Microsoft: DAT210x Programming with Python for Data Science

Bookmarks

- Start Here
- 1. The Big Picture
- 2. Data And Features
- 3. Exploring Data
- 4. Transforming Data
- 5. Data Modeling
- **6. Data Modeling II**

Lecture: SVC
Quiz

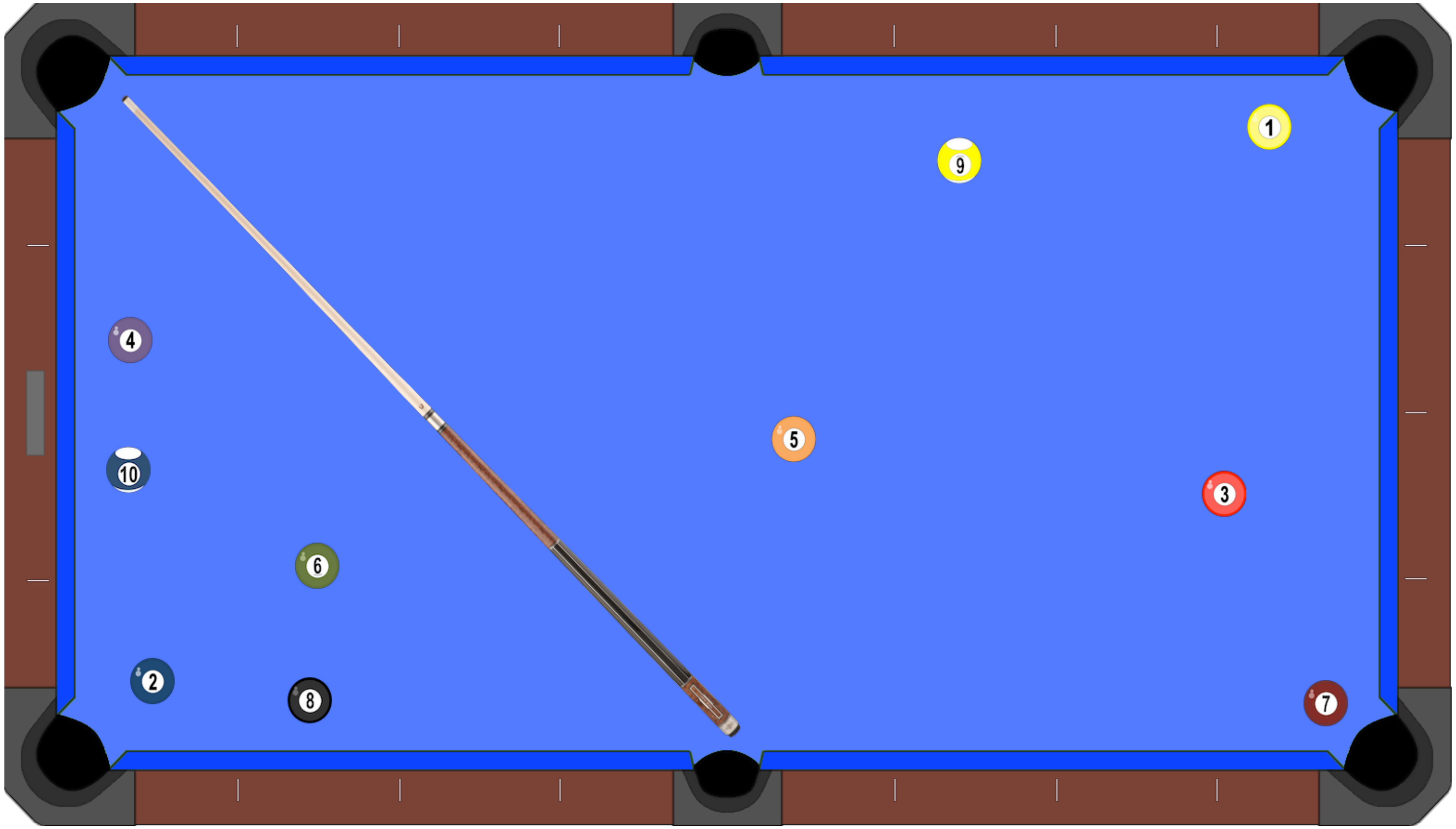
Lab: SVC
Lab

6. Data Modeling II > Lecture: SVC > SVM and SVC

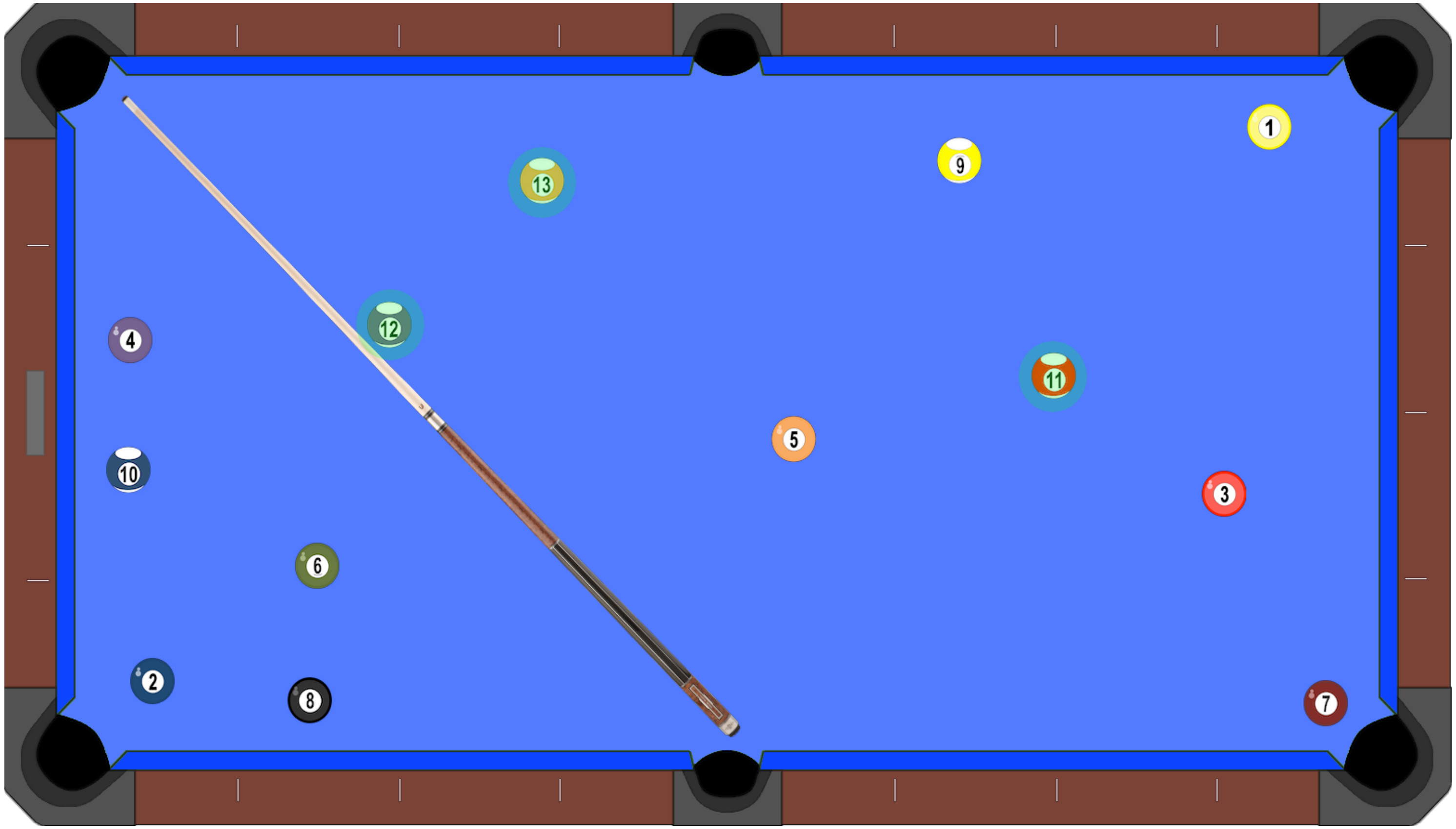
Bookmark

Support vector machines are a set of supervised learning algorithms that you can use for classification, regression and outlier detection purposes. SciKit-Learn has many classes for SVM usage, depending on your purpose. The one we'll be focusing on is Support Vector Classifier, but having understood the principles, with a little research, you'll soon be able to use the rest.

