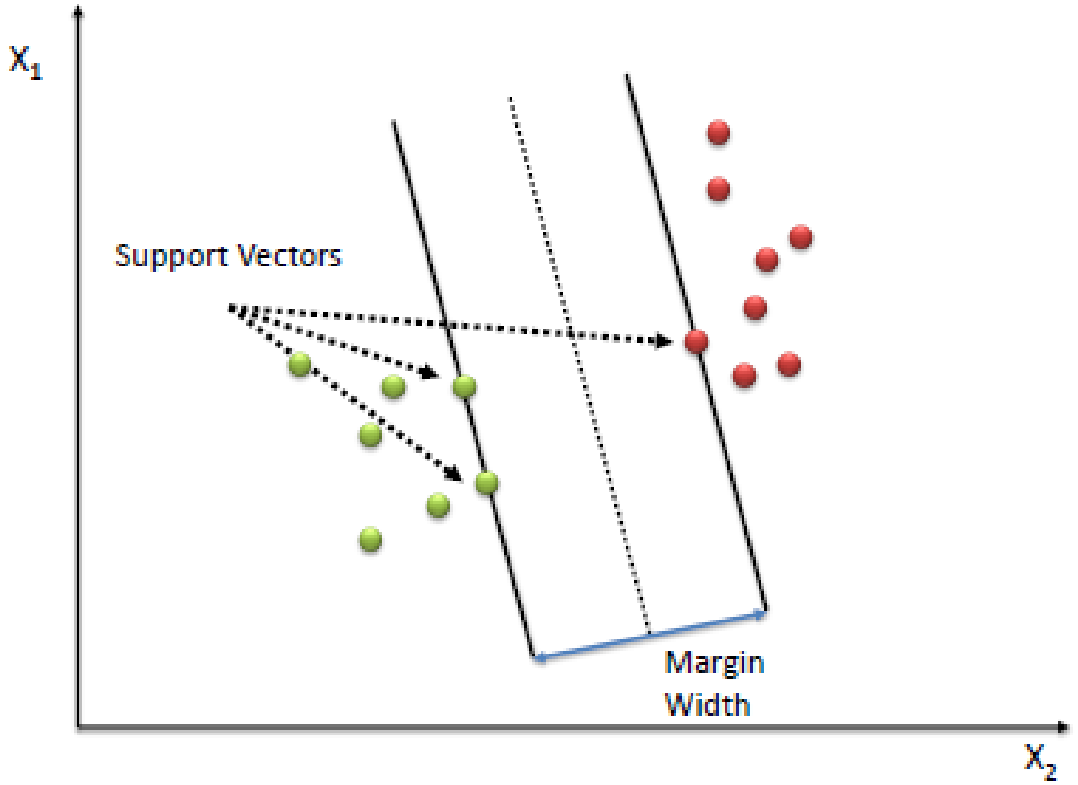
Support Vector Machine

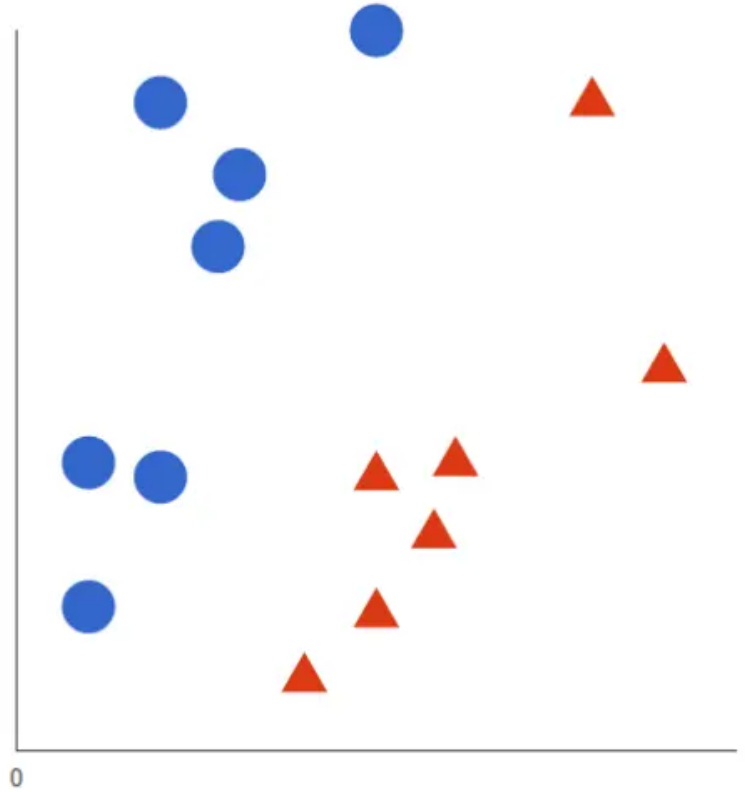
Support Vector Machine - Classification (SVM)

- A Support Vector Machine (SVM) performs classification by finding the hyperplane that maximizes the margin between the two classes.
- The vectors (cases) that define the hyperplane are the support vectors.



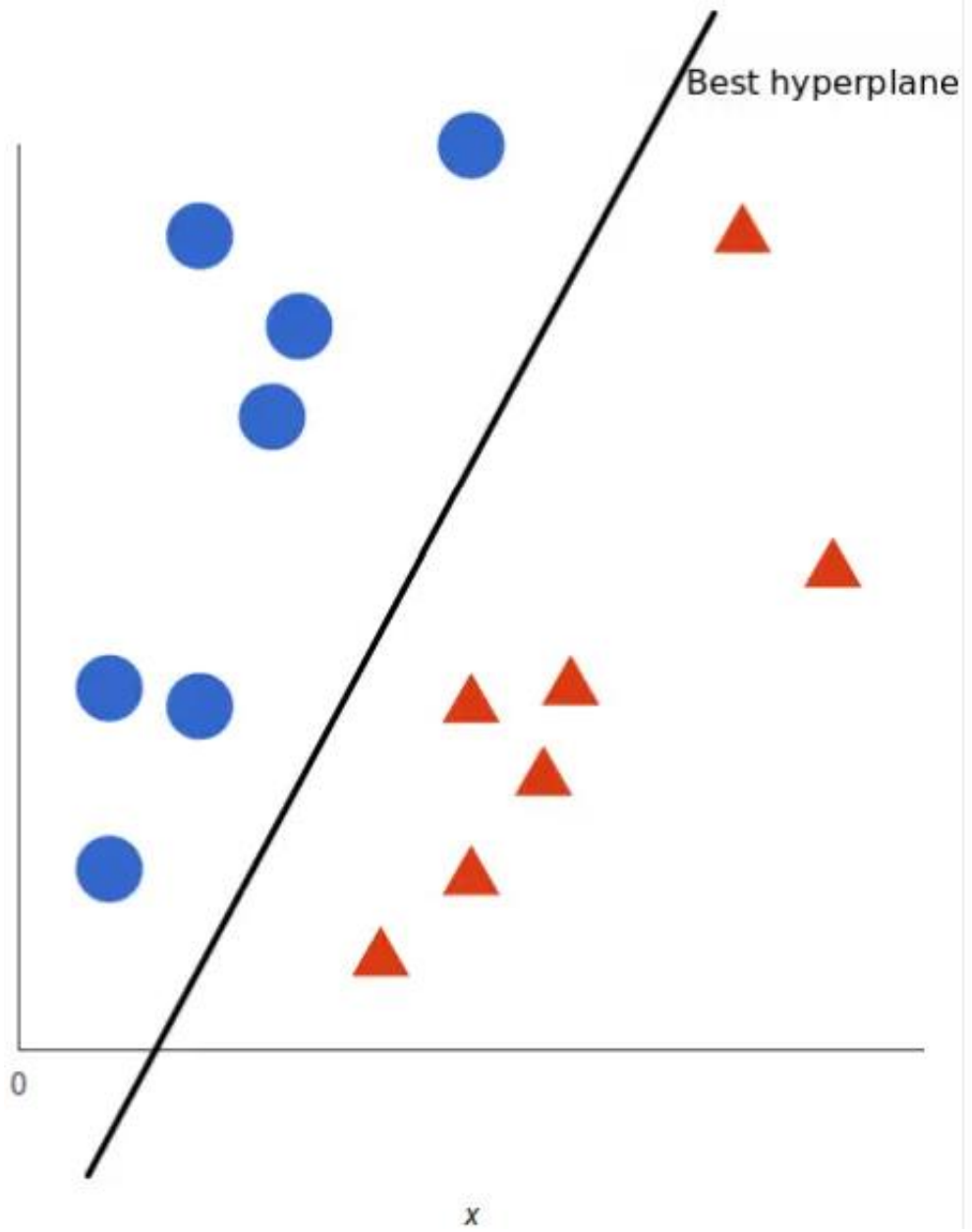
How does SVM work?

