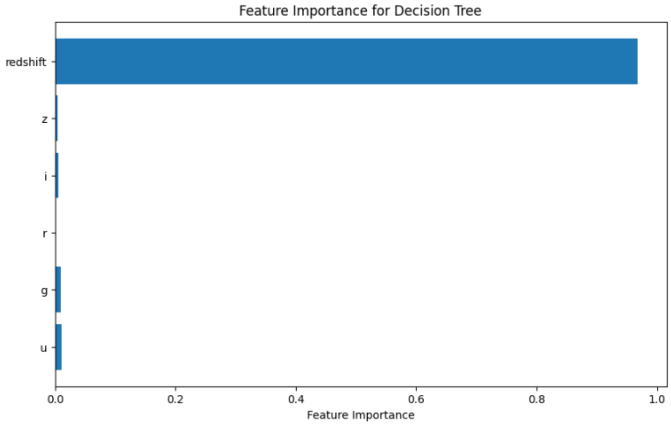



Fig.19. The importance features in Decision Tree

*Random Forest*

Result:

```
Train Result:
Accuracy: 0.99296875

Classification Report:
              precision    recall  f1-score   support

        STAR       1.00      1.00      1.00      2667
      GALAXY       0.99      0.99      0.99      3181
         QSO       0.98      0.96      0.97       552

    accuracy                           0.99      6400
   macro avg       0.99      0.98      0.99      6400
weighted avg       0.99      0.99      0.99      6400


Confusion Matrix
 [[2667    0    0]
 [  12 3157   12]
 [   1   20  531]]
Validation Result:
Accuracy: 0.990625

Classification Report:
              precision    recall  f1-score   support

        STAR       1.00      1.00      1.00       655
      GALAXY       0.99      0.99      0.99       817
         QSO       0.98      0.92      0.95       128

    accuracy                           0.99      1600
   macro avg       0.99      0.97      0.98      1600
weighted avg       0.99      0.99      0.99      1600


Confusion Matrix
 [[655   0   0]
 [  2 812   3]
 [  0  10 118]]

Test Result:
Accuracy: 0.987

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      1.00      0.99       830
      GALAXY       0.99      0.98      0.99      1000
         QSO       0.95      0.96      0.95       170

    accuracy                           0.99      2000
   macro avg       0.98      0.98      0.98      2000
weighted avg       0.99      0.99      0.99      2000


Confusion Matrix
 [[830   0   0]
 [ 10 981   9]
 [  0   7 163]]
```

Key Observations:

- Excellent accuracy across all datasets (99.3% train, 99% validation, 98.7% test).
- Much less overfitting compared to Decision Tree, as seen by similar train and test results.
- QSO recall is higher than in Decision Tree (96% train, 92% validation, 96% test), reducing misclassification.
- The confusion matrices show minimal errors, with STAR and GALAXY classification being nearly perfect.

Overall , Random Forest is one of the best-performing models, offering high accuracy, good generalization, and reduced overfitting compared to a single Decision Tree. It effectively balances precision, recall, and F1-score across all classes, making it a strong choice for robust classification in this dataset.

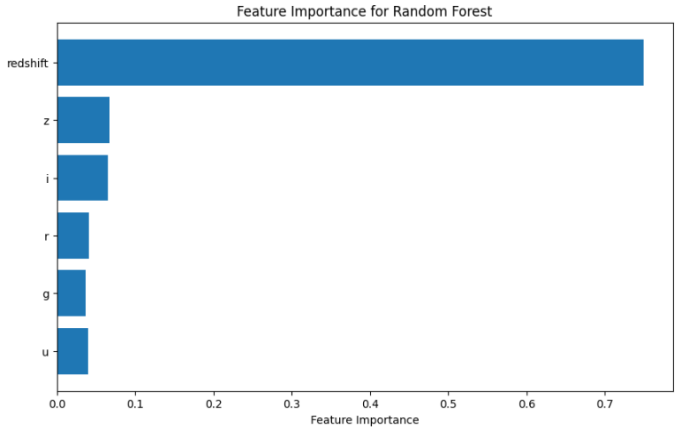



Fig.20. The importance features in Random Forest

## *Conclusions from the results*

We achieved highly accurate classification using only six features: **redshift, i, z, r, g, u**. This is a significant reduction from the original 14 features, yet the classification performance remains nearly unchanged, with accuracy differences of less than 1% across all models. Notably, KNN improved substantially, as removing noisy features allowed it to classify with only three neighbors, achieving 97% accuracy in training and 96% in testing, without overfitting as before.