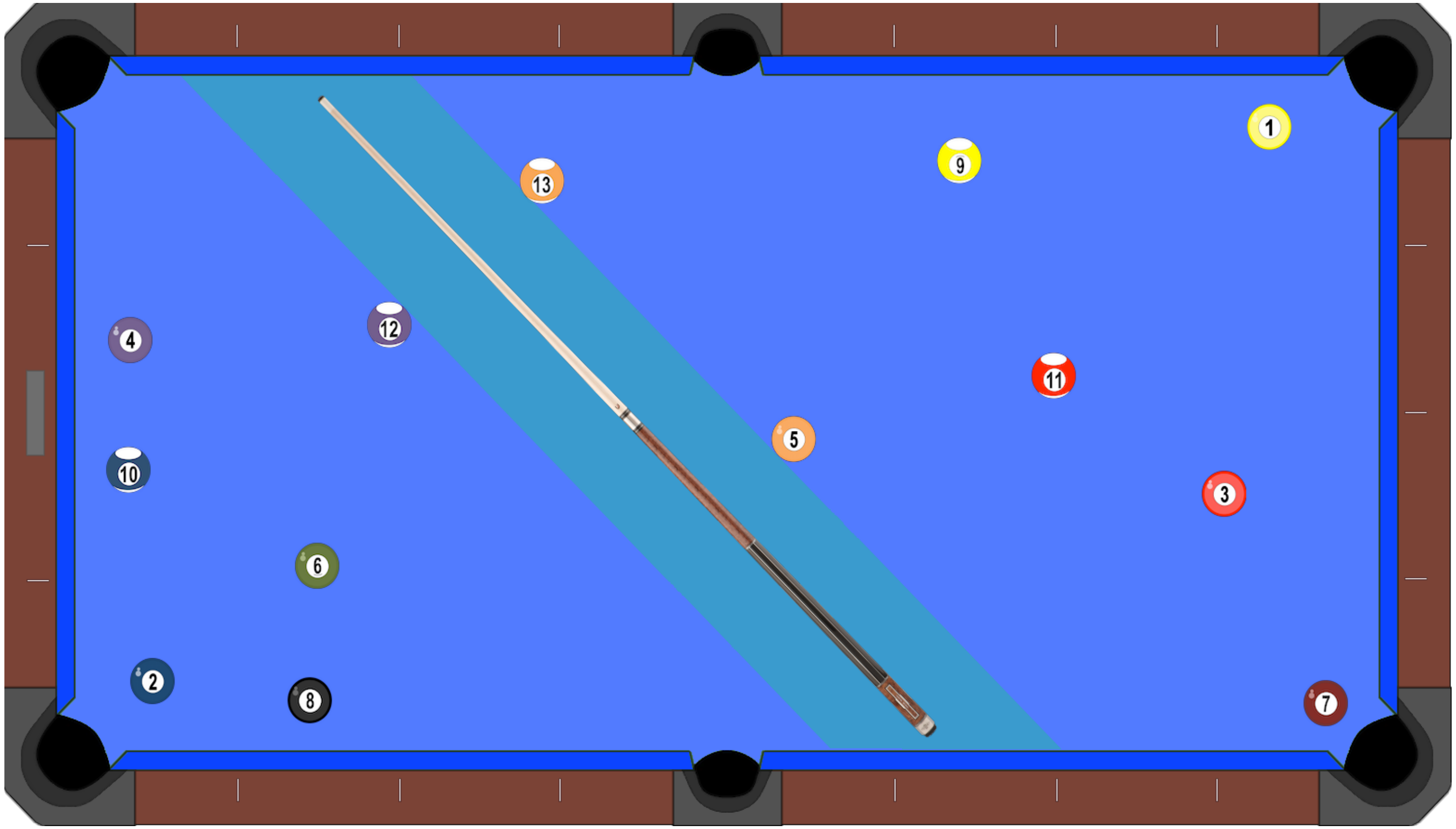
Suppose you were a student at Coding Dojo. One afternoon on sports day, your instructor challenges you to use any straight surface to separate a mixed group of billiard balls into even and odd sets. "No problem," you say, while laying a pool-stick between the two sets:



Recall, each sample in a dataset is just a point n-dimensional space, each dimension corresponding to a feature. Well, each billiard ball is a "sample" that has three features: its number, and it's X and Y-coordinates on the pool table. Stated in data-analysis lingo, you've just created a classifier that can correctly identify samples based on their features. Well done! Having impressed your instructor, she decides to challenge you further by adding more balls onto the table:



