Globular Cluster Analysis

Nga Ying Lo, Katharina Scheidt, Emily Wilbur
Supervised by Adriano Agnello

Special thanks to Zoe Ansari & Vadim Rusakov

Overview

- Goals
- Data collection and preparation
- Tensorflow NN
- Clustering
  - Data dimension reduction
  - Algorithms

Project Goals

- Use clustering algorithms to identify Globular Clusters (GCs)
  - GCs: clusters of low-mass stars.
  - How? Aperture magnitudes

- Compare clustering results with a classification output from a Tensorflow Neural Network (NN).

Data Preparation

Obtaining Data

- From HST/ACS Coma cluster Treasury Program
- Magnitudes measured in *I* and *g* filters

ID GC candidates

- Upper bound on I_{mag}
- Color: *g-I*
- Concentration index → how concentrated is the brightness?
- Weak → ~7000 GCs
- Strong → ~1000 GCs

ID galaxies

- Coordinates: RA & DEC

Data reduction

- Removing extreme values {-9999.0, 99.0} in aperture magnitude columns
- ~78,000 → ~40,000 objects

Methods

- Data is too large to use without lowering dimensions

  → 26 variables and ~78,000 objects

  → Tried: PCA, kPCA, tSNE, and 🌟UMAP🌟.

- Various clustering algorithms, including: Agglomerative Clustering, kMeans, kNN, OPTICS, OPTICS_DBSCAN, 🌟hDBSCAN🌟, 🌟Gaussian Mixture🌟, and 🌟Bayesian Gaussian Mixture🌟.

- Tensorflow NN for classification and comparison.

Dimension Reduction Methods - UMAP

Best dimension reduction and mapping was achieved with 🌟UMAP🌟:

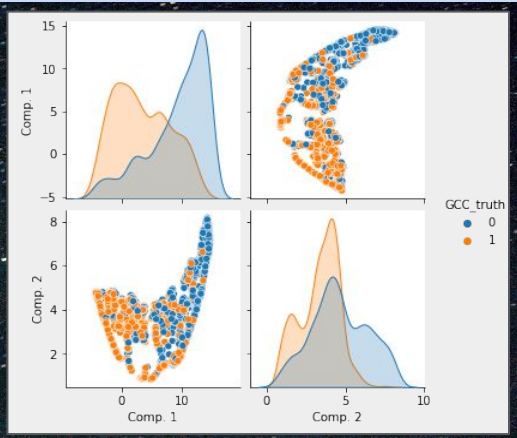
→ Required using a “balanced” dataset

→ our dataset is **highly imbalanced**:

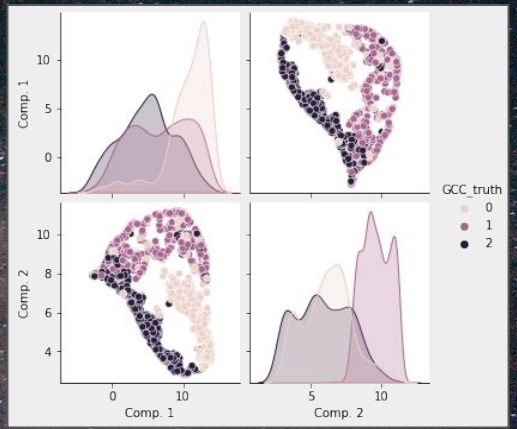
36978 non-CGs and **only 914** GCs!

