Daniele | Hossein

We want to use the variables to predictive the Pulsars.

We need the data to Train and Test the methods. Regarding to the two datasets offered, we will use all of the Train Dataset for Training and all of the Test Dataset for Testing.

|  |  |
| --- | --- |
| data\_loading.py load\_train\_data() | data\_loading.py load\_test\_data() |
| """  Load training data and divide it in labels and features.  Returns  -------  features : ndarray  labels: ndarray  """ | """  Load test data and divide it in labels and features.  Returns  -------  features : ndarray  labels: ndarray  """ |

When the probability of a new pulsar (Test Dataset) shows up, we can measure these variables and predict if the Pulsar has existed or not exist.

However, first we have to decide which machine learning method would be best. also, we will use Gaussian Model, Logistic Regression and Support Vector Machines models.

We need to do two things with the datasets:

1. Estimate the parameters for the machine learning methods.

* In other words, to use Logistic Regression, we have to use some of the data to estimate the shape of the curve. Here, estimating parameters is called Training the algorithm.

1. Evaluate how well the machine learning methods work.

* In other words, we need to find out if the curve will do a good job categorizing new data. Here, evaluating methods is called Testing the algorithm.

Then, we can compare methods by seeing how well each one categorized the test data. We assume that this data for training and testing are the best way to divide up the data.

By the way, Cross-validation uses them all one at a time to which block would be best for testing, and summarizes the results at the end. It keeps track of how well the method did with the test data. Then it uses combination of blocks to train the method.

Cross-validation allows us to compare different machine learning methods and get a sense of how well they will work in practice.

In this method, we divided the Train Dataset into 5 blocks (4 blocks for train, 1 for test). This is called Five-Fold cross-validation.

|  |
| --- |
| cross\_validation.py |
| """  Shuffle feautures and labels consistently and create samples.  Parameters  ----------  features : np.ndarray  labels : np.ndarray  samples : int  Number of partitions.  Returns  -------  sampled\_features : List[np.ndarray]  sampled\_labels : List[np.ndarray]  """ |

In the end, every block of data is used for testing and we can compare methods by seeing how well they performed.

In this case, since the Support Vector Machine did the best classifying the test datasets, we will use it.

Beside, we need to summarize how each method performed on the Training data. One way to do this is by creating a **Confusion Matrix** for each method.

The rows in a Confusion Matrix (It is especially called here true\_labels) corresponds to what the machine learning algorithm predicted and the columns (It is especially called here predicted\_labels) corresponds to the know truth.

|  |
| --- |
| confusion\_matrix.py |
| """  Compute confusion matrix from predicted labels and true labels.  Parameters  ----------  true\_labels : ndarray  Real labels to make test on.  predicted\_labels : ndarray  Labels predicted by model.  Returns  -------  confusion\_matrix : ndarray  elemnt[i, j] is predicted as part of i class, but its class is j  """ |

Since there are only two categories to choose from: “Positives” or “Negatives”, then the top left corner contains True Positives. These are the pulsars that had “Positives” that were correctly identified by the algorithm.

The True Negatives are in the bottom right-hand corner. These are the pulsars that did not have “Negatives” that were correctly identified by the algorithm.

The bottom left-hand corner contains the False Positives. These are pulsars has “Positives”, but the algorithm says they are.

Lastly, the top right-hand corner contains the False Negatives. These are when a pulsar has “Negatives”, but the algorithm said they didn’t.

The numbers along the diagonal (The True Positives and True Negatives) tell us how many times the samples were correctly classified.

The numbers not on the diagonal (the False Positives and False Negatives) are samples the algorithm messed up.

We can apply Logistic Regression to the Testing Dataset and create a Confusion Matrix.

----------------------

--------------------

The first thing we do with the Gaussian Naive Bayes classifier is making an initial guess that they are detected as Pulsar. This guess can be any probability that we want, but a common guess is estimated from the training data. That initial guesses are called Prior Probabilities.

Note: the likelihood is the y-axis coordinate on the curve that corresponds to the x-axis coordinate. And we multiply that by the maximum likelihood .

To talk about a likelihood, we assume that we have already weighed the Pulsar (or Pulsars, if it is weighed more than one). We logged transforms the individual likelihood functions.

Step 1) We have moved the log of the first likelihood function for reference.

Step 2) We have converted the multiplication into addition.

Step 3) We have converted 1 over the square root into the exponent -0.5 and convert the exponent into multiplication.

Step 4) We have converted the -0.5 exponent into multiplication and the log of e = 1

Step 5) The log can convert the multiplication of 2 into addition of and 2

Step 6) We have converted the exponent log(2) into 2 log()

Step 7) Lastly, the 2 divided by 2 term cancels out.

data\_loading

cross\_validation