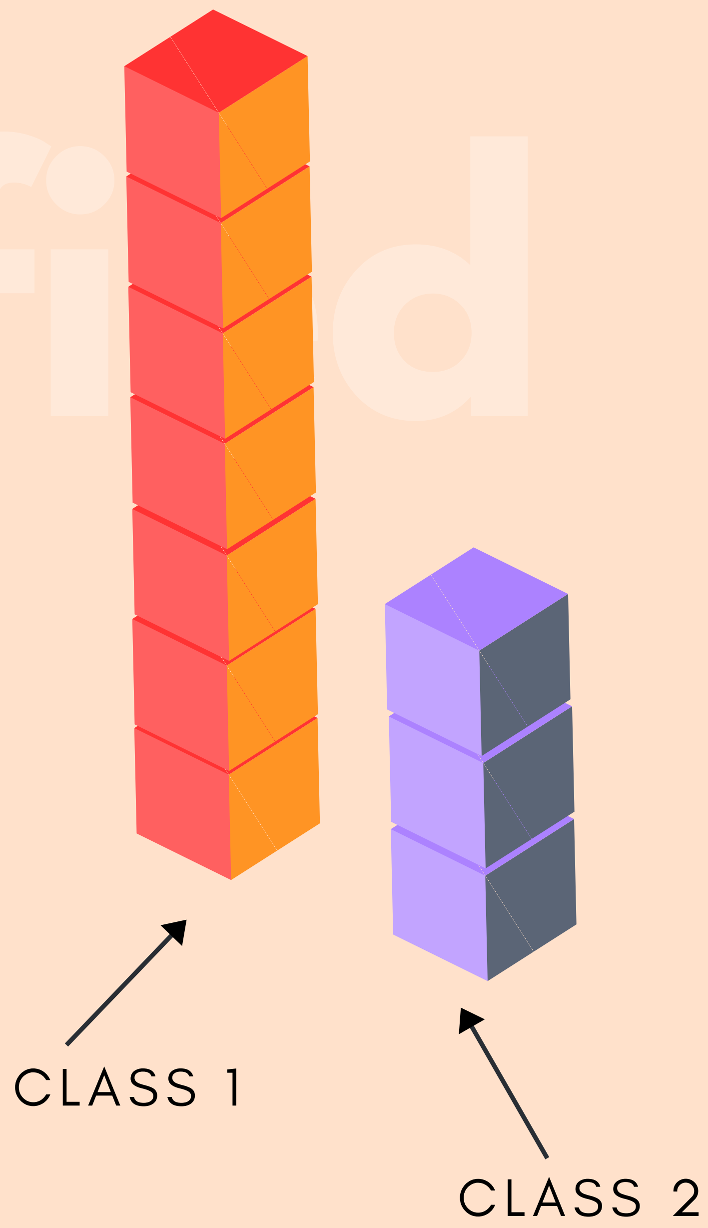
ROBOFIED

Support Vector Machines

FIXING FUNDAMENTALS
BY ROBOFIED

Support Vector Machines?