→ created dataset from 914 GCs and 914 randomly selected non-GCs

2 clusters



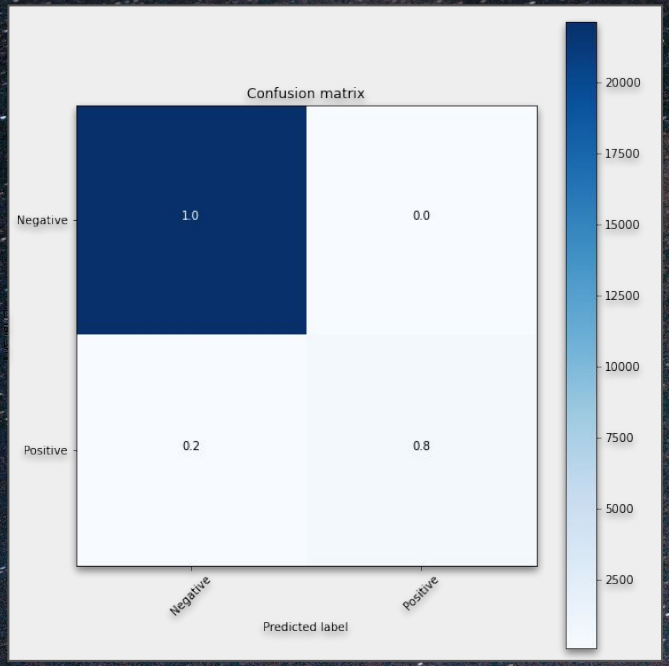
3 clusters

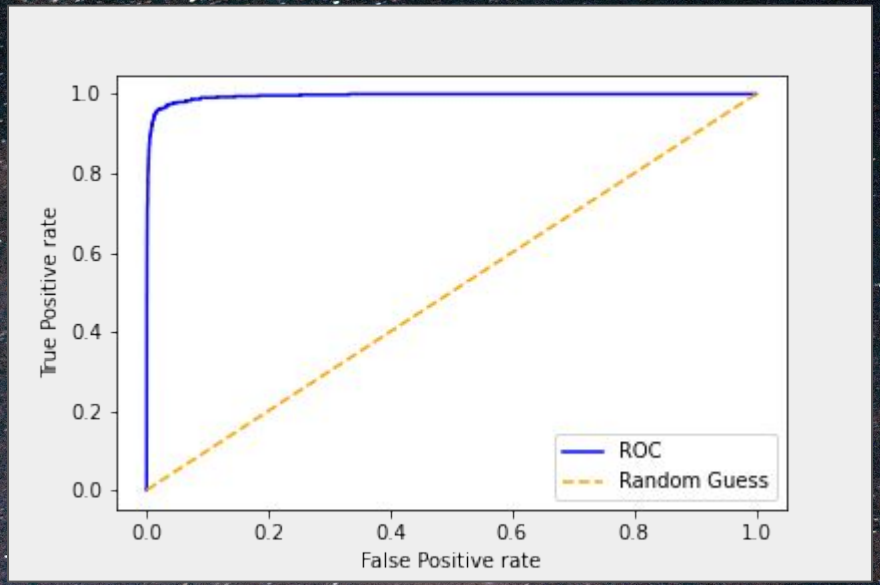


Tensorflow Classification

- Total objects (~40,000): Train (40%) and test (60%)
  - Trained on lower estimates for GC candidates
- Use 15/26 aperture magnitude features

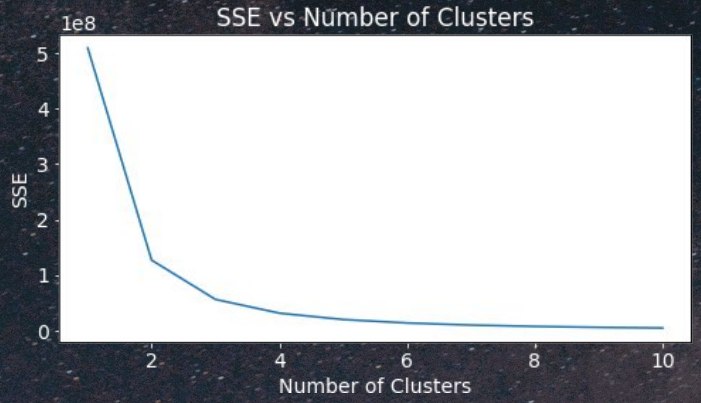
Score	Accuracy score	AUC	Cross entropy
Testing	99.29%	0.995	0.026





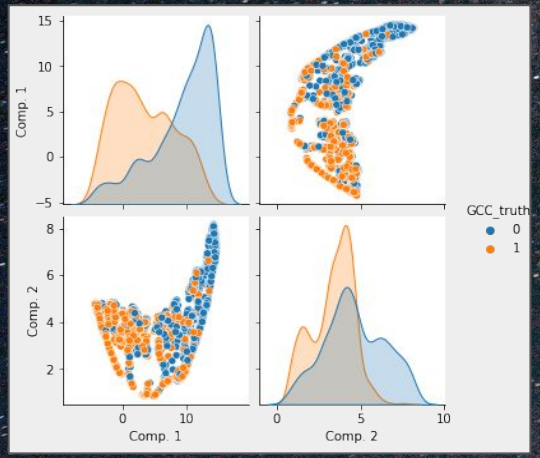
k-Means Clustering

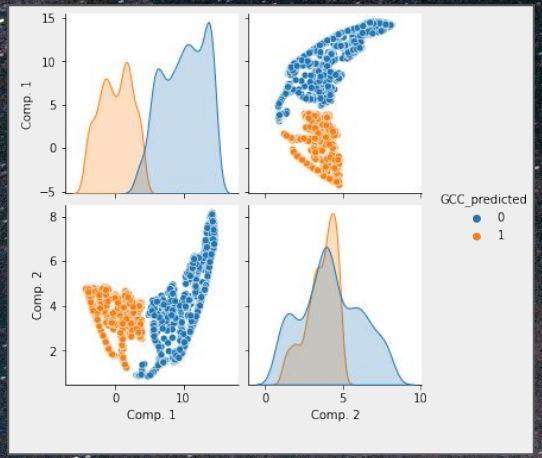
- Not used for actual clustering, but used to inform our expectations for other clustering algorithms.
- Results from k-Means showed optimal number of data clusters to be about 2 or 3:

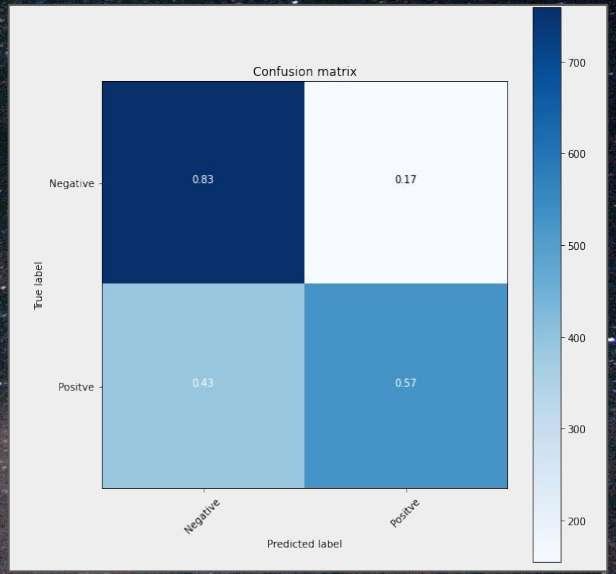


hDBSCAN Clustering

2 clusters - 0: non-GC / 1: GC
Separation visible, However we expected at least 3 clusters from kMeans



