SUPPORT VECTOR CLASSIFIER

Dr. Brian Mc Ginley

GENERALISED LINEAR MODELS

- Recall

$$y = mx + c$$

- is the equation of a line.
- Basically, if that equation is satisfied for a point (x , y) then the point falls on the line. We can also describe a line as

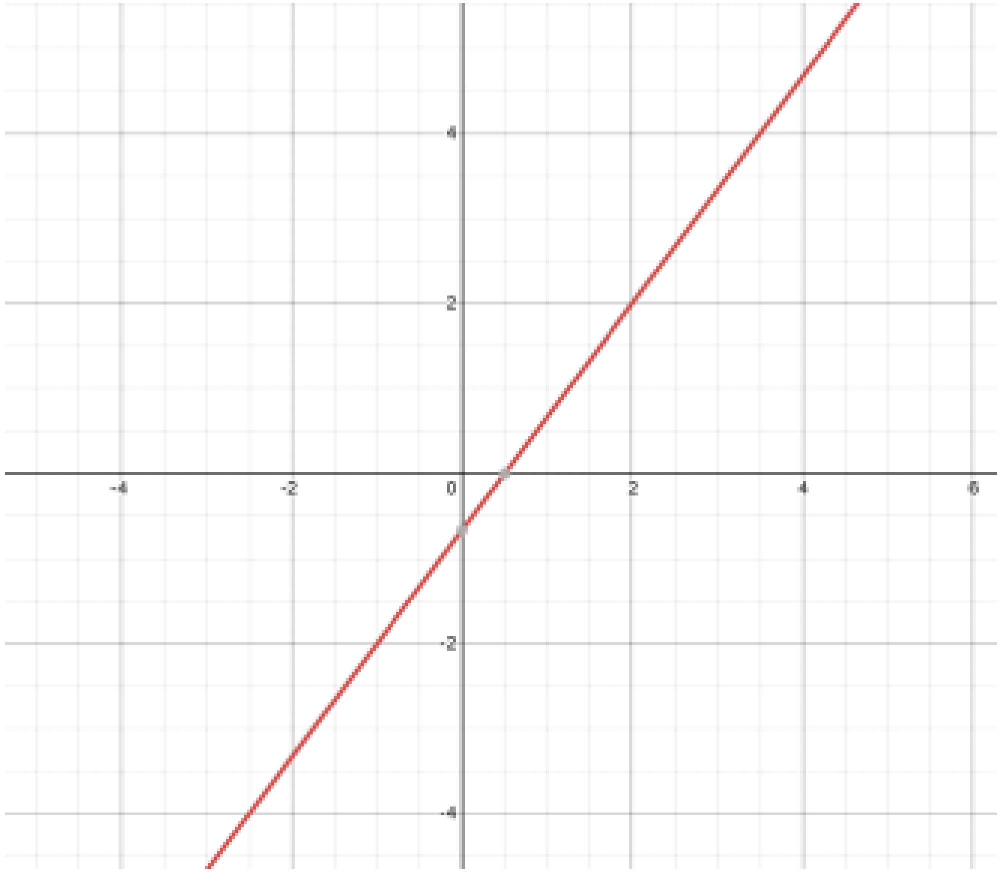
$$w_2 x_2 + w_1 x_1 + w_0 = 0$$

$$\mathbf{w}^T \mathbf{x} + w_0 = 0$$

- is also an equation of a line, a vector equation of a line. Whatever way you want to think about it, it described a line - is linear.

JUNIOR CERT GEOMETRY

- The following is the plot of the line $4x - 3y - 2 = 0$
- Clearly from the picture $(2, 2)$ falls on the line and $4(2) - 3(2) - 2 = 0$ so the equation is satisfied showing this.
- Look at the image, the point $(2, -2)$ is clearly to the right of the line. Try the equation and the result is $4(2) - 3(-2) - 2 = 12$. A positive number.
- Now, the point $(-4, 2)$ is clearly to the left of the line. Equation: $4(-4) - 3(2) - 2 = -24$. A negative number.
- So, whether the calculation of the line is positive or negative will tell us which side of the line it falls on.



LET'S EXTEND THIS

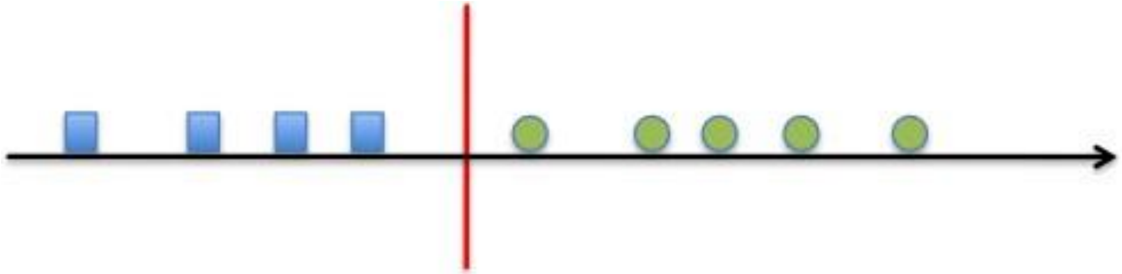
- Take $w = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$ and $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$
- Put it into the equation of the line: $\boldsymbol{w}^T\boldsymbol{x} + w_0 = (w_1 \quad w_2)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + w_0 = w_0 + w_1x_1 + w_2x_2$
- If $w = \begin{pmatrix} 4 \\ -3 \end{pmatrix}$ and $w_0 = -2$
- Then we have $4x_1 - 3x_2 - 2$
- Then if we evaluate the sample $(2, -2)$, we find it is positive, so we classify as the positive class. $(-4, 2)$ classifies in the negative class. This expands easily to larger dimensions - we just can't visualise past 3!
- The result $\boldsymbol{w}^T\boldsymbol{x} + w_0$ describes a hyperplane in any dimension

HYPERPLANE

- Linear classifier has a linear boundary (hyperplane)

$$\boldsymbol{w}^T\boldsymbol{x} + w_0$$

- which separates the space into two "half-spaces". In 1D this is simply a threshold