Fig. 16: All features contribute, though i has minimal impact on classification.

Fig. 17: All features contribute except i, which slightly degrades classification performance.

Fig. 18: All features contribute, including u, which, despite being the least important, still plays a meaningful role.

Fig. 19: Redshift is the dominant feature, while all others provide little to no contribution.

Fig. 20: All features contribute, with redshift being the most influential.

Conclusion, all features play a role in classification, with redshift being the most critical. We successfully reduced dimensionality by eliminating features that negatively impacted (or without impacted at all) classification while maintaining accuracy losses of less than 1% across all models. Additionally, KNN saw significant improvements, achieving high accuracy with fewer neighbors and no overfitting.

### *PCA - Principal Component Analysis*

As before, we will run the PCA algorithm.
We do this because we have a high correlation between the features – u, g, r, i, z.
High correlation in PCA is beneficial because it enables the principal components to capture maximum variance with fewer components, reducing dimensionality while preserving essential information.

For this analysis, **3** principal components were selected, capturing 99.4% of the total variance:
**Explained Variance Ratio:** [0.76665715 0.1472028 0.08066106]
**Final Variance Ratio:** 0.994

### *SVM – With PCA :*

Result:

```
Train Result With PCA:
Accuracy: 0.981875

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      1.00      0.99      2667
      GALAXY       0.98      0.98      0.98      3181
         QSO       0.98      0.92      0.95       552

    accuracy                           0.98      6400
   macro avg       0.98      0.97      0.97      6400
weighted avg       0.98      0.98      0.98      6400


Confusion Matrix
 [[2657   10    0]
 [  50 3118   13]
 [   1   42  509]]
```

```
Validation Result With PCA:
Accuracy: 0.985625

Classification Report:
              precision    recall  f1-score   support

        STAR       0.99      1.00      0.99       655
      GALAXY       0.99      0.99      0.99       817
         QSO       0.97      0.91      0.94       128

    accuracy                           0.99      1600
   macro avg       0.98      0.97      0.97      1600
weighted avg       0.99      0.99      0.99      1600


Confusion Matrix
 [[654   1   0]
 [  8 806   3]
 [  0  11 117]]
Test Result With PCA:
Accuracy: 0.98

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      1.00      0.99       830
      GALAXY       0.99      0.97      0.98      1000
         QSO       0.95      0.95      0.95       170

    accuracy                           0.98      2000
   macro avg       0.97      0.97      0.97      2000
weighted avg       0.98      0.98      0.98      2000


Confusion Matrix
 [[827   3   0]
 [ 20 972   8]
 [  0   9 161]]
```

Key Observations:

- High accuracy across all datasets: 98.19% (train), 98.56% (validation), and 98% (test).
- The classification performance remains strong after reducing dimensions to three.
- QSO recall is slightly lower (92% train, 91% validation, 95% test), indicating minor difficulty distinguishing quasars.
- The confusion matrices show minimal misclassifications, mainly between GALAXY and QSO, but STAR classification is nearly perfect.
- The macro and weighted F1-scores remain consistently high, suggesting that PCA did not significantly degrade performance.

Overall , SVM maintains excellent classification performance even after dimensionality reduction. The accuracy remains nearly identical to the full-feature model, proving that PCA effectively preserves key information. The model generalizes well, with small performance drops (less then 1%), making it a computationally efficient and robust choice.
Fig.13. Important eigenvectors in Logistic Regression