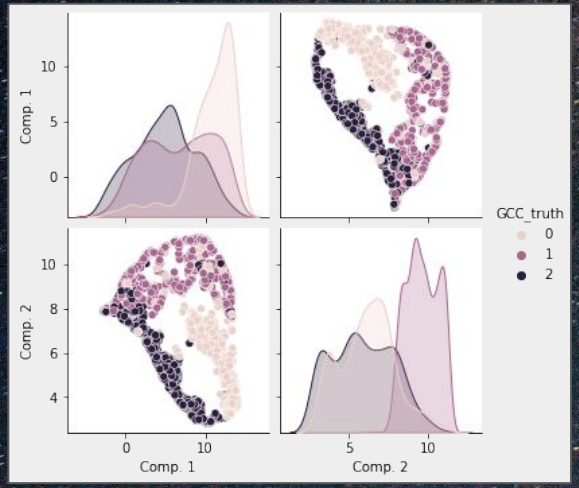


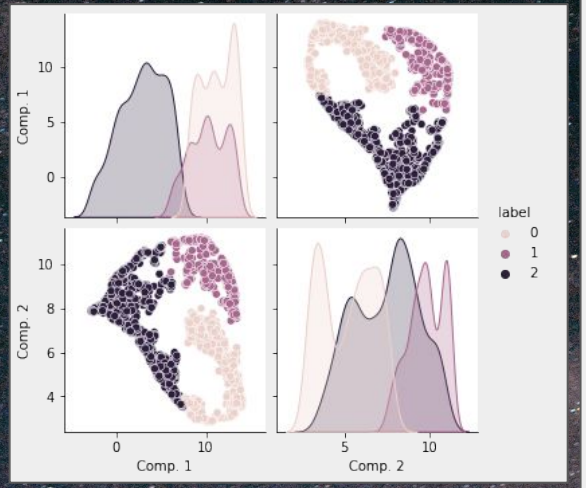


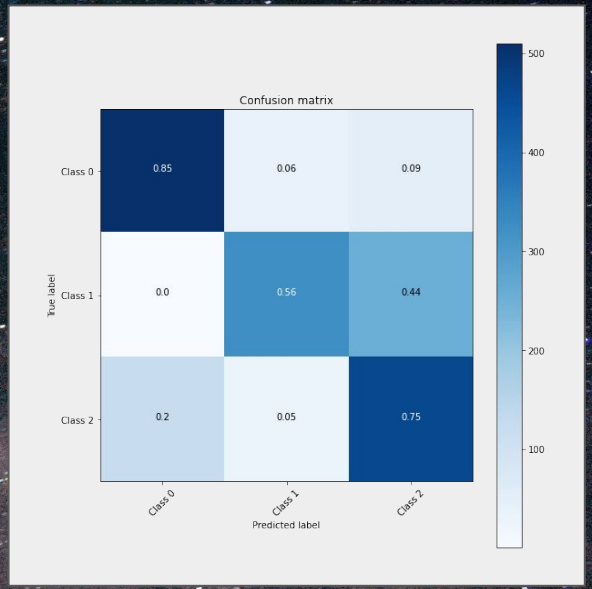
hDBSCAN Clustering

3 clusters - 0: non-GC/non-galaxy / 1: GC / 2: galaxy

Separation less visible,
All objects were clustered







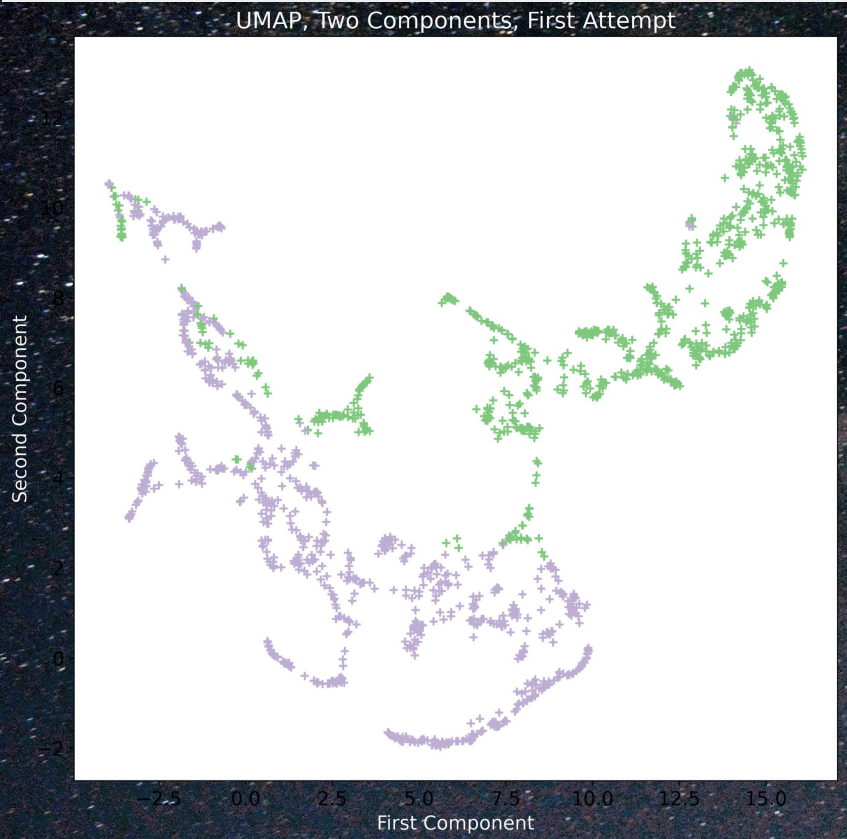
Gaussian Mixture Model Clustering

