Exoplanets

Master degree in Computer Engineering for Robotics and Smart Industry

Valentini Michel VR472456
Canella Giacomo VR481630

September 2022

Contents

1 Motivation and rational

All the planets in our solar system orbit the sun. Planets orbiting other stars are called exoplanets under NASA's Exoplanet Exploration Program.

Exoplanets are very difficult to see directly with telescopes. They are hidden by the brightness of the star they orbit. It can take some time for a exoplanet to be confirmed as such.

Nowadays, scientists and astronomers have attached great importance to the task of discovering new exoplanets, even more so if they are in the habitable zone. Most of the exoplanets discovered so far are found in a relatively small region of our galaxy, the Milky Way. To date, NASA has confirmed 4301 exoplanets, using a variety of discovery techniques, including planetary transits, radial velocities, gravitational microlensing and direct imaging from databases provided by space and ground-based telescopes, e.g. NASA's Kepler space telescope and the NASA's Transiting Exoplanet Survey Satellite.

The discovery of new exoplanets has taken a high degree of importance during the last few years. Since the amount of data provided by telescopes is enormous, the only way to analyze it is using Machine Learning techniques.

2 State of art

In the literature, different approaches that use artificial intelligence techniques to detect exoplanets can be found. One example is detecting exoplanet transits by applying the k-nearest neighbors (kNN) method to determine whether a given signal is sufficiently similar to known transit signals. Another example is the use of the Random Forest Classifiers (RFCs) algorithm for exoplanets classification. They achieve an overall error rate of 5.85% and an error rate in the classification of exoplanet candidates of 2.81%. A further step shows a combination of RFCs and Convolutional Neural Networks (CNNs) to distinguish between the different types of signals. The authors say that the combination of both methods offers the best approach to identify exoplanets correctly in the test data approximately 90% of the time. Another example is a CNN based approach that is capable of detecting Earth-like exoplanets in noisy time series data with a greater accuracy than a least-squares method.

3 Objectives

The aim of this work is to perform a comparison of different classical Machine Learning algorithms with the objective to understand which is the most effective (in terms of accuracy) in the classification of exoplanets.

4 Methodology

4.1 Dataset

The dataset is composed by 9564 samples, each one representing an object acquired by a satellite. Every samples has 50 feature. One of this, labeled by "koi_disposition", is defined by three wordings: "confirmed", "candidate", "false positive". This specific feature establishes the 3 class' labels. From the remaining ones (49) we decided to discard all the features containing a string value. Subsequently, observing the disparity between the number of samples of the 3 classes, we decided to delete 2500 samples from the "false positive" class, in order to balance the dataset. As a result we end up with a dataset of 7064 samples, 42 feature and 3 classes. Finally we splitted the dataset in 80% of samples for the training and 20% of samples for testing.

source for dataset: `https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=koi`

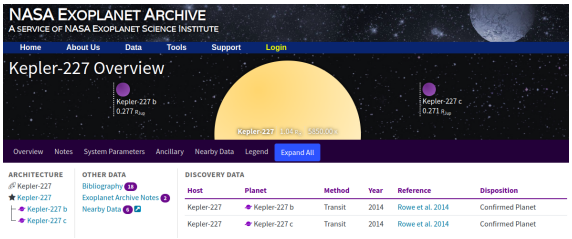

Figure 1: Confirmed exoplanet

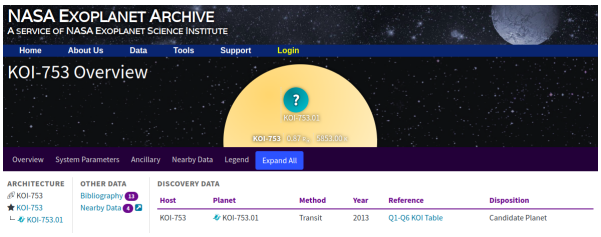

Figure 2: Candidate exoplanet

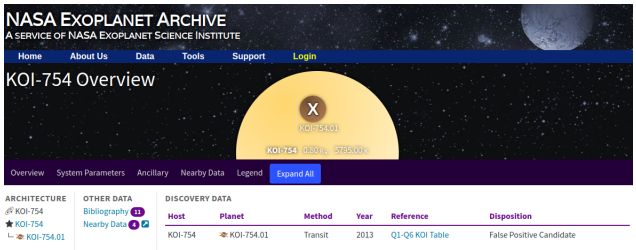

Figure 3: False Positive

4.2 Algorithms

In this section there is an overview of the algorithms we applied to our data-set, the specific results are in the results‘ section, also refer to the conclusions‘ section for furthers comments.

