Support Vector Machines: The Flexible One

by Dane Brown

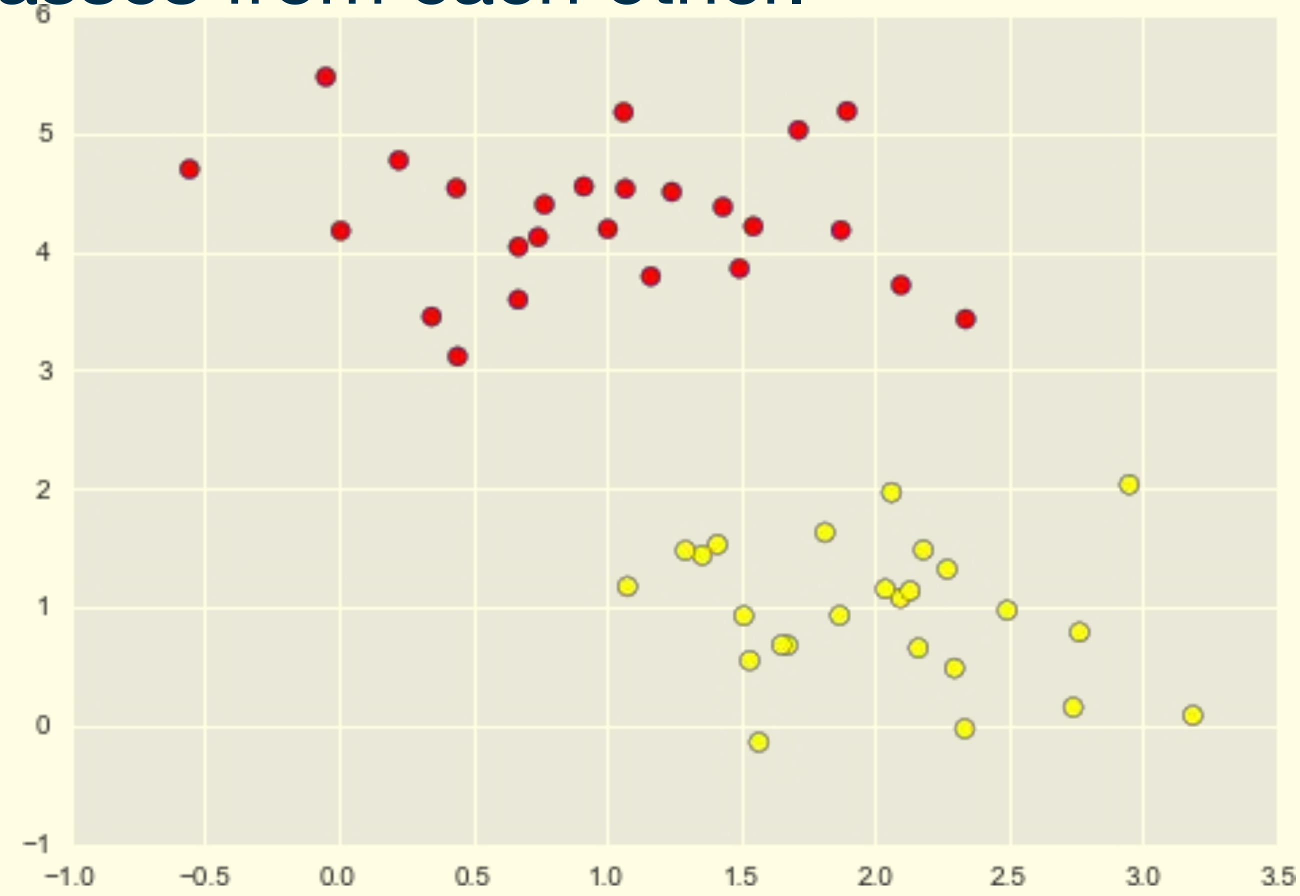
ML that Just Works

Support Vector Machine (SVM)

- Particularly powerful and flexible supervised learning algorithm
- Both **classification** and regression
- The theory also involves some complex maths/stats that you can read up if you interested

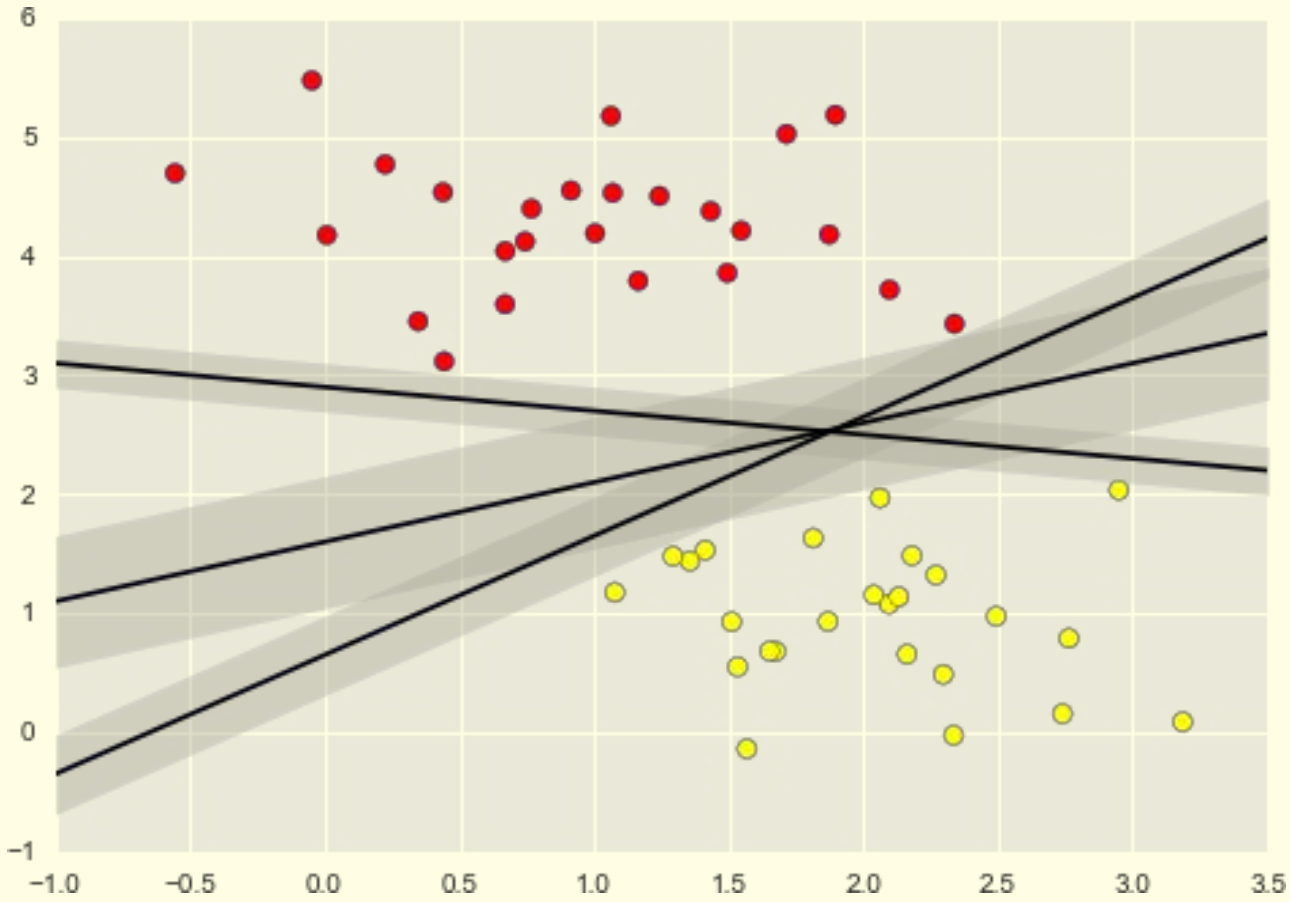
How SVMs Work

- **Discriminative classification:** instead of focussing on the detail of the data and modeling each class, it finds lines or curves that **divide** all the classes from each other.



