The Support Vector Machine Algorithm

The **Support Vector Machine (SVM)** algorithm is a classifier that finds the hyperplane that effectively separates the observations into their class labels. It starts by positioning each instance into a data space with *n* dimensions, where *n* represents the number of features. Next, it traces an imaginary line that clearly separates the instances belonging to a class label from the instances belonging to others.

A support vector refers to the coordinates of a given instance. According to this, the support vector machine is the boundary that effectively segregates the different support vectors in a data space.

For a two-dimensional data space, the hyperplane is a line that splits the data space into two sections, each one representing a class label.

How Does the SVM Algorithm Work?

The following diagram shows a simple example of an SVM model. Both the triangles and circular data points represent the instances from the input dataset, where the shapes define the class label that each instance belongs to. The dashed line signifies the hyperplane that clearly segregates the data points, which is defined based on the data points' location in the data space. This line is used to classify unseen data, as represented by the square. This way, new instances that are located to the left of the line will be classified as triangles, while the ones to the right will be circles.

The larger the number of features, the more dimensions the data space will have, which will make visually representing the model impossible:

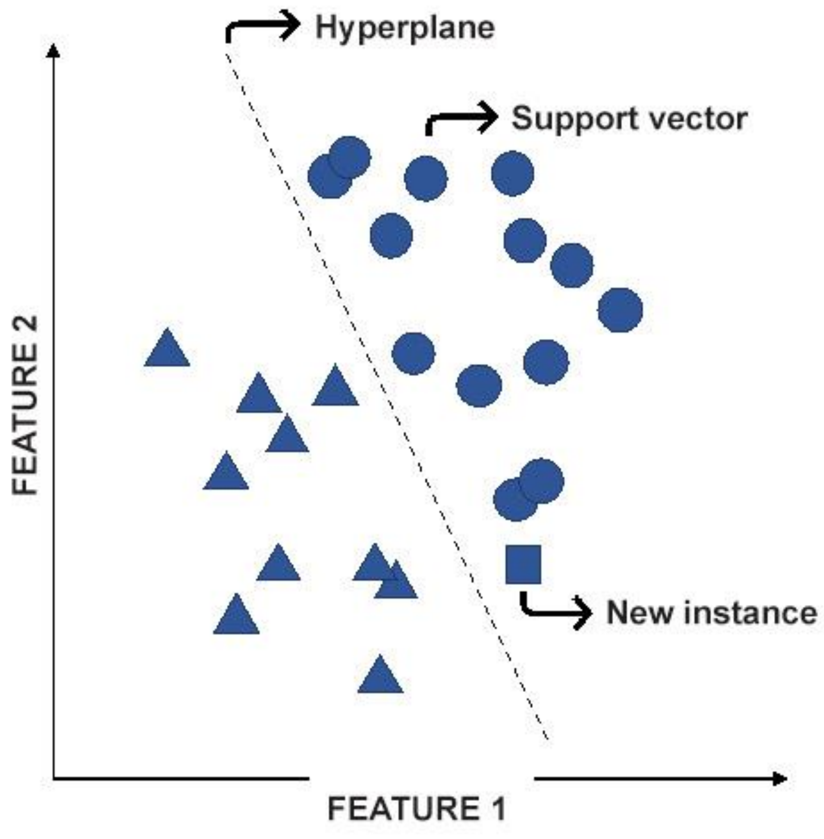

Figure 1: Graphical example of an SVM model

Although the algorithm seems to be quite simple, its complexity is evident in the algorithm's methodology for drawing the appropriate hyperplane. This is because the model generalizes to hundreds of observations with multiple features.

To choose the right hyperplane, the algorithm follows the following rules, wherein *Rule 1* is more important than *Rule 2*:

- **Rule 1**: The hyperplane must maximize the correct classification of instances. This basically means that the best line is the one that effectively separates data points belonging to different class labels while keeping those that belong to the same one together.
  For instance, in the following diagram, although both lines are able to separate most instances into their correct class labels, line A would be selected by the model as the one that segregates the classes better than line B, which fails to classify two data points:

