SVC Gotchas!

One of SVC's strong points is that since the core of the algorithm is based on a small subset of your dataset's samples, namely the support vectors, even if you have fewer samples than dimensions, so long as the samples you do have are close to the decision boundary, there's a good chance your support vector classifier will do just swell.

Intuitively, those samples that are further away from the decision boundary are more clearly identifiable as belonging to their respective classes. The samples closer to the decision boundary are more vague, and could easily be mistaken as belonging to the wrong class. If you wanted to train a child how to recognize cats from dogs, good training samples would include the most "catly" cat you could find, and the most "dogly" dog. By showing them samples from the two classes that are far away from the decision boundary, they are less likely to look for characteristics and features that might accidentally be misconstrued. Support vector machines behave counter intuitively; they don't care about the samples that are clearly cats, or that are clearly dogs. Rather, they focus on those samples, or support vectors, closest to the decision boundary, so they can compute precisely the smallest change of features that differentiate between the two, perchance it's able to properly classify all of them.

Although SVC can run on a subset of your features, if you get rid of too many, that is, if the dimensionality of your features is *much* greater than the number of samples, the quality of your decision boundary may still suffer. Again, depending on how far away those samples are from it.

Support vector machines, unfortunately, do not *directly* give probability estimates for what class a sample belongs to. If a sample is further from the decision boundary than the margin, then the algorithm is intuitively 100% sure of its classification. Any testing found within the margin has some probability of belonging to either class. In SciKit-Learn, to calculate the probability of belonging to either class, you actually have to use an expensive five-fold cross-validation, which we won't discuss at all until the next module.

Since SVC is one of SciKit-Learn's highly configurable predictors, it's easy to start overfitting your models if you're not careful. Furthermore, unlike KNeighbors that does all its processing at the point of predicting, SVC does the majority of its heavy lifting at the point of training, so large training sets can result in sluggish training. If the ability to do realtime training and updating of your model is of great concern to you, you might have to consider another algorithm, depending on the size of your dataset. That said, there are a few mechanisms to speed it up.


Download on the
App Store
ANDROID APP ON
Google play