Except for Parzen Window, each algorithm has been tested both with and without a previous dimensionality reduction (principal component analysis (PCA) and linear discriminant analysis (LDA)).

4.2.1 SVM

Tested with different kernels, we got the best result (82% of accuracy) with polynomial of 5th degree, this without a previous dimensionality reduction (generally all kernels worked better without LDA and PCA). Sigmoid gave worst results in terms of accuracy, but this is reasonable due to the fact that this kernel is more suitable for binary classification, in our case we have to deal with three classes.
Kernels used: "Rbf", "Poly3", "Poly5", "Sigmoid", "Linear".

4.2.2 KNN

Tested with different values for k parameter (number of neighbors) and different kernels. Comparing the results we obtained the best possible accuracy value of 78%, with kernel = “Manhattan” and k = 9. As the SVM algorithm, KNN worked better without a previous dimensionality reduction of the data.
Kernels used: "Euclidian", "Manhattan", "Chebyshev", "Minkowski".
Number of neighbours: 7, 9, 11, 13, 15.

4.2.3 K-Means

We analyzed the clustering generated by the algorithm in terms of accuracy (see conclusion‘s section for a more detailed explanation) obtaining a value of 43%. We also decided to evaluating the resulting cluster in terms of silhouette (average silhouette 0.61 for 3 clusters) and adjusted random index (resulting ARI = 0.16).

4.2.4 Parzen Window

We tested this algorithm with three different kernels and different values for bandwidth "h", best result (accuracy 42,4%) with kernel = rectangle and h = 0.01.
Kernels used: "rectangle", "gaussian", "exponential".
Bandwidth h: 0.01, 0.1, 1.

4.2.5 Multi-layer Perceptron

Tested with different types of solver and number of hidden layers, best result (85% accuracy) with solver lbfgs and 200 hidden layers.
Solvers used: "lbfgs", "sdg", "adam".
Number of hidden layers: 100, 200, 500, 800

5 Conclusions

Exoplanet classification is a new field of research, the available data keep increasing and changing with respect new discovers. The work done in this field aims to provide researchers a more valid method for discerning between an astronomical object and an exoplanet. What we have done was to follow this path by applying different types of machine learning algorithms and analyzing the results. We tested both supervised and unsupervised learning methods:
- SVM: analyzing our results we observed that SVM has worked better without a previous dimensionality reduction. We tried both PCA and LDA, the latter worked better. Our opinion is that, in this specific case, the high dimensionality of dataset's space helps in the discrimination of the classes through kernels functions.
- Knn: as for SVM, applying PCA leads to worst results, we got better performances considering all the features but also reducing the dimensionality through LDA.
- K-means is more suitable for clustering purpose rather than classification, the algorithm has no knowledge of dataset's classes, it only groups samples depending on their similarity. Nevertheless you can still exploit your labels: once you have the clusters created by the algorithm (in our case forcing the number of clusters to be equal to the number of our classes), what you can do is create a rule to determine which cluster corresponds to which classes, for example based on occurrence of each classes in a specific cluster. We did this in order to analyze the results in terms of accuracy and also to understand how well separated are the classes in our dataset. The low value obtained for accuracy implies that our data are not well separated in the space. However we decided also to verify the goodness of the clusters through some specific tools such as silhouette and ARI (adjusted rand index). Both values are considerably low, as a matter of fact, following the latter discussion, our dateset does not have have a clear separation of the classes.
- Parzen window is a non-parametric method to estimate a probability density function for a group of data, but it can also be used as a classifier (leaning to the Bayes' theory). We only tested it with a previous reduction of the number of features (forcing it to be 4) obtaining not as good results as for the other classifier, probably we didn't find a proper way to estimate the distribution of our data.
- Mlp: according to the state of art, artificial neural network are the optimal classifier for this type of data. We decided to give a try exploiting sklearn function for MLP classifier. As expected this classifier gives the best result in terms of accuracy.

A future work could be to test other supervised algorithms, maybe focusing on artificial neural networks and training the classifiers with the currently growing dataset downloadable at the site:

`https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=koi`

6 Results

In this sections are shown the results in terms of accuracy and confusion matrix. For convenience, we reported just the confusion matrix of the optimal case for each algorithm. For SVM we reported a graph with an overview of the accuracy for the different kernels. For KNN we reported a graph with an overview of the accuracy for different numbers of neighbours. For K-means we reported the graph representing the silhouette analysis for 3 clusters.

