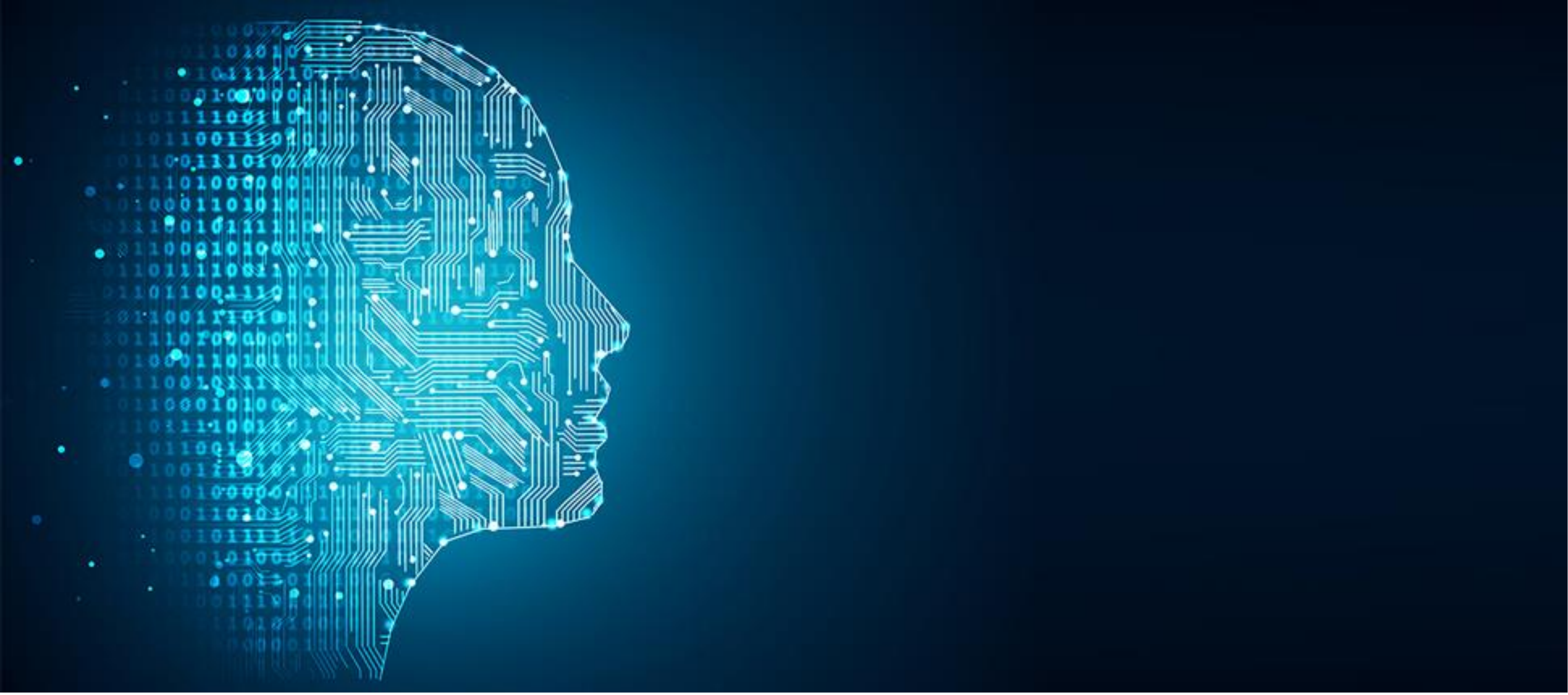


COMP SCI 1400
AI Technologies

Dr. Kamal Mammadov

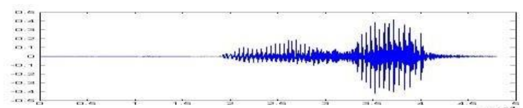
Outline

- Classification
  - ➢ Support Vector Machine

- Machine Learning
  - ✓ supervised learning
  - ✓ unsupervised learning
  - ➢ semi-supervised learning
  - ➢ reinforcement learning

Machine Learning --- Finding Functions

- Speech Recognition

$f($  $)=$ "How are you"

- Image Recognition

$f($  $)=$ "Cat"

- Playing Go

$f($  $)=$ "5-5" (next move)

- Dialogue System

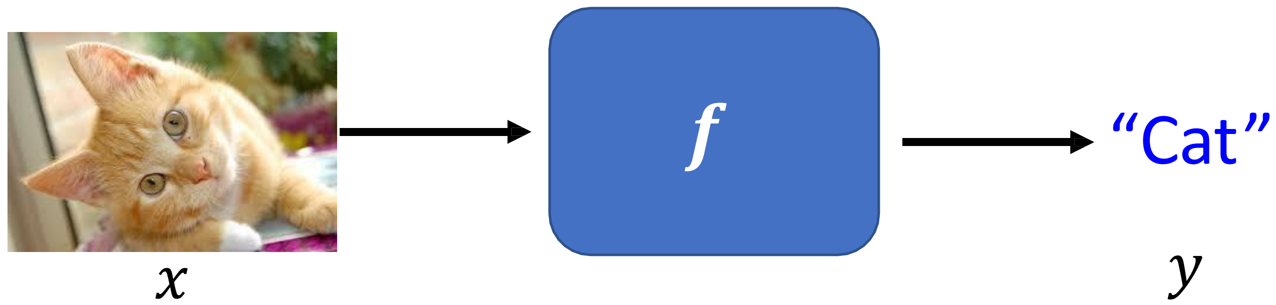
$f($ "How are you?" $)=$ "I am fine."

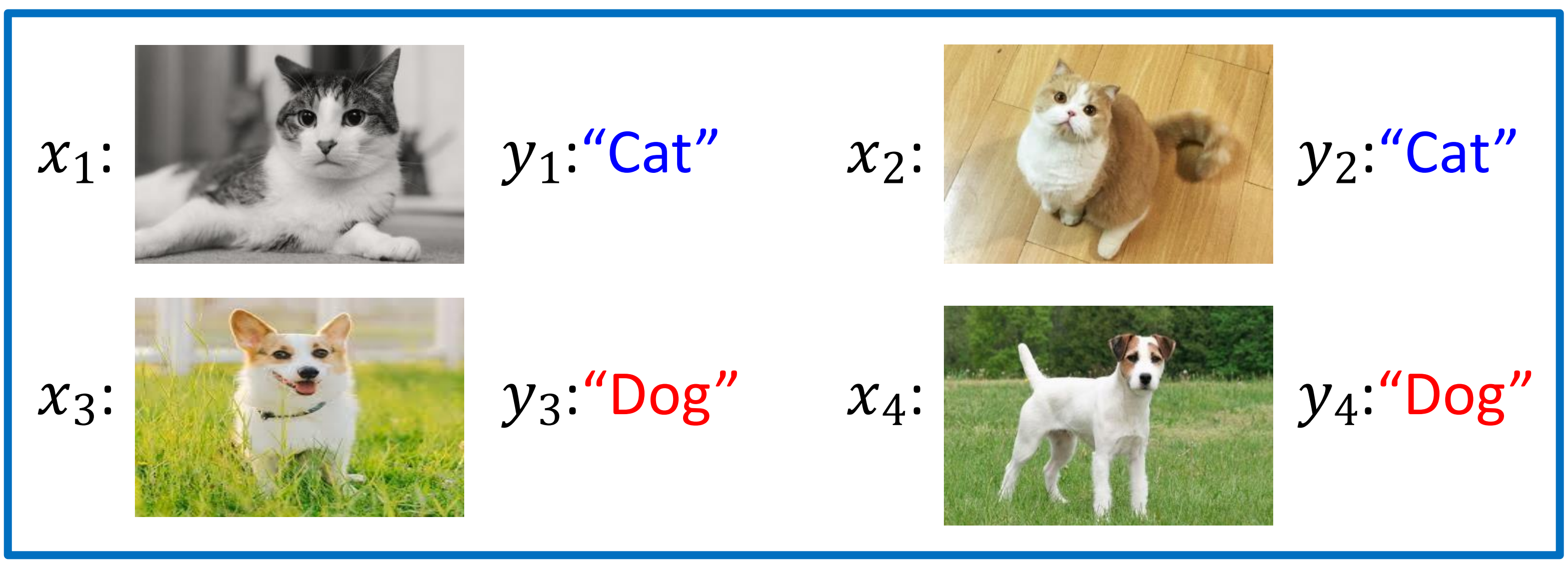
(what the user said) (system response)