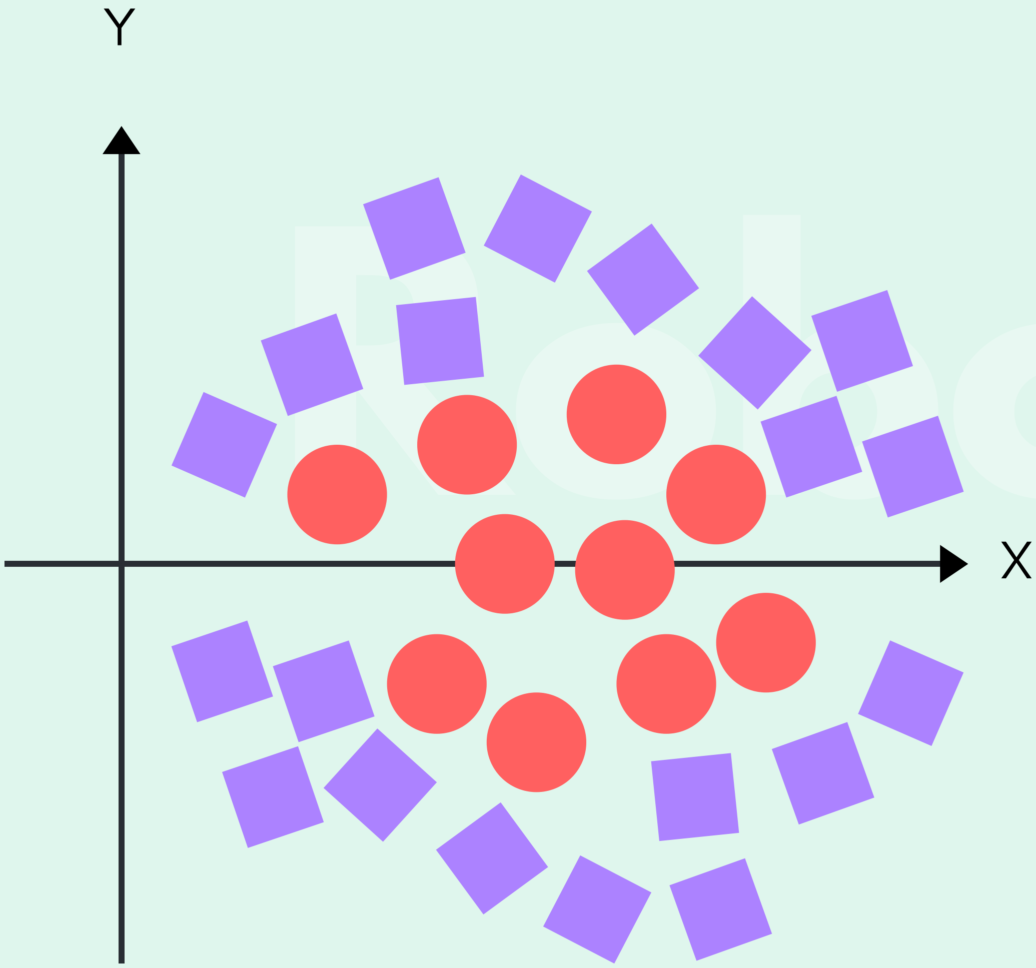
Support Vector Machines or SVMs makes use of kernel to transform your data and then based on these transformations it aims to find an optimal hyperplane such that it distinctly classifies the data points.