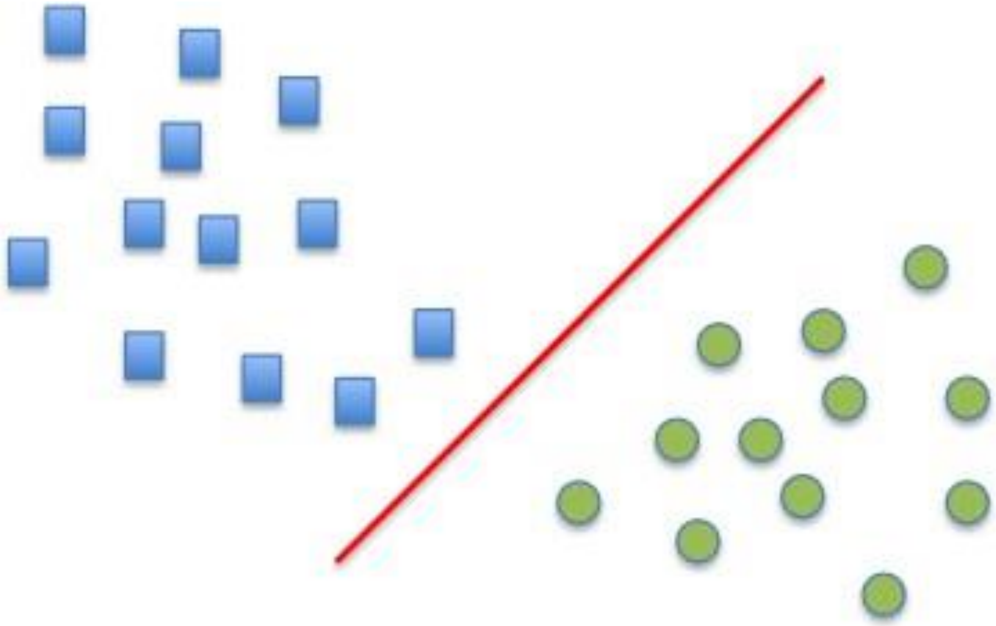


HYPERPLANE

- Linear classifier has a linear boundary (hyperplane)

$$\boldsymbol{w}^T \boldsymbol{x} + w_0$$

- which separates the space into two “half-spaces”. In 2D this is a line

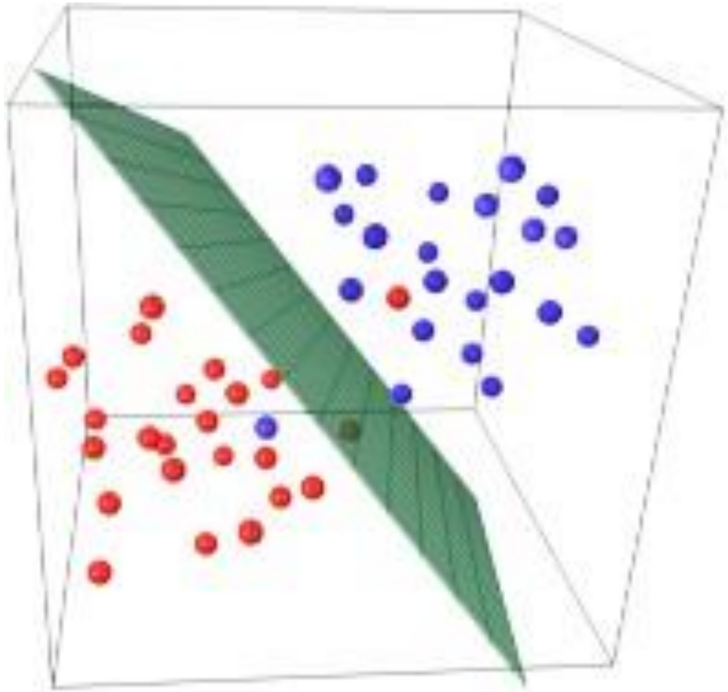


HYPERPLANE

- Linear classifier has a linear boundary (hyperplane)

$$\boldsymbol{w}^T\boldsymbol{x} + w_0$$

- which separates the space into two "half-spaces". In 3D this is a plane

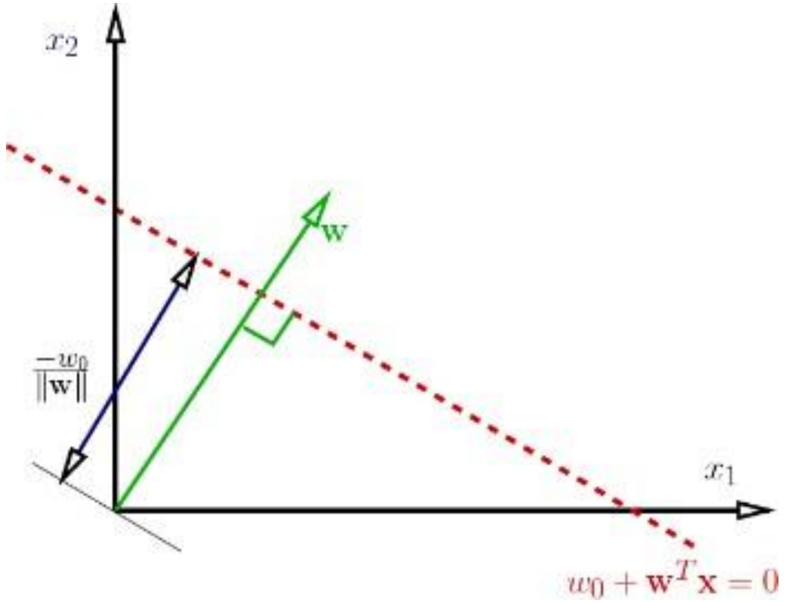


GEOMETRY

- $\boldsymbol{w}^T\boldsymbol{x} = 0$ is a line/hyperplane passing through the origin and is orthogonal to **w**
- The reason that **w** is orthogonal (perpendicular) to the hyperplane is that – the dot product of any 2 vectors can be 0 only if they're orthogonal (90 degrees)
- Dot product review:

$$\boldsymbol{a}.\boldsymbol{b} = a_1b_1 + \cdots + a_nb_n \qquad \boldsymbol{a}.\boldsymbol{b} = |\boldsymbol{a}||\boldsymbol{b}|cos\theta$$