- Let’s imagine we have two classes: red and blue, and our data has two features: x and y.
- We want a classifier that, given a pair of (x,y) coordinates, outputs if it’s either red or blue. We plot our already labeled training data on a plane:



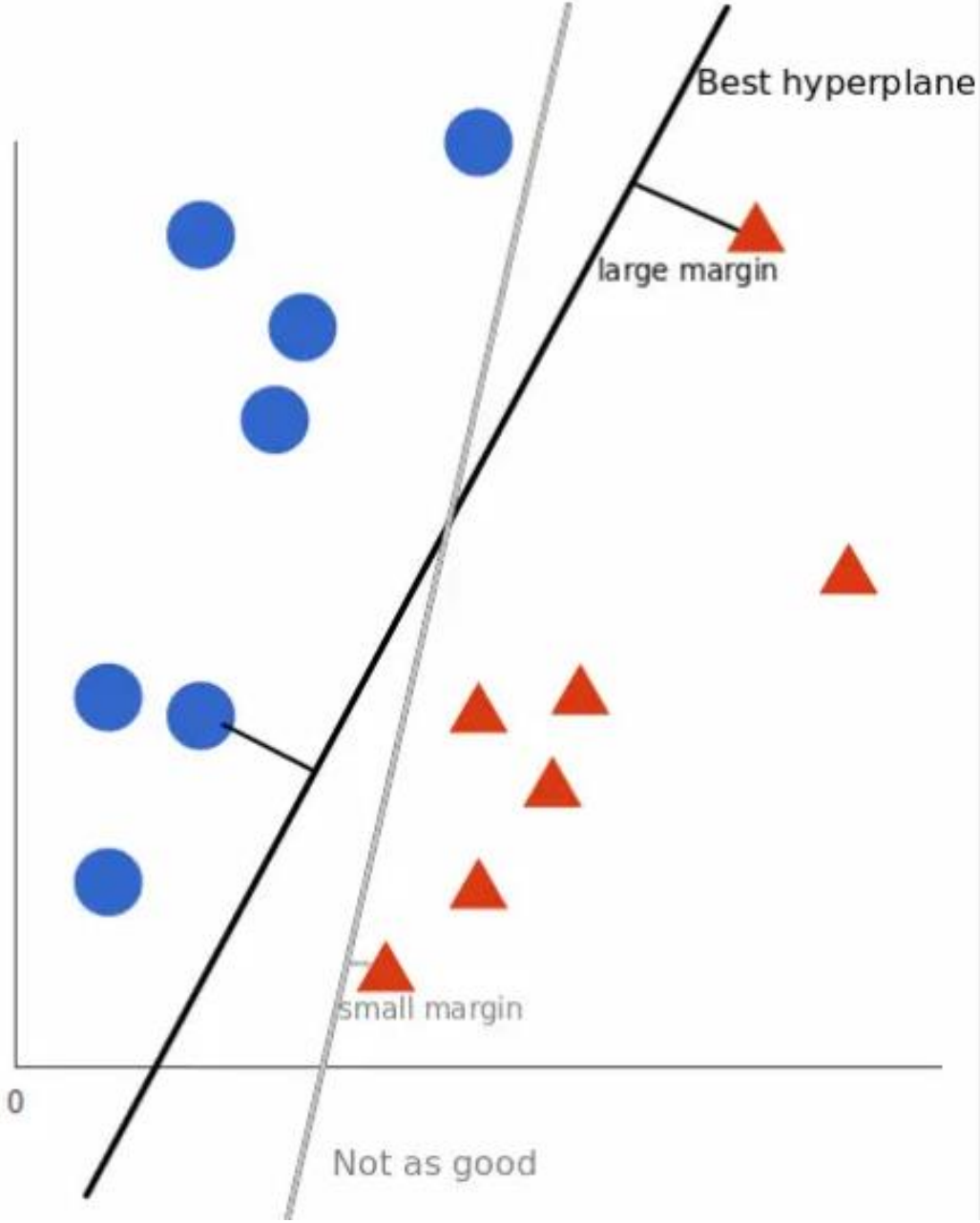
How does SVM work?

- A SVM takes these data points and outputs the hyperplane that best separates the classes.
- This line is the decision boundary: anything that falls to one side of it we will classify as blue, and anything that falls to the other as red.



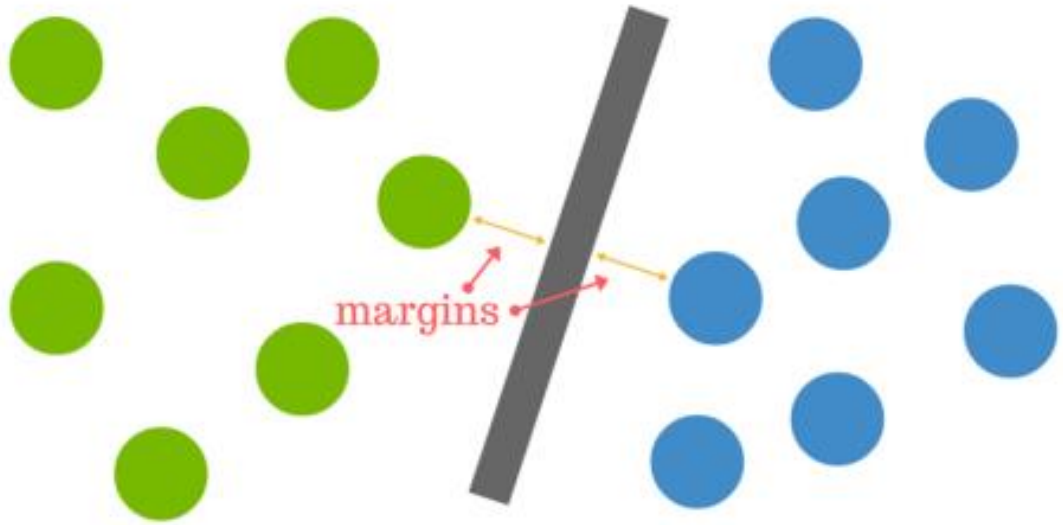
How does SVM work?

