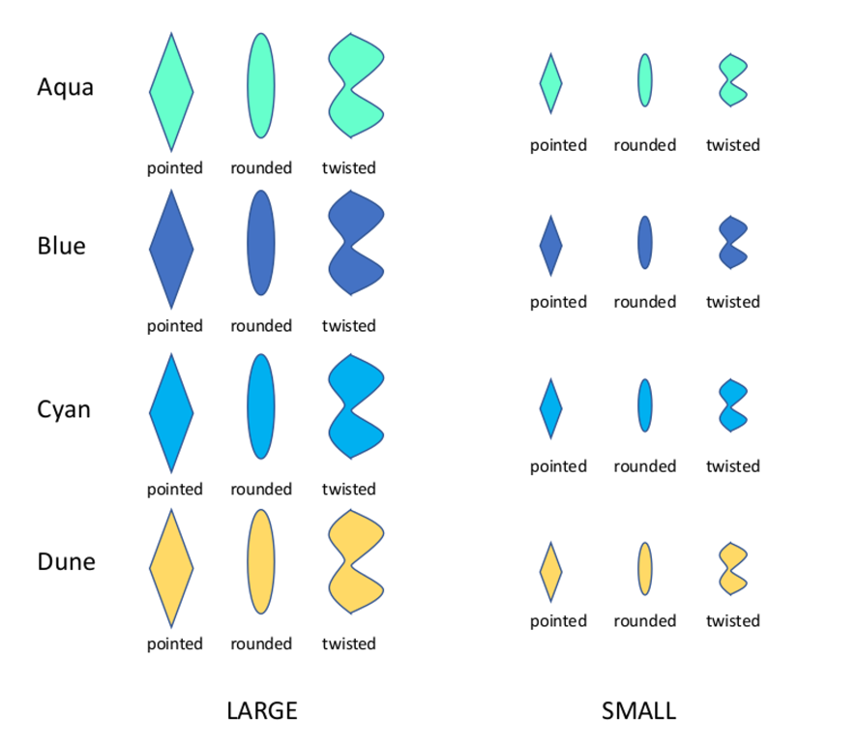
As discussed in the last column, classification can be a tricky thing. Much of the machine learning buzz centers on classification problems. Typical examples include things like optical character recognition (classify a set of pixels in a image as a particular character), computer vision and image processing (classify a region on the ground as flooded or not), and so on.

This column focuses on one of the more common classification algorithms: [naïve Bayes classifier](https://en.wikipedia.org/wiki/Naive_Bayes_classifier) (NBC). Paraphrasing the Wikipedia article, the NBC is a simple technique that produces a model that acts as an agent, which by looking at some collection of features associated with an object, can place that object within the appropriate ‘bucket’.

To create a concrete example, we’ll use the scheme used by James McCaffrey in his June 2019 Test Run column entitled [*Simplified Naive Bayes Classification Using C#*](https://docs.microsoft.com/en-us/archive/msdn-magazine/2019/june/test-run-simplified-naive-bayes-classification-using-csharp). One can imagine that we are pawn brokers in McCaffrey’s universe. People frequently come in hawking jewelry and, given that we run a pawn shop, we should expect that some of our clientele are less than trustworthy. We want build a model that allows us to classify a gemstone as being real or fake based on its color, size, and style of the cut.

These three attributes of the gemstone will be factors used to make the prediction and they are typically arranged in a list or array that is euphemistically called a vector (it is only euphemistically so as these list don’t obey the accepted definitions for a vector space). The gemstone vector will have 3 dimensions for color, size, and style. Each attribute has various realizations as shown in this figure



To develop our model we first have to pool together what we know based on the gemstones we’ve seen. For example, if a kind-hearted woman who had fallen on hard times came in with a small, twisted aqua-colored stone that we verified was authentic then we would enter into our database the entry:

Aqua,Small,Twisted,1

where the ‘1’ means authentic or good. If some shady character, acting all tough came in with a small, blue, pointed stone that we reluctantly took and found out later was fake we would amend our database to read:

Aqua,Small,Twisted,1

Blue,Small,Pointed,0

where the ‘0’ means fake or bad. Proceeding in this fashion, we produce a training set for our agent to gain experience with as it develops its own internal mode. For this initial prototype, I used the 40 element training set provided by McCaffrey (of which the first two points are as shown above).

This kind of training is called supervised since we actually label each feature vector with the category into which it belongs. It is worth noting that there isn’t a single Bayesian classifier but rather a family of related algorithms. The basic concepts are the same but the particular way in which the training set is characterized leads to better or worse performance based on context. In particular, all NBCs assume that a given attribute is independent of the value of any other attribute.

Anyway, returning to McCaffrey’s NBC, the structure of his algorithm is most easily summarized in the following steps (the names of my Python routines to implement these steps are shown in parentheses):

1. Training data is digested, the dimension of the feature vector is deduced, and the types of each attribute uniquely cataloged (find\_distinct\_values)
2. The marginals of the distributions are determined (calculate\_Laplace\_smoothed\_marginals) with an added nuance to handle if a feature combination is not present
3. Additional statistics are computed to facilitate the classification scheme (characterize\_data\_set)
4. Finally the model is able to classify on the feature vector of a new instance of a gemstone (calculate\_evidence)

The primary data structure is the python dictionary which builds up around each attribute discovered in the training set. Obviously, this limits the NBC to classifying on known attributes. In other words, if a ruby-colored gemstone came on the scene the agent/model wouldn’t know how to classify it. This situation would be the same for us manning the pawn shop when a person who don’t know whether to trust of not comes in with such a stone.

The code for each function is listed here:

Happily, the code reproduces the results of McCaffrey’s original article but preliminary tests with more varied training sets have been disappointing. Next month’s column will compare the results from this implementation with some of those found in scikit-learn.