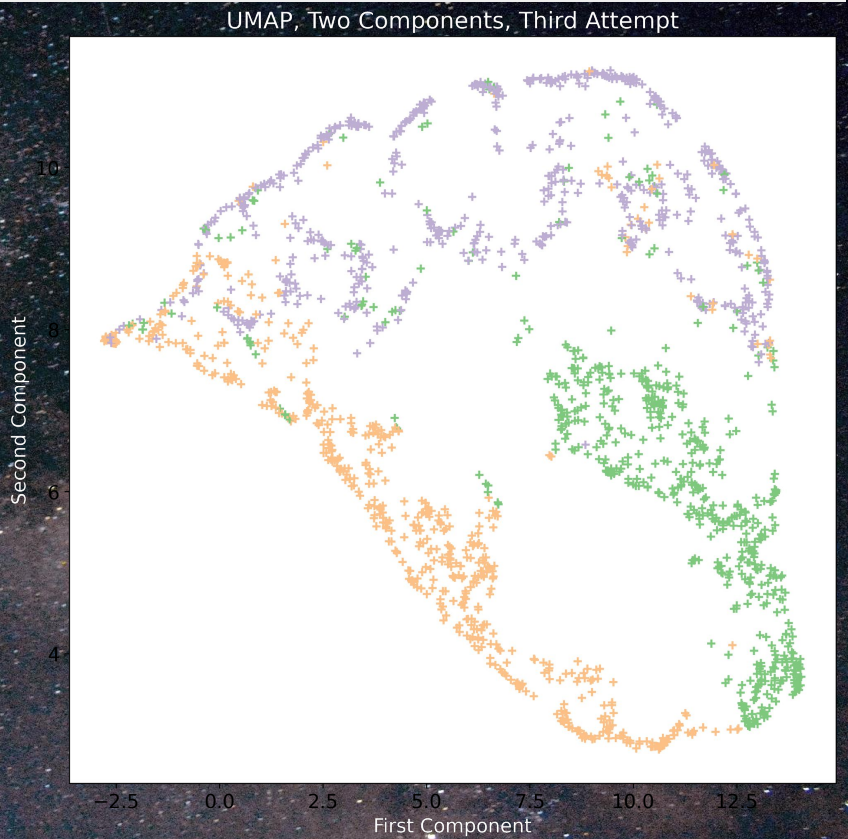
- Used on UMAP reduced components with balanced, smaller dataset.

- Employed two versions: simple GMM and Bayesian GMM with a Dirichlet Prior. Adding the Dirichlet prior increases our accuracy.

- These algorithms use (gaussian) probability distribution to determine clusters.