Supervised Learning

Unsupervised Learning

Supervised Learning

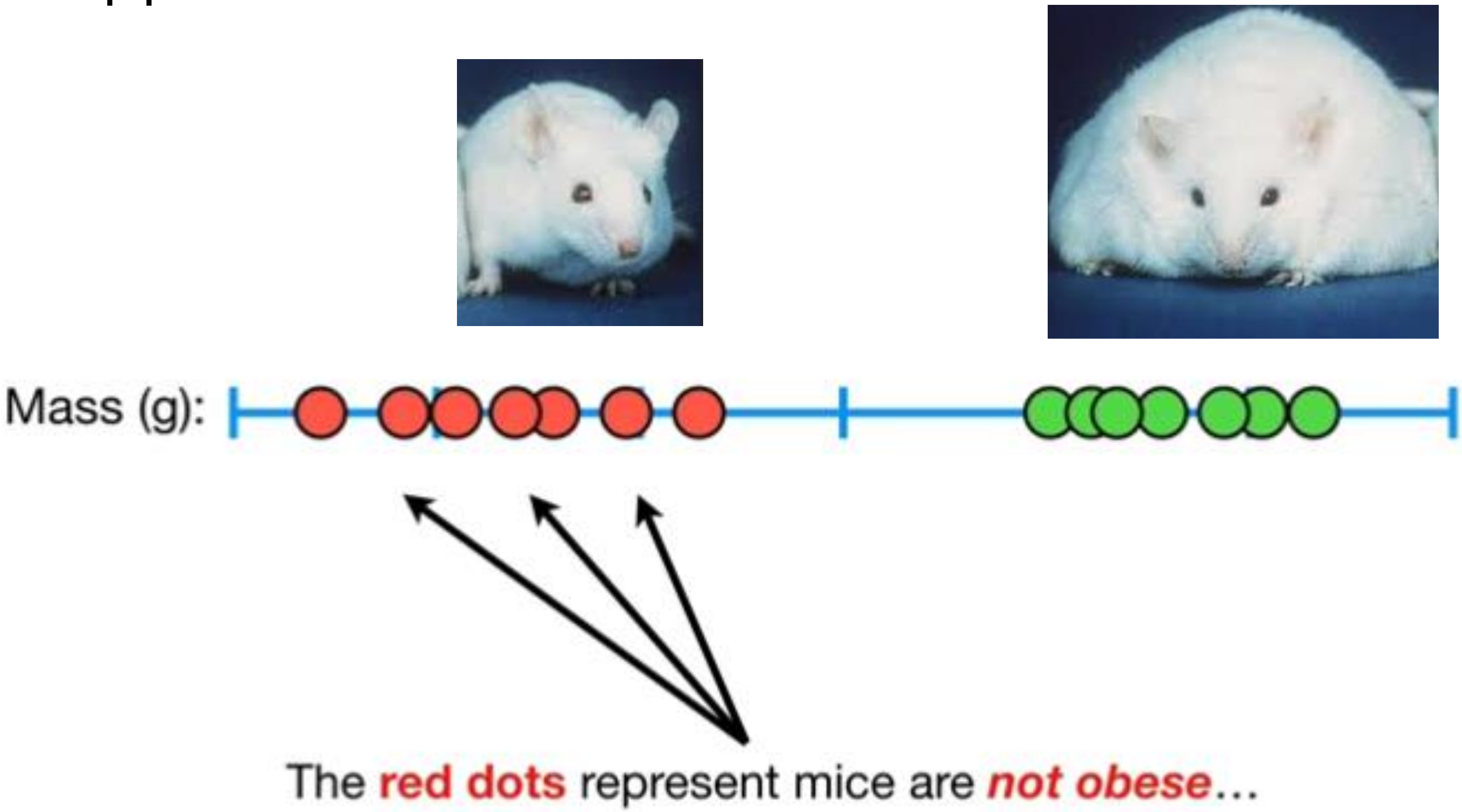




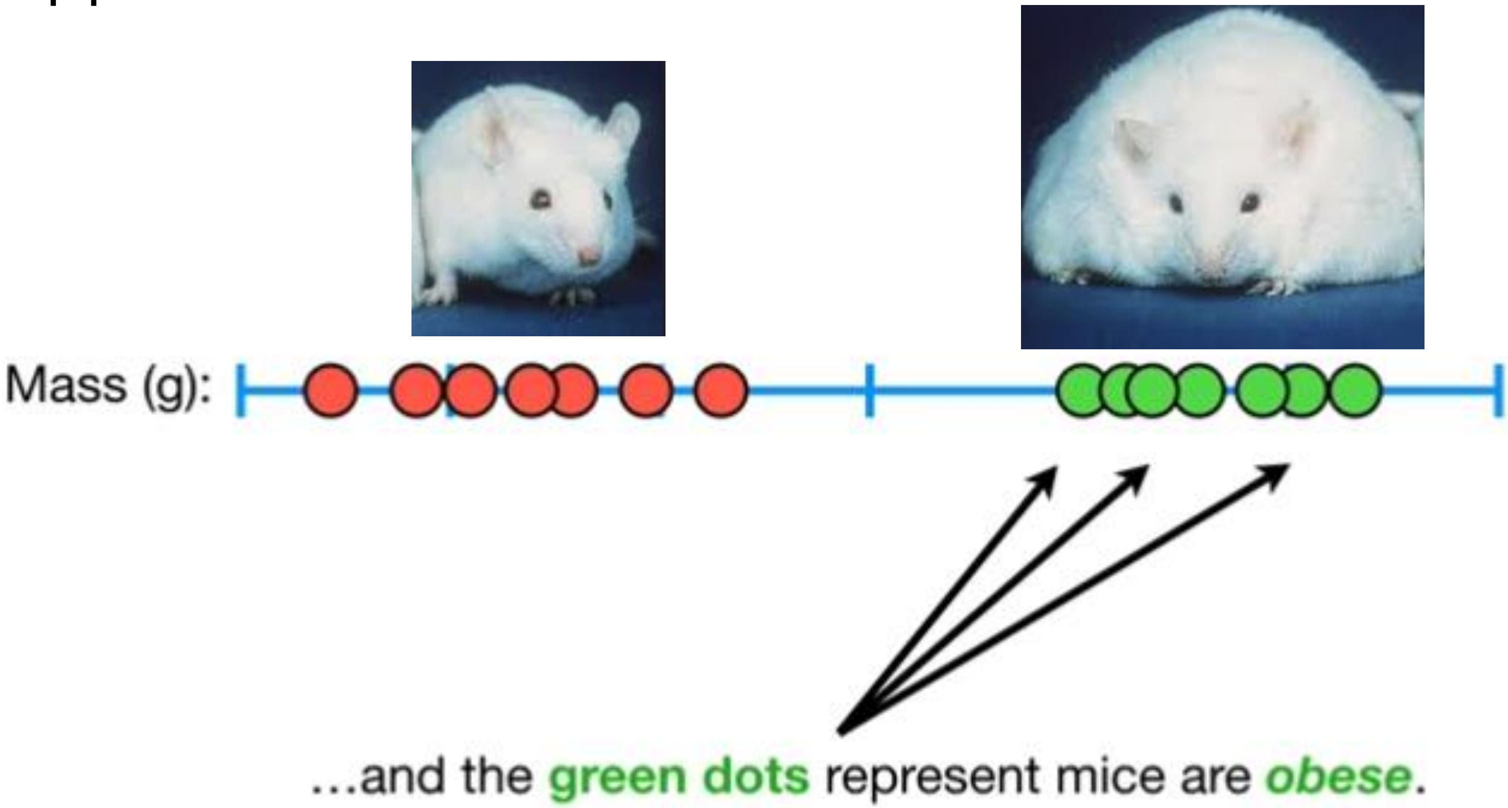
Labelled Data

Support Vector Machine

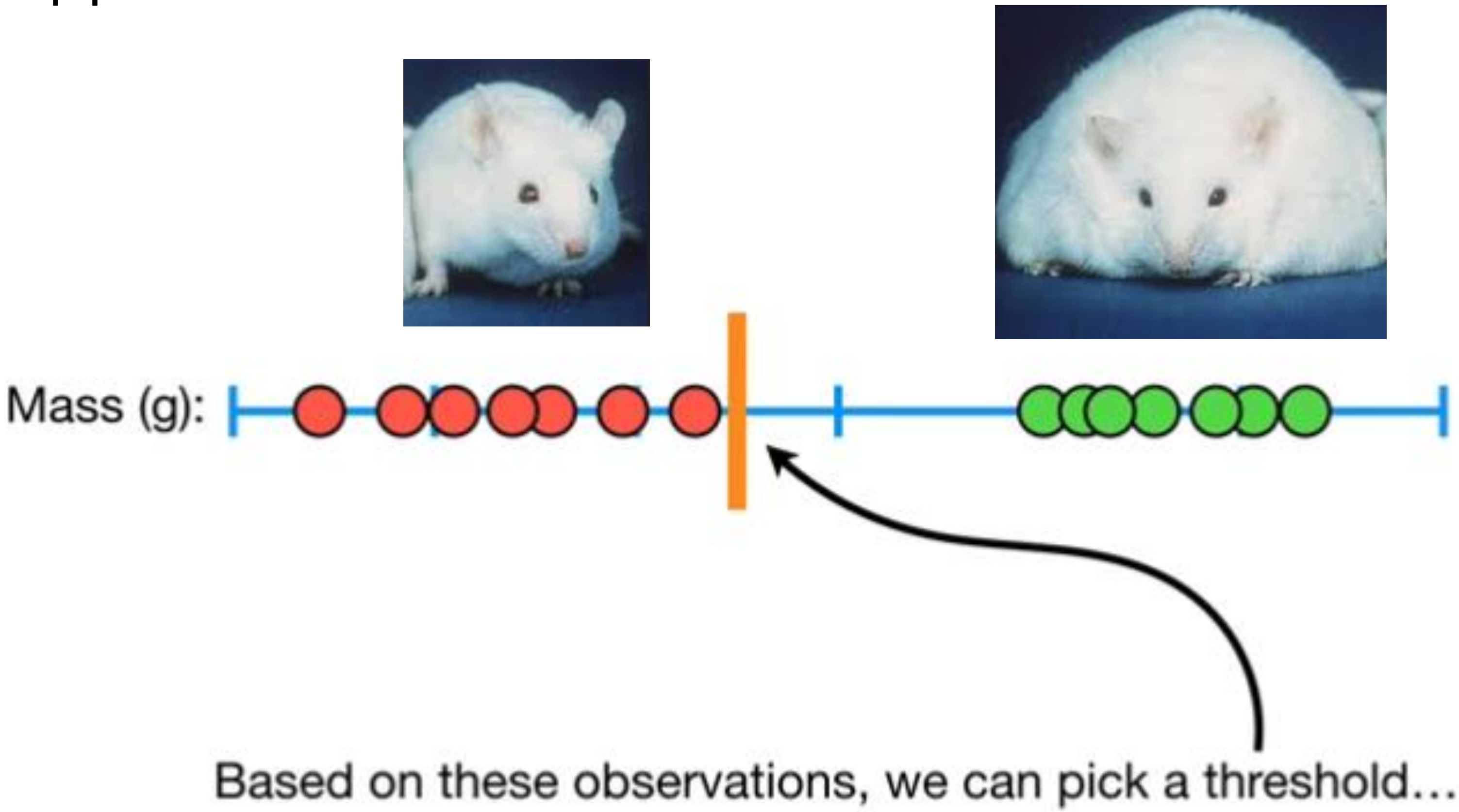
- Support Vector Machine --- 1D



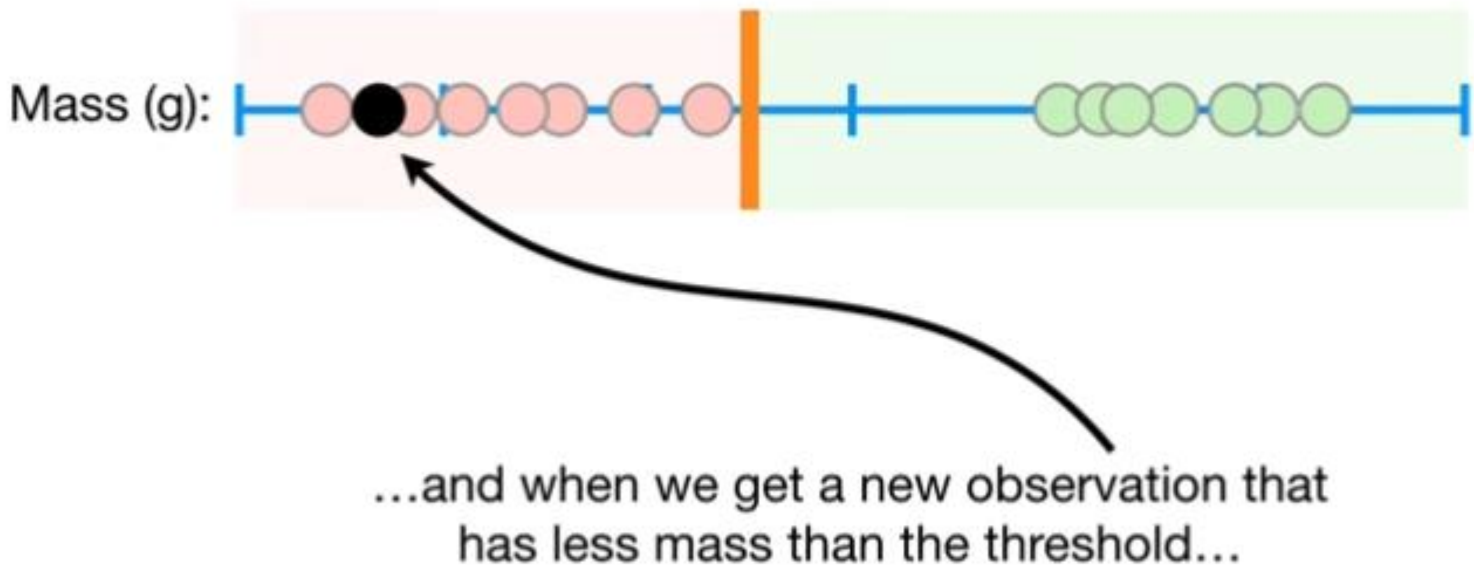