6.1 SVM without pca and lda

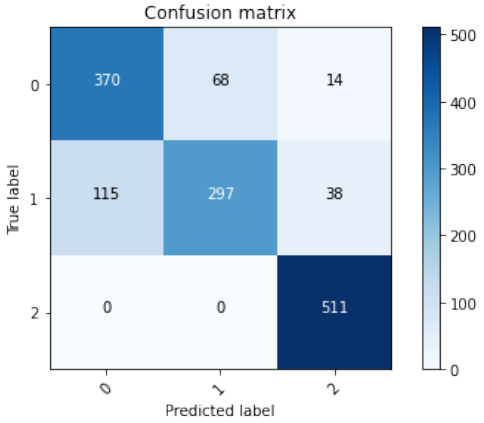

Figure 4: Best Confusion Matrix "RBF"

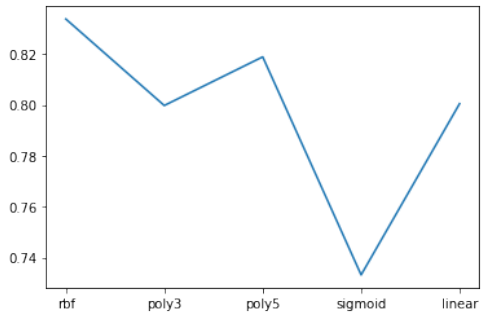

Figure 5: Accuracy graph

6.2 SVM after pca

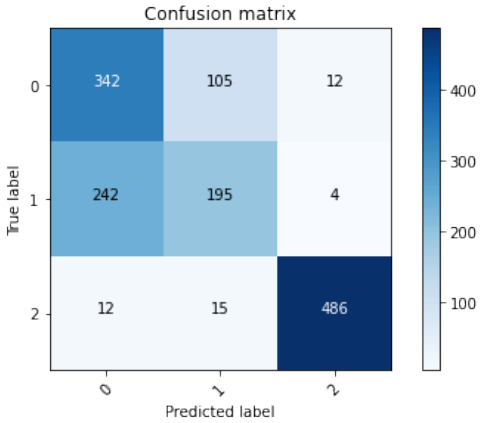

Figure 6: Best Confusion Matrix "poly5"

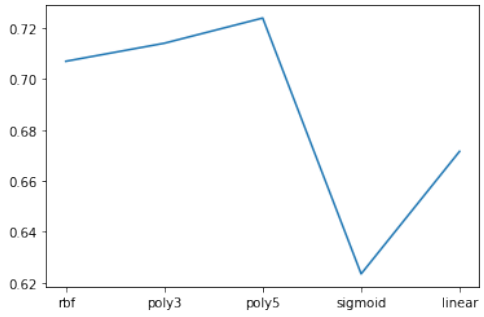

Figure 7: Accuracy graph

6.3 SVM after LDA

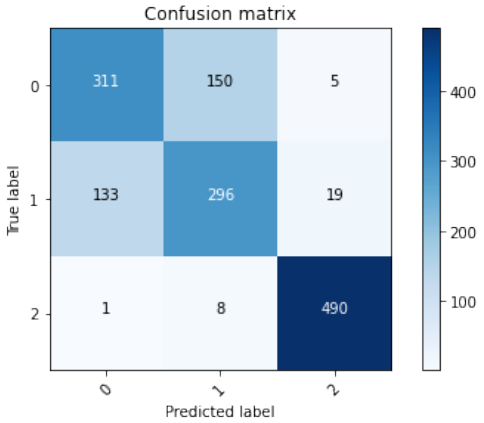

Figure 8: Best Confusion Matrix "poly3"

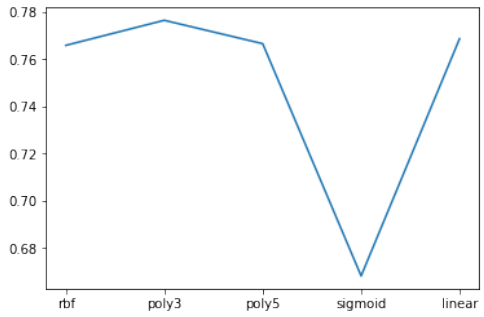

Figure 9: Accuracy graph

6.4 KNN without pca and lda

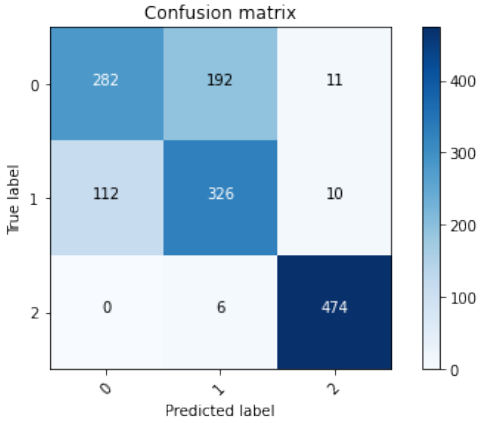

Figure 10: Confusion Matrix "Manhattan k=13"