- But, what exactly is the best hyperplane?
- For SVM, it's the one that maximizes the margins from both classes.
- In other words: the hyperplane (remember it's a line in this case) whose distance to the nearest element of each tag is the largest.
- Support vectors are the data points nearest to the hyperplane.

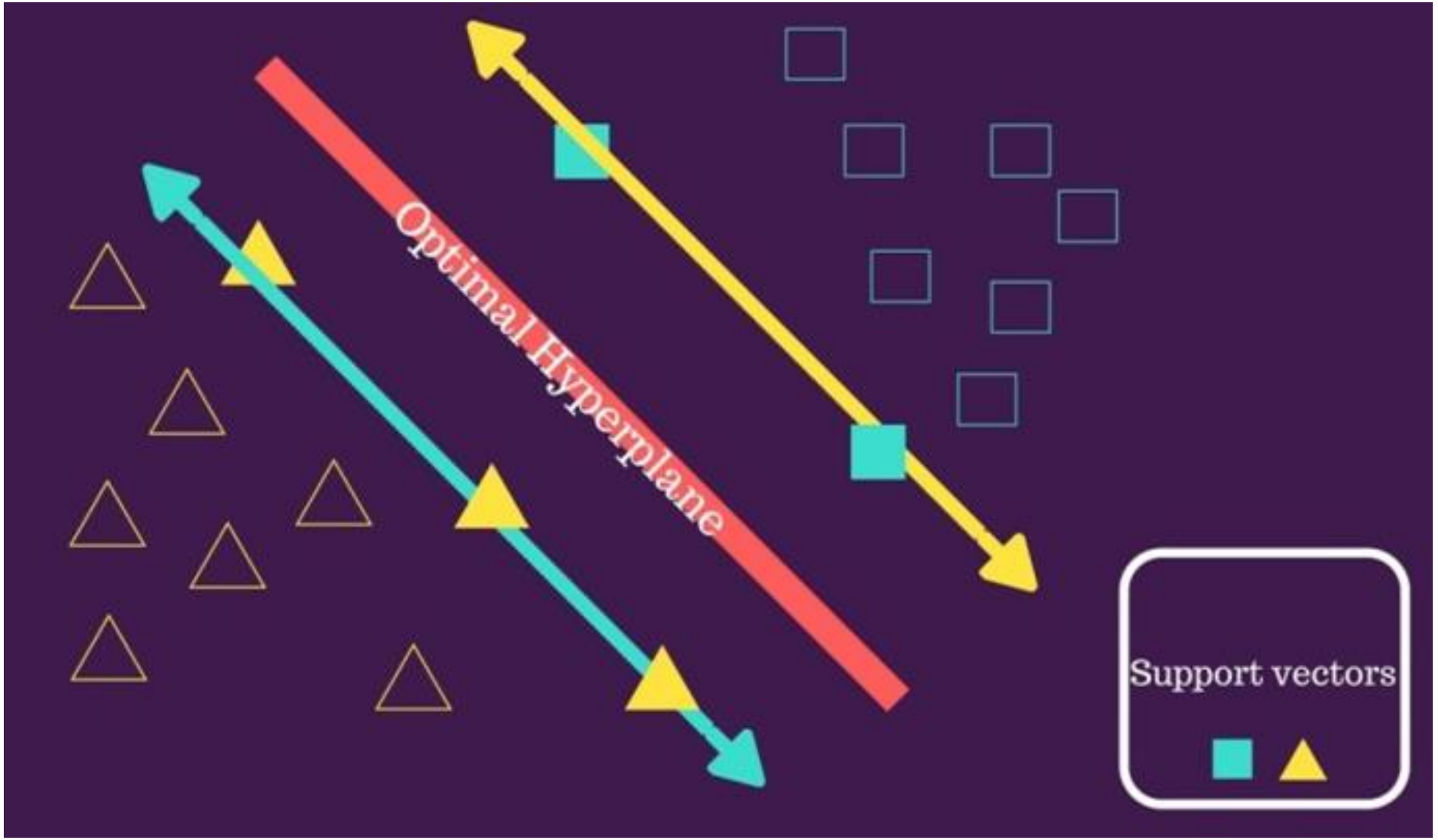


How does SVM work?

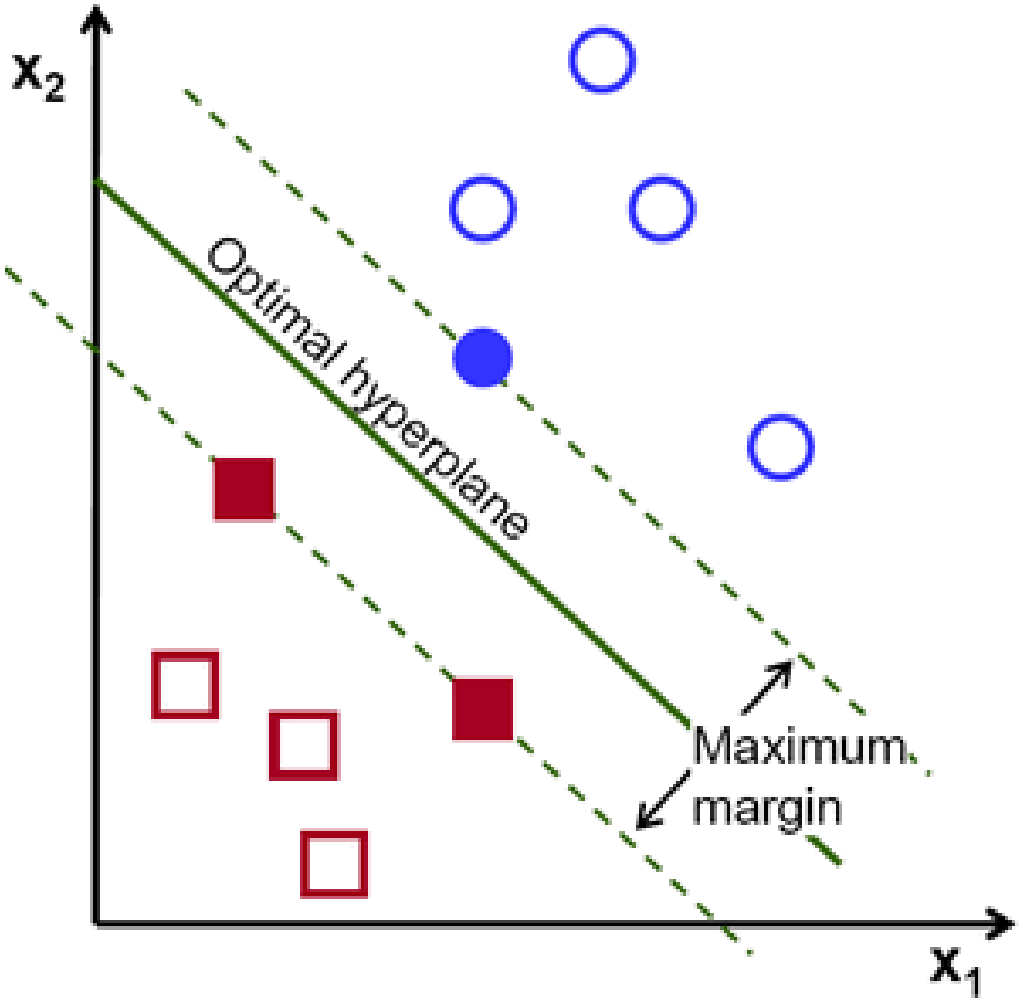
- You can think of a hyperplane as a line that linearly separates and classifies a set of data.
- Intuitively, the further from the hyperplane our data points lie, the more confident we are that they have been correctly classified.
- We therefore want our data points to be as far away from the hyperplane as possible.