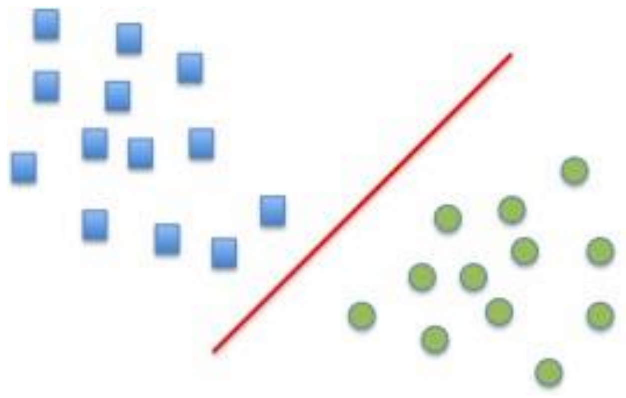
- $\boldsymbol{w}^T\boldsymbol{x} + w_0 = 0$ shifts the hyperplane by w_0

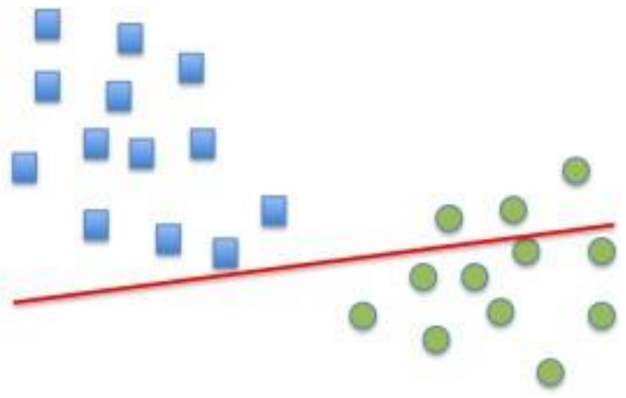


Recall $|\boldsymbol{a}|$ corresponds to the length (magnitude/modulus) of vector a.

LEARNING LINEAR CLASSIFIERS

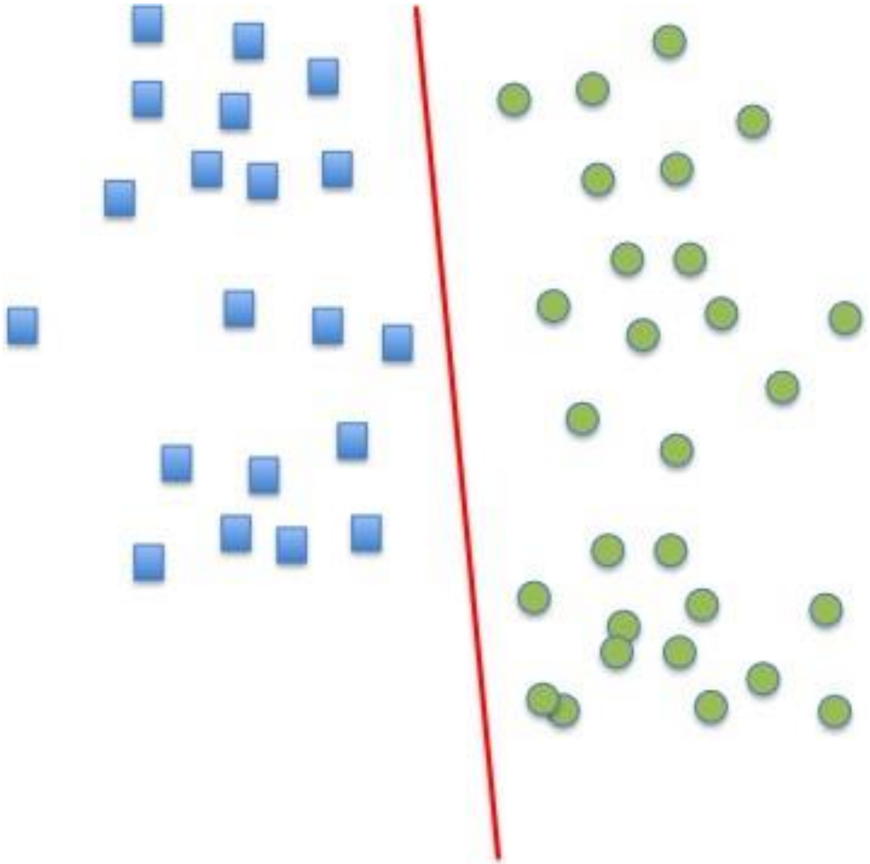
- Learning consists in estimating a "good" decision boundary
- We need to find $\boldsymbol{w}$ (direction) and w_0 (location) of the boundary.
- What does "good" mean?
- Is this boundary good? - We need a criteria that tell us how to select the parameters - use a loss function.