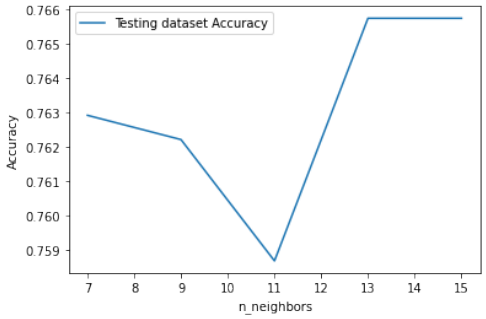

Figure 11: Accuracy matrix Manhattan

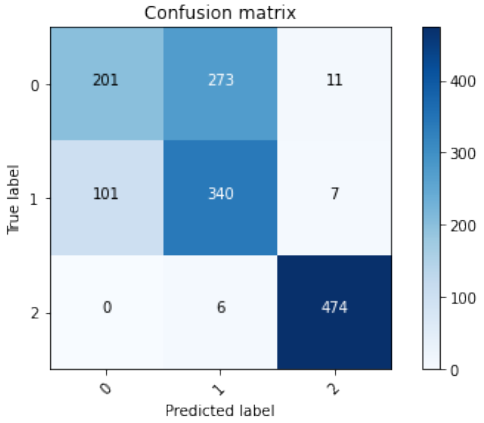

Figure 12: Confusion Matrix "Euclidian k=13"

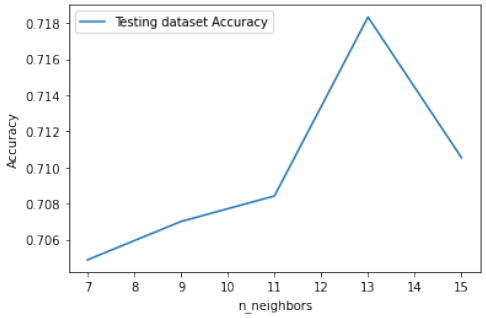

Figure 13: Accuracy graph

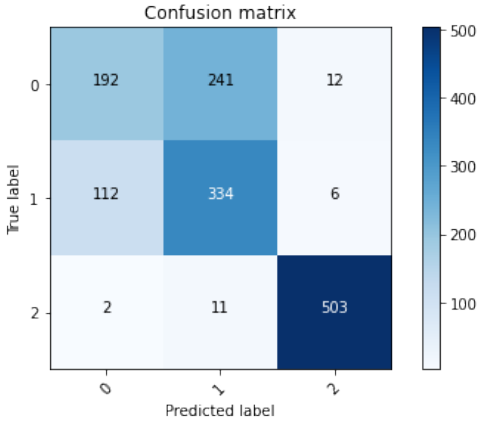

Figure 14: Confusion Matrix "Chebyshev k=13"

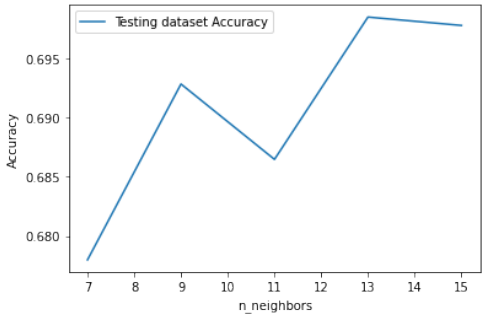

Figure 15: Accuracy graph

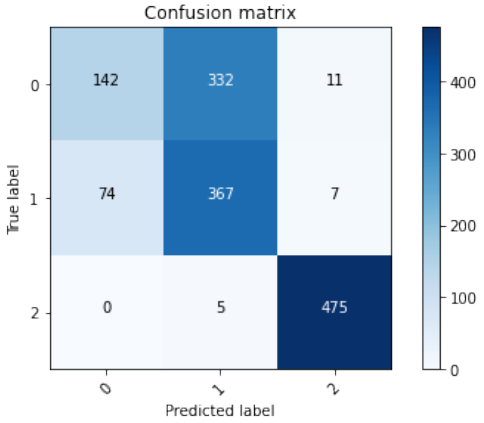

Figure 16: Confusion Matrix "Minkowski k=15"