- Support Vector Machine --- 1D



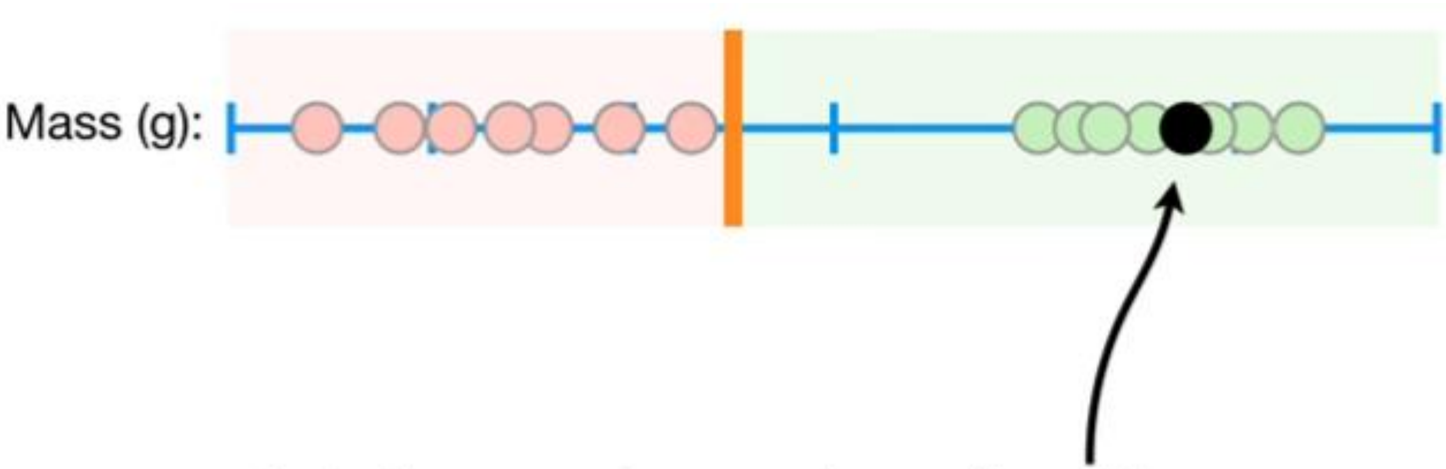
- Support Vector Machine --- 1D



- Support Vector Machine --- 1D

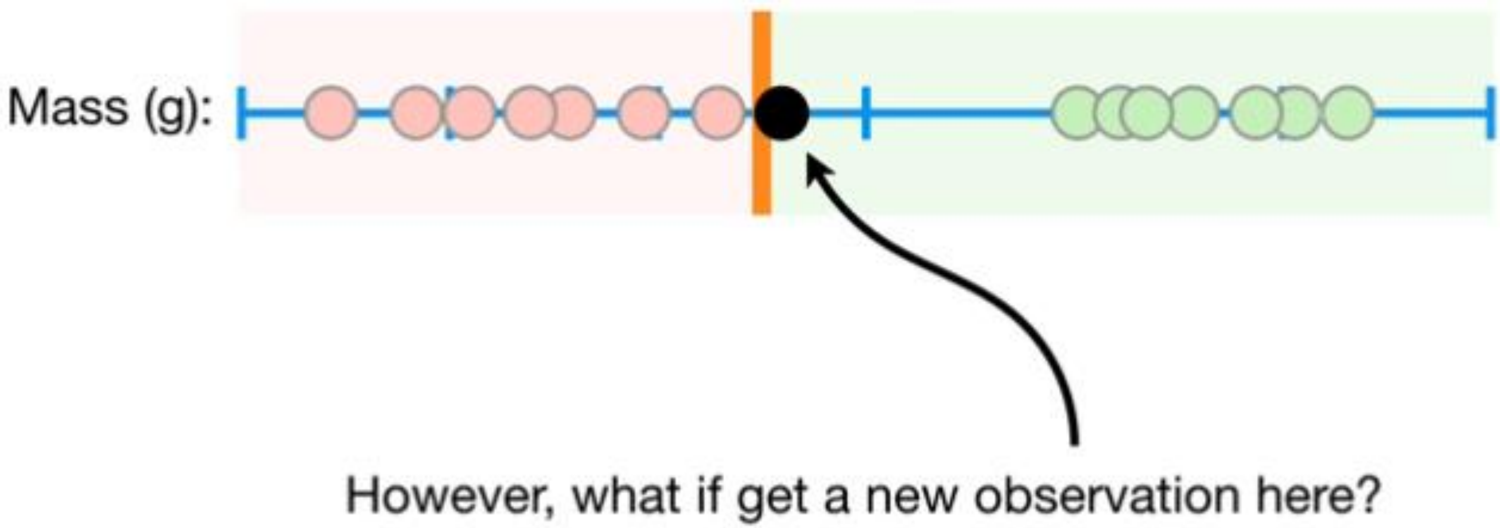


- Support Vector Machine --- 1D

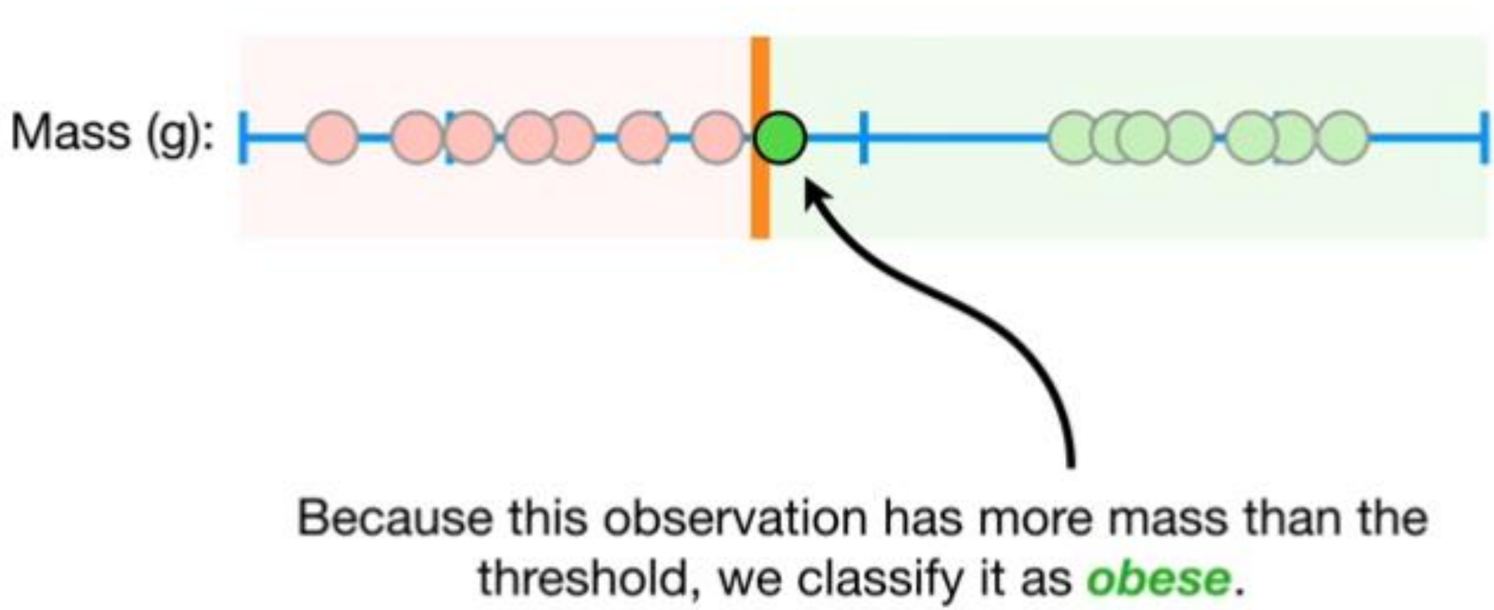


