Exoplanets Classification E4

SUPPORT VECTOR MACHINE

Our objective here was to use a SVM classification algorithm shown in class. We wanted to use it in the merged_pca_df dataset and look after the PC1, PC2, PC3, PC4, PC5 and PC6 variables obtained in the PCA and the 'esi' variable obtained in the formula. The main problem is that 'esi' is a continuous variable and we can't perform a SVC in it. We decided to create a binary variable 'is_habitable' in order to perform a SVM. After creating it we performed the code in order to obtain best_score, best_params and final_score. We will use the results of best_params (C and gamma) to calculate the score with the kernels.

DECISION TREE

In our model due to our dataset properties we have considered that making a decision tree is so important. So with sklearn libraries we have created the SelectionTreeClassifier in order to get the accuracy and the recall of our model. Once we have got them, as we have seen, it is necessary to get the values that fit the most in our model. To achieve that goal, we have used the GridSearchCV to get the perfect parameters and then we only had to change to another model and we see that our accuracy and also the recall have improved.

RANDOM FOREST

We have considered doing a Random Forest for our model, because it is usually a more accurate model. A large number of trees acting as Committees go beyond individual composition models. We have imported the RandomForestClassification Library to work with it. And the 'gridSearchCSV' searches through a grid of hyperparameters (param_grid) and finds the combination that gives the best performance on cross-validated data. Adjust the param_grid values based on your specific needs.
Why did i use this hyperparameters?: max_depth=20 allows the tree to capture intricate relationships without any overfitting. max_features=4 limits feature consideration to avoid over-reliance on specific features. min_samples_leaf=3 encourages finer splits for capturing detailed patterns and min_samples_split=12 controls split granularity.

NAIVE BAYES

I'd like to start by addressing the main "difficulty" I've found and how I managed to work around it. After some study and consultation, mainly from the SciKit-Learn libraries' documentation, I applied the algorithms over our original data, X or features: PCA's PC1 to PC6 components and y or labels with values 1 or 0. It's important to note down that the Naive Bayes algorithm classifies data based on its attributes into different classes, which means our target vector must include instances of those classes. However when using the is_habitable variable of our dataset (binary column) the classification fails to correctly address all the existing classes, maybe due to the fact that our positive planet cases are few and not evenly distributed.

To work around it I created a class variable with 3 different classes to better represent the data ranges. This variable goes as follow, inhospitable if its’ esi is below 0.3, possible when in between 0.3 and 0.7, and likely if the esi is larger than 0.7 as likely. This proved to be useful as the algorithm now detects the 3 classes and their belonging entries and its accuracy its not flawed. I tried the several possible algorithms concluding that the Bernoulli Naive Bayes was the best approach to take from the initial accuracy checks)