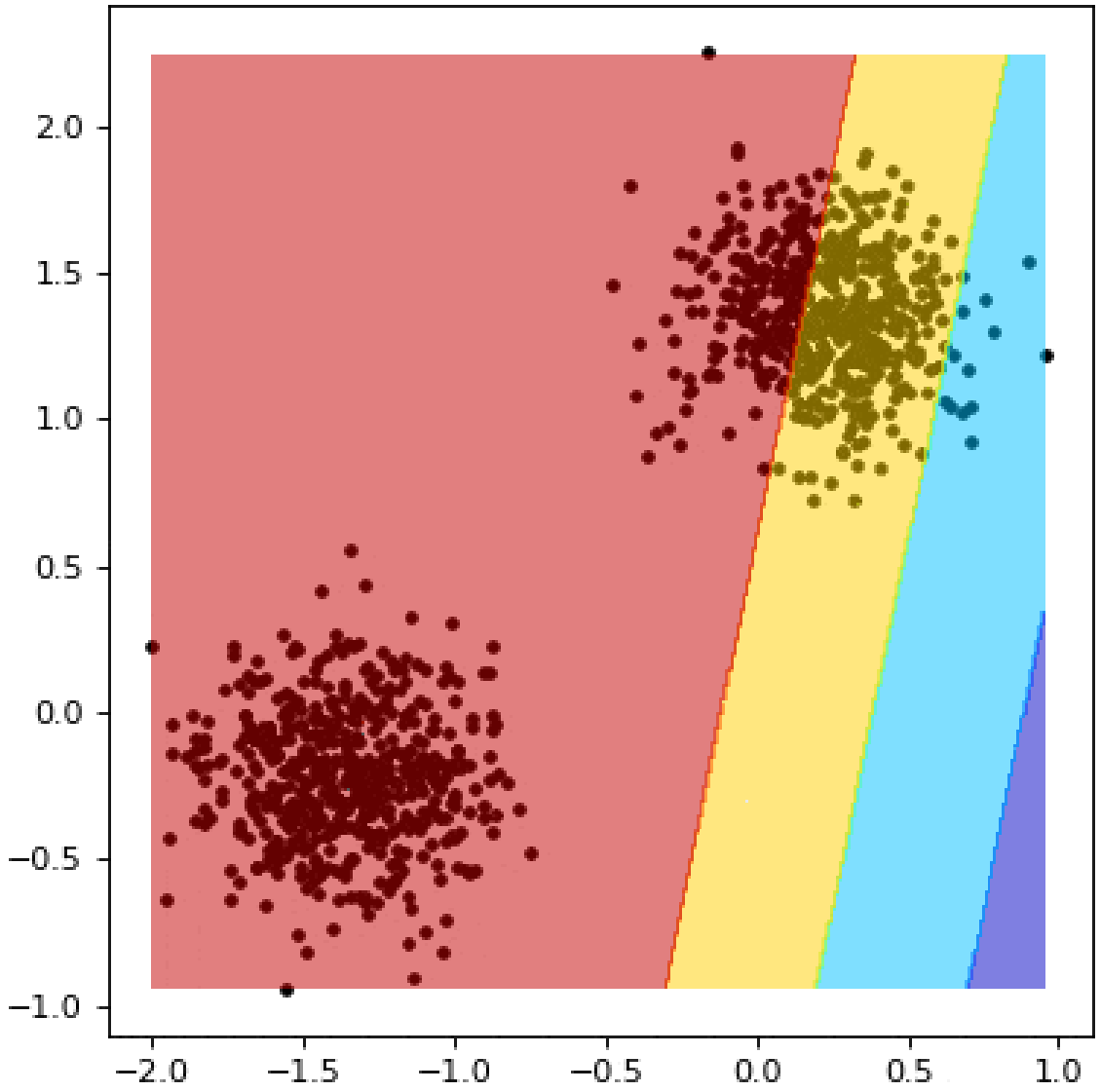
How does SVM work?



How does SVM work?

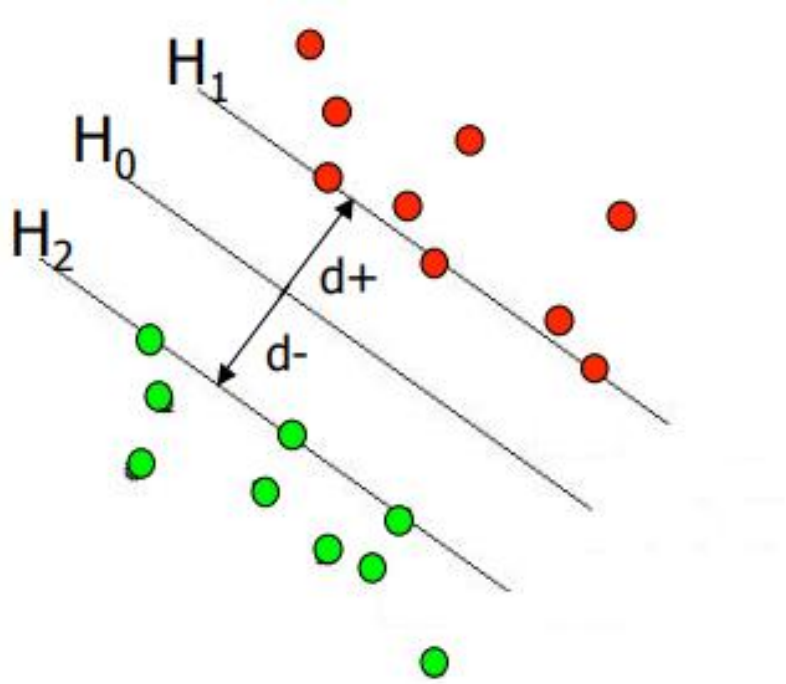


How does SVM work?