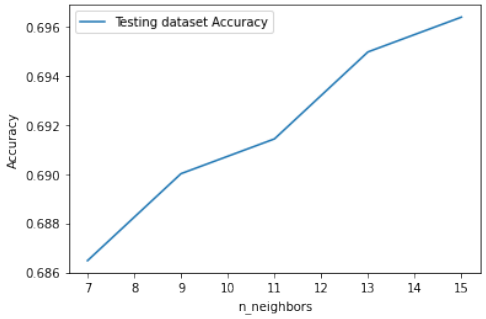

Figure 17: Accuracy graph

6.5 KNN after pca

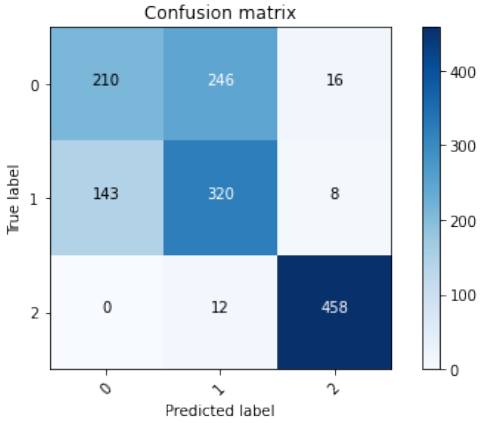

Figure 18: Confusion Matrix "Manhattan k=15"

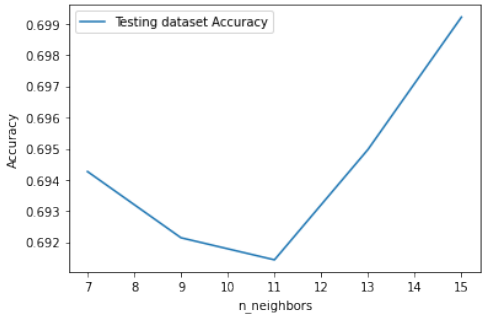

Figure 19: Accuracy matrix

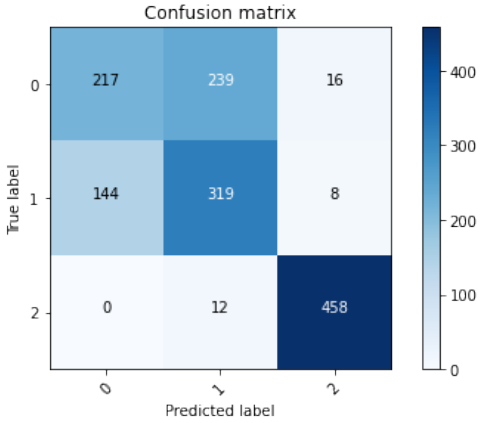

Figure 20: Confusion Matrix "Euclidian k=15"

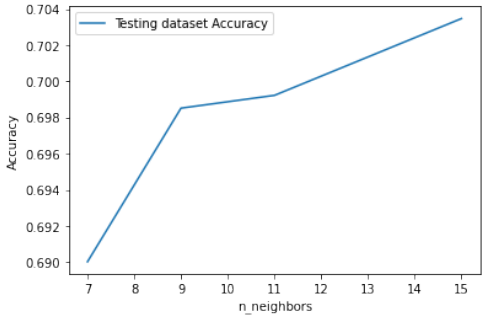

Figure 21: Accuracy graph

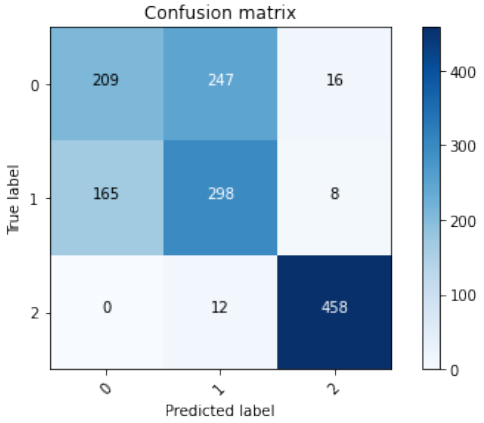

Figure 22: Confusion Matrix "Chebyshev k=15"

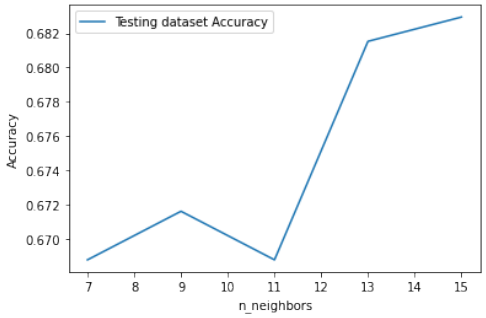

Figure 23: Accuracy graph

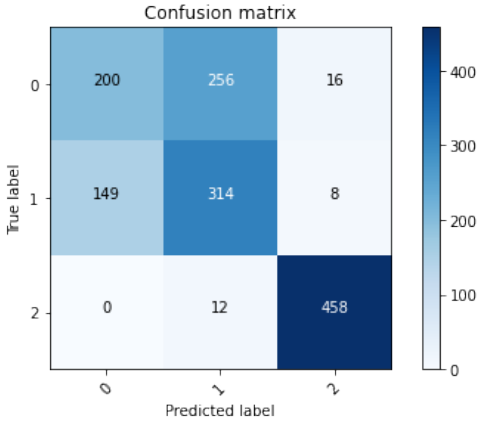

Figure 24: Confusion Matrix "Minkowski k=11"