And when we get a new observation with more mass than the threshold…

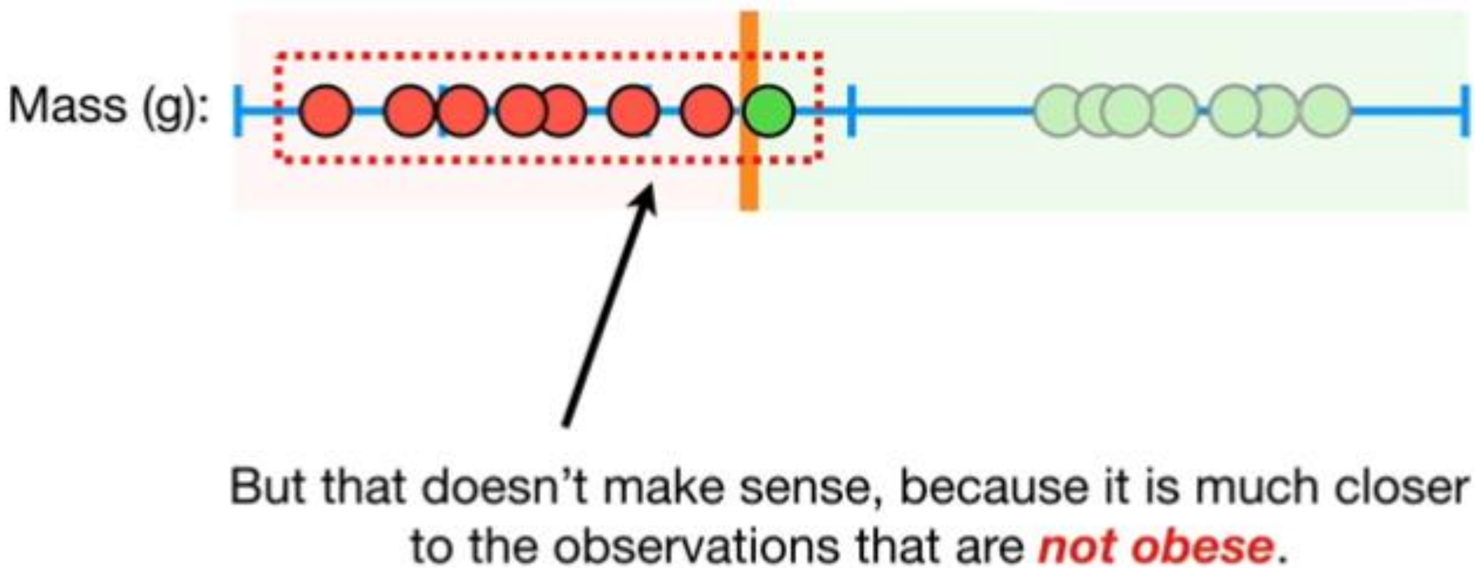
- Support Vector Machine --- 1D



- Support Vector Machine --- 1D

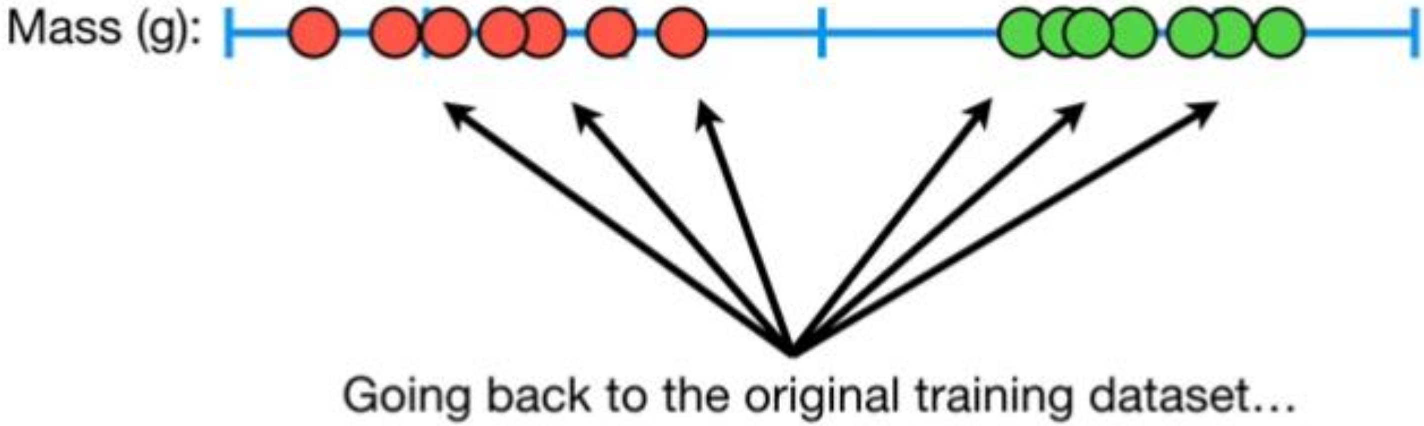


- Support Vector Machine --- 1D

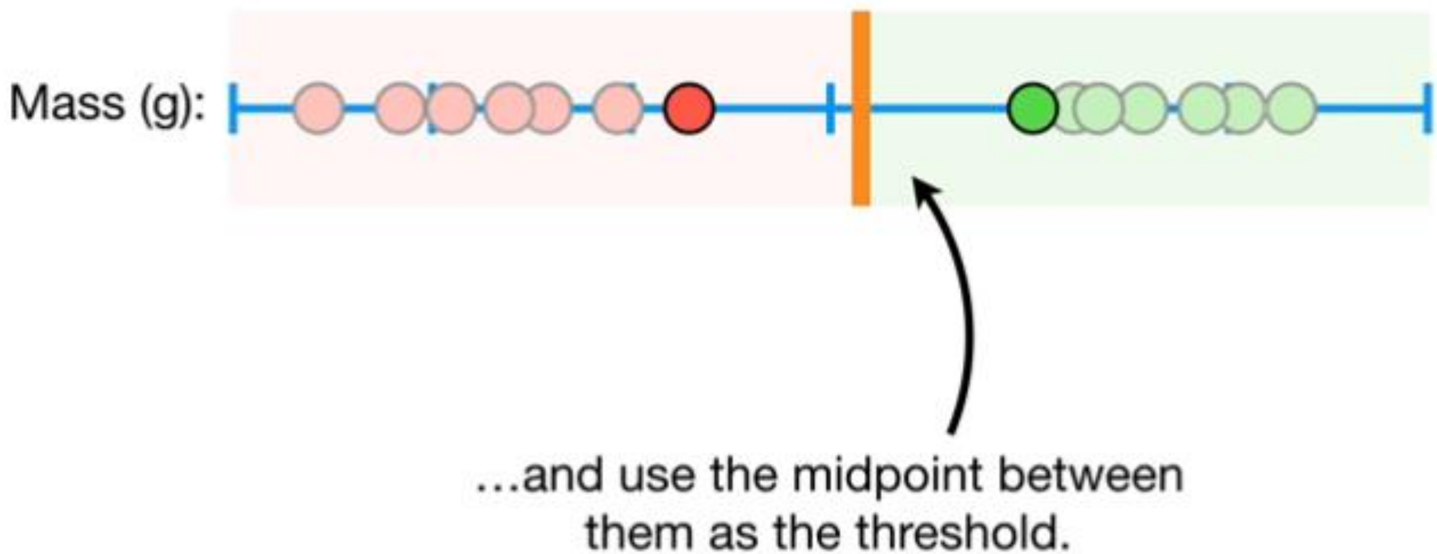


- Support Vector Machine --- 1D