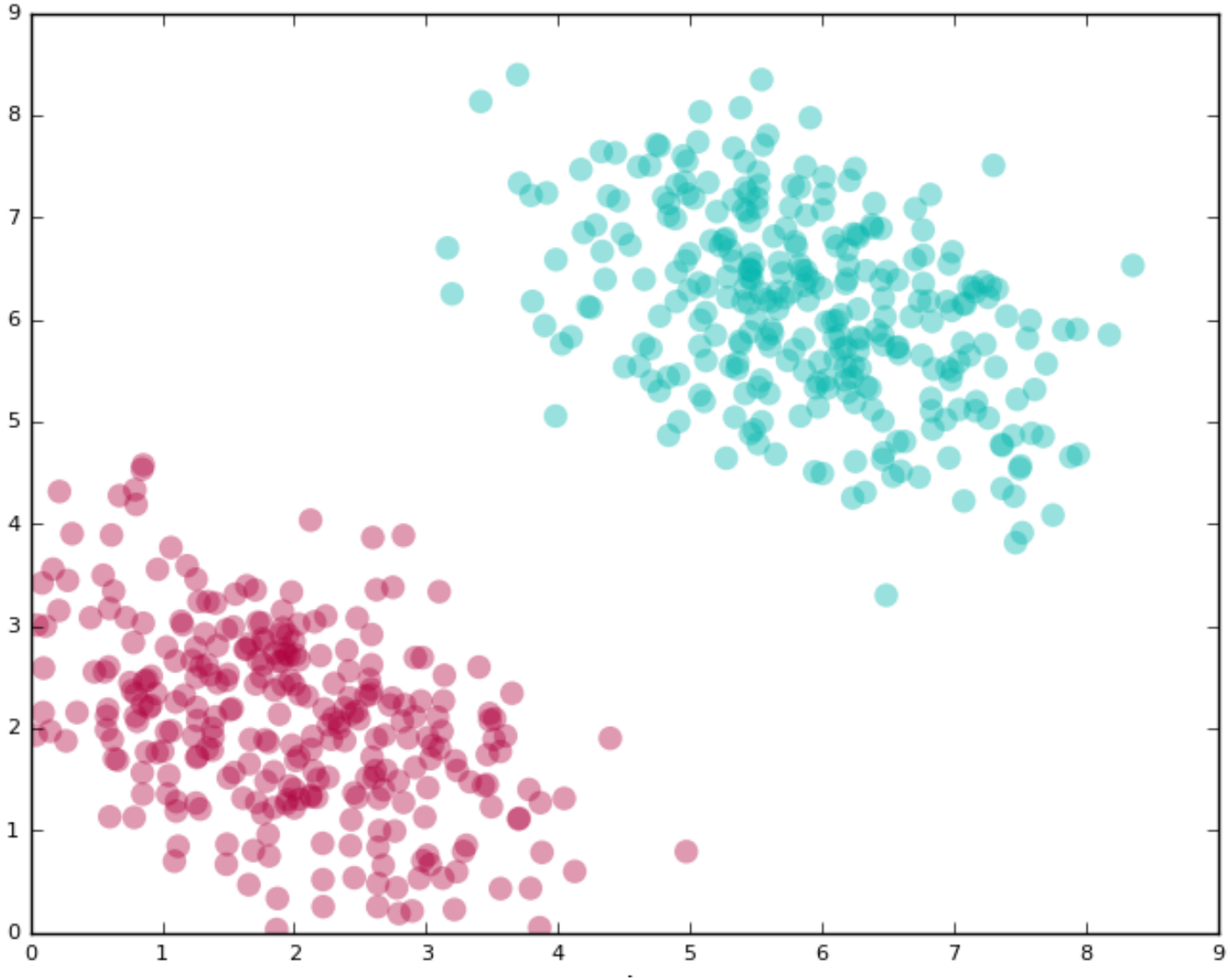
Some definitions

- Margin of Separation (d): the separation between the hyperplane and the closest data point.
- Optimal Hyperplane (maximal margin): the particular hyperplane for which the margin of separation d is maximized.
- We want a classifier (linear separator) with as big a margin as possible.