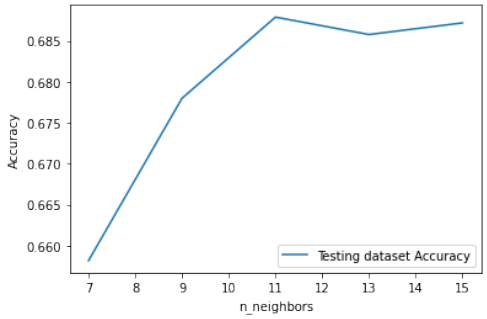

Figure 25: Accuracy graph

6.6 KNN after LDA

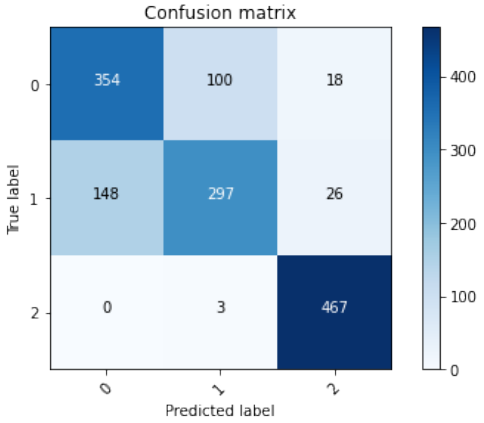

Figure 26: Confusion Matrix "Manhattan k=15"

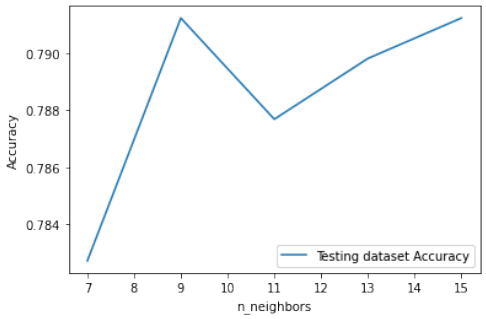

Figure 27: Accuracy matrix

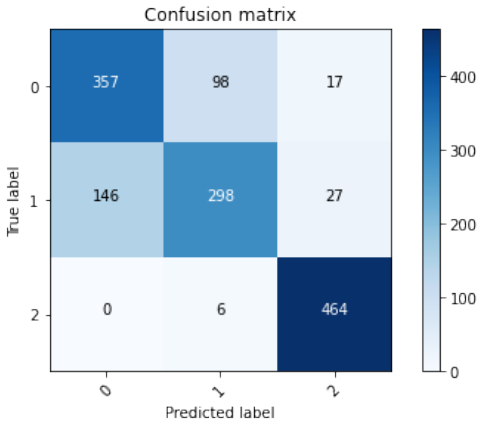

Figure 28: Confusion Matrix "Euclidian k=13"

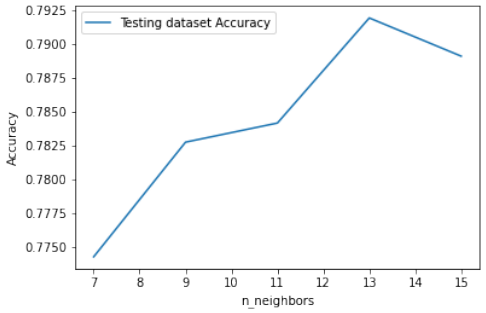

Figure 29: Accuracy graph

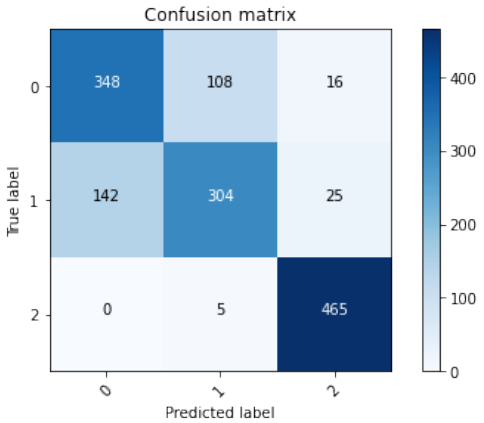

Figure 30: Confusion Matrix "Chebyshev k=11"