SEPARATING CLASSES

- If we can separate the classes, the problem is **linearly separable**

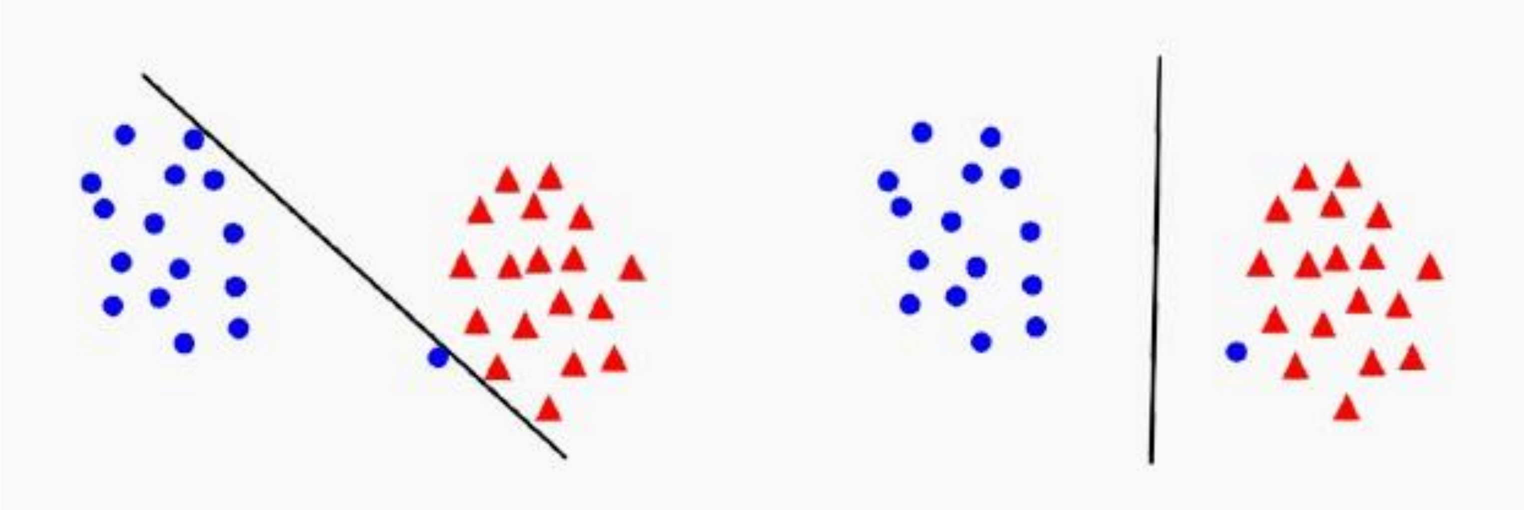


SEPARATING CLASSES

- Causes of non-perfect separation:
  - Model is too simple
  - Noise in the inputs (i.e., data attributes)
  - Simple features that do not account for all variations
  - Errors in data targets (mis-labellings)
- Should we make the model complex enough to have perfect separation in the training data?

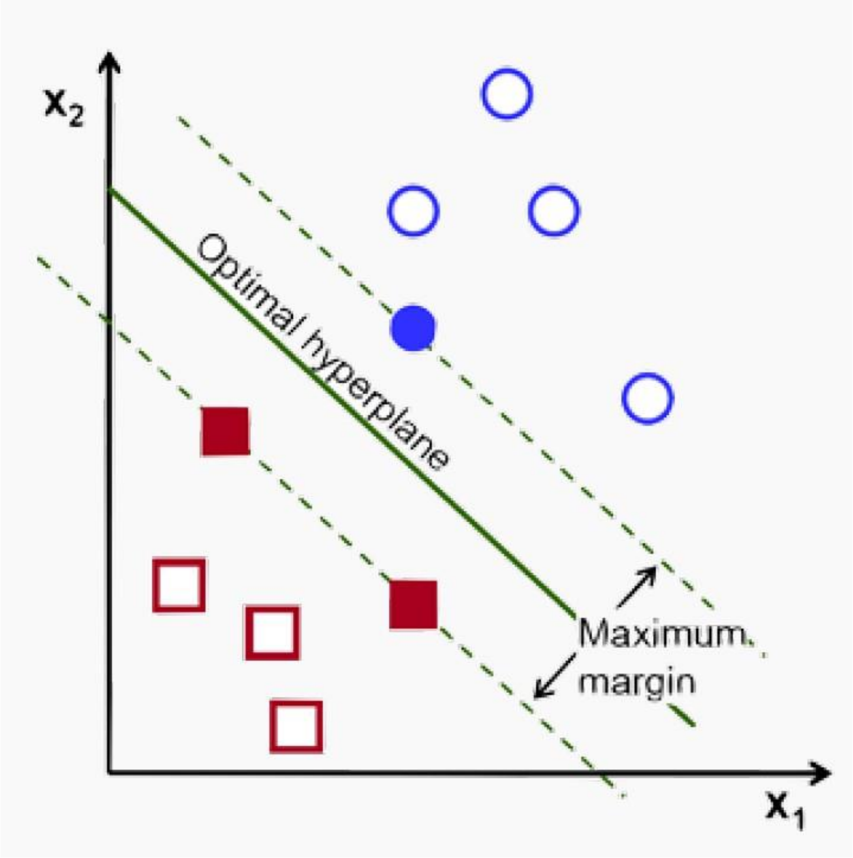
DECISION BOUNDARIES

- For example, we may select a decision boundary that maximises the margin between both classes
  - Geometrically, this means choosing a boundary that maximizes the distance or margin between the boundary and both classes.
  - This is known as a Maximal Margin/Hard Margin Classifier
  - However, what if the data looks like this?
  - Maximal Margin /Hard Margin Classifiers are very sensitive to outliers and are prone to over-fitting
  - We can consider alternative/relaxed constraints that prevent overfitting.



MARGIN

- Definition: The shortest distance between the observations and the hyperplane is called the margin

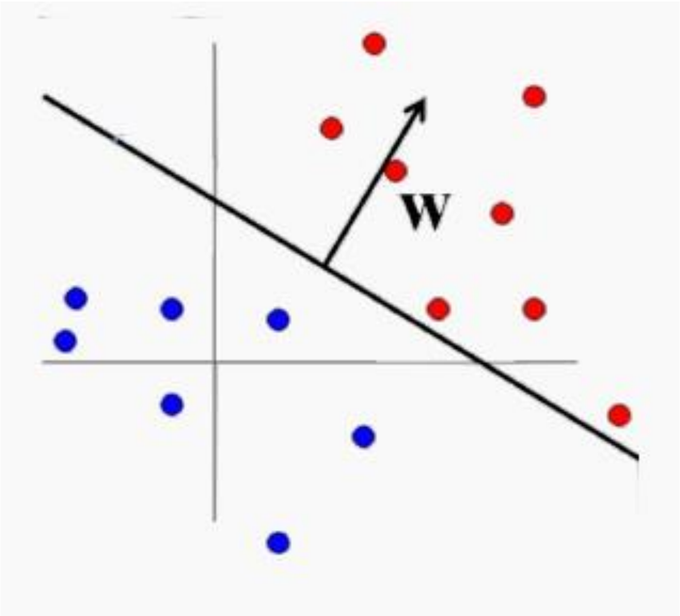


GEOMETRY TO DECISION BOUNDARY

- Recall that the decision boundary is defined by some equation in terms of the predictors. A linear boundary is defined by