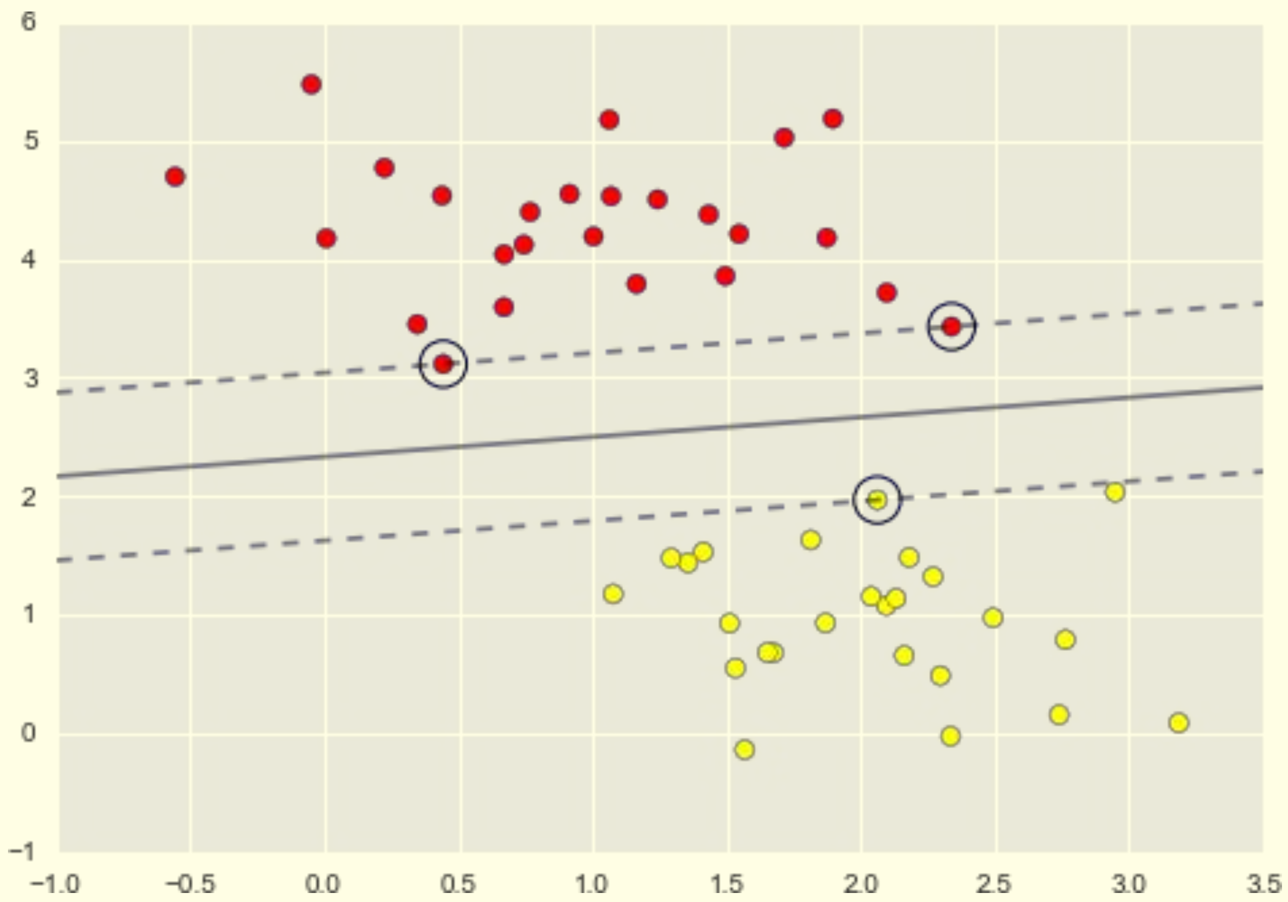
How SVMs Work

- the dividing line that separates classes is known as the **hyperplane**



How SVMs Work

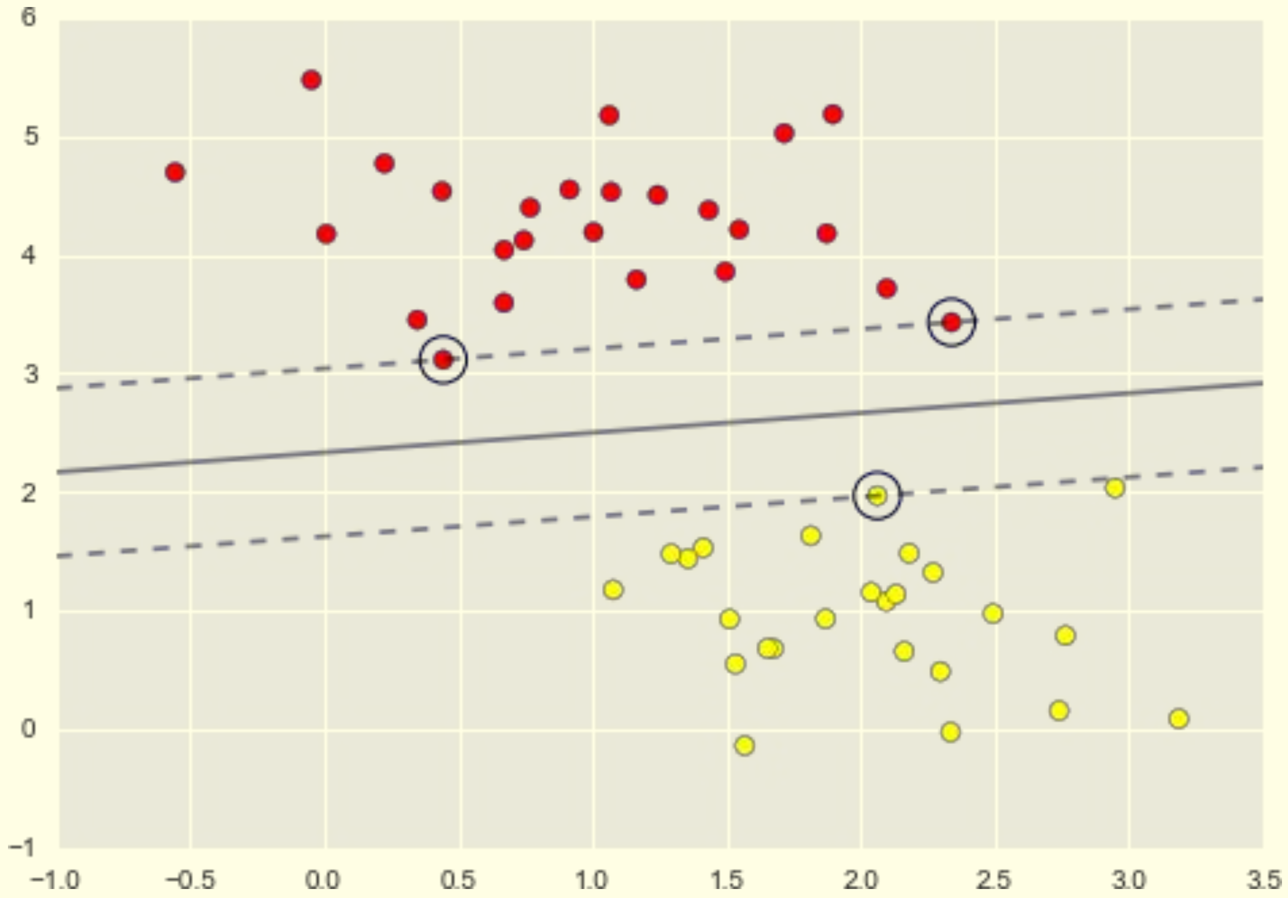
- the line that exists in the centre of the maximum margin is known as the optimal **hyperplane**