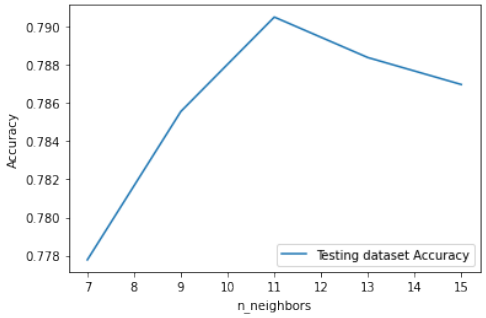

Figure 31: Accuracy graph

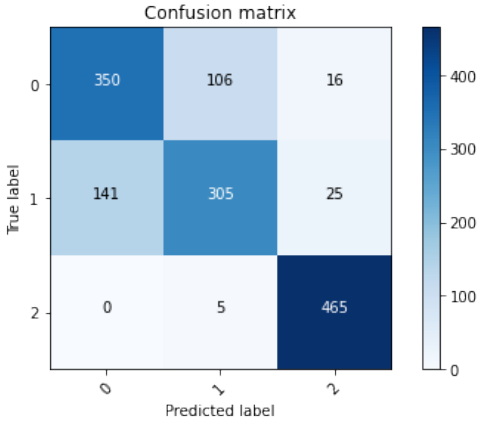

Figure 32: Confusion Matrix "Minkowski k=11"

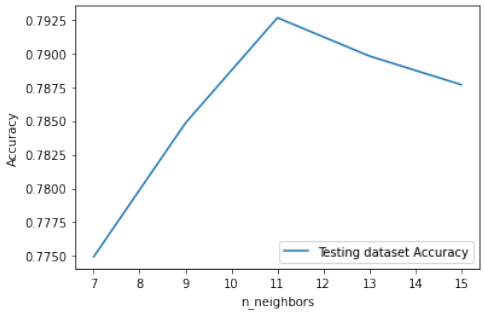

Figure 33: Accuracy graph

6.7 K-Means

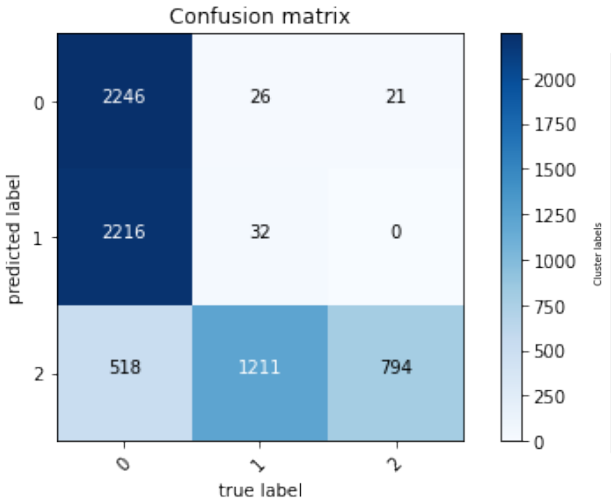

Figure 34: Confusion Matrix

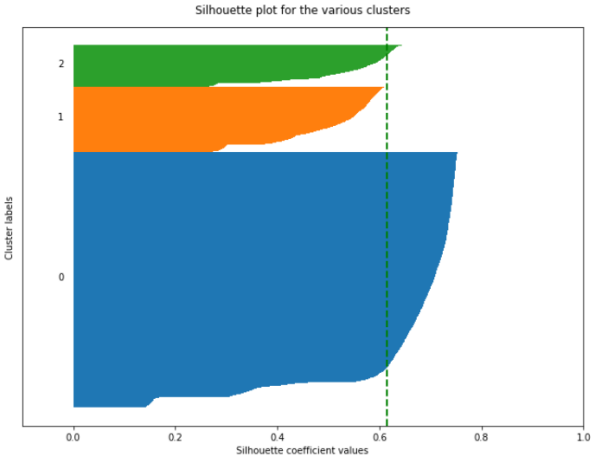

Figure 35: Silhouette with 3 clusters

6.8 MLP

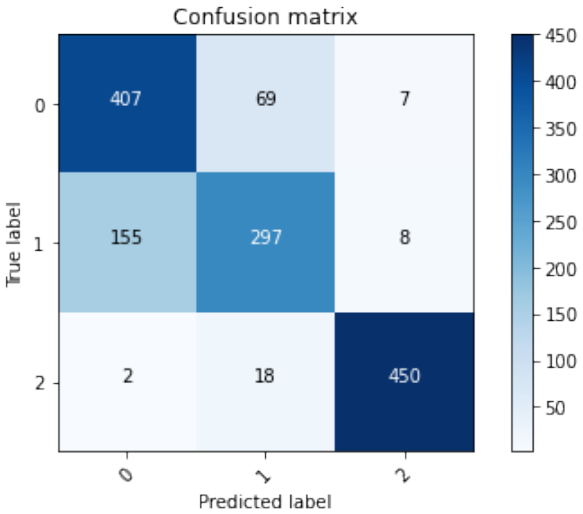

Figure 36: Confusion Matrix

6.9 Overview of algorithm‘s accuracy (in decimals)

Accuracy matrix KNN algorithm without dimensionality reduction :