$$\boldsymbol{w}^T\boldsymbol{x} + w_0 = 0$$

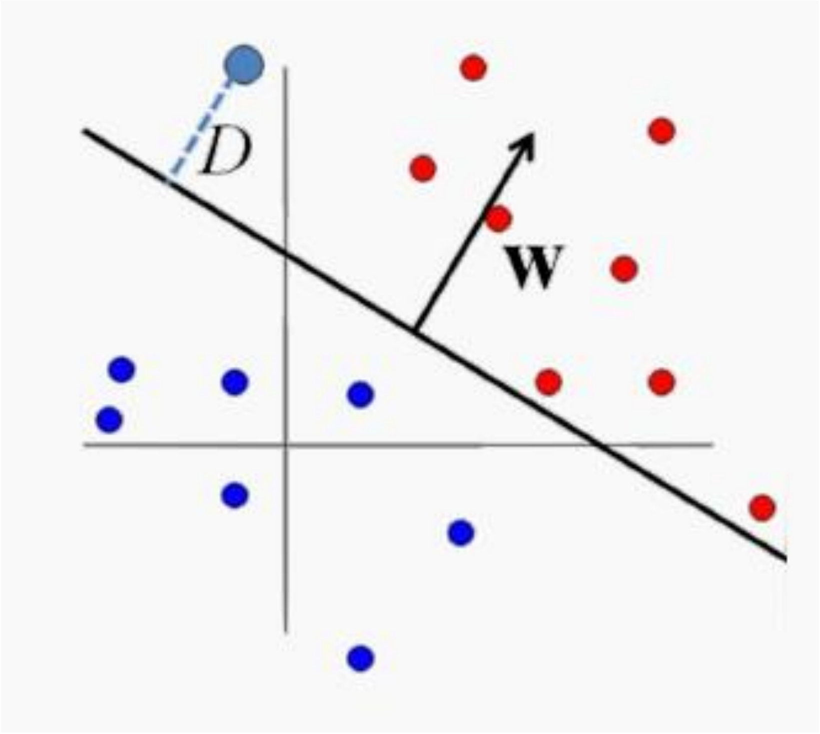
- The non-constant coefficients, **w**, represent a **normal vector**, pointing orthogonally away from the plane



GEOMETRY TO DECISION BOUNDARY

- Now, using some geometry, we can compute the distance between any point to the decision boundary using $\boldsymbol{w}$ and w_0.
- The signed distance from a point $\boldsymbol{x} \in R^n$ to the decision boundary is

$$D(x) = \frac{\boldsymbol{w}^T \boldsymbol{x} + w_0}{||\boldsymbol{w}||}$$



- Note: we need the signed distance because we care which side of the hyperplane the observation is on.
- E.g. in 2D (standard equation for distance from point to line):

$$D(x) = \frac{w_0 + w_1 x_1 + w_2 x_2}{\sqrt{w_1^2 + w_2^2}}$$

MAXIMISING MARGINS

- So our goal. Find a decision boundary that maximises the distance to both classes.
- A hard margin classifier doesn't maximise the distance of all points to the boundary. Instead, it only maximises the distance to the **closest** points.
- The points closest to the decision boundary are called support vectors.
- This means that only support vectors impact position of the hyperplane. Which training samples are used as support vectors is decided by cross-validation
- For any plane, we can always scale the equation

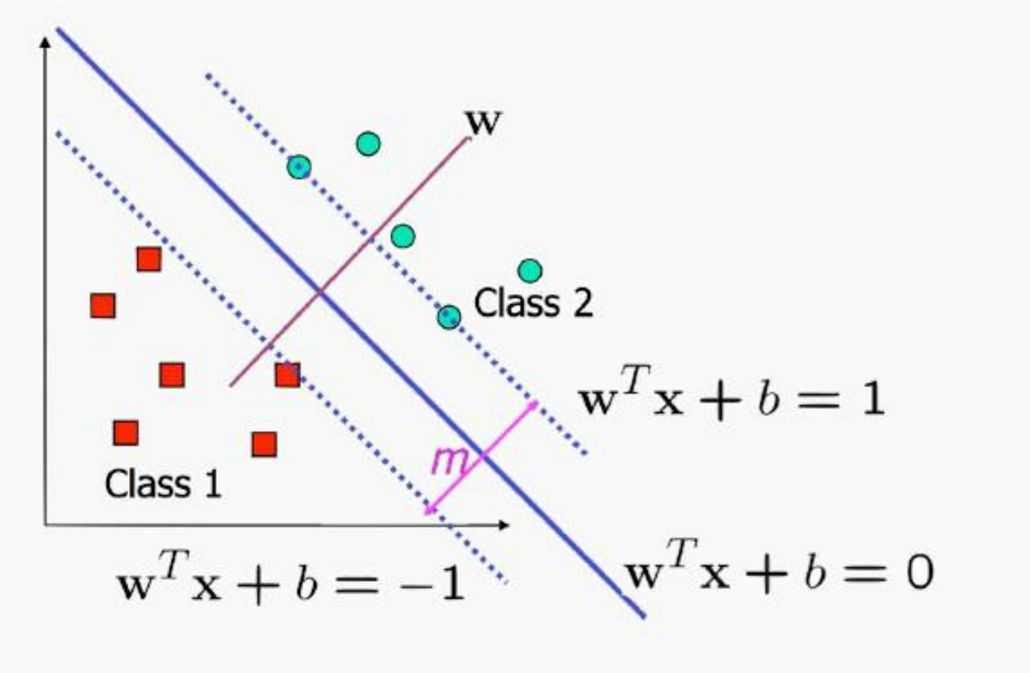
$$\boldsymbol{w}^T\boldsymbol{x} + w_0 = 0$$

  - so that the support vectors lie on the planes (depending on their classes)

$$\boldsymbol{w}^T\boldsymbol{x} + w_0 = \pm 1$$

MAXIMISING MARGINS

- For points on planes $\mathbf{w}^{\mathrm{T}}\mathrm{x} + \mathrm{w}_0 = \pm 1$, their distance to the decision boundary is $\pm \frac{1}{\|w\|}$.
- So we can define the **margin** of a decision boundary as the distance to its support vectors: $\mathrm{m} = \frac{2}{\|w\|}$



SUPPORT VECTOR CLASSIFIER: HARD MARGIN

- Finally, formulate our optimization problem: Find a decision boundary that maximises the distance to both classes – i.e. maximises the margin, M, while maintaining zero misclassifications

$$\begin{cases} max_w \dfrac{2}{\|w\|} \\ such\ that\ y^{(i)}\left(w^T x^{(i)} + w_0\right) \geq 1, \qquad i = 1, \ldots, N \end{cases}$$