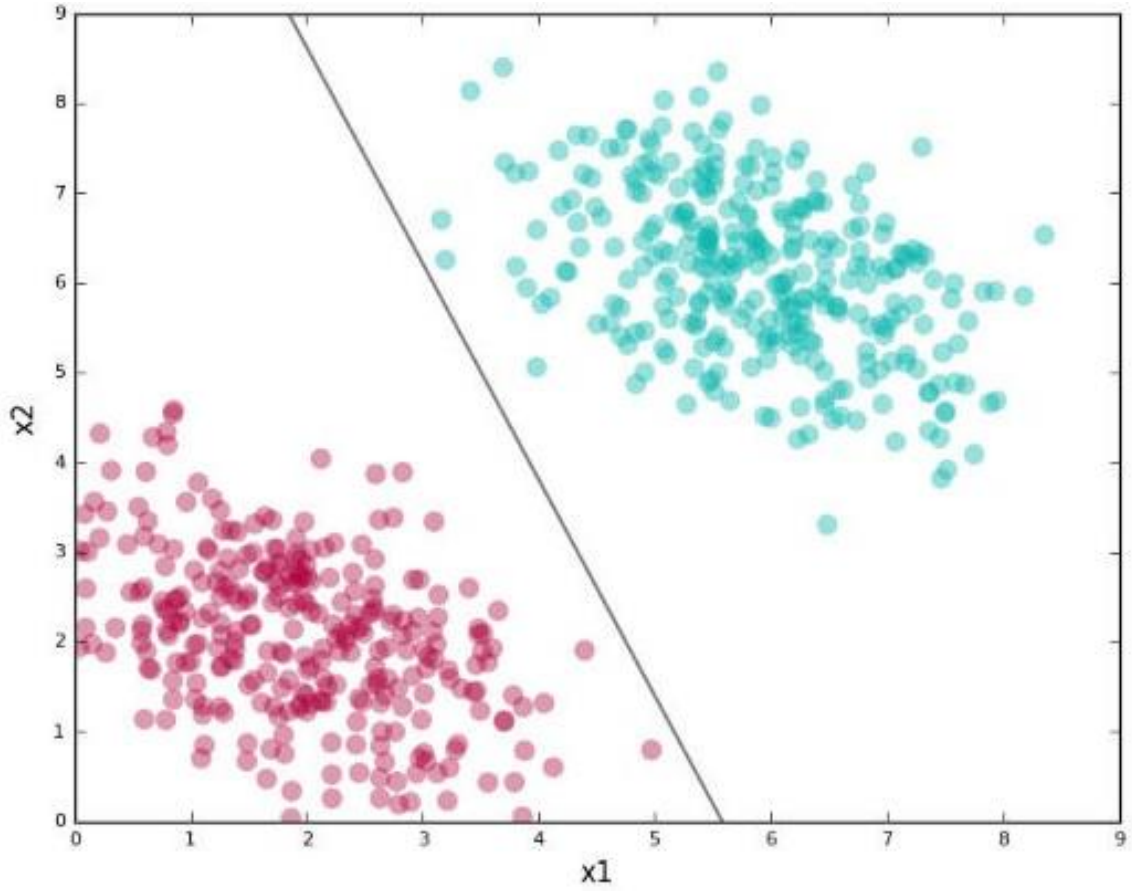
Good vs Bad Classifiers

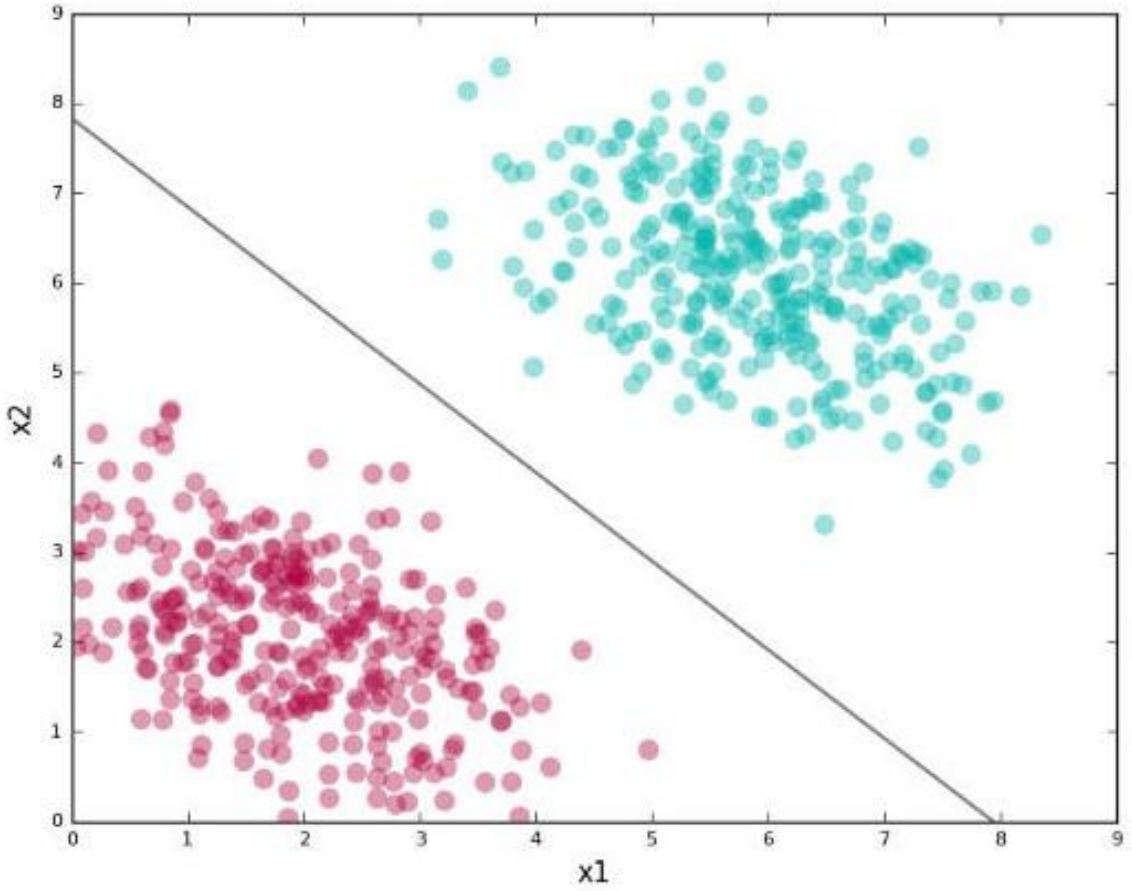
- In this case, finding a line that passes between the red and green clusters, is a good algorithm.



Good vs Bad Classifiers

- The line here is our separating boundary.
- The figure shows two possible classifiers for our problem.