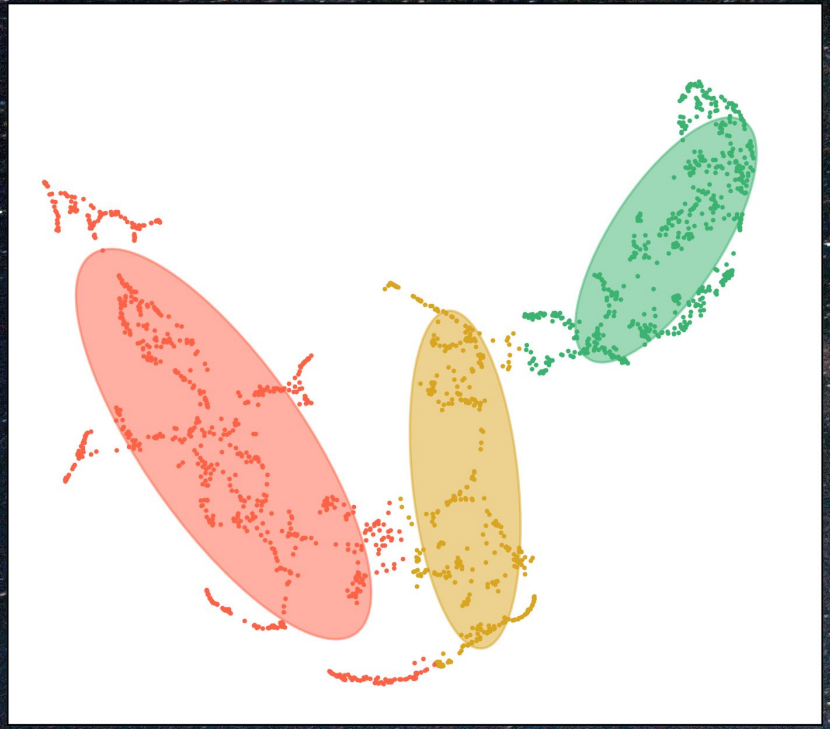
Gaussian Mixture Model Clustering



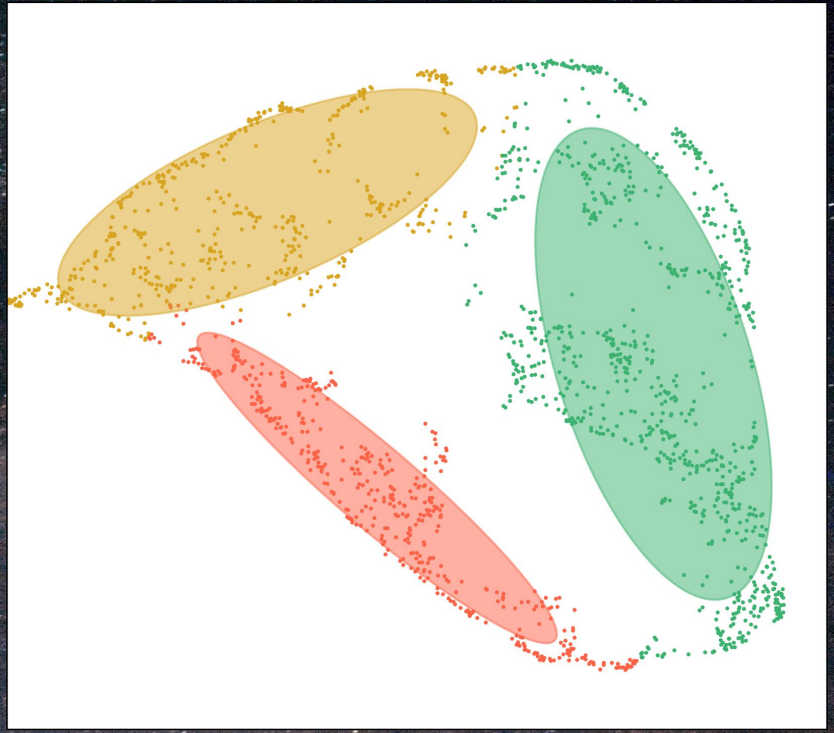


Gaussian Mixture Model Clustering

Bayesian GM, 3 clusters



Bayesian GM, 3 clusters



Conclusions

- Total methods of dimension reduction: 4
- Total of 9 algorithms: 8 clustering (unsupervised), 1 NN (supervised)
- NN works well on dataset with strong constraint on GCs → overall, underestimates the true number of GC.
- **UMAP gave the most clear data separation from dimension reduction.**
- HDBSCAN could identify 2 / 3 clusters, however just on a reduced dataset.
- **Bayesian GMM algorithm seems to have been best able to cluster our data.**

Next Steps

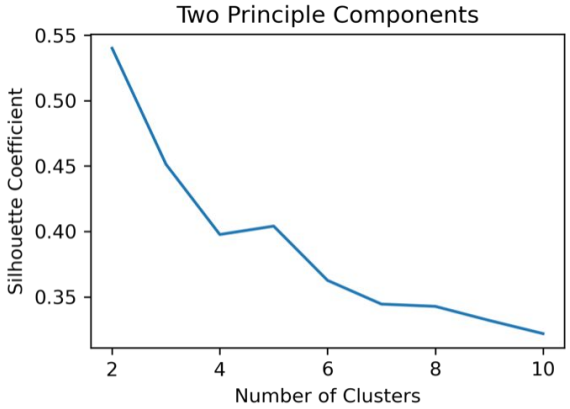
- Better-constrain the truth labels for GC candidates.
- Train NN on larger set of GC candidates
- Use other methods of dimension reduction and clustering.
  - Maybe a Self Organized Map?
- Use GPUs to make time-consuming computations more accessible.

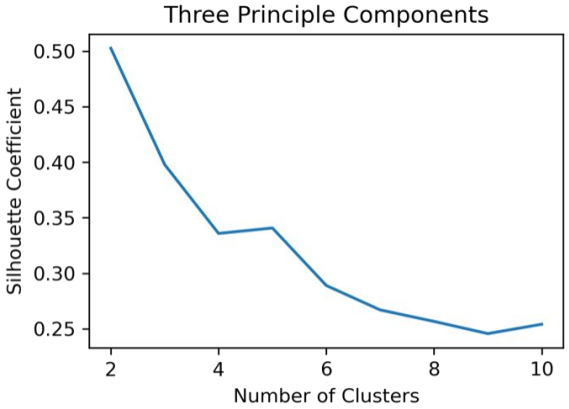
Comments

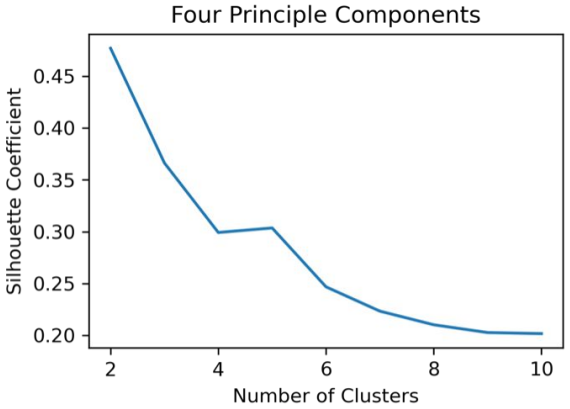
- Difficulties with data preparation:
  - Estimating GC candidates
  - Coordinate cross-validation of galaxy ID
  - Many meaningless aperture magnitude values
- Difficulties with dimension reduction:
  - Cluster separation: most dimension reductions gave one cluster of GCs mixed with galaxies
- Final clustering analysis used only 2000 of 78,000 objects
- We didn’t sleep for a week.