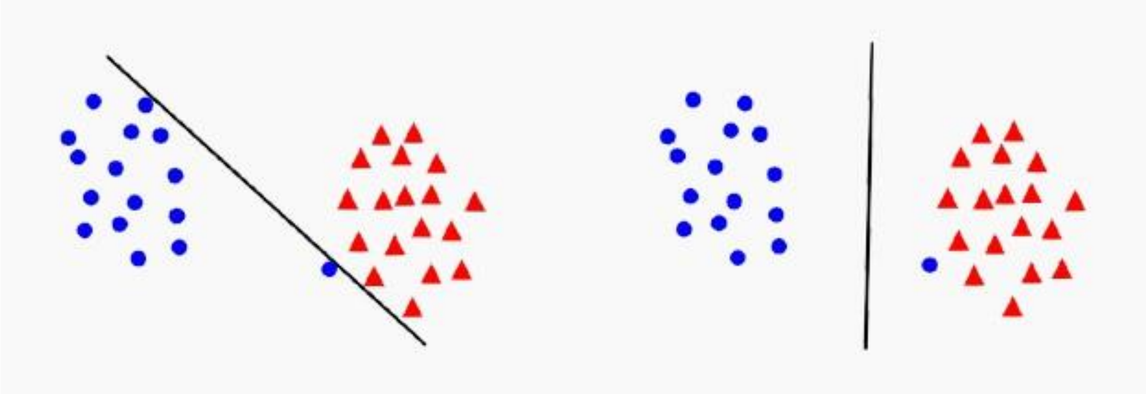
- Maximising $\frac{2}{\|w\|}$ is the same as minimizing $\|w\|$. However L2 optimisations are more stable. Therefore:

$$\begin{cases} min_w \|w\|^2 \\ such\ that\ y^{(i)}\left(w^T x^{(i)} + w_0\right) \geq 1, \qquad i = 1, \ldots, N \end{cases}$$

- This is a quadratic optimisation problem, has linear constraints and there is a unique solution.
- Calculus again! (Lagrange multipliers if you want to look up the maths)

MARGIN ERROR/TRADE OFF

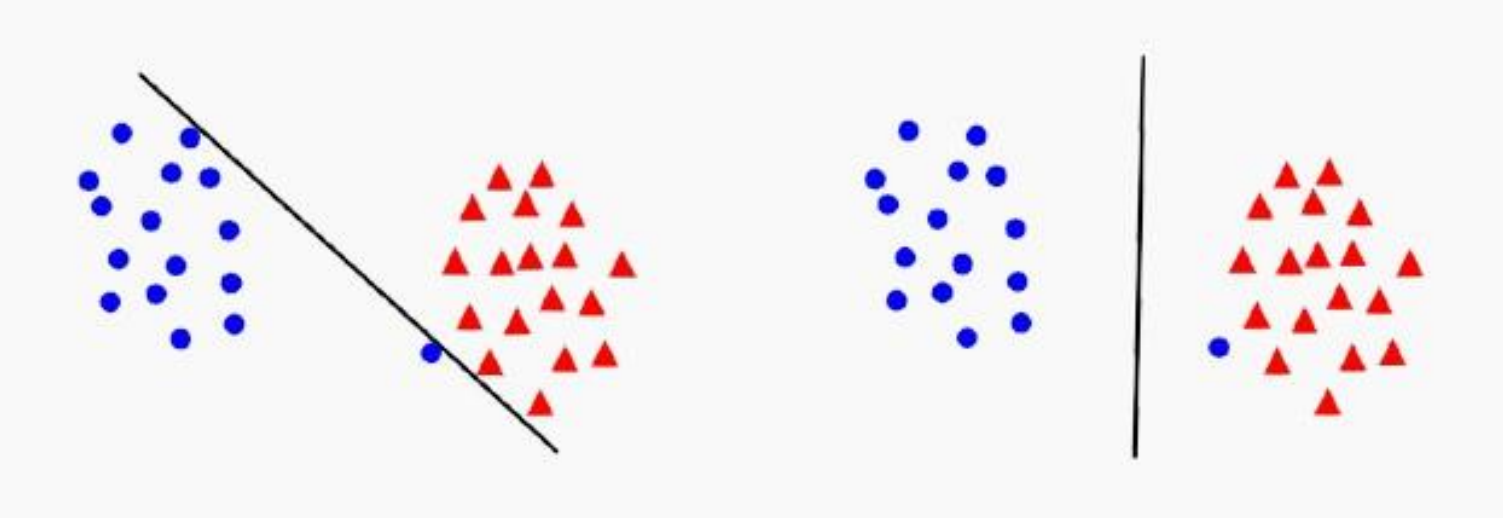
- Which one of these lines is a better generalisation?



- In the first one the points can be linearly separated but there is a very narrow margin
- But possibly the large margin solution is better, even though one constraint is violated (this is known as a soft margin classifier)