Support Vectors

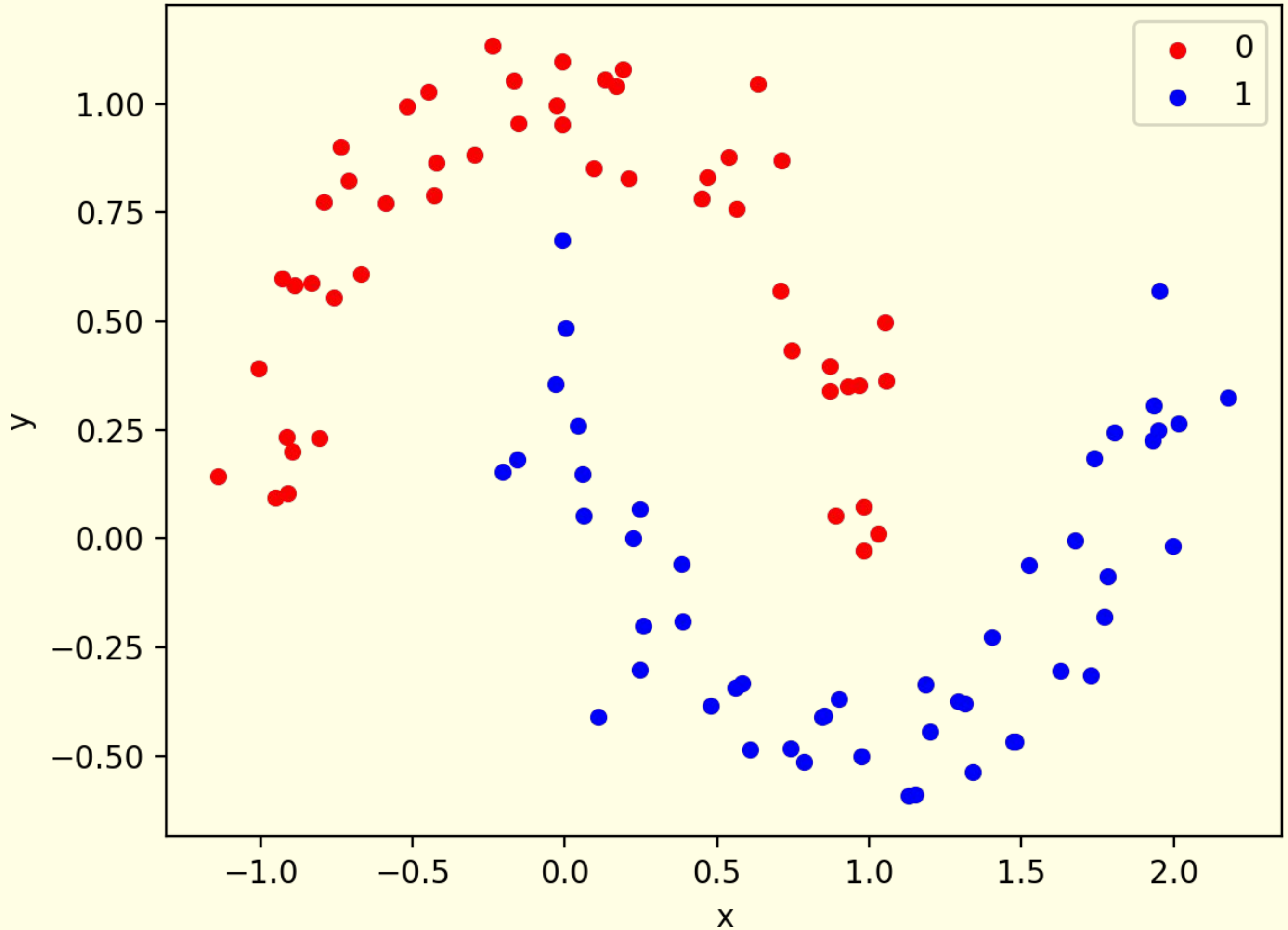
Check it out -> CV_ML

- Only the position of the support vectors matter
- Other points do not affect model fitting



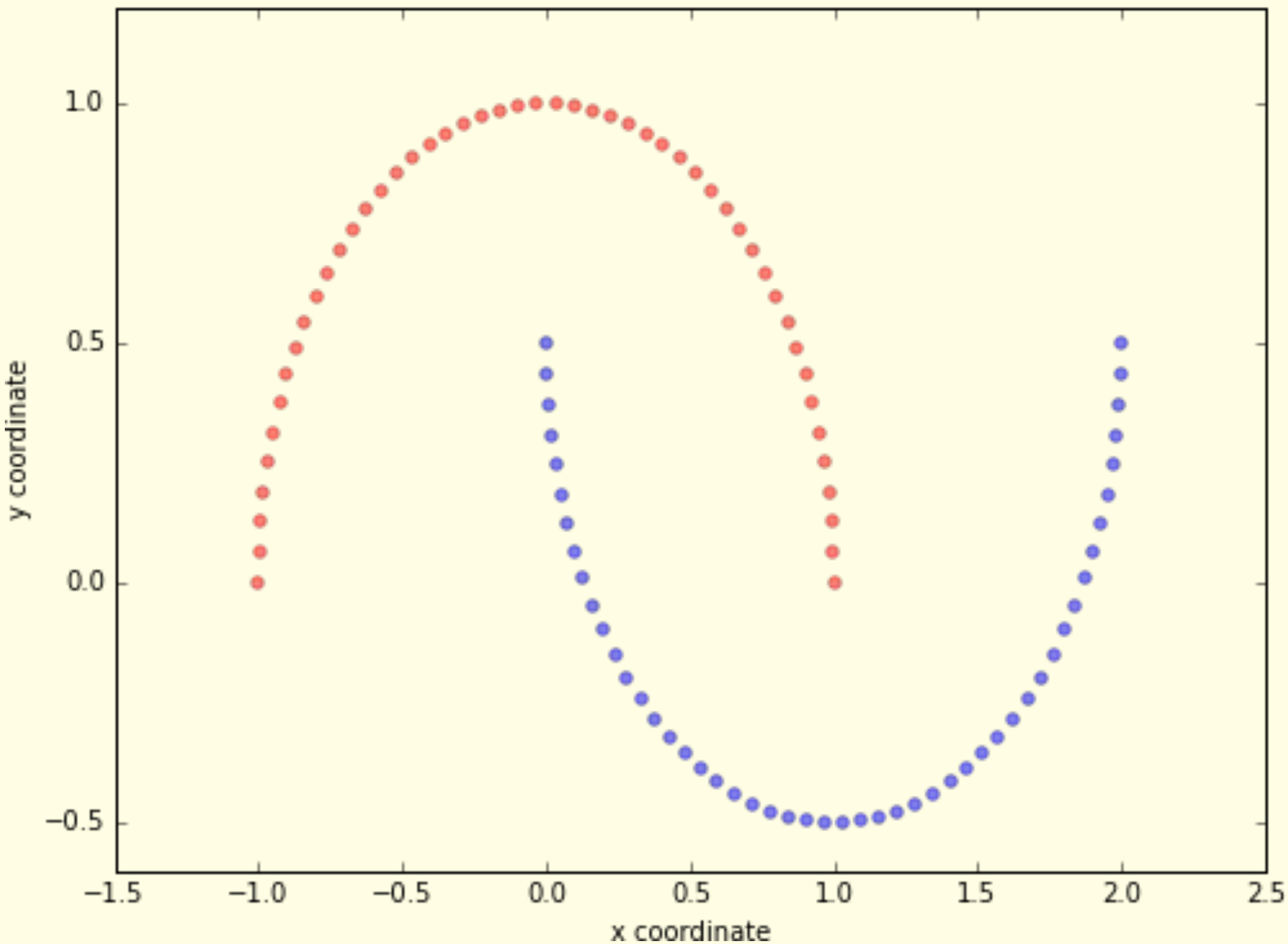
Why SVMs are Flexible?

- The **kernel trick** transforms data into higher-dimensional space that suits non-linear patterns of data up to the n^{th} dimension!