ROBOFIED

02

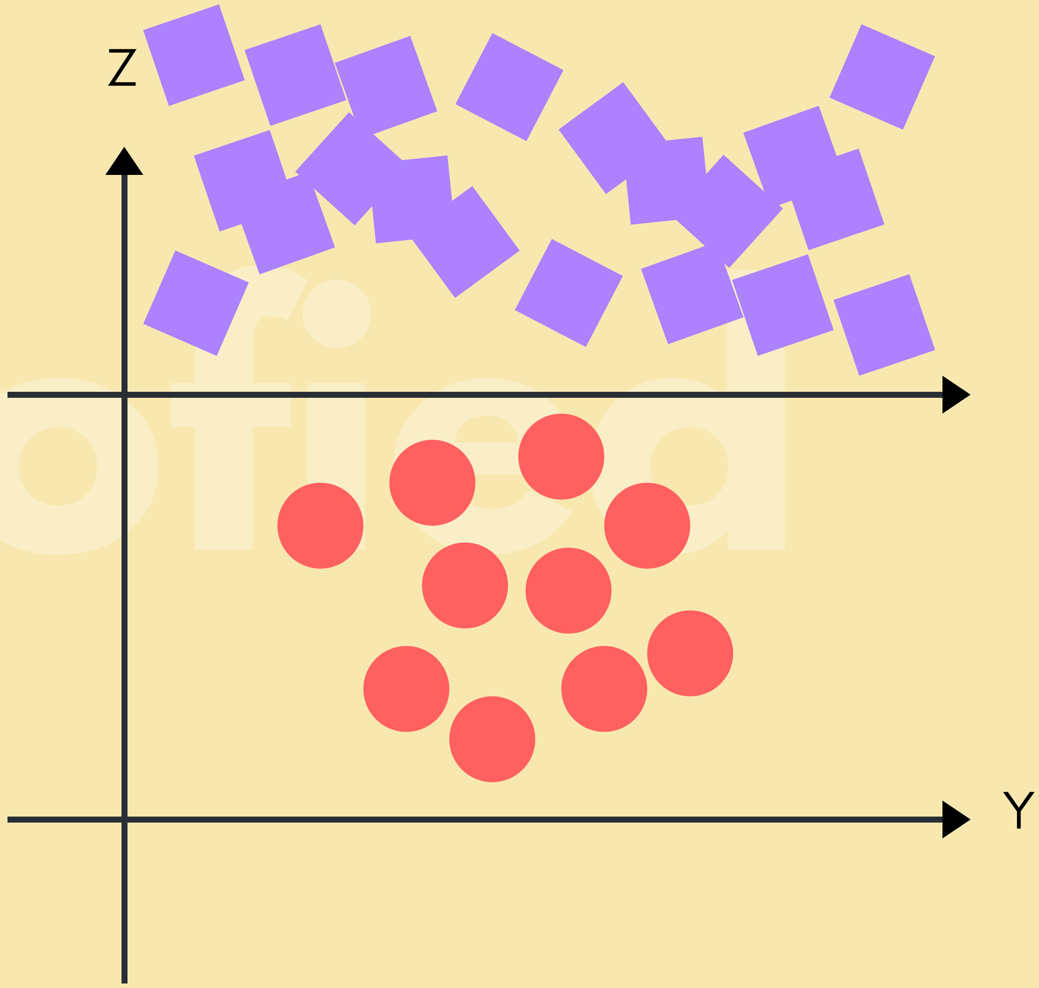
Data Exploration



Inspect the dataset that you have. In the adjoining figure, we have some data points which are to be classified into two groups. Clearly, we can see that there is no possible straight line which can separate them into two groups.

Data Transformation 03

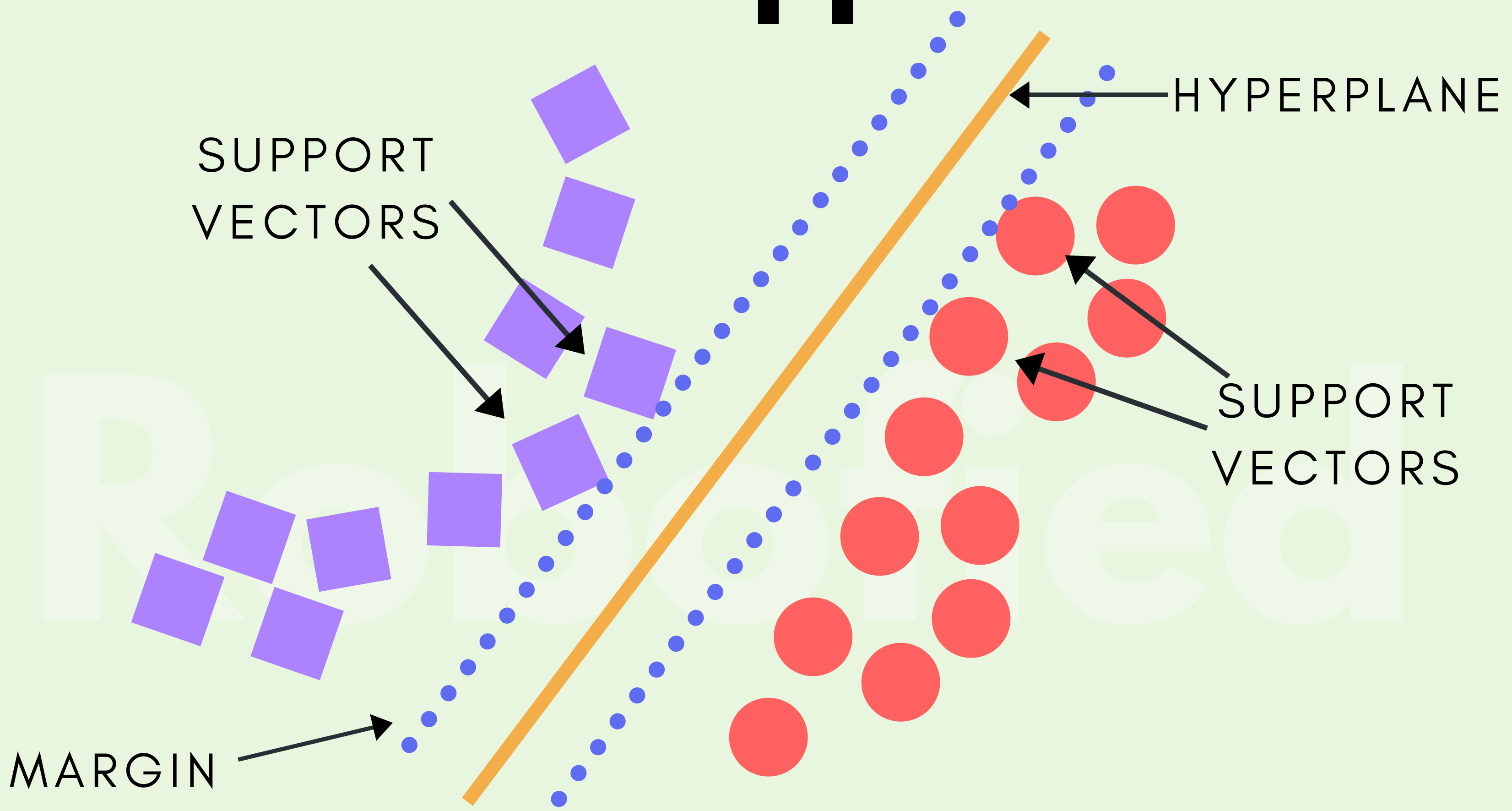
Now, you need to transform your data by adding more dimension 'z' apart from x and y as z = x.x + y.y. If we plot in y-z axis, a clear separation is visible and hence a line can be easily drawn. The line that we used here for separation is known to be as a decision boundary or hyperplane whereas the transformation to project our data into higher dimensions is known to be as kernel.