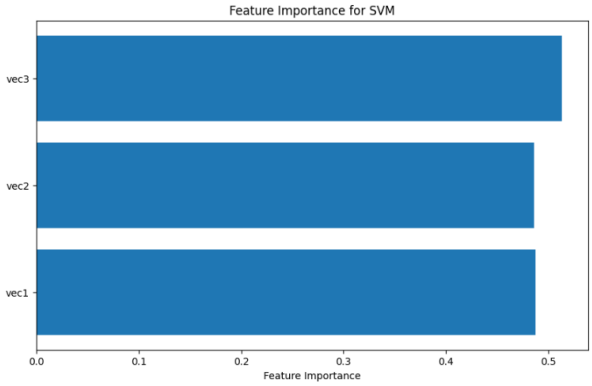



Fig.21. Important eigenvectors in SVM

*KNN – With PCA :*

Result:

```
Train Result With PCA:
Accuracy: 0.9703125

Classification Report:
              precision    recall  f1-score   support

        STAR       0.95      0.99      0.97      2667
      GALAXY       0.98      0.96      0.97      3181
         QSO       0.98      0.95      0.97       552

    accuracy                           0.97      6400
   macro avg       0.97      0.97      0.97      6400
weighted avg       0.97      0.97      0.97      6400


Confusion Matrix
 [[2643   24    0]
 [ 129 3043    9]
 [   3   25  524]]
Validation Result With PCA:
Accuracy: 0.954375

Classification Report:
              precision    recall  f1-score   support

        STAR       0.92      0.99      0.96       655
      GALAXY       0.98      0.93      0.95       817
         QSO       0.98      0.92      0.95       128

    accuracy                           0.95      1600
   macro avg       0.96      0.95      0.95      1600
weighted avg       0.96      0.95      0.95      1600


Confusion Matrix
 [[649   6   0]
 [ 54 760   3]
 [  0  10 118]]
```

```
Test Result With PCA:
Accuracy: 0.954

Classification Report:
              precision    recall  f1-score   support

        STAR       0.92      0.99      0.95       830
      GALAXY       0.98      0.93      0.95      1000
         QSO       0.96      0.95      0.95       170

    accuracy                           0.95      2000
   macro avg       0.95      0.95      0.95      2000
weighted avg       0.96      0.95      0.95      2000


Confusion Matrix
 [[820  10   0]
 [ 66 927   7]
 [  2   7 161]]
```

Key Observations:

- Accuracy is slightly lower than SVM but still strong: 97.03% (train), 95.44% (validation), and 95.4% (test).
- The model generalizes well, with validation and test results closely matching train performance.
- QSO recall improves compared to previous KNN models (95% train, 92% validation, 95% test), indicating that PCA reduced noise.
- The confusion matrices show slightly higher misclassification rates compared to SVM, especially between GALAXY and QSO.
- STAR classification remains highly accurate, with only minor misclassifications.

Overall, applying PCA significantly improved KNN's generalization, reducing overfitting compared to previous models. Despite the loss of some feature information, KNN still achieves high accuracy (95.4%) with only three dimensions. PCA enhanced KNN is now a viable alternative, especially when interpretability and efficiency are prioritized.

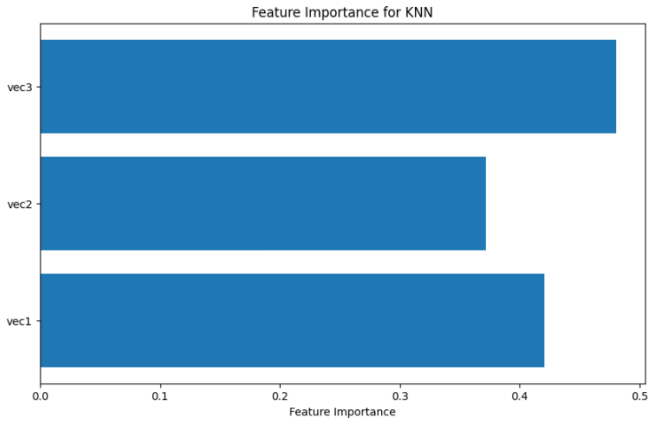



Fig.22. Important eigenvectors in KNN

*Logistic Regression – With PCA :*

Result:

```
Train Result With PCA:
Accuracy: 0.98109375

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      1.00      0.99      2667
      GALAXY       0.99      0.98      0.98      3181
         QSO       0.97      0.94      0.95       552

    accuracy                           0.98      6400
   macro avg       0.98      0.97      0.97      6400
weighted avg       0.98      0.98      0.98      6400


Confusion Matrix
 [[2656   10    1]
 [  60 3104   17]
 [   1   32  519]]
Validation Result With PCA:
Accuracy: 0.981875

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      1.00      0.99       655
      GALAXY       0.99      0.98      0.98       817
         QSO       0.94      0.92      0.93       128

    accuracy                           0.98      1600
   macro avg       0.97      0.97      0.97      1600
weighted avg       0.98      0.98      0.98      1600


Confusion Matrix
 [[653   1   1]
 [ 11 800   6]
 [  0  10 118]]

Test Result With PCA:
Accuracy: 0.978

Classification Report:
              precision    recall  f1-score   support

        STAR       0.97      1.00      0.98       830
      GALAXY       0.99      0.97      0.98      1000
         QSO       0.94      0.96      0.95       170

    accuracy                           0.98      2000
   macro avg       0.97      0.97      0.97      2000
weighted avg       0.98      0.98      0.98      2000


Confusion Matrix
 [[827   3   0]
 [ 24 966  10]
 [  0   7 163]]
```