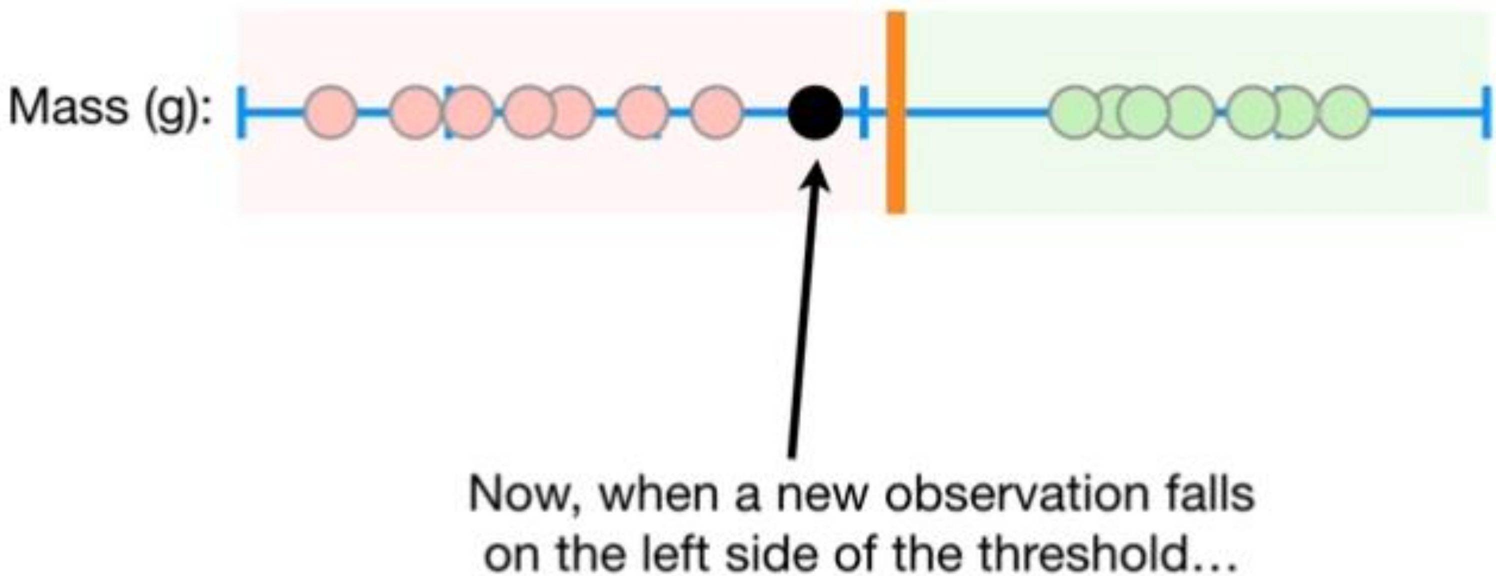
Find a new threshold



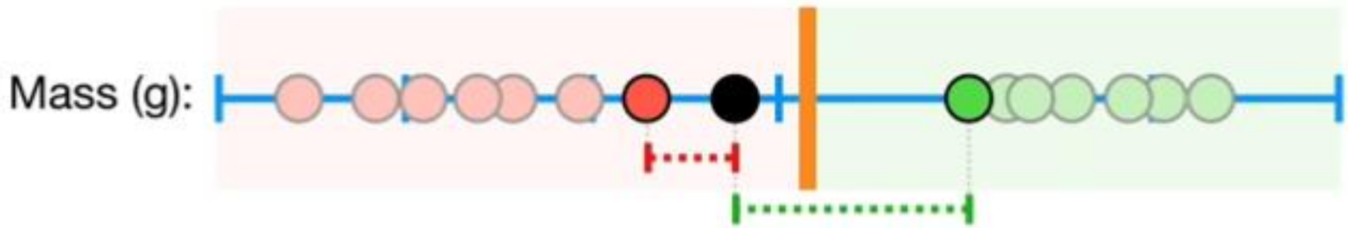
- Support Vector Machine --- 1D



- Support Vector Machine --- 1D

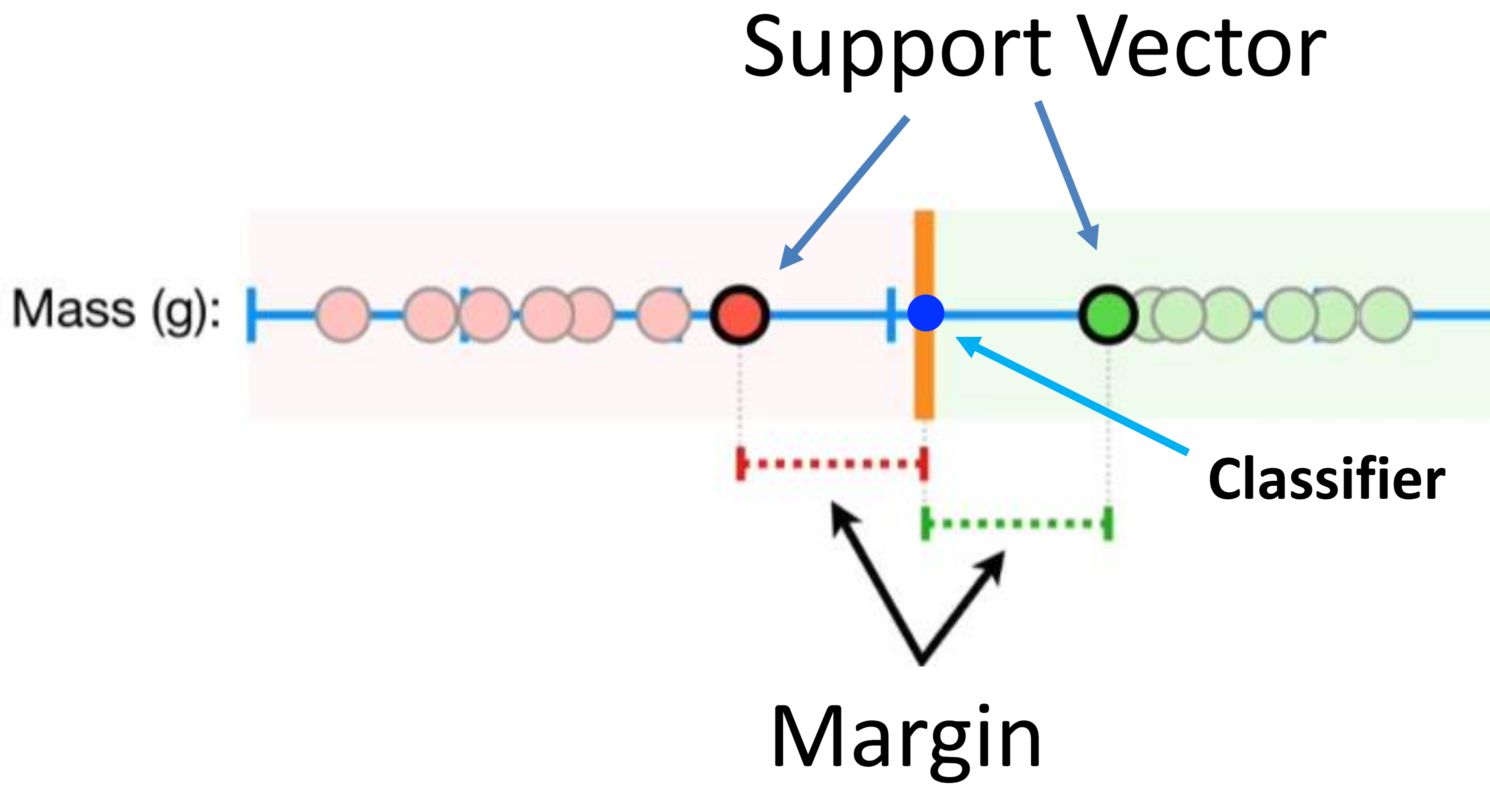


- Support Vector Machine --- 1D

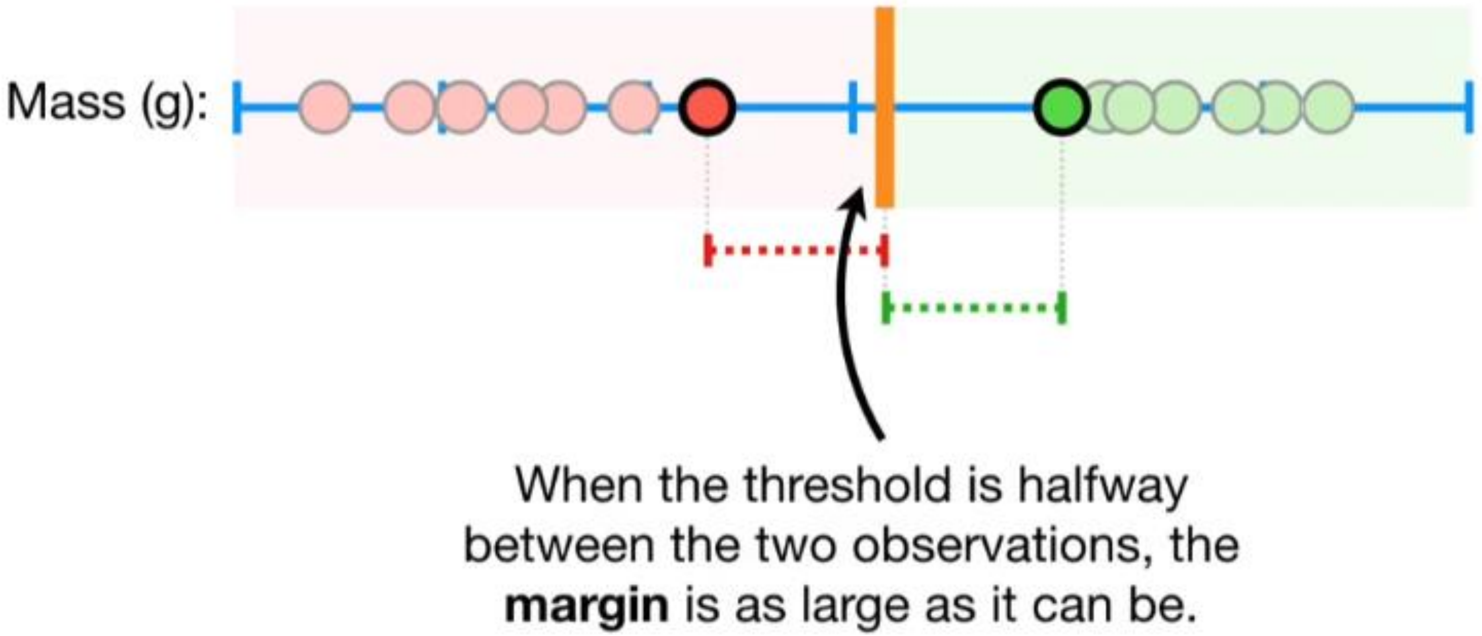