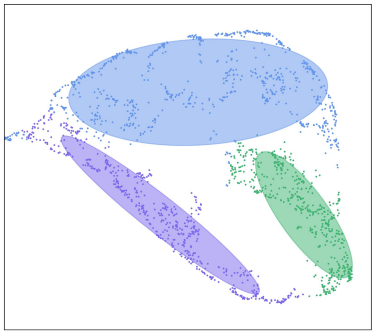

Questions or comments?
Thanks to Zoe & Adriano

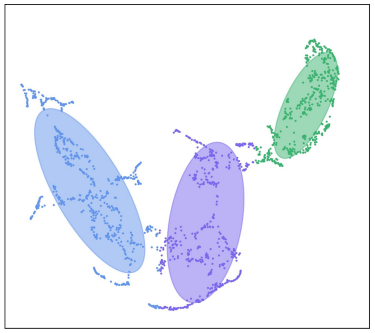
Appendix

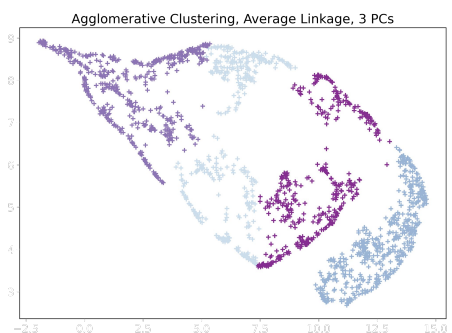










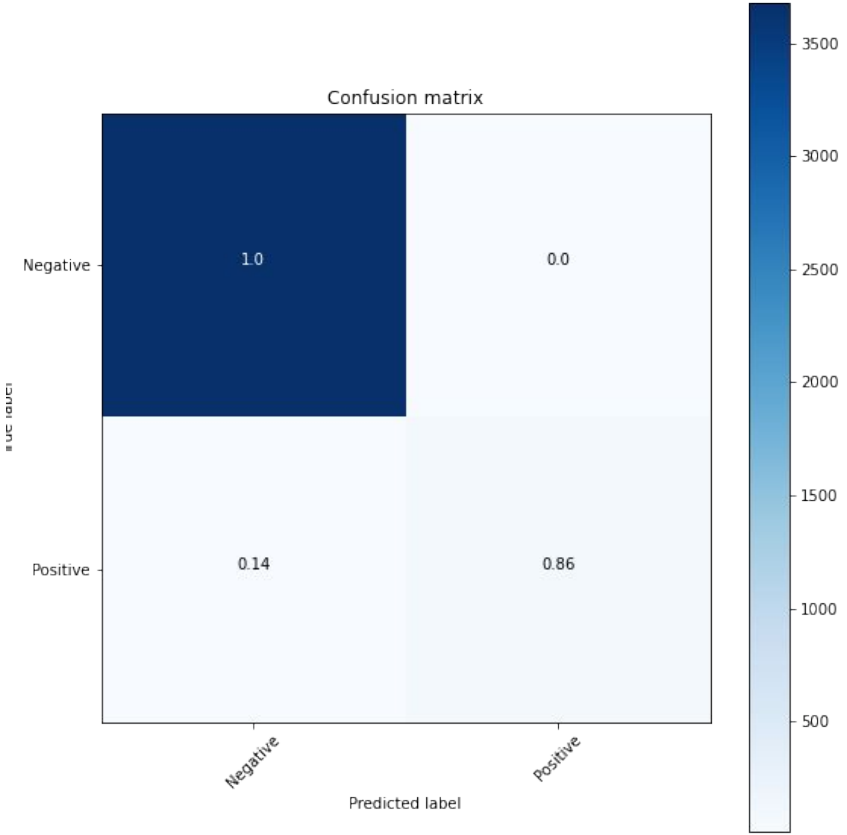


Top: silhouette scores for kMeans, each plot for a different # of principal components.

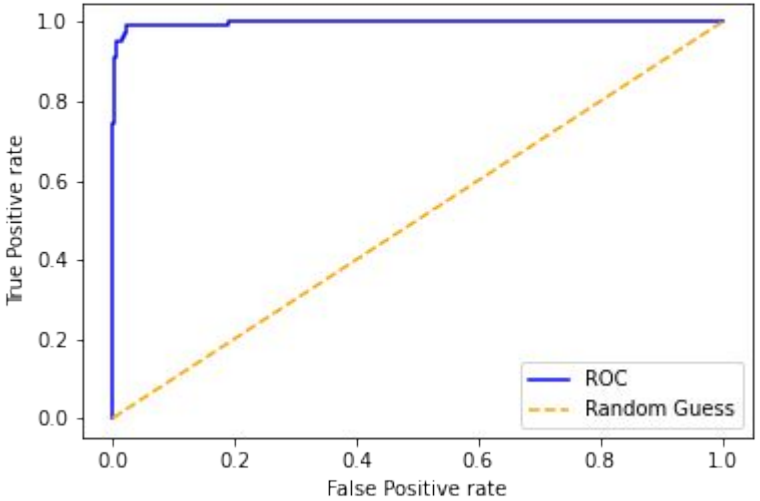
Left: simple Gaussian MM without Dirichlet process prior for the two cases shown previously.

Right: Attempt at Agglomerative Clustering on a UMAP reduction not shown here (it was actually surprisingly accurate!).