Key Observations:

- High accuracy across all datasets: 98.1% (train), 98.19% (validation), and 97.8% (test).
- Performance remains strong despite dimensionality reduction.
- QSO recall is slightly lower (94% train, 92% validation, 96% test), indicating minor misclassification issues.
- STAR classification is nearly perfect, with only a few misclassified instances.
- Confusion matrices show minor misclassifications, mainly between GALAXY and QSO.

Overall , Logistic Regression maintains excellent performance after PCA, achieving high accuracy with only three dimensions. The slight drop in QSO recall suggests some information loss, but the model remains highly effective and computationally efficient.

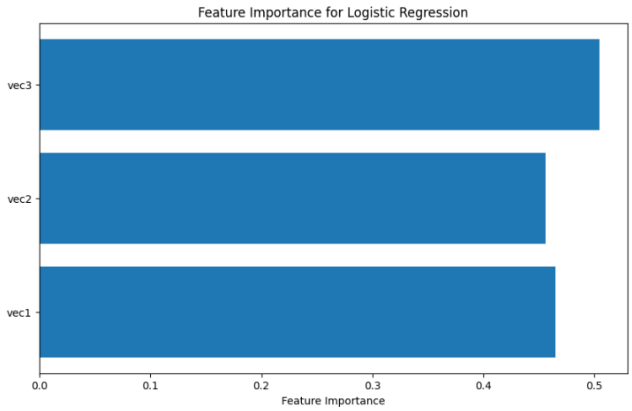



Fig.23. Important eigenvectors in Logistic Regression

*Decision Tree – With PCA:*

Result:

```
Train Result With PCA:
Accuracy: 0.98515625

Classification Report:
              precision    recall  f1-score   support

        STAR       0.98      0.99      0.99      2667
      GALAXY       0.99      0.98      0.99      3181
         QSO       1.00      0.97      0.98       552

    accuracy                           0.99      6400
   macro avg       0.99      0.98      0.98      6400
weighted avg       0.99      0.99      0.99      6400


Confusion Matrix
 [[2652   14    1]
 [  60 3120    1]
 [   1   18  533]]
```

```
Validation Result With PCA:
Accuracy: 0.920625

Classification Report:
              precision    recall  f1-score   support

        STAR       0.89      0.94      0.92       655
      GALAXY       0.94      0.91      0.92       817
         QSO       0.96      0.91      0.94       128

    accuracy                           0.92      1600
   macro avg       0.93      0.92      0.92      1600
weighted avg       0.92      0.92      0.92      1600


Confusion Matrix
 [[616  39   0]
 [ 72 740   5]
 [  1  10 117]]
Test Result With PCA:
Accuracy: 0.927

Classification Report:
              precision    recall  f1-score   support

        STAR       0.90      0.96      0.93       830
      GALAXY       0.95      0.91      0.93      1000
         QSO       0.96      0.91      0.93       170

    accuracy                           0.93      2000
   macro avg       0.93      0.92      0.93      2000
weighted avg       0.93      0.93      0.93      2000


Confusion Matrix
 [[793  37   0]
 [ 87 906   7]
 [  3  12 155]]
```

Key Observations:

- Train accuracy is extremely high (98.5%), but validation (92%) and test (92.7%) accuracy drop significantly, indicating overfitting.
- QSO classification remains strong, but GALAXY and STAR misclassifications increase.
- The confusion matrices show a noticeable increase in misclassifications, especially between GALAXY and STAR.

As analyzed earlier, PCA does not work well with Decision Trees. The model overfits on the training data but generalizes poorly, leading to a significant accuracy drop in validation and test sets. Decision Trees rely heavily on feature space structure, and PCA disrupts this, making it an ineffective approach for this model.