So it makes sense to classify this new observation as ***not obese***.

- Support Vector Machine --- 1D



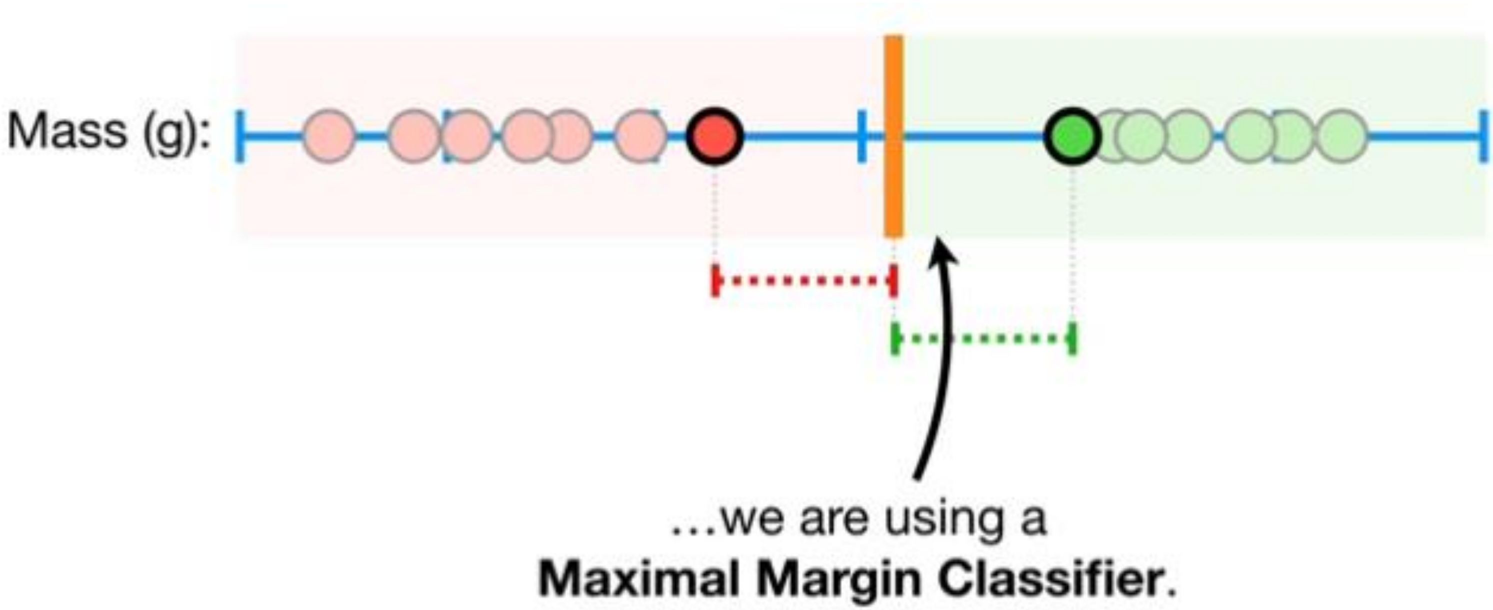
- Support Vector Machine --- 1D



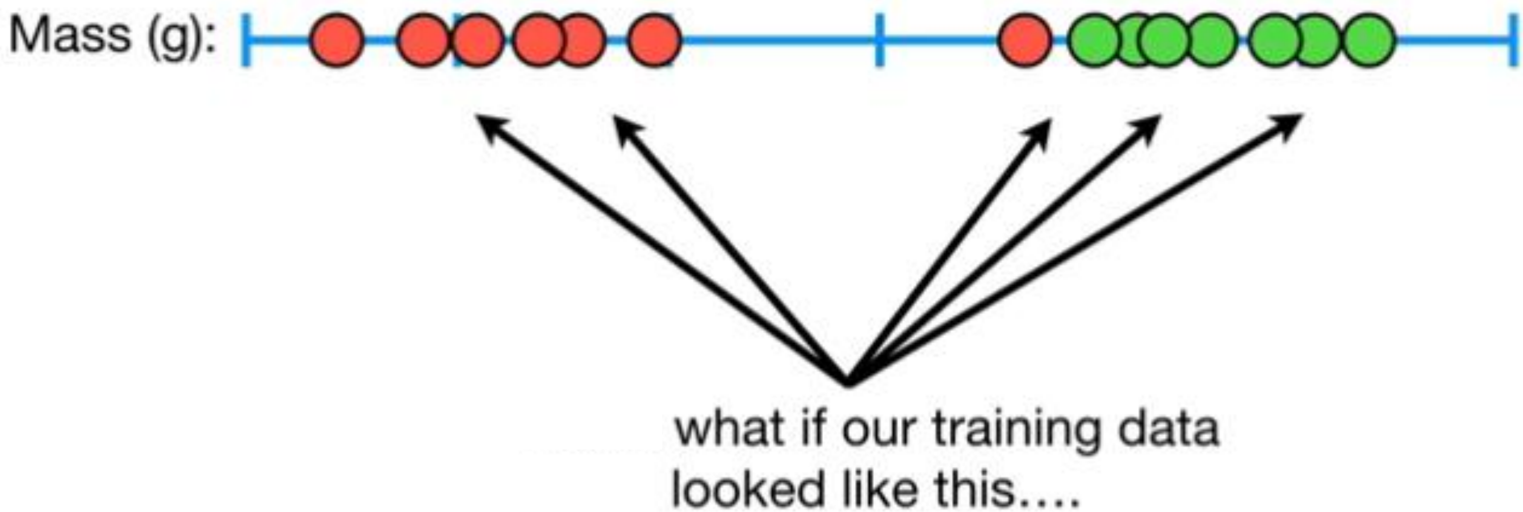
We <u>maximize the margin</u> to both classes to find the best classifier