_	Euclidian	Manhattan	Chebyshev	Minkowski
K=7	0.74	0.75	0.71	0.71
K=9	0.75	0.79	0.71	0.71
K=11	0.75	0.79	0.71	0.71
K=13	0.74	0.78	0.69	0.71
K=15	0.75	0.78	0.70	0.70

Accuracy matrix KNN algorithm with PCA reduction:

_	Euclidian	Manhattan	Chebyshev	Minkowski
K=7	0.70	0.69	0.67	0.66
K=9	0.70	0.69	0.67	0.68
K=11	0.70	0.69	0.68	0.69
K=13	0.70	0.70	0.68	0.69
K=15	0.69	0.67	0.68	0.69

Accuracy matrix KNN algorithm with LDA reduction:

_	Euclidian	Manhattan	Chebyshev	Minkowski
K=7	0.75	0.75	0.75	0.75
K=9	0.75	0.75	0.76	0.76
K=11	0.76	0.76	0.76	0.76
K=13	0.75	0.76	0.76	0.76
K=15	0.76	0.76	0.77	0.76

Accuracy matrix SVM algorithm:

_	RBF	Poly3	Poly5	Sigmoid	Linear
LDA	0.72	0.71	0.69	0.45	73
PCA	0.47	0.46	0.44	0.51	42
NO PCA	0.80	0.80	0.82	0.71	80

Accuracy matrix MLP algorithm without dimensionality reduction :

_	Lbfgs	Sgd	adam
layers = 100	0.85	0.80	0.77
layers = 200	0.85	0.80	0.73
layers = 500	0.85	0.81	0.78
layers = 800	0.83	0.81	0.74

Accuracy matrix MLP algorithm with PCA reduction :

_	Lbfgs	Sgd	adam
layers = 100	0.63	0.67	0.73
layers = 200	0.61	0.68	0.71
layers = 500	0.60	0.68	0.73
layers = 800	0.66	0.68	0.70

Accuracy matrix MLP algorithm with LDA reduction :

_	Lbfgs	Sgd	adam
layers = 100	0.30	0.31	0.31
layers = 200	0.30	0.32	0.32
layers = 500	0.31	0.31	0.31
layers = 800	0.31	0.31	0.31

Accuracy matrix Parzen Windows algorithm with PCA reduction :

_	rect	gaussian	exponential
h=0.01	0.42	0.28	0.32
h=0.1	0.21	0.34	0.34
h=1	0.27	0.47	0.32

[*]

References

[N S] D J A Brown Faedi Hay Hebb Kiefer Mancini Maxted Palle Pollacco Queloz Smalley Udry R West Wheatley N Schanche A Collier Cameron G Hébrard L Nielsen Triaud Almenara Alsubai Anderson Armstrong Barros Bouchy P Boumis. *Machine-learning approaches to exoplanet transit detection and candidate validation in wide-field ground-based surveys.* URL: https://academic.oup.com/mnras/article/483/4/5534/5199219.

[Obe] Abhishek Malik Benjamin P. Moster Christian Obermeier. *Exoplanet Detection using Machine Learning.* URL: https://arxiv.org/abs/2011.14135.

[Ren] Hippke Michael Heller René. *A box-fitting algorithm in the search for periodic transits.* URL: https://ui.adsabs.harvard.edu/abs/2019A%26A...623A..39H/abstract.

[Ris] Leon Ofman Amir Averbuch Adi Shliselberg Idan Benaun David Segev Aron Rissman. *Automated identification of transiting exoplanet candidates in NASA Transiting Exoplanets Survey Satellite (TESS) data with machine learning methods.* URL: https://arxiv.org/abs/2102.10326.

[] *Selecting the number of clusters with silhouette analysis on KMeans clustering.* URL: https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html#:~:text=Silhouette%20analysis%20can%20be%20used,like%20number%20of%20clusters%20visually.

[Per18] Michael Perryman. *The Exoplanet Handbook.* August 2018. ISBN: 9781108419772.