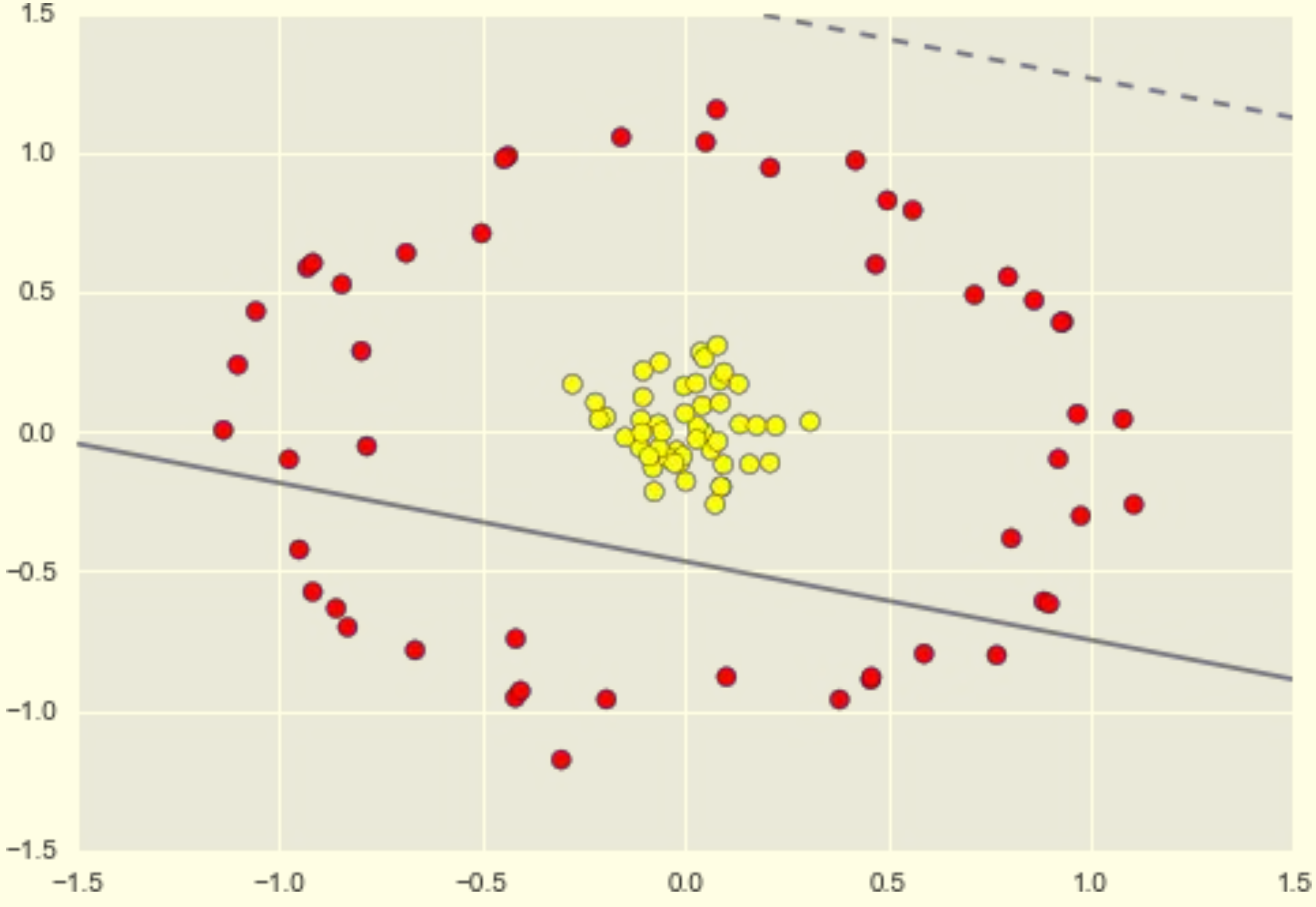
Transcend that which is Non-Linear (soon)

- Even without scatter this problem is still non-linear



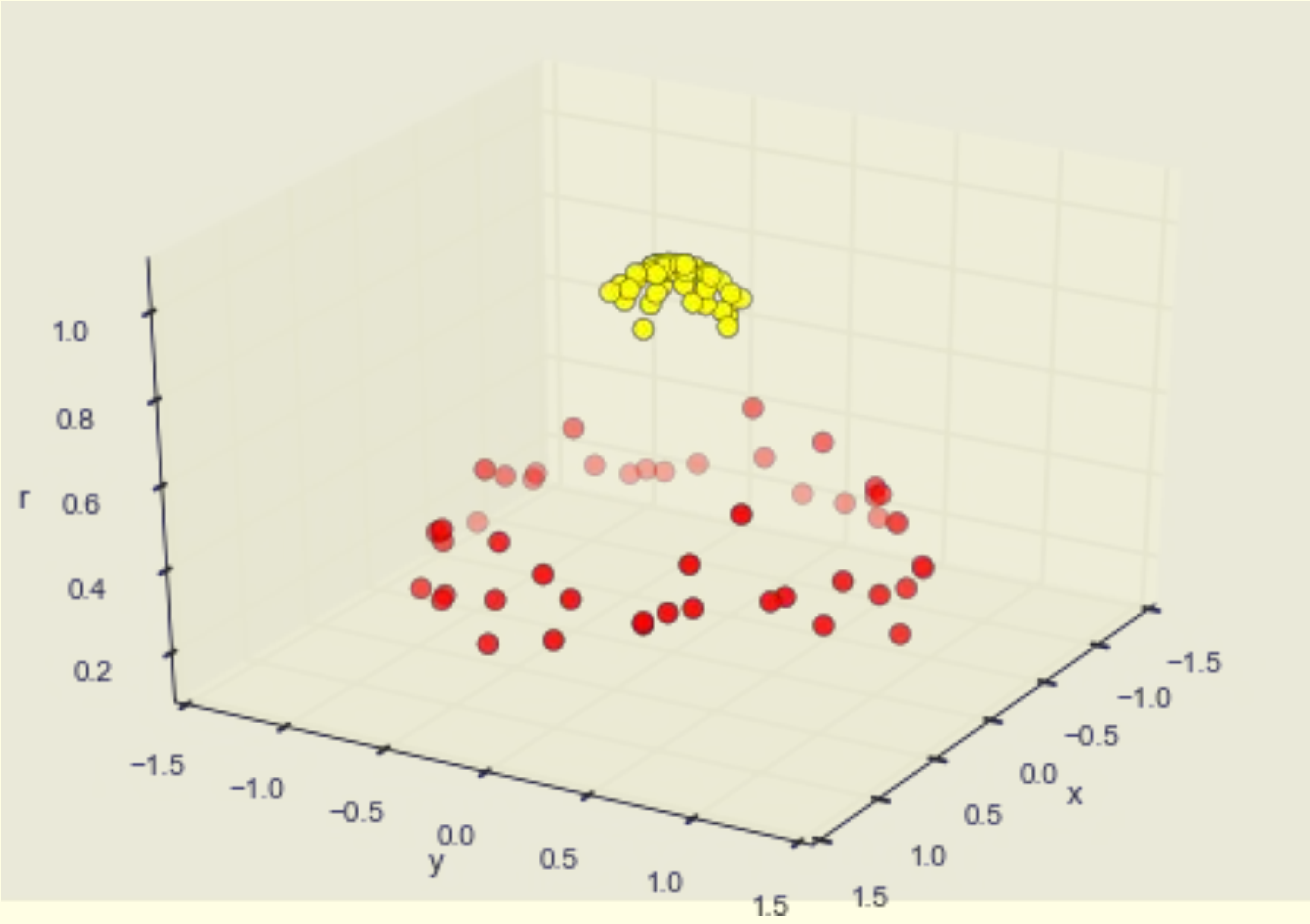
Find the Maximum Margin in 2D???

- Surely this can be solved by adding a dimension?



Find the Maximum Margin in 3D Space

- Easy to see that r = 0.7 gives optimal margin



Fitting the Kernel in N Dimensions

- Compute a basis function centred at every point in feature space
- But projecting N points into N dimensions becomes extremely computationally intensive as N grows large.