Appendix



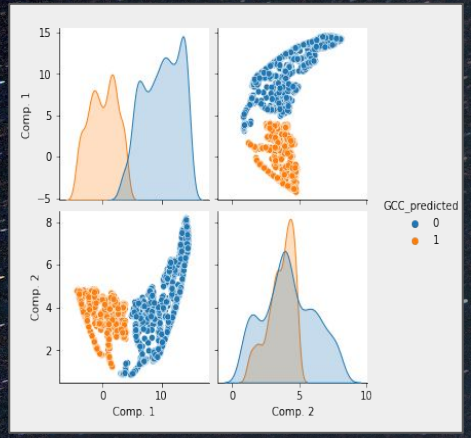
Score	Accuracy score	AUC	Cross entropy
Training	99.37%	0.997	0.023



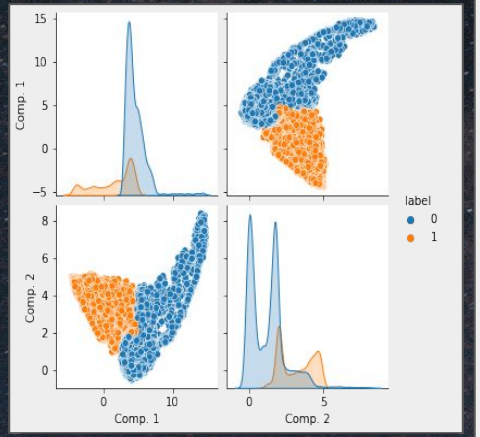
Appendix

- **kNN:** attempt to use the labels achieved by hDBSCAN and the reduced dimension dataset (using UMAP) to train a kNN

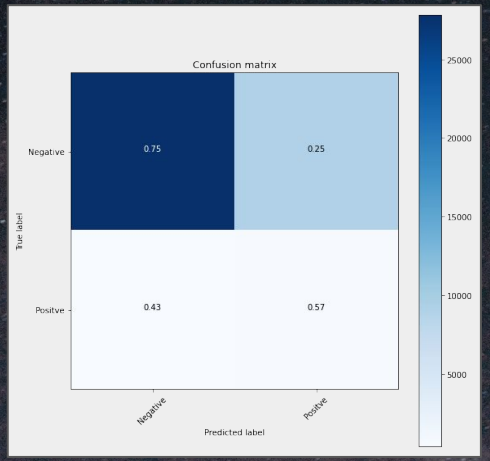
  → this worked out “nicely” using two clusters

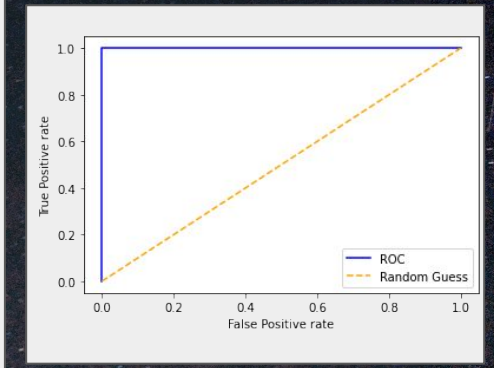


hDBSCAN prediction



kNN





Appendix B: Project Description

Clustering on Coma Cluster Data

Aim of the study

- Identifying globular clusters (GCs) among other astronomical objects using machine learning on photometric data from Hubble Space Telescope/Advanced Camera for Surveys (HST/ACS) Coma cluster Treasury Program
- perform clustering analysis in aperture magnitude space in order to possibly separate galaxies, isolated stars, isolated GCs, stars in front of a galaxy and GCs on a galaxy → this may flag up unknown UDGs

Dataset

- Catalogs of observed objects detected and measured by SExtractor From HST/ACS Coma cluster Treasury Program [1]
- A Catalog of structural parameters of 8814 galaxies in the 25 observed field-of-view [1]
- Objects we were looking at:
    - the measured and known galaxies from the catalog
    - GCs
    - other astronomical objects such as stars, clusters in front of galaxies, etc.

Methods

1. Getting an approximate set of 'truth' data for GCs by setting the following criteria:
    a. I magnitude < 26.5 [3]
    b. color: 0.6 < *(g-I)* < 1.5 [3]
    c. concentration index:
        i. $C_{3-9} = 0.45 \pm 0.2$[3][1]
        ii. $C_{4-7} < 0.3$[4][2]
2. Cleaning the data, by
    a. removing rows/objects with aperture magnitudes of values -9999 OR 99
    b. (i) removing rows/objects with aperture magnitudes of values -9999 AND 99, (ii) for rows containing either -9999 or 99, replace those values with the column mean[3]
3. Reducing the number of features to 10/15 by
    a. looking at pairplots from **seaborn** (10): MAG_AUTO, MAG_AUTO_CORRA, MAG_AUTO_CORRB, MAG_PETRO, MAG_APER_1, MAG_APER_2, MAG_ISO, MAG_ISO_F475, MAG_ISOCOR, MAG_AUTO_F475