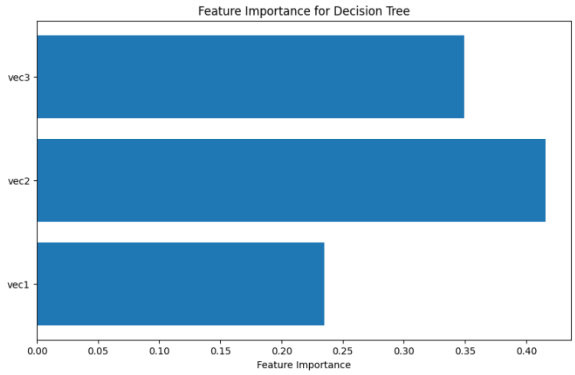



Fig.24. Important eigenvectors in Decision Tree

### *Random Forest – With PCA:*

Result:

```
Train Result With PCA:
Accuracy: 1.0

Classification Report:
              precision    recall  f1-score   support

        STAR       1.00      1.00      1.00      2667
      GALAXY       1.00      1.00      1.00      3181
         QSO       1.00      1.00      1.00       552

    accuracy                           1.00      6400
   macro avg       1.00      1.00      1.00      6400
weighted avg       1.00      1.00      1.00      6400


Confusion Matrix
 [[2667    0    0]
 [   0 3181    0]
 [   0    0  552]]
Validation Result With PCA:
Accuracy: 0.9325

Classification Report:
              precision    recall  f1-score   support

        STAR       0.91      0.95      0.93       655
      GALAXY       0.95      0.92      0.93       817
         QSO       0.97      0.91      0.94       128

    accuracy                           0.93      1600
   macro avg       0.94      0.93      0.93      1600
weighted avg       0.93      0.93      0.93      1600


Confusion Matrix
 [[625  30   0]
 [ 63 751   3]
 [  2  10 116]]
```

```
Test Result With PCA:
Accuracy: 0.9425

Classification Report:
              precision    recall  f1-score   support

        STAR       0.90      0.98      0.94       830
      GALAXY       0.98      0.91      0.94      1000
         QSO       0.96      0.94      0.95       170

    accuracy                           0.94      2000
   macro avg       0.95      0.94      0.95      2000
weighted avg       0.95      0.94      0.94      2000


Confusion Matrix
 [[814  16   0]
 [ 83 911   6]
 [  4   6 160]]
```

Key Observations:

- Train accuracy is perfect (100%), but validation (93.25%) and test (94.25%) accuracy drop significantly, indicating overfitting.
- STAR classification is nearly perfect, with some misclassifications between GALAXY and QSO.
- The confusion matrices show more errors in validation and test sets, especially between GALAXY and STAR.

As analyzed earlier, PCA does not work well with tree-based models like Decision Trees and Random Forests. The model overfits on the training data but generalizes poorly, leading to a noticeable drop in validation and test performance. Since tree-based models rely on the original feature space structure, PCA disrupts their ability to split data effectively, making it an ineffective approach for Random Forests as well.