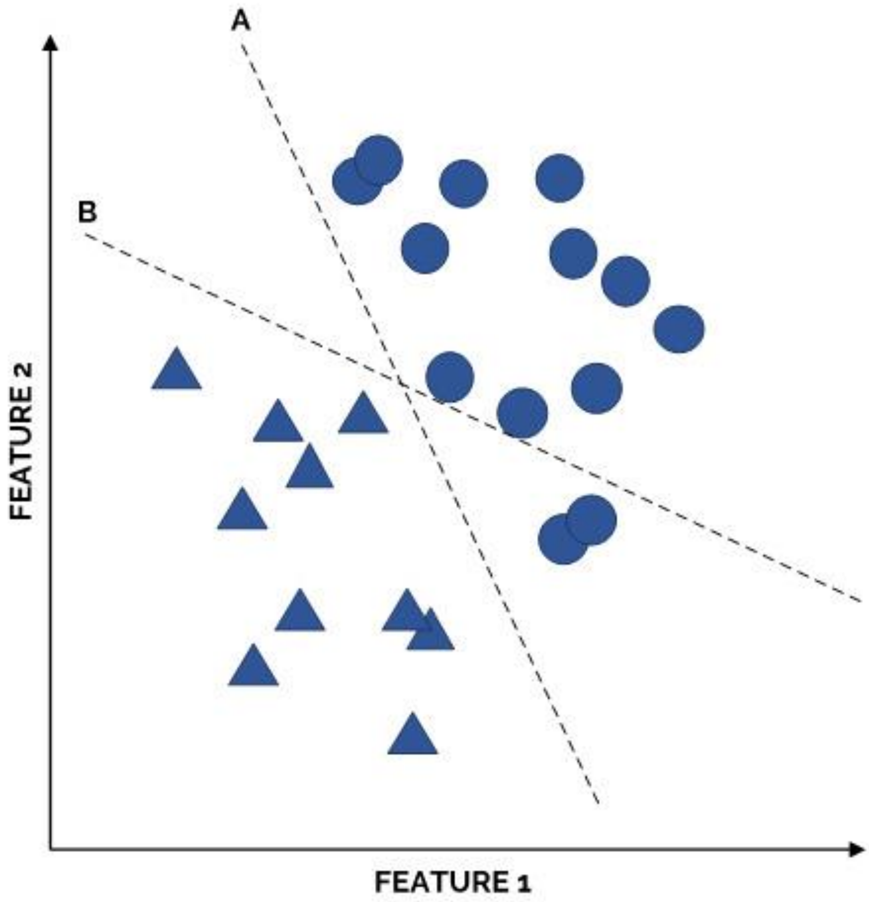

Figure 2: Sample of hyperplanes that explain Rule 1

- **Rule 2**: The hyperplane must maximize its distance to the nearest data point of either of the class labels, which is also known as the **margin**. This rule helps the model become more robust, which means that the model is able to generalize the input data so that it works efficiently on unseen data. This rule is especially important in preventing new instances from being mislabeled.
  For example, by looking at the following diagram, it is possible to conclude that both hyperplanes comply with *Rule 1*.

Nevertheless, line A is selected, since it maximizes its distance to the nearest data points for both classes in comparison to the distance of line B to its nearest data point:

