Implementation of tree models

Vishnu Sharma [1]
AI23MTECH11005
Indian Institute of Technology, Hyderabad
Email: ai23mtech11005@iith.ac.in

Pratik Yawalkar [2]
AI23MTECH11006
Indian Institute of Technology, Hyderabad
Email: ai23mtech11006@iith.ac.in

Code can be found here here.

Abstract

With the proposed methodology, star types of galaxies are roughly classified by what can be called a hybrid approach involving K-Nearest Neighbors (KNN), K-Dimensional Trees (KD-Trees), and decision tree systems. The KD Trees are being integrated into the method that is specifically designed to be a good processor of spatial query handling and that allows massive, multidimensional datasets used in astronomy to be handled efficiently This integration results into tremendous saving of computational complexity of the problem O(n) to O(log n) which gives fast classification process. What's more, the approach that we have adopted is a mixed one which entails the pivotal elements of dimensionality reduction and data normalization, each one being carefully thought out for the purpose of improving the accuracy and the efficiency. Furthermore, we infuse decision tree concepts in to our methodology through the use of Knighted Nearest Neighbors and Knighted Distance Tree. With the help of the inclusion, the examination of characteristic spaces could be more thorough, which will amplify the precision of classification results. Classification procedures of our hybrid method were compared to traditional KNN techniques. Thus, it can be concluded that the method speeds up the process and still maintains an impressive average accuracy of around 92%. The above results turn out to be a great exhibit of the effectiveness and reliability of this model in enriching our understanding of galaxy constituents and behavior on the whole undercover of astrophysics.

1. Problem Statement

in astronomy, stellar classification is the categorization of stars according to their specific spectral appearances. The star's electromagnetic radiation is studied by the use of a prism or a diffraction grate (spectrometers) that divide it into a spectrum of the rainbow colors interspersed with the spectral lines Each line reflects a definite chemical element or molecule, the line strength shows the quantity of this element that is present. The differences in spectral line strengths are mostly caused by the temperature of the photosphere though in some cases, a true abundance difference can also disrupt the balance. The spectroscopic class is a succinct code that chiefly conveys the degree of ionization which leads to an objective measuring of the surface temperature of the photo-sphere.

Although some stars are are not yet classified as O type stars (the hottest) or M type stars (the coolest), most stars follow the Morgan–Keenan (MK) system using the letter sequence of O, B, A, F, G, K, and M. Every object belonging to a letter class will then be subdivided into a digit number starting from 9 (hottest) to 0 (coolest) (e.g. A8, A9, F0, and F1 go from hotter to cooler). The series has been amended with the lessons for other stars and star-like objects that do not comply with the classical system. White dwarfs are included into Class D and carbon stars are added to Classes S and C.

The goal is to apply this passage by classifying stars depending on data presented in the dataset.

2. Description of the dataset

The dataset contains one file, 'star type.csv', with 240 rows and seven columns. Each row represents a specific transaction. Details of each column are listed below:

- 'Temperature' (Integer): It represents temperature of star in kelvin.
- 'Luminosity' (Float64): It represents amount of light coming out from surface of star. Defined by L/Lo Watts.
- 'Absolute Magnitude' (Float64): Radius of the star. Defined by R/Ro m

– ‘Color‘ (String): Visible color of the star (General Color of Spectrum)

– ‘Spectral Class‘ (String): An asteroid spectral type is assigned to asteroids based on their emission spectrum, color, and sometimes albedo. These types are thought to correspond to an asteroid’s surface composition. Types are O,B,A,F,G,K,M / SMASS

– ‘Star type‘ (Integer): 0 to 5 labelled

  – Red Dwarf (0): A star having substantially lower surface temperature, intrinsic luminosity, mass, and size than the sun

  – Brown Dwarf (1): A celestial object that is much smaller than a normal star and has insufficient mass to sustain nuclear fusion but that is hot enough to radiate energy especially at infrared wavelengths

  – White Dwarf (2): A star that is at the end of its life and is very hot, small, and dense

  – Main Sequence (3): The group of stars that on a graph of spectrum versus luminosity forms a band comprising 90 percent of stellar types and that includes stars representative of the stages a normal star passes through during the majority of its lifetime

  – Super Giants (4): A star of very great intrinsic luminosity and enormous size

  – Hyper Giants (5): A star that has an extremely high luminosity, mass, size and mass loss because of its extreme stellar winds

(where, Lo = 3.828 x 10$\hat{2}$6 Watts (Avg Luminosity of Sun) and Ro = 6.9551 x 10$\hat{8}$ m (Avg Radius of Sun))