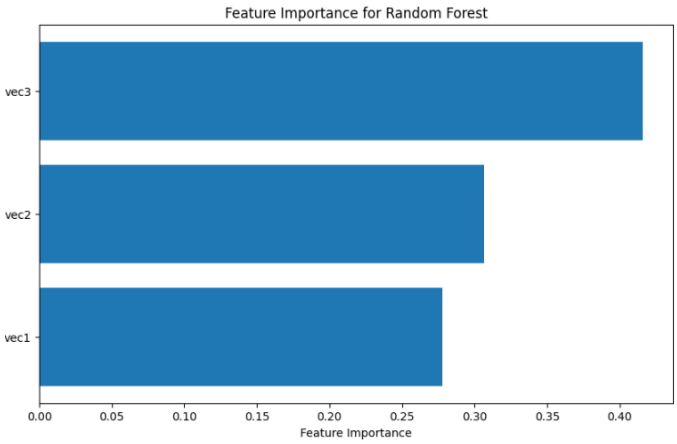


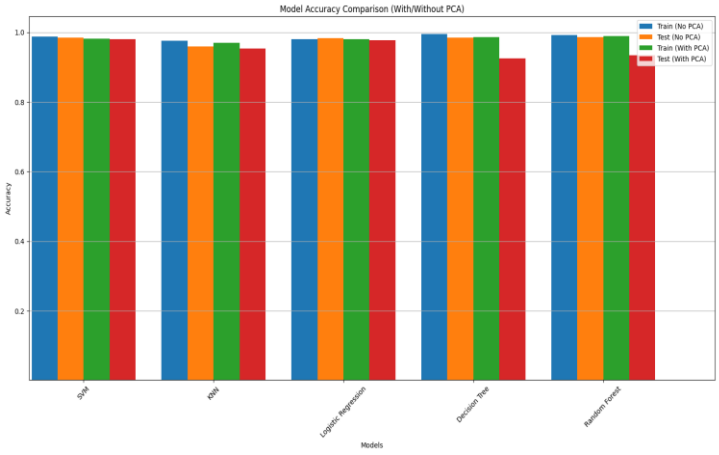

Fig.25. Important eigenvectors in Random Forest

*Conclusions from the results*

SVM, Logistic Regression, and KNN maintained high accuracy across train, validation, and test sets. Unlike before, KNN also demonstrated strong generalization, with its accuracy dropping by less than 1% between training and test sets, indicating no overfitting.

Random Forest had the highest training accuracy but experienced a drop in test accuracy, suggesting some memorization of the training data. Decision Trees also struggled with PCA, reinforcing the idea that tree-based models are less suited for dimensionality reduction techniques like PCA. For these models, feature selection techniques may be a better alternative.

PCA significantly improved the performance of SVM, Logistic Regression, and KNN while negatively impacting Decision Trees and Random Forests. This highlights the importance of choosing dimensionality reduction techniques based on model characteristics. We successfully reduced the dimensionality from 14 features to only 3, achieving effective compression while maintaining strong classification performance. Additionally, as observed in Fig. 21 to Fig. 25, all eigenvectors play a crucial role in the reduced feature space.




**Best Model: Random Forest (with or without PCA)**
The best-performing model is Random Forest, demonstrating high accuracy and strong generalization.

- **Train Accuracy:** 99.30%
- **Validation Accuracy:** 99.06%
- **Test Accuracy:** 98.70%

Random Forest outperforms other models by

maintaining high accuracy across all sets with minimal overfitting. The classification metrics show high precision and recall across all classes, making it the most reliable model for this task.

**Best Model with PCA**
After applying PCA, the best model is SVM with PCA, which reduces the feature space from 14 to 3 while maintaining high accuracy.

- **Train Accuracy:** 98.18%
- **Validation Accuracy:** 98.56%
- **Test Accuracy:** 98.00%

SVM performs well with fewer dimensions, preserving classification performance while improving efficiency. While Random Forest without PCA remains the best overall, SVM with PCA provides an excellent balance between accuracy and dimensionality reduction.

**Comparison of Random Forest with Full Feature Set vs. Selected Features (Redshift, i, u, r, z, g)**
To assess whether reducing the number of features improves performance, we compare Random Forest trained on all 14 features with Random Forest trained on just the 6 selected features (redshift, i, u, r, z, g).
Random Forest with 6 Features:

- **Train Accuracy:** 99.29%
- **Validation Accuracy:** 99.00%
- **Test Accuracy:** 98.7%

Random Forest with 14 Features:

- **Train Accuracy:** 99.99%
- **Validation Accuracy:** 99.00%
- **Test Accuracy:** 98.75%

Both models achieve nearly identical accuracy, indicating that the additional features do not significantly contribute to performance. The reduced feature set (6 features) provides a more efficient model while maintaining accuracy, making it a preferable choice for practical applications where computational efficiency is important.

**Comparison of SVM with PCA: Reducing 14 Features to 9 vs. Reducing 6 Features to 3**
To evaluate the impact of dimensionality reduction, we compare the performance of SVM with PCA applied to two different feature sets: one where the feature space is reduced from 14 to 9 dimensions and another where it is reduced from 6 to 3 dimensions.
**SVM with PCA (Reduction from 14 to 9 Features)**

- **Train Accuracy:** 98.34%
- **Validation Accuracy:** 98.56%
- **Test Accuracy:** 98.15%

**SVM with PCA (Reduction from 6 to 3 Features)**

- **Train Accuracy:** 98.18%
- **Validation Accuracy:** 98.56%
- **Test Accuracy:** 98.00%

Both models maintain high accuracy while significantly reducing the dimensionality, enhancing efficiency without sacrificing classification performance. The slight decrease in accuracy for the model with three features suggests that further reduction beyond this point may start to impact predictive power. Nonetheless, SVM with PCA using three features still provides a strong balance between computational efficiency and accuracy, making it a viable option for resource-constrained environments.