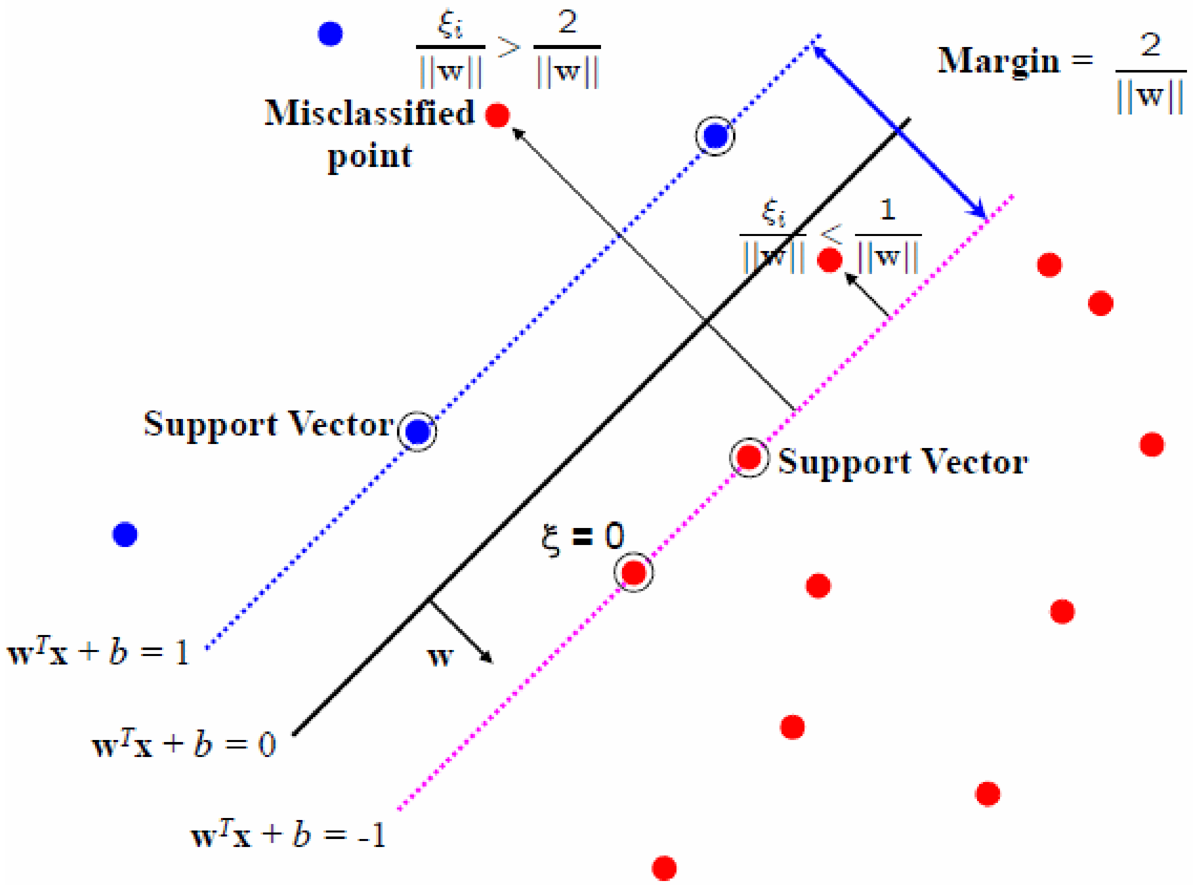
MARGIN ERROR/TRADE OFF

- Maximising the margin is a good idea as long as we know that the underlying classes are linear separable and that the data is noise free.
- If data is noisy, we might be sacrificing generalisation in order to minimise classification error with a very narrow margin
- With every decision boundary, there is a trade-off between maximising margin and minimising the error.



SLACK VARIABLES

- We can add a variable $\xi_i \geq 0$ for each point/sample.
  - For $0 < \xi \leq 1$ point is between margin and correct side of hyperplane. This is called a margin violation
  - For $\xi \geq 1$ point is misclassified
  - For $\xi = 0$ point is the correct side of the margin.



SOFT MARGIN SOLUTION

- To relax the constraints, our problem is re framed as

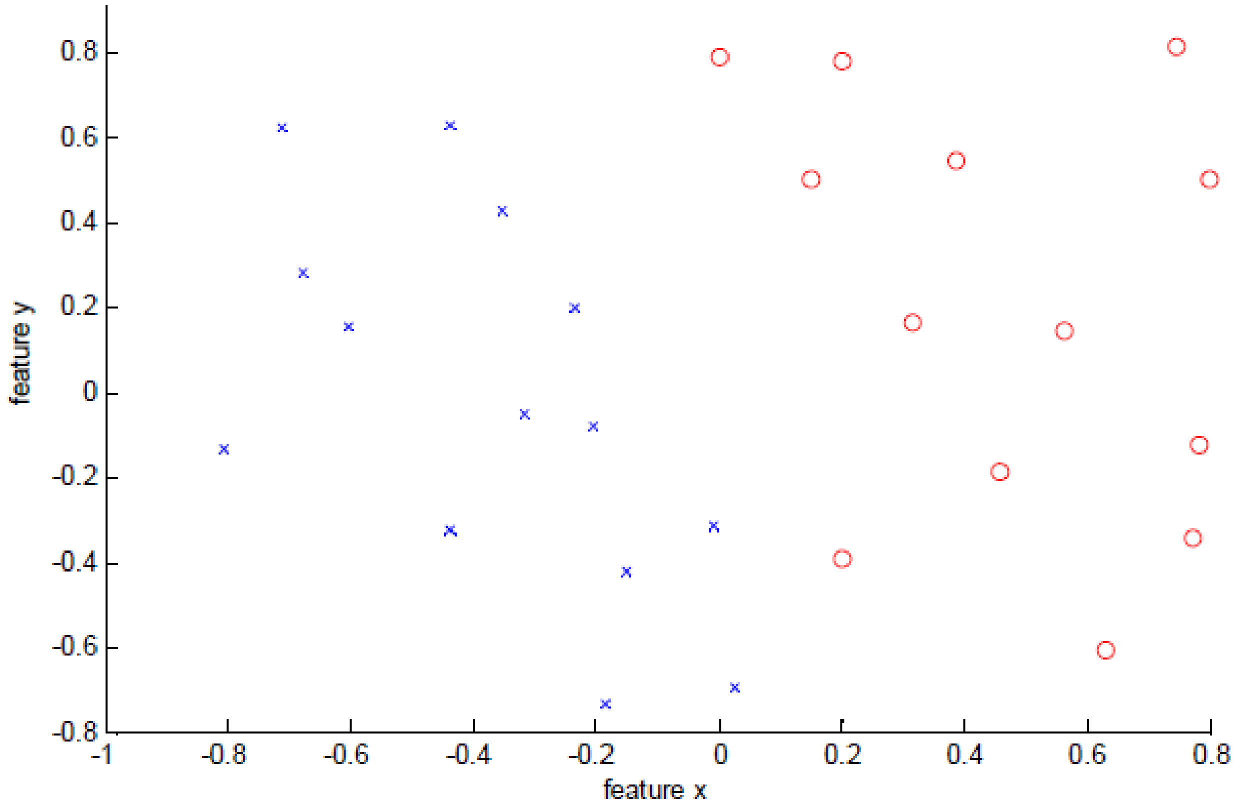
$$\begin{cases} min_w \|w\|^2 + C \sum_{i=1}^{N} \xi_i \\ such\ that\ y^{(i)}\left(w^T x^{(i)} + w_0\right) \geq 1 - \xi_i\ and\ \ \xi_i \geq 0, \qquad \forall i \end{cases}$$