Support vector determines the classifier

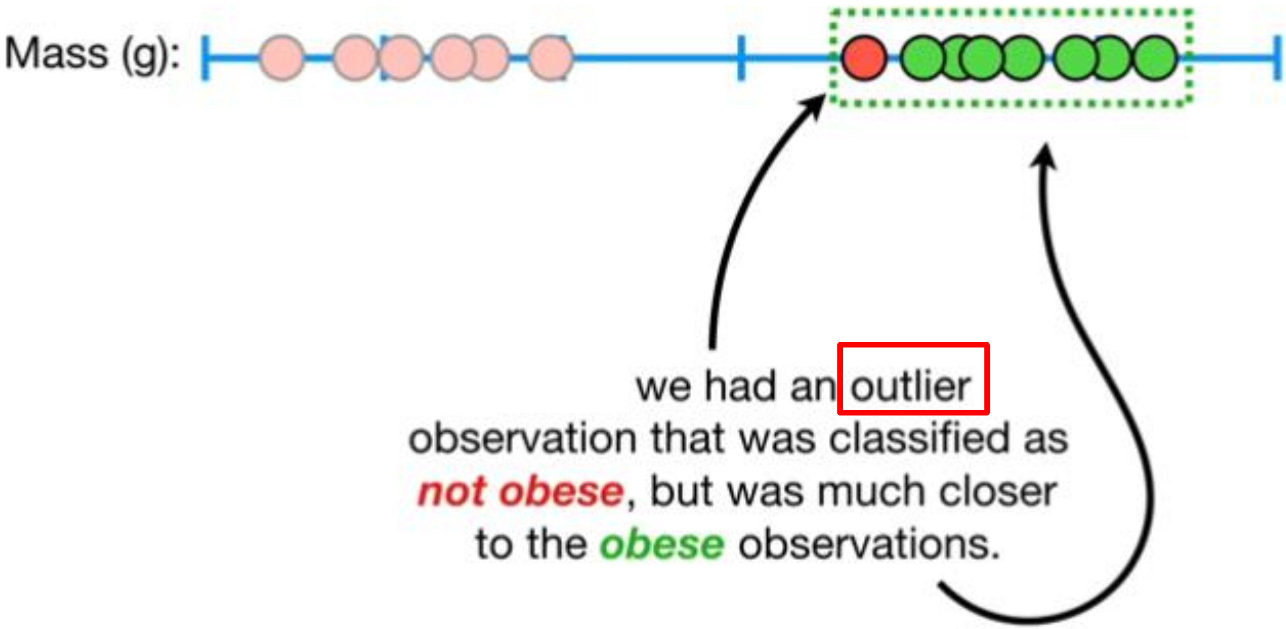
- Support Vector Machine --- 1D



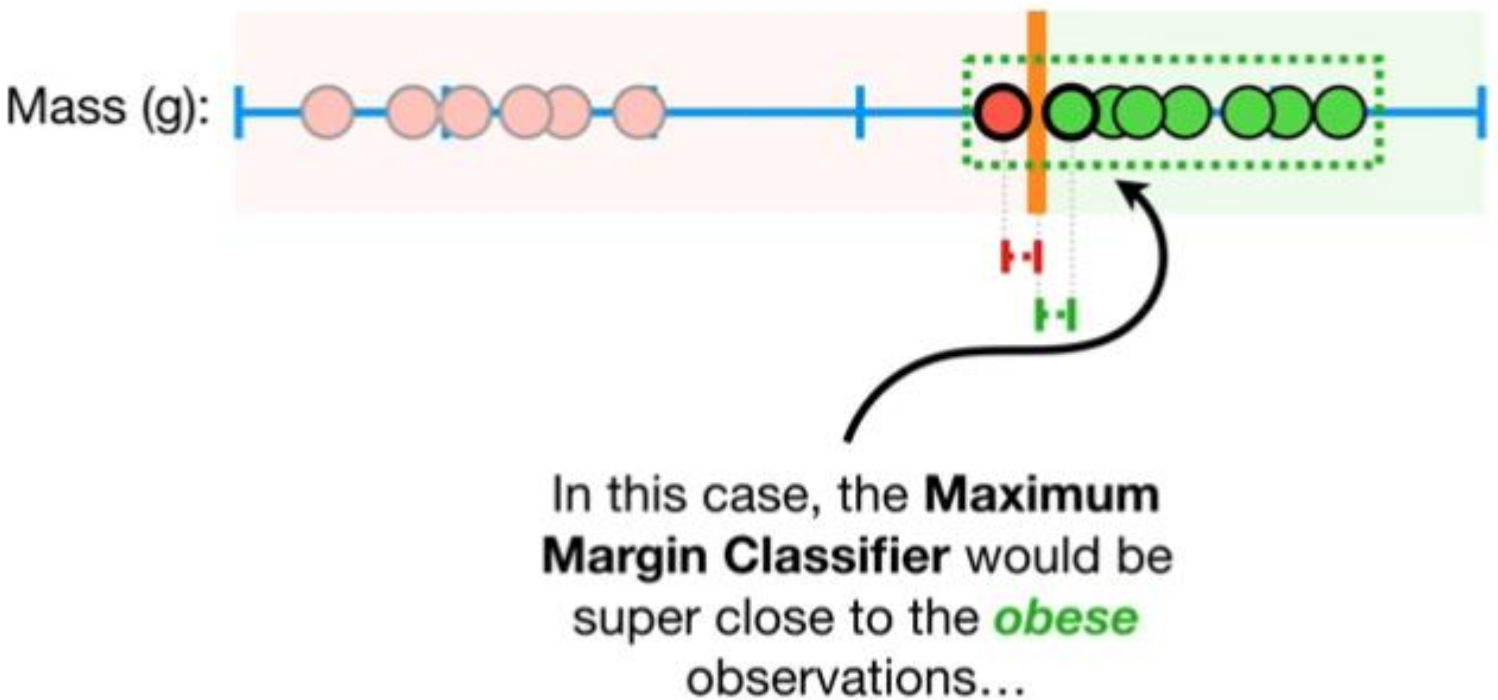
- Support Vector Machine --- 1D



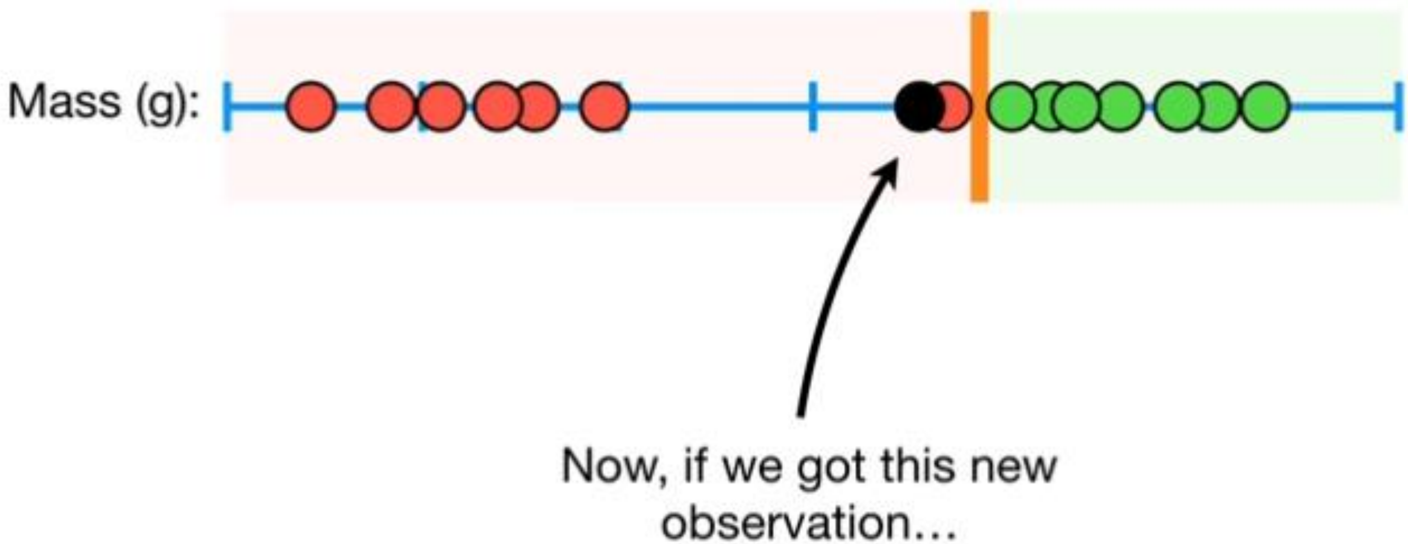
- Support Vector Machine --- 1D



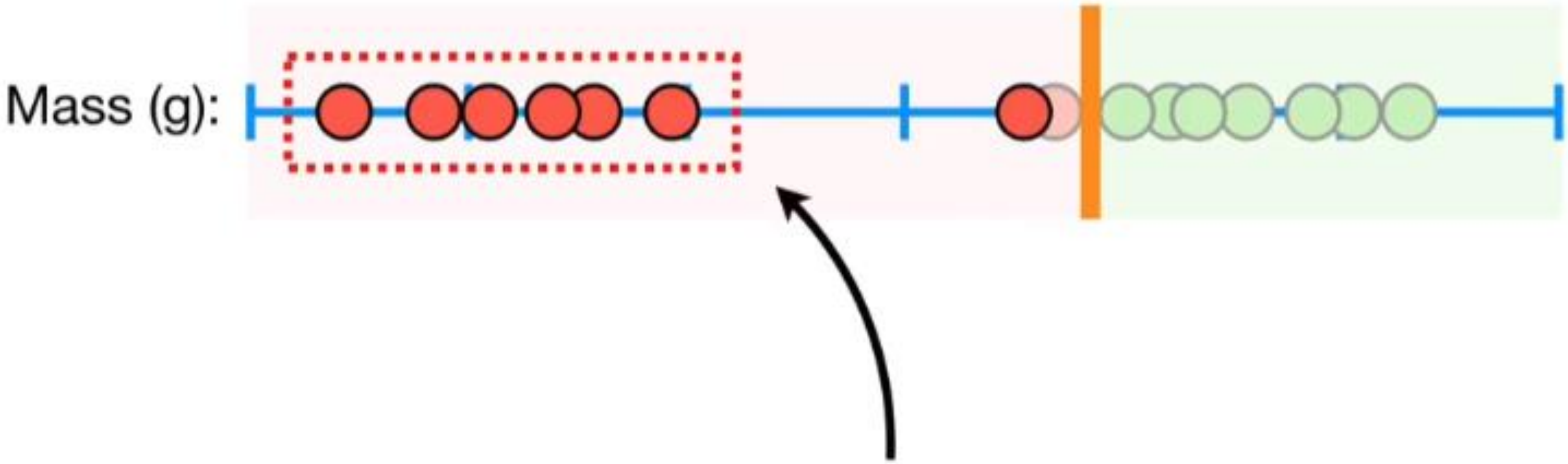
- Support Vector Machine --- 1D



- Support Vector Machine --- 1D

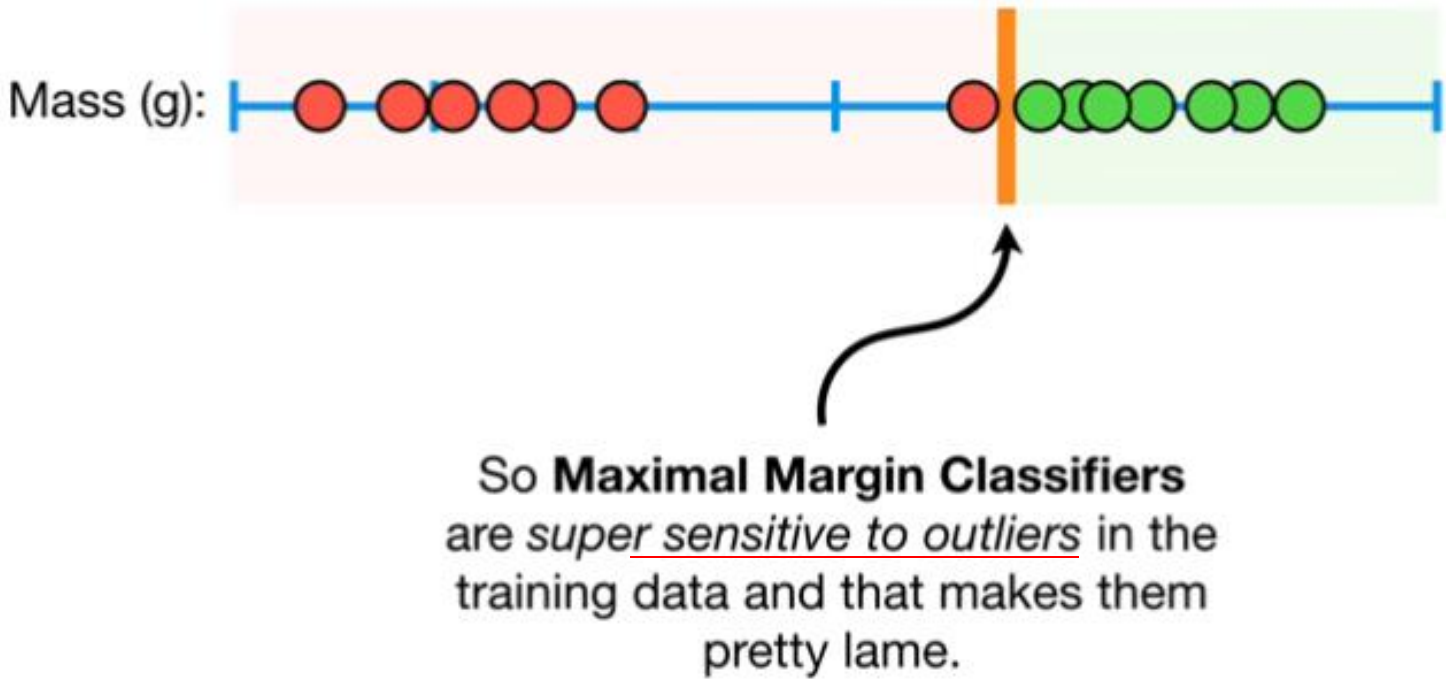