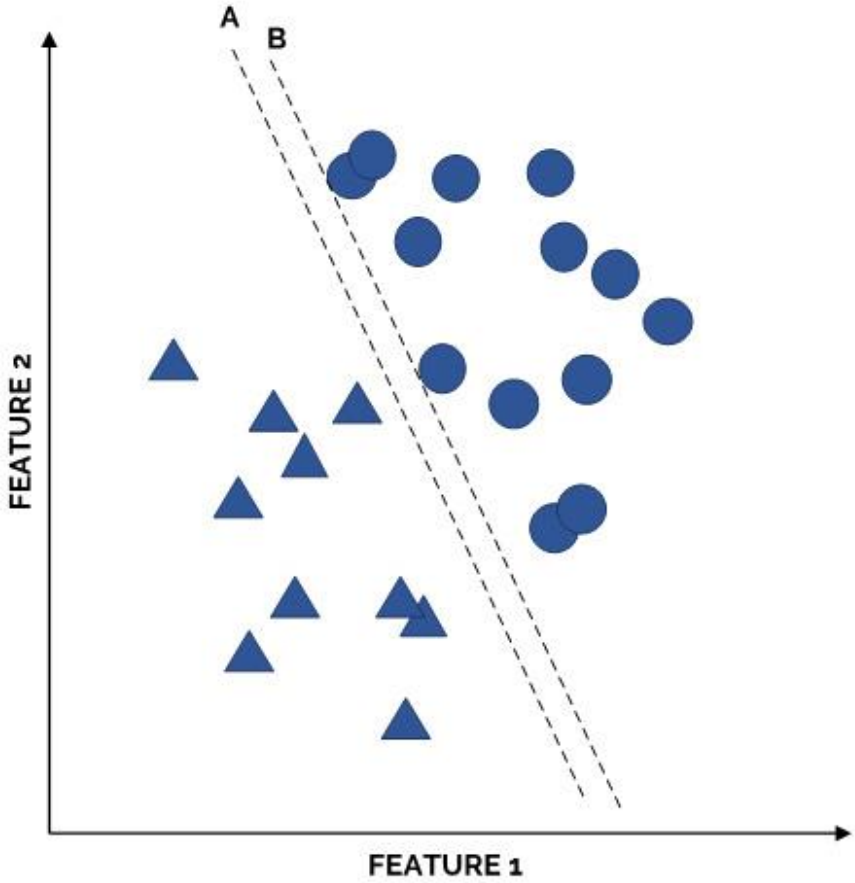

Figure 3: Sample of hyperplanes that explain Rule 2

By default, the SVM algorithm uses a linear function to split the data points of the input data. However, this configuration can be modified by changing the kernel type of the algorithm. For example, consider the following diagram:

Note

For scikit-learn's SVM algorithm, the kernel refers to the mathematical function to be used to split the data points, which can be linear, polynomial, or sigmoidal, among others. To learn more about the parameters for this algorithm, visit the following URL: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC.

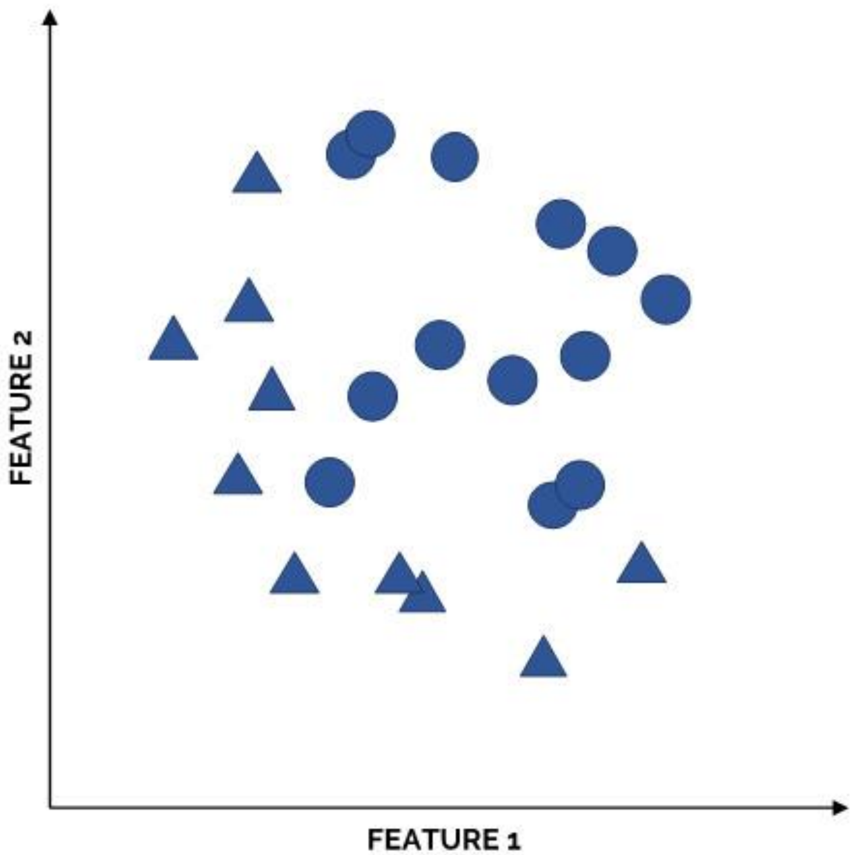

Figure 4: Sample observations

To segregate these observations, the model would have to draw a circle or another similar shape. The algorithm handles this by using kernels (mathematical functions) that can introduce additional features to the dataset in order to modify the distribution of data points into a form that allows a line to segregate them. There are several kernels available for this, and the selection of one should be done by trial and error so that you can find the one that best classifies the data that's available.

However, the default kernel for the SVM algorithm in scikit-learn is the **Radial Basis Function** (**RBF**) kernel. This is mainly because, based on several studies, this kernel has proved to work great for most data problems.