Note: $\forall i$ means for all i

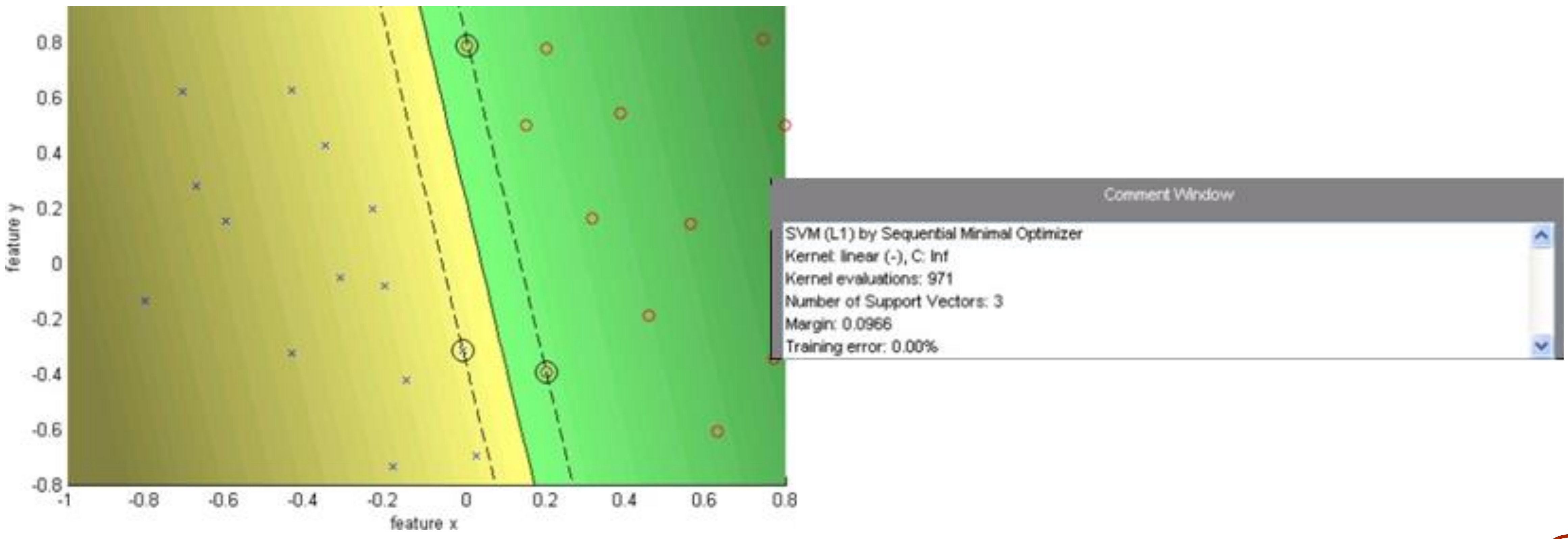
- C is a regularisation parameter: (some notes will use λ instead of C , sklearn uses C)
  - Small C allows constraints to be easily ignored → large margin
  - Large C makes constraints hard to ignore → narrow margin
  - C → ∞ enforces all constraints: hard margin
- This is still a quadratic optimization problem and there is a unique minimum. Note: there is only one parameter, C (that you choose/cross-validation).
- In general, the best C parameter depends on the situation. Experiment (Cross-Validation). One note: larger C takes more computation to train.

EXAMPLE:

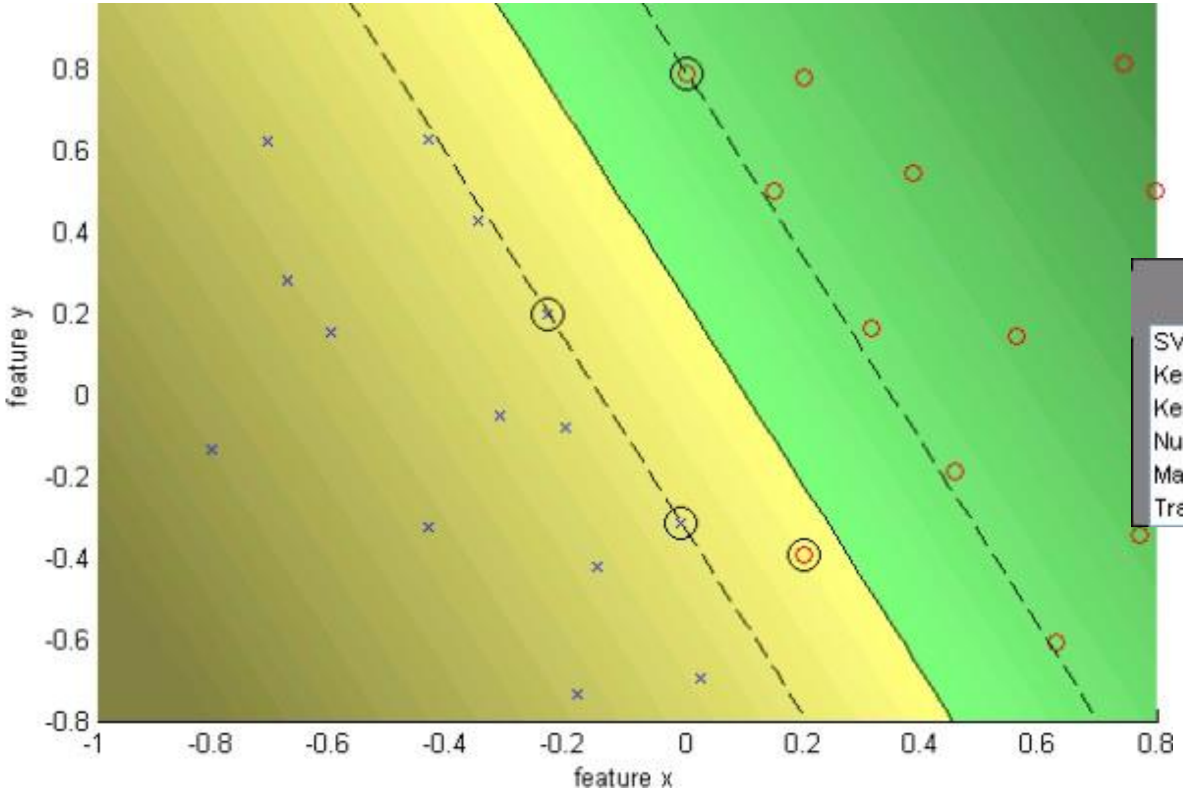
- Data is linearly separable - but only with a narrow margin



INFINITY C - HARD MARGIN



C = 10 - SOFT MARGIN



Comment Window

SVM (L1) by Sequential Minimal Optimizer
Kernel: linear (-), C: 10.0000
Kernel evaluations: 2645
Number of Support Vectors: 4
Margin: 0.2265
Training error: 3.70%

PREVIOUS PROBLEM - BREAST CANCER DATA SET

- SVM classifiers often do better on the ”hard” problems. If we return to the breast cancer data set:

```
model = make_pipeline(
    StandardScaler(),
    SVC(kernel='linear', C=2.0)
)
model.fit(X_train, y_train)
print(model.score(X_test, y_test))

0.993006993006993
```

- That’s better than either GaussianNB or KNeighborsClassifier.