## Conclusion & Research Questions

Throughout this project, we successfully overcame numerous challenges to achieve highly accurate classification of astronomical objects - stars, galaxies, and quasars. By carefully analyzing the dataset and optimizing feature selection, we were able to classify objects with an impressive **98.7% accuracy** using only **six features**, compared to the original dataset containing **17 features** (excluding class labels).
We conducted an in-depth analysis of the dataset and leveraged **Principal Component Analysis (PCA)** effectively. By reducing the feature space to only **three dimensions**, and utilizing **SVM with PCA**, we achieved a classification accuracy of **98.00%**. This was made possible by thorough correlation analysis and an understanding of feature relationships, allowing us to maintain high performance while simplifying the model.
Despite the **imbalanced class distribution** with quasars (QSO) being the least represented class (only 800-900 samples, constituting less than 10% of the dataset), our models still achieved a remarkable classification performance. Using only six features with **Random Forest**, we were able to classify QSO objects with a precision of **0.95**. Similarly, using **SVM with PCA**, we attained a QSO classification accuracy of **0.95** with only **three dimensions**.
By overcoming these challenges, we developed a high-performing model that achieves excellent classification accuracy while ensuring simplicity with minimal dimensions.

### *Research Questions & Answers*

**1. Can we classify astronomical objects (stars, galaxies, and quasars) based on their features, such as redshift and magnitude?**

Yes, we successfully classified astronomical objects with high accuracy using their features. Our analysis demonstrated that key astronomical attributes, such as **redshift and magnitude across different spectral bands**, provide enough discriminative power to

differentiate between stars, galaxies, and quasars. By applying feature selection and dimensionality reduction techniques, we confirmed that a subset of features is sufficient for accurate classification.

**2. Which combination of features provides the most accurate classification for distinguishing stars, galaxies, and quasars?**

The optimal set of features for classification includes: **redshift, i, u, r, z, g**. These six features were selected based on their correlation with class labels and their contribution to model performance. The inclusion of multiple spectral magnitudes, along with redshift, ensures that the model captures essential astrophysical properties of each object type.

**3. Can we achieve high classification accuracy using machine learning models on this dataset?**
Yes, we achieved a **classification accuracy of 98.75%**, demonstrating the effectiveness of machine learning for this task. Random Forest, in particular, performed exceptionally well, leveraging feature importance and ensemble learning to generalize across different object classes.

**4. Can Principal Component Analysis (PCA) improve classification accuracy?**

While PCA did not improve classification accuracy, it significantly simplified the model by reducing dimensionality with **minimal loss in performance**. Algorithms like SVM, KNN, and Logistic Regression benefited from PCA, as reducing the feature space improved computational efficiency while maintaining high accuracy. We managed to **reduce the number of dimensions significantly while experiencing less than a 1% drop in accuracy.**

**5. Which of the five machine learning algorithms Logistic Regression, Decision Tree, SVM, KNN, and Random Forest performs best for classifying astronomical objects from the SDSS dataset?**

**The best performing model: Random Forest**, achieving the following results using only the six selected features (redshift, i, u, r, z, g):

- **Train Accuracy:** 99.29%
- **Validation Accuracy:** 99.00%
- **Test Accuracy:** 98.7%

Additionally, **SVM also performed exceptionally well** when combined with PCA. By reducing the feature space to just **three dimensions**, we achieved:

- **Train Accuracy:** 98.18%
- **Validation Accuracy:** 98.56%
- **Test Accuracy:** 98.00%

These findings highlight that Random Forest remains the top model in terms of raw accuracy, while SVM with PCA offers a highly efficient alternative with minimal performance tradeoff.

In summary, this project successfully addressed the classification of astronomical objects by leveraging machine learning techniques. Through feature selection, dimensionality reduction, and algorithm optimization, we developed models that achieve high accuracy while maintaining computational efficiency. The combination of Random Forest with selected features and SVM with PCA provides a robust framework for future astronomical classification tasks, demonstrating that machine learning is a powerful tool in astrophysical data analysis.