SVM uses the **Kernel Trick**

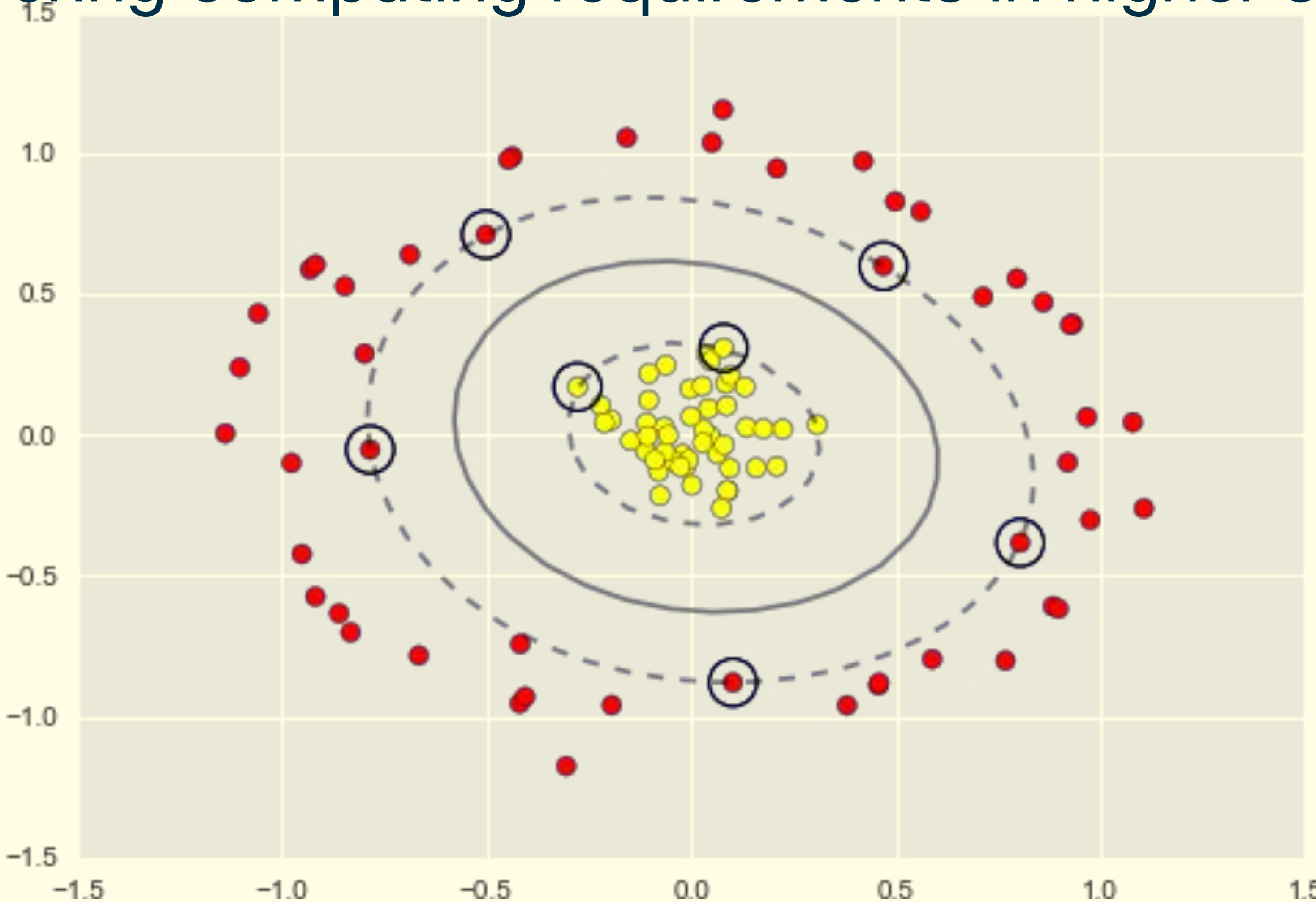
- Adding dimensions can thus become computationally expensive

- Kernel methods can operate in a high-dimensional space without explicitly computing the coordinates of the data in that space. **SVM Kernel Trick**:
  - rather compute the inner products between the images of all pairs of data in the feature space
  - Substantially cheaper than the explicit computation of the coordinates
  - transforms data into another dimension so that it has a clear class-dividing line

Kernel: Find the Maximum Margin

Check it out -> CV_ML

- Result: Finds a maximum margin in 2D space, lowering computing requirements in higher space

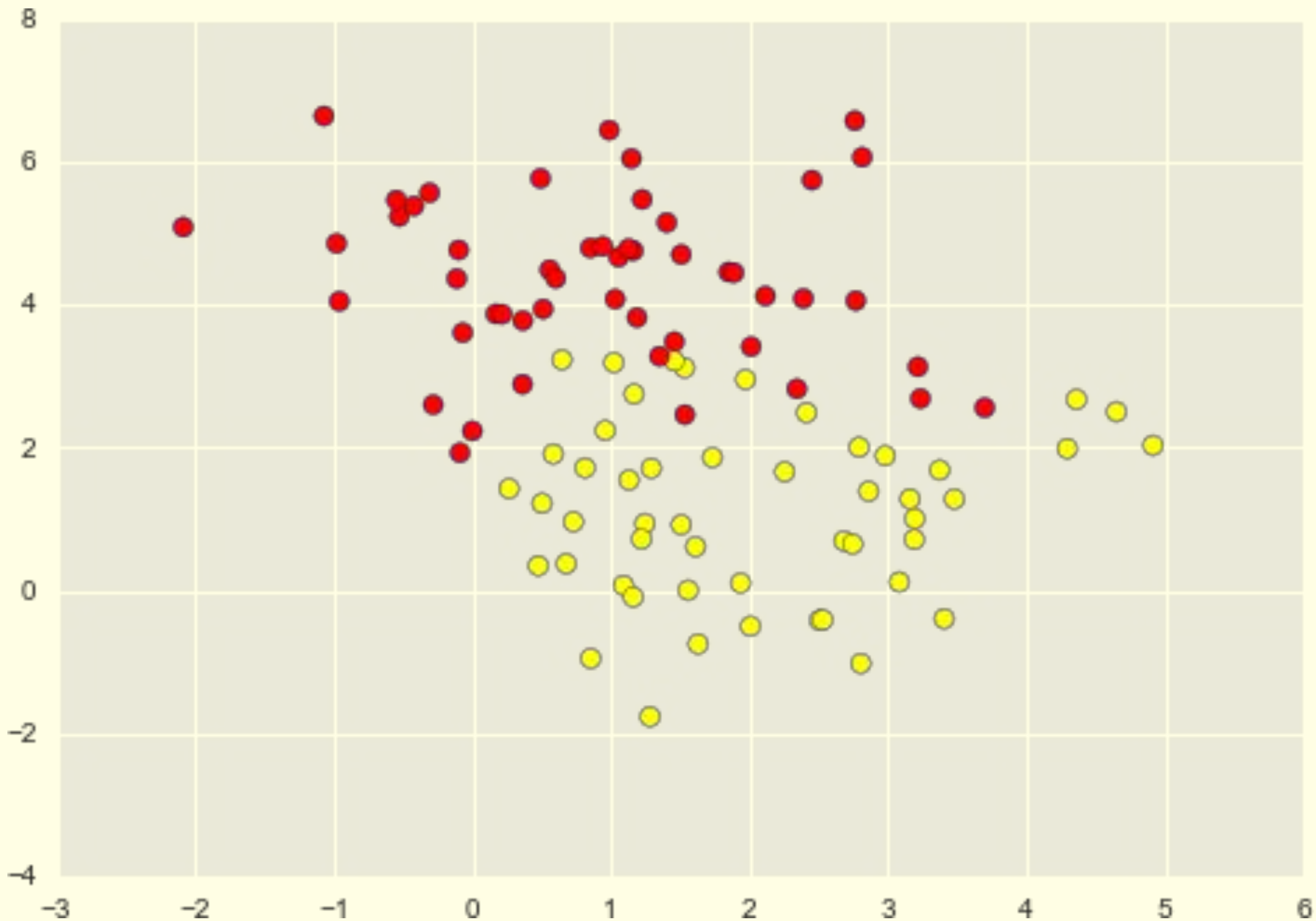


What is the Best Kernel?

- The RBF kernel normally works best for your average problem, but..
- For sharp features such as letters or edge features such as fingerprints, the linear kernel works well
- Other and custom kernels are quite particular
- Mechanical Solution???
- **Parameter estimation using grid search with cross-validation**