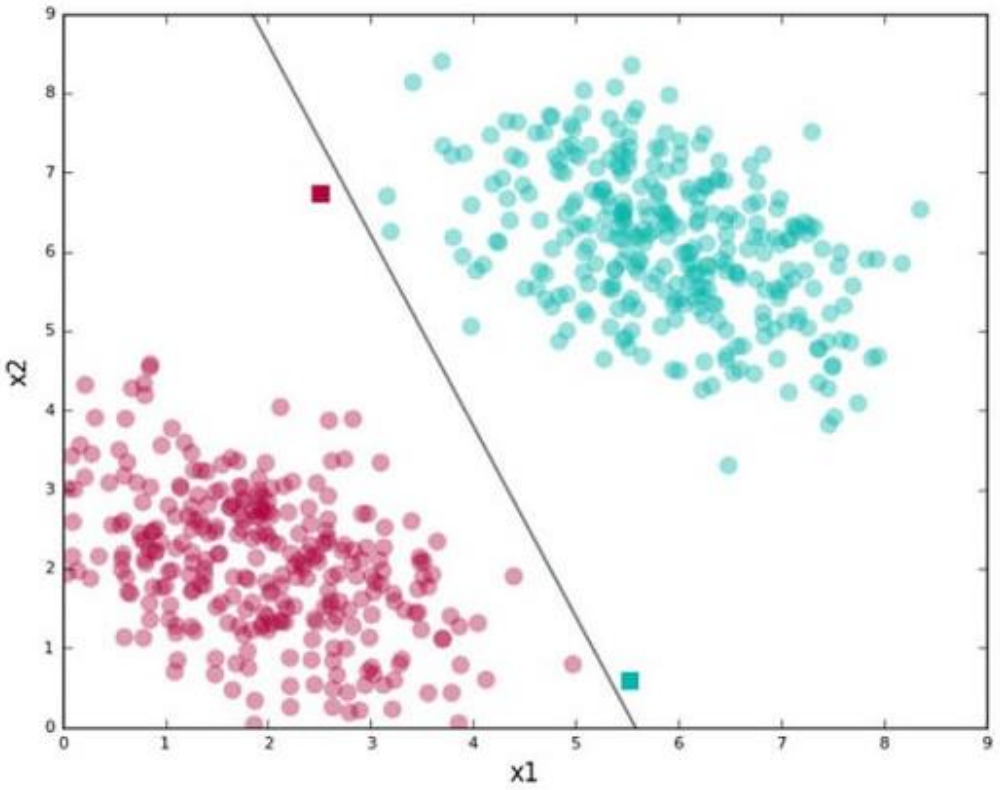


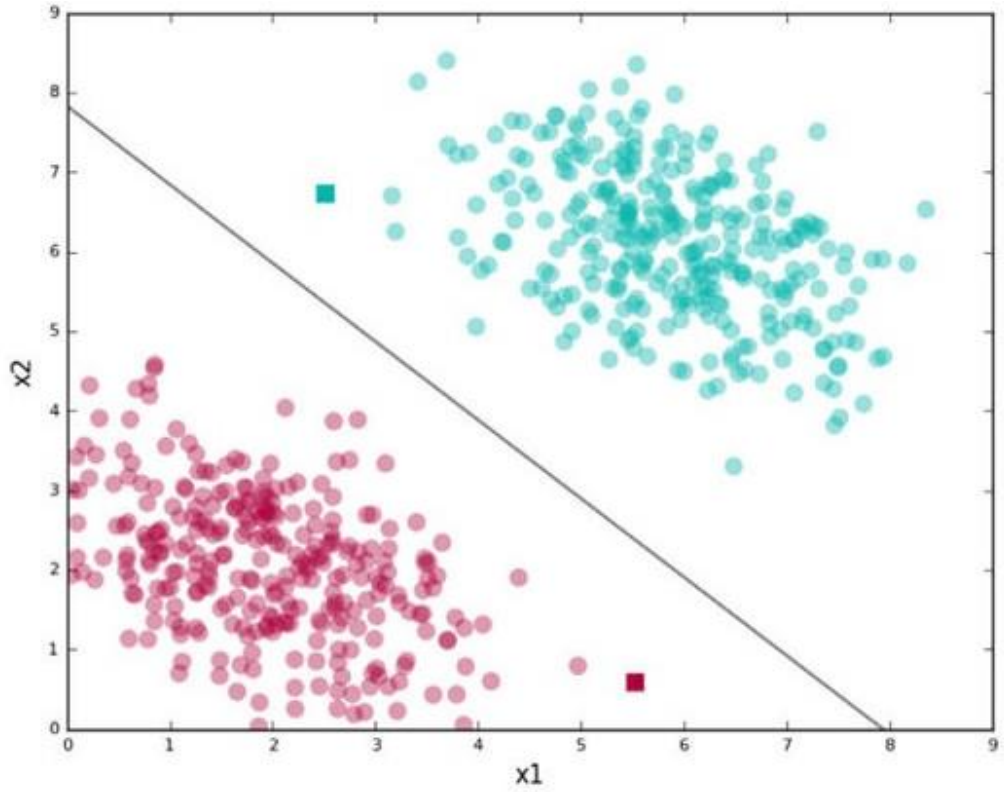
Good vs Bad Classifiers

- Here's an interesting question: both lines above separate the red and green clusters.
- Is there a good reason to choose one over another?
- The first line seems a bit "skewed."
- Sure, it separates the training data perfectly, but if it sees a test point that's a little farther out from the clusters, there is a good chance it would get the label wrong.
- The second line doesn't have this problem. For example, look at the test points shown as squares and the labels assigned by the classifiers in the figure next.

Good vs Bad Classifiers

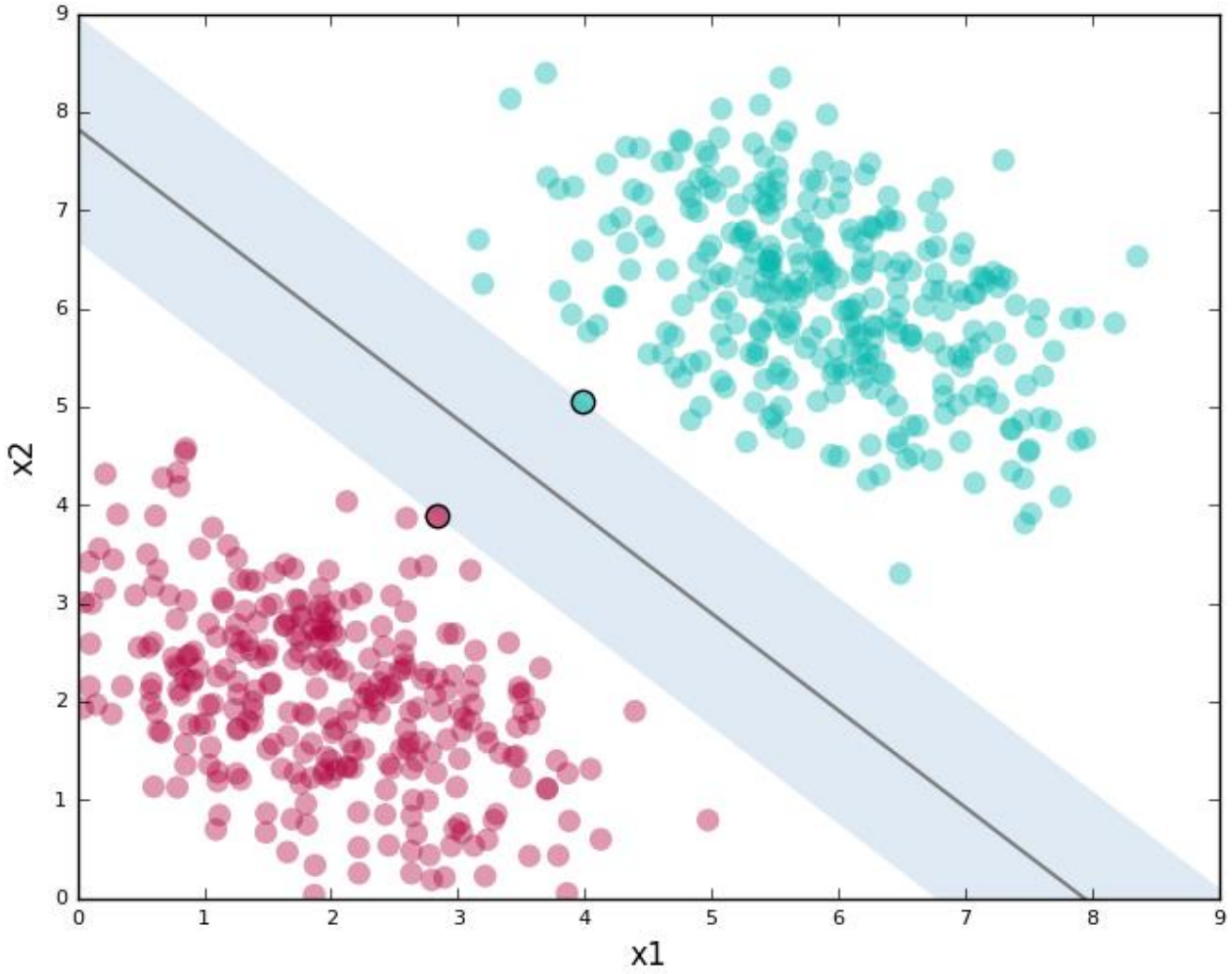
- The second line stays as far away as possible from both the clusters while getting the training data separation right.
- By being right in the middle of the two clusters, it is less “risky,” gives the data distributions for each class some scope, and thus generalizes well on test data.