---

[1] 1.a, 1.b, 1.c.i is a stronger constraint criteria set which gives a low estimate of GC candidates
[2] 1.a, 1.b, 1.c.ii is a weaker constraint criteria set which gives a higher estimate of GC candidates
[3] This doesn't really help in the clustering analysis since the dataset was really big for the algorithms to handle; using a reduced dataset by deleting all rows containing -9999 or 99 enhanced the clustering significantly

    b. calculating feature importance from the NN model using **eli5** (15): MAG_APER_9, MAG_APER_3, MAG_APER_4, MAG_APER_2, MAG_APER_3_F475, MAG_APER_1_F475, MAG_ISO_F475, MAG_AUTO_F475, MAG_AUTO_CORRA, MAG_APER_2_F475, MAG_APER_1, MAG_APER_4_F475, MAG_AUTO_CORRB, MAG_AUTO, MAG_PETRO
4. Balancing the data: Our data is originally highly unbalanced, since number of galaxies is a lot more higher than the number of GCs → this was only applied to the data we treated with the UMAP and t-SNE
5. Some dimension reducing algorithms before clustering:
    a. PCA: reduced data dimensions to 2, 3, and 4. Did not show data separation.
    b. kPCA: run on Erda on 40% of the data and reduce dimensions to 2, 3. Did not show good separation.
    c. **t-SNE:** can deal with a high dimensional space (but only with several thousands of data points), couldn't separate data to satisfactory degree [5]
    d. UMAP: can also deal with high dimensional space; faster than t-SNE and preserves more of the global structure. [6]
6. Clustering methods:
    a. **kMeans**: this gave us an approximate number of expected clusters (3 clusters).
    b. OPTICS_DBSCAN: widely used for similar problems. [7]
    c. hDBSCAN: extension of DBSCAN (converts it into hierarchical clustering algorithm)
    d. kNN: non parametric classification method, trained with labels achieved with hDBSCAN
    e. Gaussian Mixture Model Clustering: can use a training period for "learning" combined with traditional clustering approach for greater accuracy. Scores samples based on cluster assignment. [7]
    f. Bayesian Gaussian Mixture Model Clustering: uses a dirichlet process prior to raise the accuracy of the Gaussian MM clustering method. Scores samples based on cluster assignment. [7]
7. Classification method:
    a. Tensorflow Neural Network: Train on 40% of the data and test on 60%. Used strong constraint criteria set (Methods 1.a, 1.b, 1.c.ii)
        i. Accuracy: 99.30% (sklearn.metrics.accuracy_score)
        ii. AUC: 0.993 (sklearn.metrics.auc)
        iii. Cross entropy: 0.020 (sklearn.metrics.log_loss)

Short summary of results

- Neural Network: Perform well for predicting GCs and non GCs objects using 15 aperture magnitude features in our data set and our stronger constraints for GC candidates, which yields a lower number of 'true' GCs. Underestimates the true number of GCs.
- Best clustering results were achieved with the Gaussian Mixture Model and the hDBSCAN which identified 2-3 clusters on a UMAP dimension reduced (to two components) balanced dataset.
- Accuracy scores compared to our approximate truth were:
    - Gaussian Mixture Model:

        i. Accuracy: 90.0% (sklearn.accuracy_score), full dataset
    - hDBSCAN:
        i. Accuracy: 70.0% (sklearn.accuracy_score) for 2 clusters, balanced dataset
        ii. Accuracy: 72.1% (sklearn.accuracy_score) for 3 clusters, balanced dataset
    - kNN: Accuracy: 67.3 % (compared to original truth data), n_neighbors =15

- From confusion matrices we saw that more objects were classified as Globular Clusters than present in our approximate truth data form which we could infer that there are more Globular Clusters than primarily estimated
- Problems we faced during our studies:
    - No truth data for GCs are available, our estimated truth labels were obtained following the criteria in Peng et al. 2011 and Lim et al. 2018
    - Cross-validate coordinates of galaxies from the galaxy catalog and catalogs of all observed objects was time-consuming
    - our data contains a large amount of data we couldn't use (-9999, and 99 values) and we had to discard a big part of our data, which in general one would like to avoid for Machine Learning Approaches
    - even after using dimension reducing algorithms it was quite difficult to separate clusters from each other; in the most cases we found one big "cluster" with several agglomerations of GCs and galaxies
    - for our final clustering we only used a minor part of the dataset, since we used a balanced data set containing only approximately 1000 clusters and 1000 galaxies (compared to a dataset of 77000 samples)

Future Work

- Train Neural Network on identifying GCs using weaker constraints
- iteratively cluster all galaxies, by selecting several subsets of galaxies in the dataset
- compare the prediction scores from NN and Bayesian Gaussian Mixture Model: this was not done because output of GMM was a balanced dataset (equal number of objects of possibly GCs and non-GCs objects) whilst the output of NN was not
- Use better constraints for GCs as potential truth label
- Try more variation of dimension reduction methods
- Parallelization and utilization of GPU to save computation time

References

- [1] 2011 Mikulski Archives for Space Telescopes https://archive.stsci.edu/prepds/coma/datalist2.2.html
- [2] Amorisco N.C. and Leob A. 2016 MNRAS 459 L51
- [3] Eric W. Peng et al 2011 ApJ 730 23
- [4] Sungsoon Lim et al 2018 ApJ 862 82
- [5] Wattenberg, et al., "How to Use t-SNE Effectively", Distill, 2016. http://doi.org/10.23915/distill.00002
- [6] McInnes L. and Healy J. 2018 *ArXiv e-prints 1802.03426*

- [7] Pedregosa, F. et al, “Scikit-learn: Machine Learning in Python.” Journal of Machine Learning Research. Vol. 12. p.2825-2830.