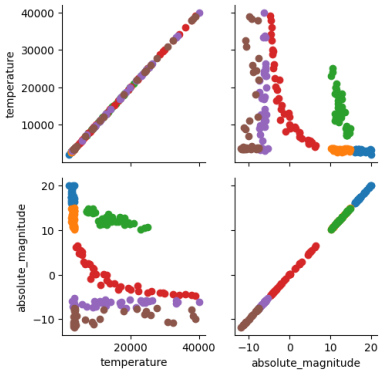

Figure 1. Pair-plot of Absolute magnitude Vs temperature

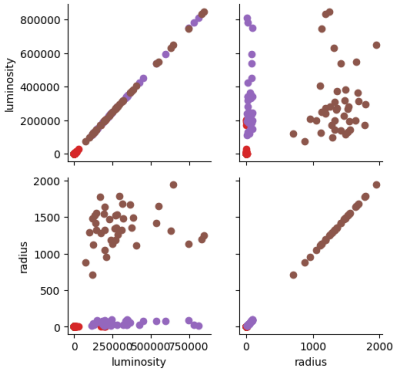

Figure 2. Pair-plot of Radius Vs Luminosity

3. Exploratory Data Analysis

Statistical analysis was applied by us in order to get more insights of the data. Achieved through Exploratory Data Analysis (EDA) to uncover the deeper and the more complex aspects of our data set. Through EDA, we were able not only to enhance our approach but also make our machine learning method interpret-able so in the end, we had a rather deep insight into stars and their characteristics.

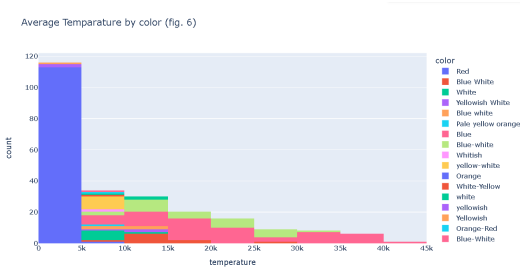

Figure 3. Average Temperature by color

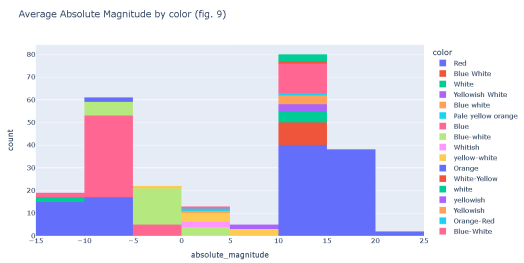

Figure 4. Average Absolute Temperature by color

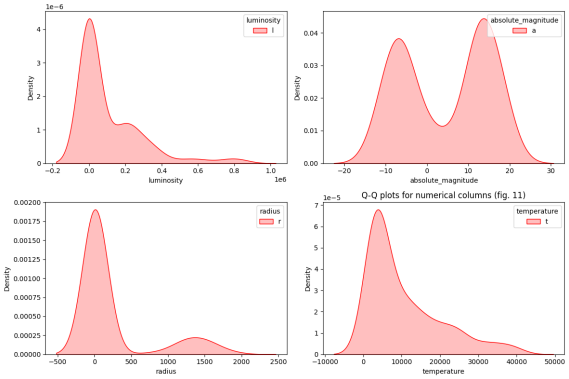

Figure 5. Density plots of Numerical columns

4. Methods

In this section, we will explore different methods to perform claassification of star type.

4.1. K-Nearest Neighbors (KNN) Algorithm

The one of the most simple, yet powerful, machine learning technique K-Nearest Neighbors (KNN) is used for the problems like the classification and regression. It is a the non-parametric method, which means that there are no assumptions drawn about the data distribution. "K-NN"'s concept is based on to determining a new data point's label by taking the average (for classification) or the "k" closest neighbors (for regression) from the data that has already been labeled and then having the majority decide the label.

Steps of KNN Algorithm:

1. Choose the number 'k' of neighbors: By setting this, we give the algorithm a clue how many closest neighbors to take into account when the prediction will be made. The selection of 'k' would, in its turn, impact on the degree of accuracy of the forecasts and, thus, result in the over-fitting.

2. Calculate the distance: Calculate the length of the line that may be drawn from the query point to all other points of the dataset. The most wide-spread distance measures are Euclidean, Manhattan and Min-k-ow-ski distance.

3. Sort the distances and identify the nearest neighbors: Create the list starting with the closest point to the last as they are distant.

4. Aggregate the labels of nearest neighbors: When category is to determine, simply count the number of neighboring unit stat in each class among the 'k' neighbors close to unit and predict whether the class it belongs to is the same as that in which the majority of the unit is found. To find out regression, the mean of the k nearest neighbors is being calculated.