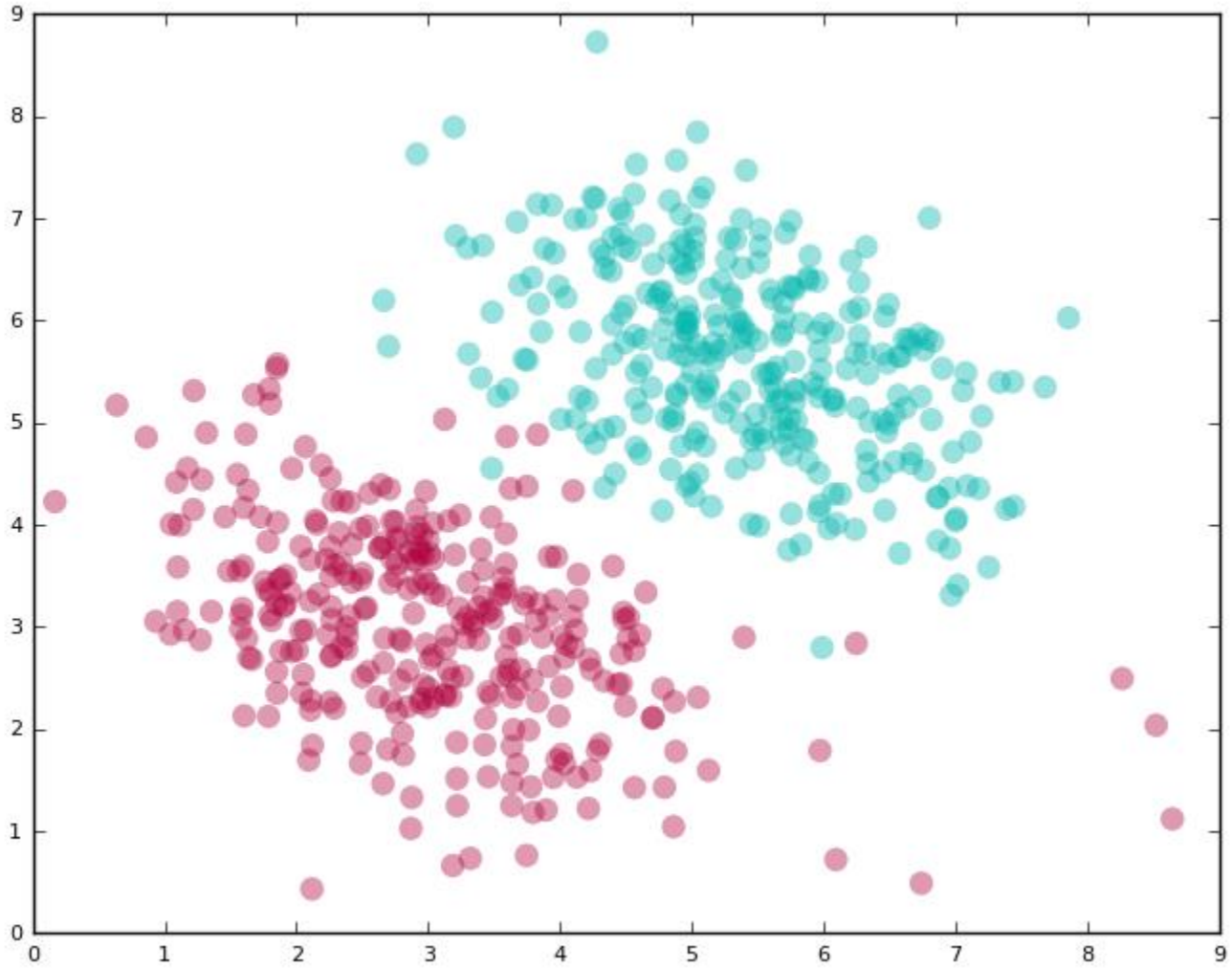
Good vs Bad Classifiers

- SVMs try to find the second kind of line.
- Here's the second line shown with the support vectors:
  - points with black edges (there are two of them) and
  - the margin (the shaded region).



Allowing for Errors

- We looked at the easy case of perfectly linearly separable data in the last section.
- Real-world data is, however, typically messy.
- You will almost always have a few instances that a linear classifier can't get right.
- Here's an example of such data

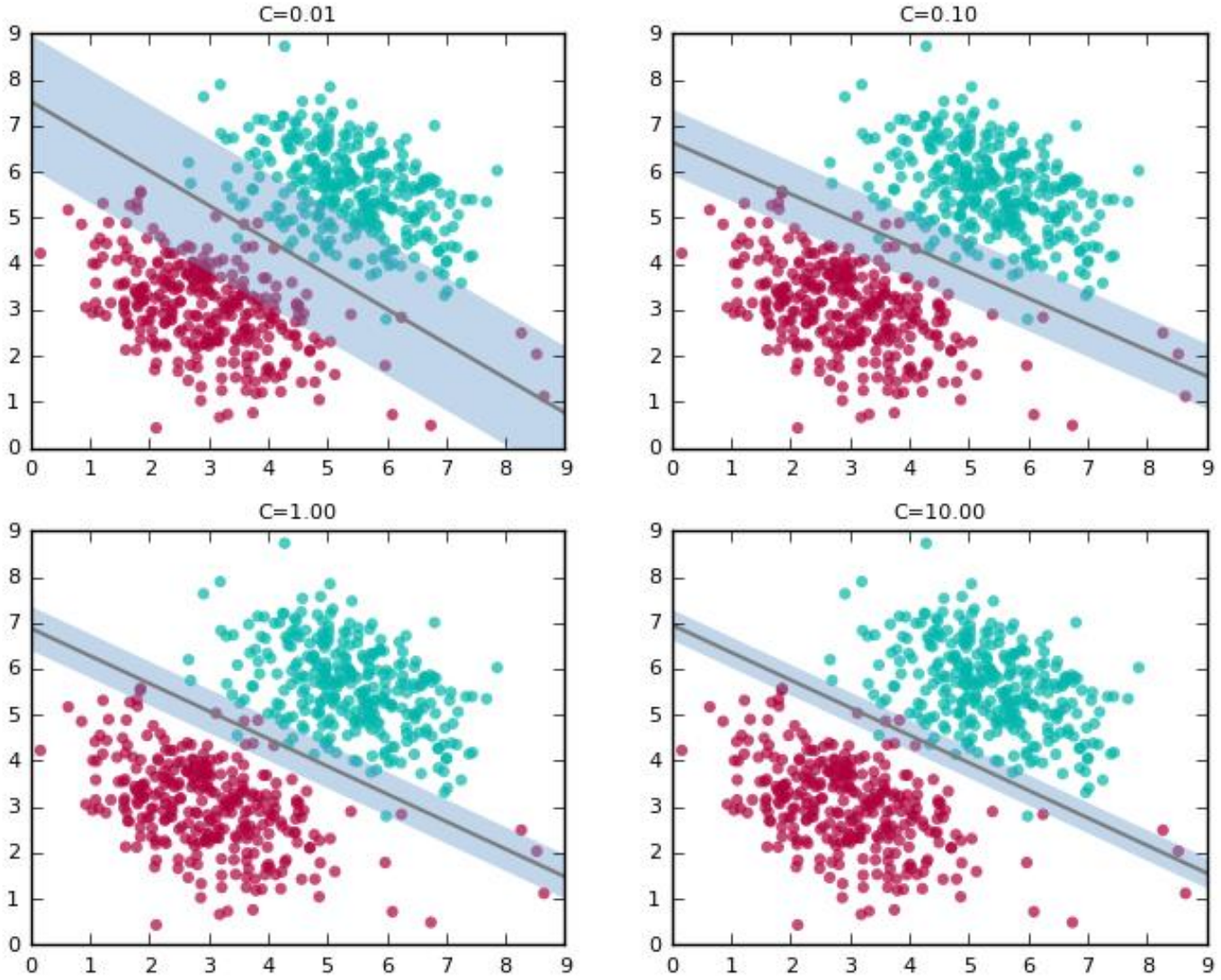


Allowing for Errors

- You can provide a parameter called “C” to your SVM; this allows you to dictate the tradeoff between
  - Having a wide margin.
  - Correctly classifying training data. A higher value of C implies you want lesser errors on the training data.
- You get better classification of training data at the expense of a wide margin.
- Plots on next slide show how the classifier and the margin vary as we increase the value of C

Allowing for Errors

- Note how the line “tilts” as we increase the value of C.
- Since this is a trade-off, note how the width of the margin shrinks as we increase the value of C.



Pros & Cons of Support Vector Machines

- Pros
  - Accuracy
  - Works well on smaller cleaner datasets
- Cons
  - Isn’t suited to larger datasets as the training time with SVMs can be high
  - Less effective on noisier datasets with overlapping classes

Thanks