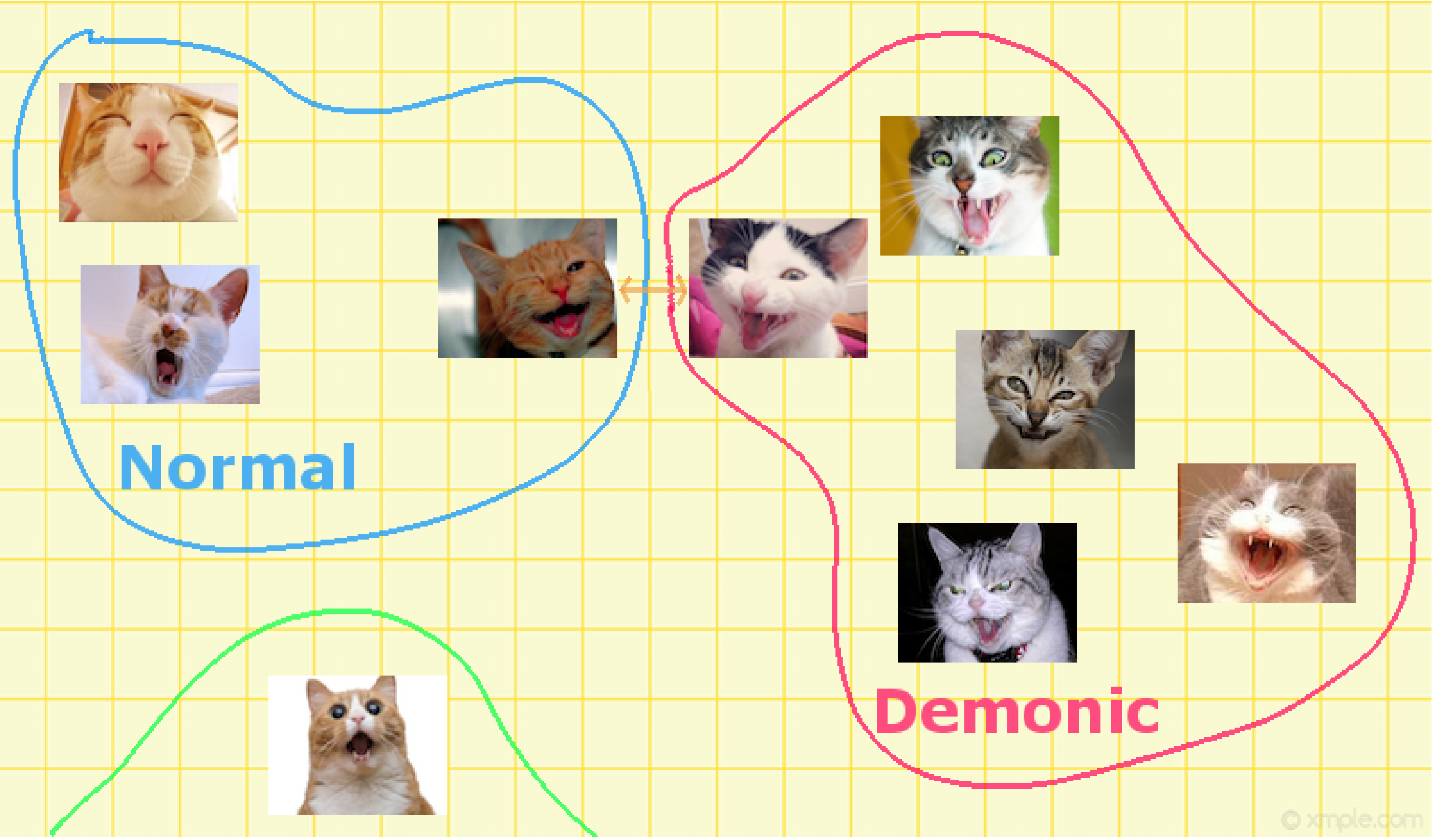
Your initial pool-stick placement still works for the most part, but there is probably a better spot to place it that might produce even better separation between the two sets. This is still, no issue for you. You simply adjust the placement of your stick and voilà, it once again works perfectly!



Having moved the stick to a new position, something dawns on you. Where all the balls are on the table doesn't really matter. In fact, even if you had no idea where the 2-ball, the 7-ball, the 10-ball, and the 1-ball, etc. were placed, it wouldn't really make much of a difference in how you placed the stick. The only balls that you really need to be aware of when doing stick-placement, are those balls of either class that are closest to one another.

Support vector classifier behaves just like your stick placement. The two things it wishes to fulfill, in order of priority, are first finding a way to separate your data. It does this by looking at the balls from either class that are closest to one another, or in machine learning lingo, the support vectors. The algorithm then ensures it separates your data in the *best way possible* by orienting the boundary such that the gap, or *margin* between it and your support vector samples is maximized.

To understand why SVC does this, think back to K-Means clustering; do you remember how it was possible for samples in separate groups to actually be closer, or more similar than even samples from the same group, depending on their positioning within the group?



That means that to classification algorithms, samples closer to the decision boundaries, in the image above, the blue, pink, and green lines, actually seem more similar to one another than samples further away from it. Stated differently, by finding a way to separate the two sets of samples in a manner that maximizes the distance from sample's of either class to the decision boundary, an algorithm will be more *certain* of the class of each sample. As a result, you're guaranteed to get better classification accuracy.

In a nutshell, SVC solves the classification problem by finding the equation of the hyperplane (linear surface) which results in the most separation between two classes of samples. This allows you to confidently label your samples in a *very fast* and efficient way.