ROBOFIED

04

Hyperplanes and Support Vectors

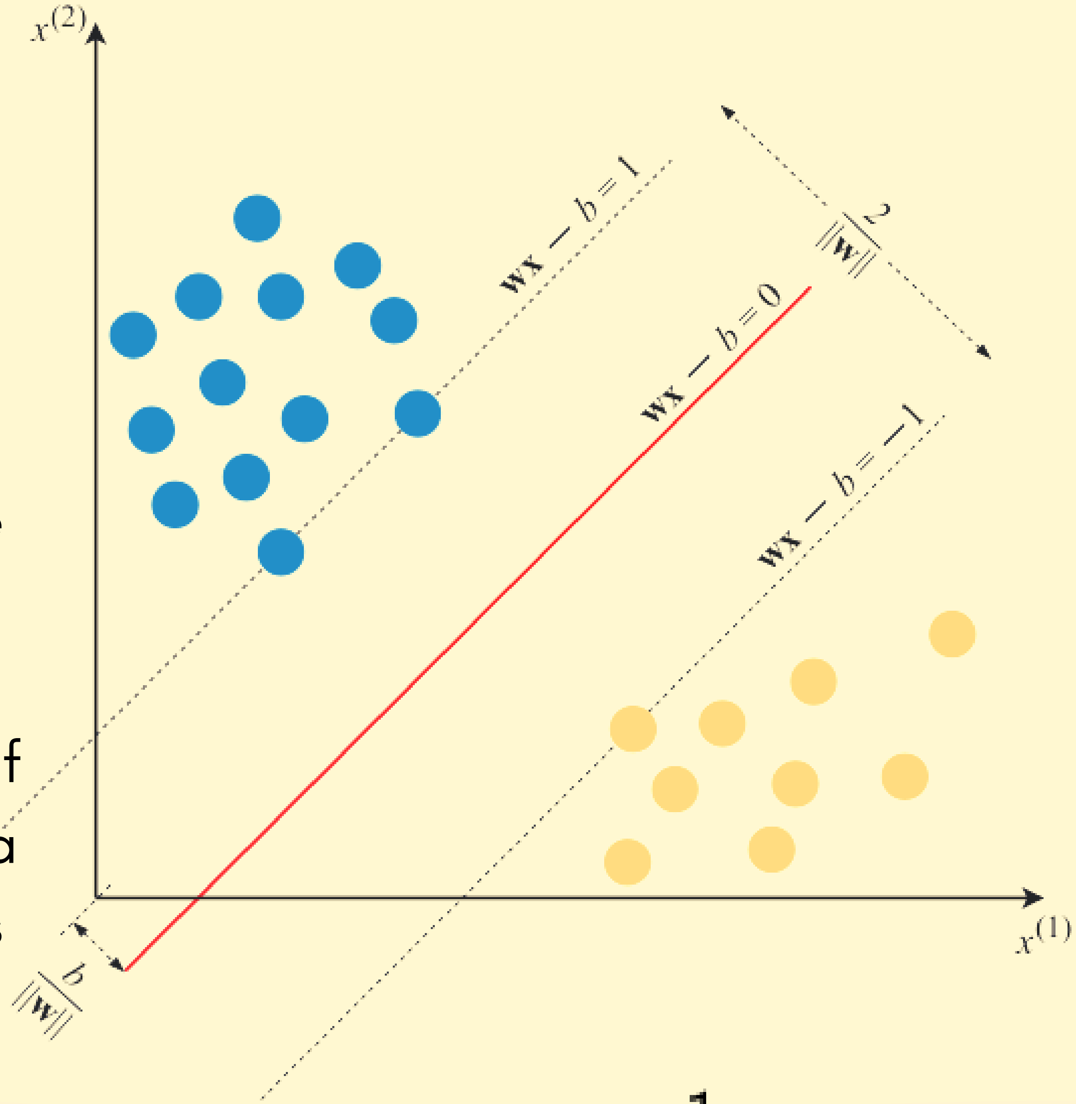


Hyperplanes is the decision boundary that helps to classify the data points. Those data points which are closer to this hyperplane are referred as Support Vectors. The distance between the closest examples of two classes as defined by the hyperplane is known as margin. Support Vectors together with margin help us find the optimal hyperplane and build the Support Vector Machine.

Soft SVM vs Hard SVM

The goal of SVM is to learn a hyperplane which maximizes the margin between the different class labels.

If your data is linearly separable, use Hard SVMs and if it's not use Soft SVMs in which we penalize each instance if it falls on the other side of the margin. Soft SVMs make use of a hyperparameter C which determines the tradeoff between increasing the size of the decision boundary and ensuring that each data point lies on the correct side of the decision boundary.



HARD SVM $\longrightarrow$ $$min \frac{1}{2}\|w\|_2^2$$

SOFT SVM $\longrightarrow$ $$min \frac{1}{2}\|w\|_2^2 + C\sum_i \epsilon_i$$

06

Pros & Cons

Pros:

- SVM works well when there is clear margin of separation between classes.
- Works well in high-dimensional spaces.

Cons:

- Doesn't work well when dataset is large.
- When number of features > number of training data samples, SVM will underperform.

Code


```
# scikit-learn implementation of Support Vector Machines (SVM)
import numpy as np
from sklearn import svm
X = np.array([[0, 0], [1, 1]])
y = np.array([0, 1])
clf = svm.SVC(gamma = 'auto')
clf.fit(X, y)
print(clf.predict([[2., 2.]]))
```

ROBOFIED

Tips from Team at Robofied

08

- If there is noise in your data and no hyperplane is able to separate positive examples from the negative ones, it's better to use hinge loss function. SVMs which optimize hinge-loss function are called Soft Margin SVMs. More is the value of C, SVM will try to find highest margin by ignoring misclassification. And if, the value of C is less, making classification mistakes is very costly and SVM does so by sacrificing margin size.

- If the data cannot be separated using a plane, but could be separated using a higher-order polynomial, it's the indication of using kernel trick. Functions which help us do so are called kernels. Multiple functions exist, most popular of which is the RBF kernel.

ROBOFIED

At **Robofied,**
we aspire to democratize **AI**
Education

Join our channels:

www.robofied.com
https://instagram.com/robofied
https://twitter.com/Robofied