5. Output the assigned class.

4.2. KD-Tree Algorithm

A K-Dimensional Tree (KD-Tree) is a space-partitioning data structure for organizing points in a k-dimensional space. KD-Trees are particularly useful in applications that require efficient nearest neighbor searches, such as with the KNN algorithm. They help to significantly reduce the number of comparisons needed to find the nearest neighbors in large datasets.

Steps of KD-Trees:

1. Initialization: Start with the entire data set and go for the node on the root dimension.

2. Splitting: Pick a pan-ne-shift along that selected dimension which will divide the data into two halves.

3. Recursion: Use recursion and repeat steps 1 and 2 for the partitioning of each set of data until each set contains a single data point or the pre-defined number of points.

4. Construction: Create the KD-Tree nodes by inserting each partition and connecting them in the form of the hierarchy.

5. Querying: When given a query point, go through the tree by using the tree structure to collapse the search space, and then find the leaf node closest to the query point.

6. Backtracking: During your descent through the tree check other partitions if they contain closer points with respect to the query point, adjusting the nearest neighbor when necessary.

4.3. Decision Tree Algorithm

A decision tree is a non-parametric supervised learning method which can be applied to both classification and regression tasks. A hierarchical, tree structure is what it is. It comprises of a root node, branches, internal nodes and leaf nodes.

Steps of Decision Tree:

1. Initialization: Tree has a root node comprising all dataset.

2. Feature Selection: For best performance, pick the best attribute out of the dataset to create subsamples. This criterion can be exploited by using the concepts such as entropy, information gain or Gini impurity.

3. Splitting: Perhaps, partition the data into more clusters for the feature to get better results. Choose the cluster that has the majority of the data to represent the feature.

4. Recursion: Follow steps 2 and 3 while subdividing the data for each subset, and keep going until you reach the maximum depth is met or the minimum number of samples in a node.

5. Leaf Node Creation: Once the recursion has been halted, label the leaves with class labels and start with each leaf subset. The majority class class found inside the subsets should be assigned as the leaf class label.

6. Pruning (optional): Trim the tree by removing unnecessary branches to bypass the overfitting and hence increasing the generalization accuracy of the model.

7. Prediction: On a new input data, go down the tree from the root node until a leaf node is reached that was determined by properly matching the feature values of the input to the classes assignment at the leaf node.

5. Results

5.1. Results from KNN

After successfully training , we got an accuracy of about 92 % with nearly 4 false positives

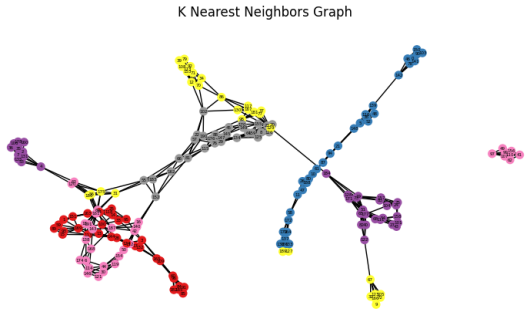

Figure 6. Visualization of clusters/graphs obtained by KNN

5.2. Results from KD-Trees

After successfully training , we got an accuracy of about 85 % with nearly 2 false positives

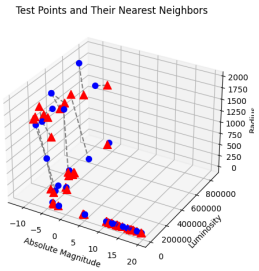

Figure 7. Visualization of test points with nearest train points

5.3. Results from Decision Trees (depth=3)

After successfully training , we got an accuracy of about 77 % with nearly 11 false positives

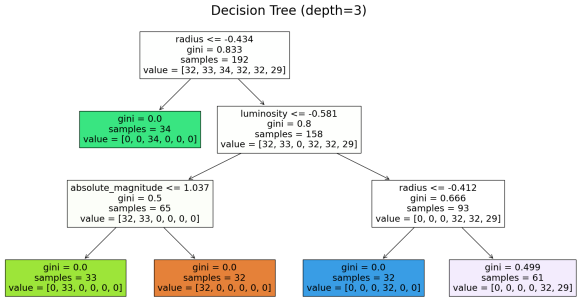

Figure 8. Decision Tree with $depth = 3$

6. Conclusion

The approach used in the paper merges KNN, KD-Trees, decision trees and statistical analysis of these methods in the galaxy star type classification leading to its high efficiency and applicability to the high dimensional data. In this way of encapsulating the entire process we can invent a scalable solution for astronomical research that will speed-up the process and help us with attaining the data on the composition of galaxies and their dynamics which then in turn will enable the development of astrophysics.

References

[1] Martin Skrodzki , "The k-d tree data structure and a proof for neighborhood computation in expected logarithmic time ", March-2019

[2] Types of Stars by NASA https://science.nasa.gov/universe/stars/types/