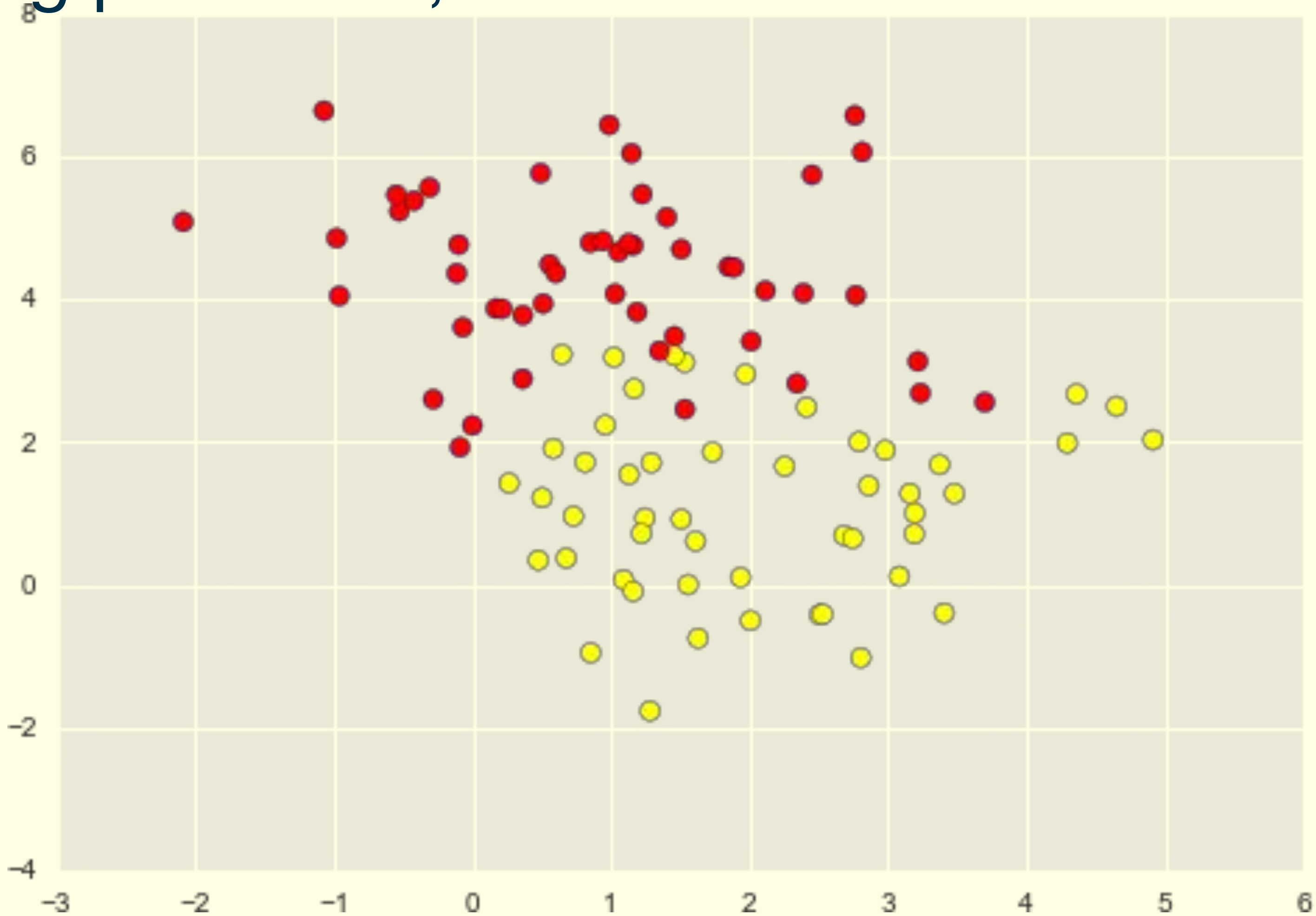
Tuning the SVM: Softening Margins

- But what about overlapping data?



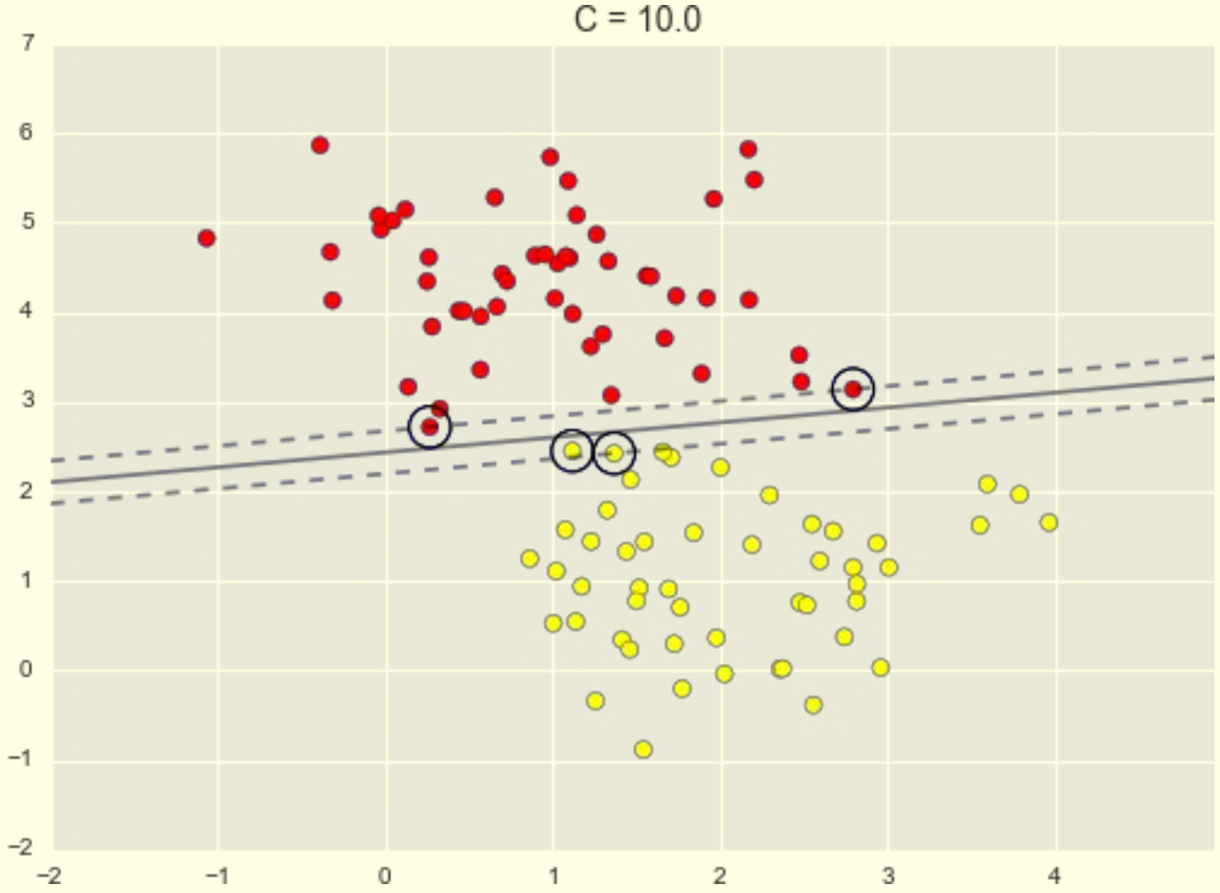
Tuning the SVM: Softening Margins

- The hardness of the margin is controlled by a tuning parameter, most often known as C.



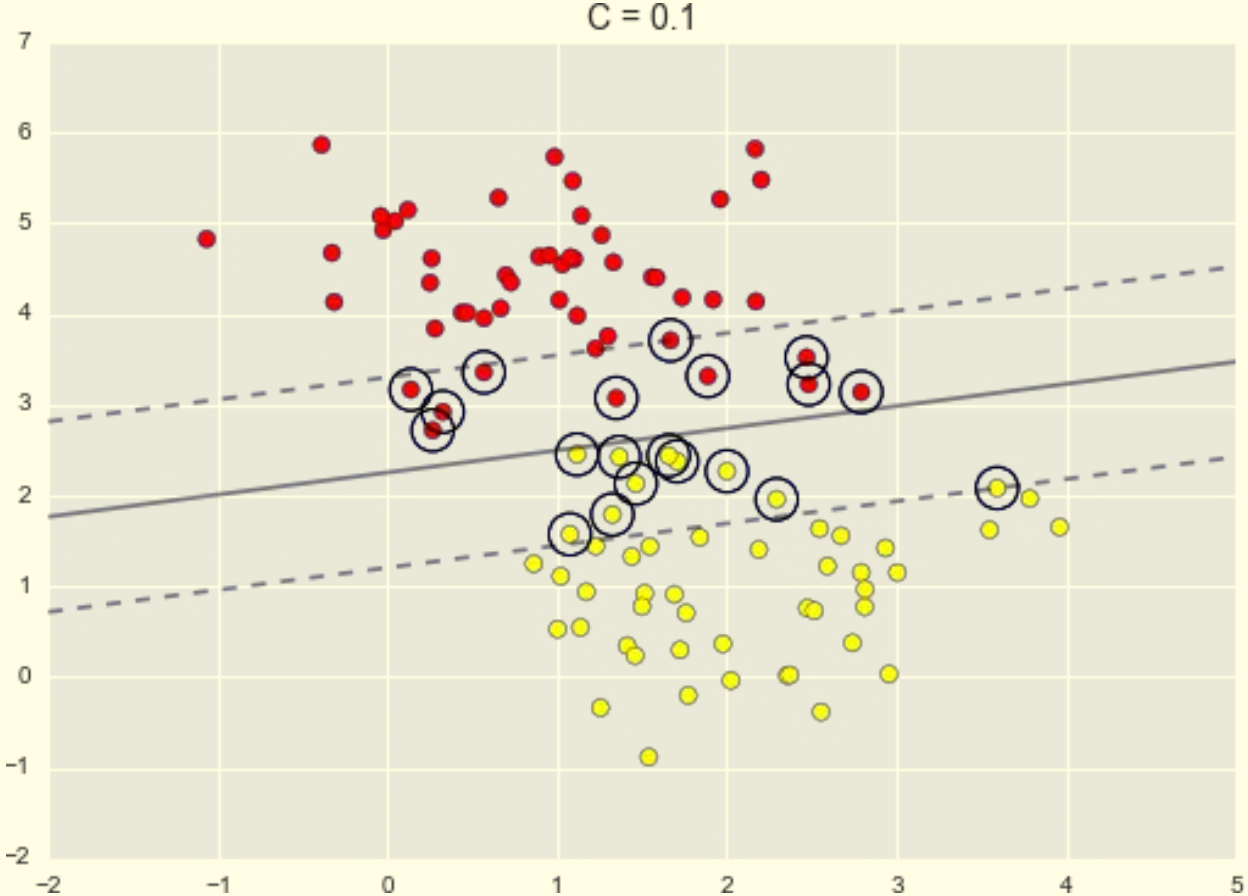
Tuning the SVM: Softening Margins

- Large C: hard margin, and points can't lie in it.



Tuning the SVM: Softening Margins

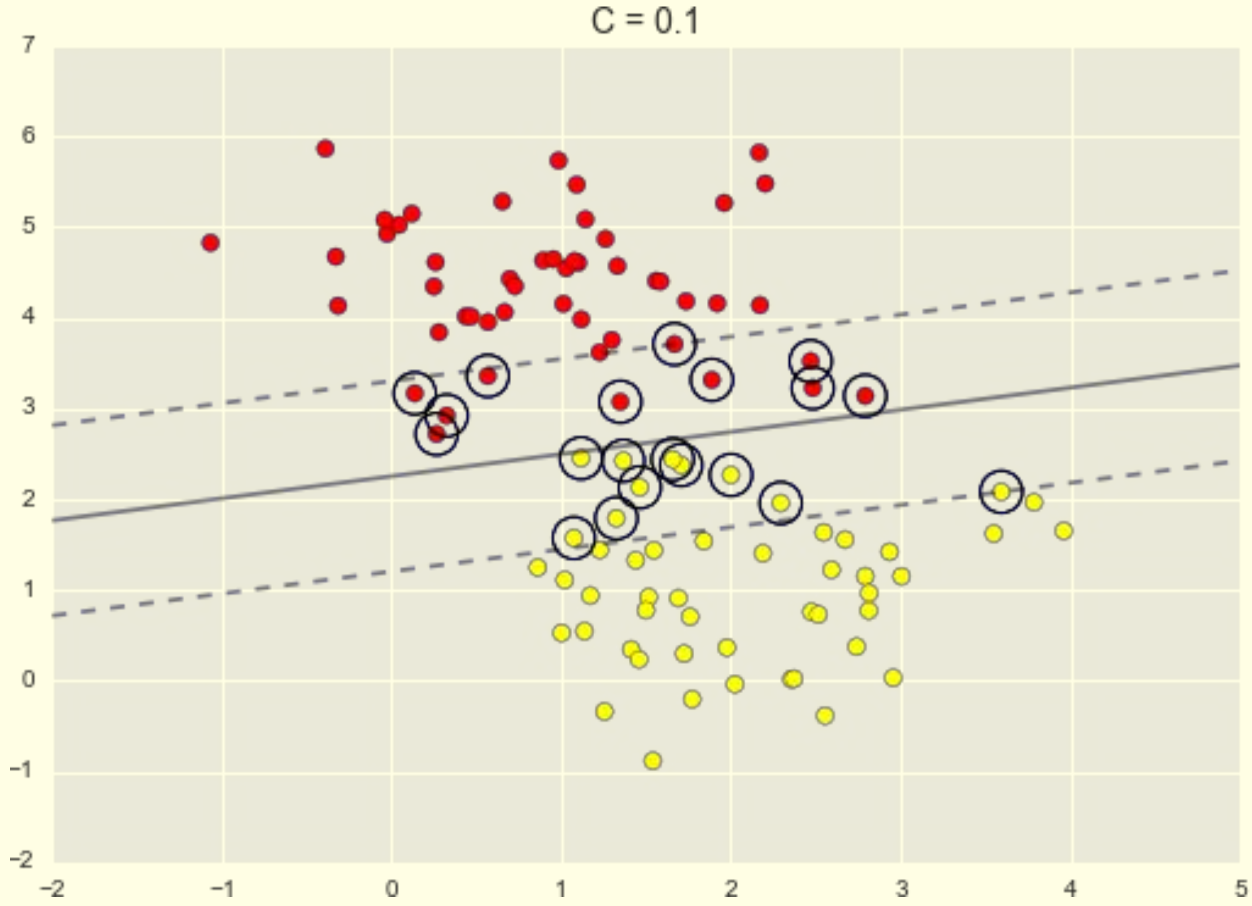
- These conflicting support vectors are avoided to reduce borderline cases but might introduce false predictions on unseen data



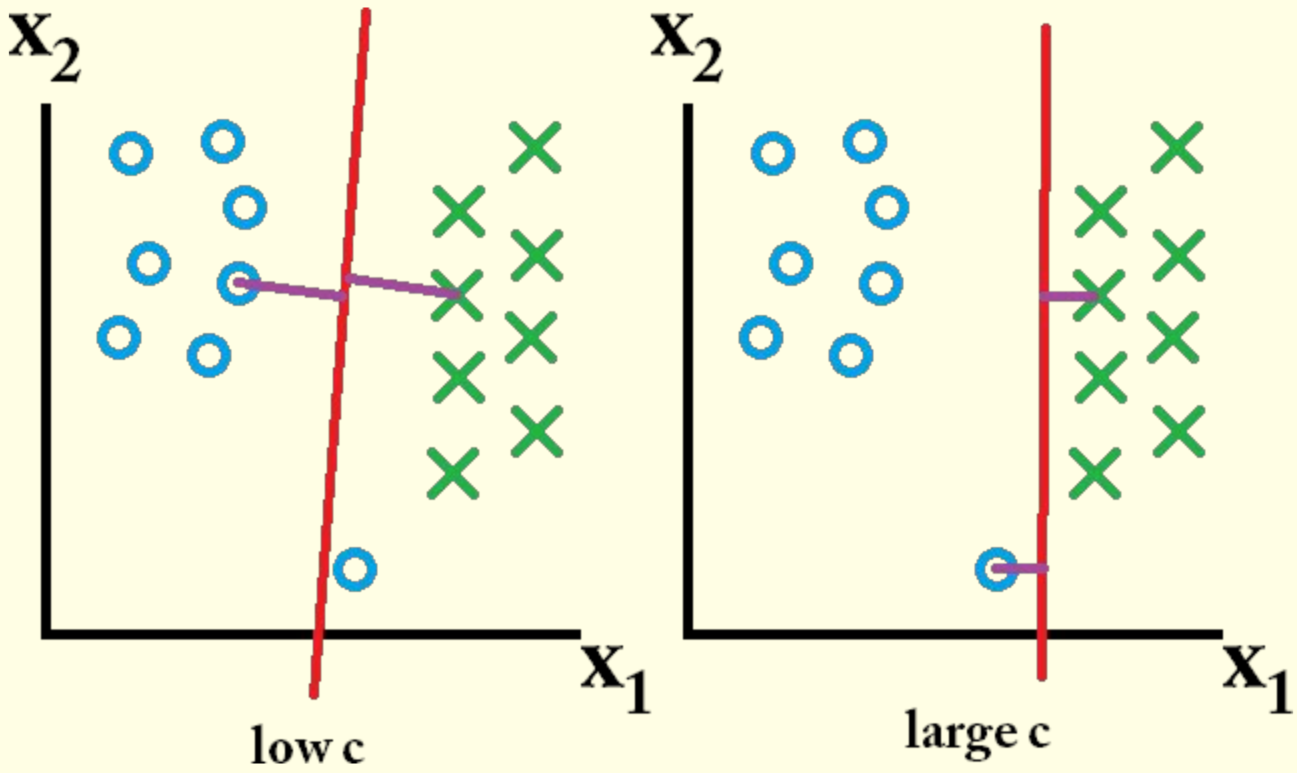
Tuning the SVM: Softening Margins

Check it out -> CV_ML

- Smaller C: the margin is softer, and can grow to encompass some support vectors

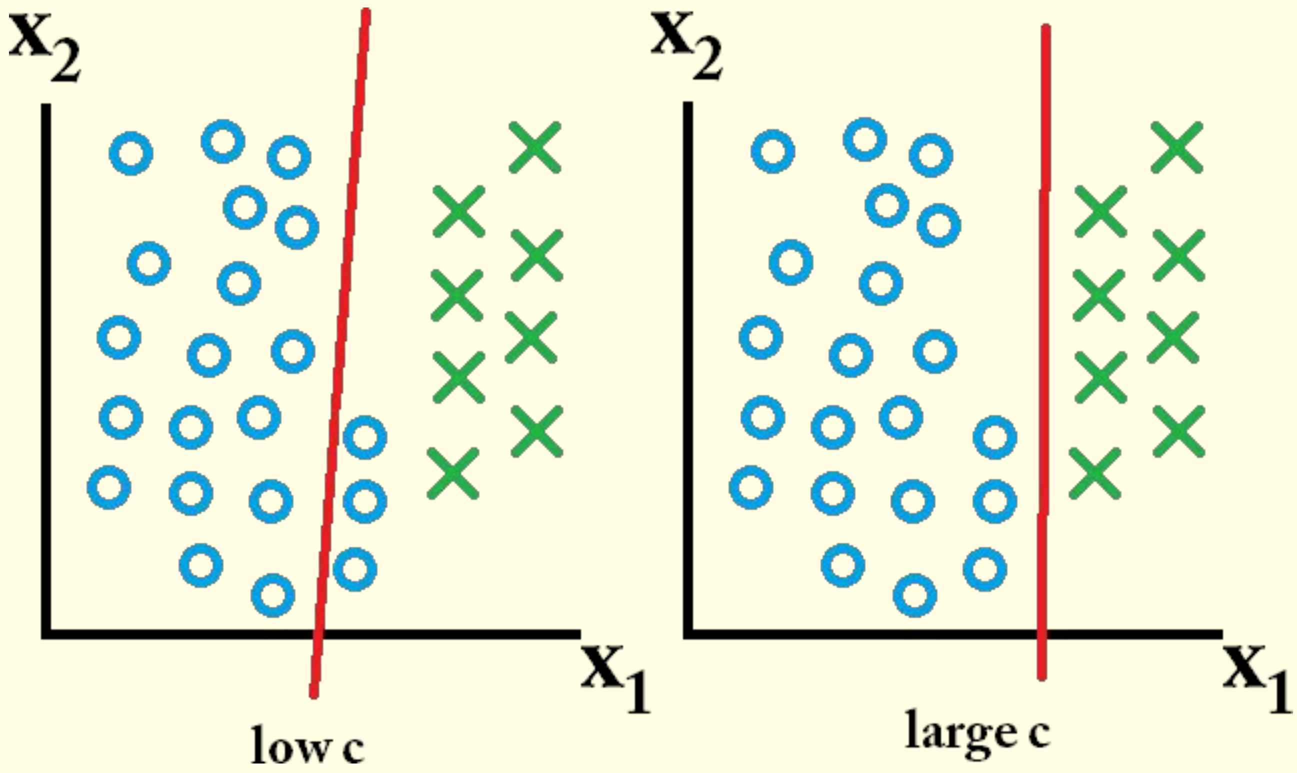


Softening Margins Example



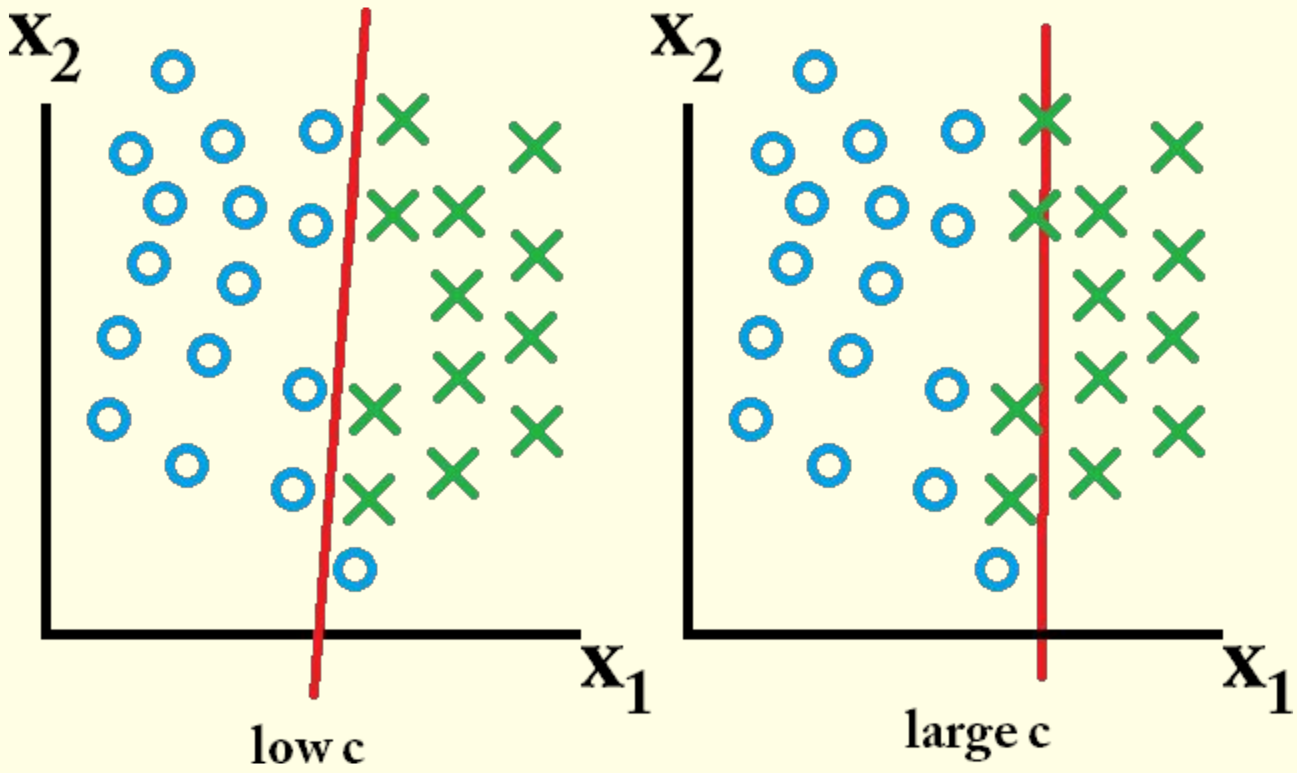
Softening Margins Example

- Here large *C* works best



Softening Margins Example

- Now small *C* works best



Tuning the SVM: Grid Search + Cross Validation

`Check it out -> CV_ML`

- Even with the kernel trick high-dimensionality problems sometimes require preprocessing
- PCA again

Disadvantages of SVM

- SVMs rely on a suitable choice for the softening parameter C using grid search + cross-validation

- On large datasets, this becomes time consuming

- Newer CNNs and DNNs classifiers generally outperform it in accuracy (but are ultra expensive)

Advantages of SVM

- Support vectors are very compact models
  - take up very little memory
- Work well with high-dimensional data
  - even data with more dimensions than samples
  - the above is challenging for most other algorithms
- Once the model is trained (often slow), the prediction phase is fairly fast
- Their integration with kernel methods makes them very versatile

Using your Own Datasets

by Dane Brown

Let’s do what we came here for