- Support Vector Machine --- 1D



…we would classify it as ***not obese***, even though most of the ***not obese*** observations are much further away than the ***obese*** observations.

- Support Vector Machine --- 1D

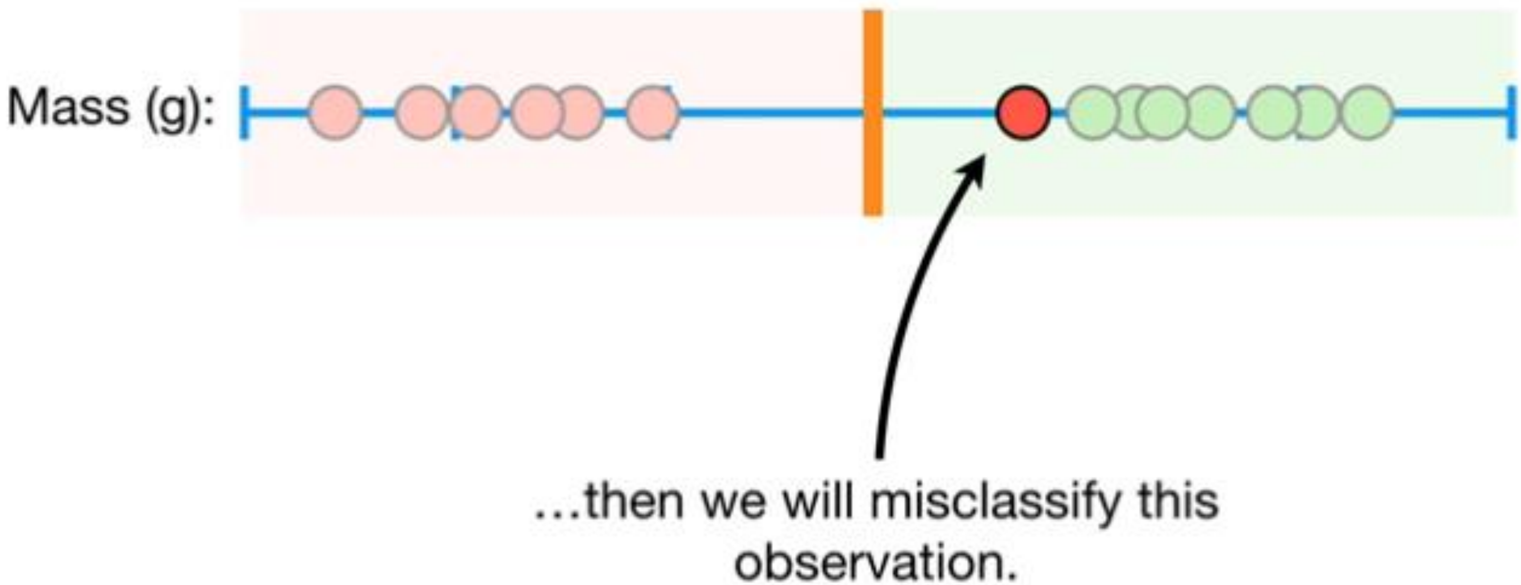


- Support Vector Machine --- 1D

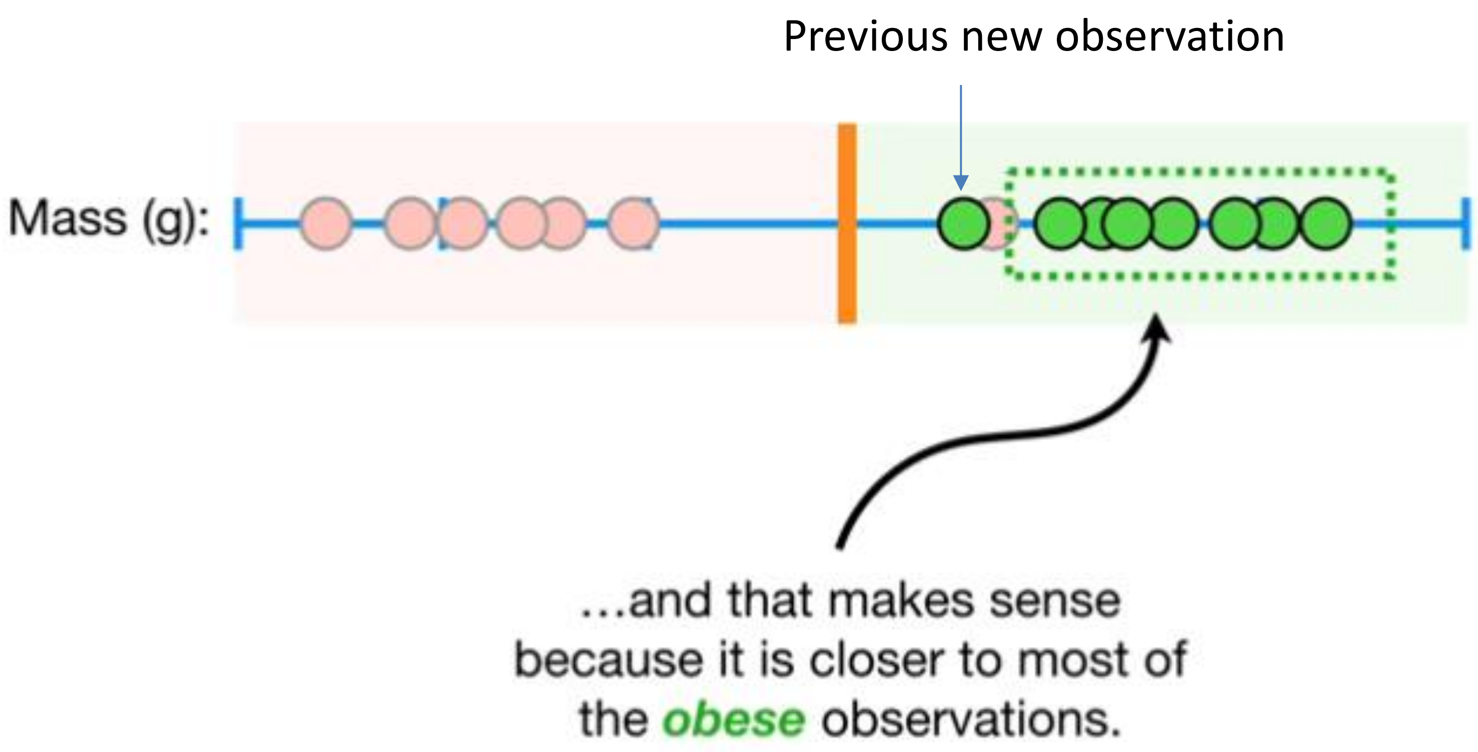
Solution

To make a threshold that is not so sensitive to outliers we must **allow misclassifications**.

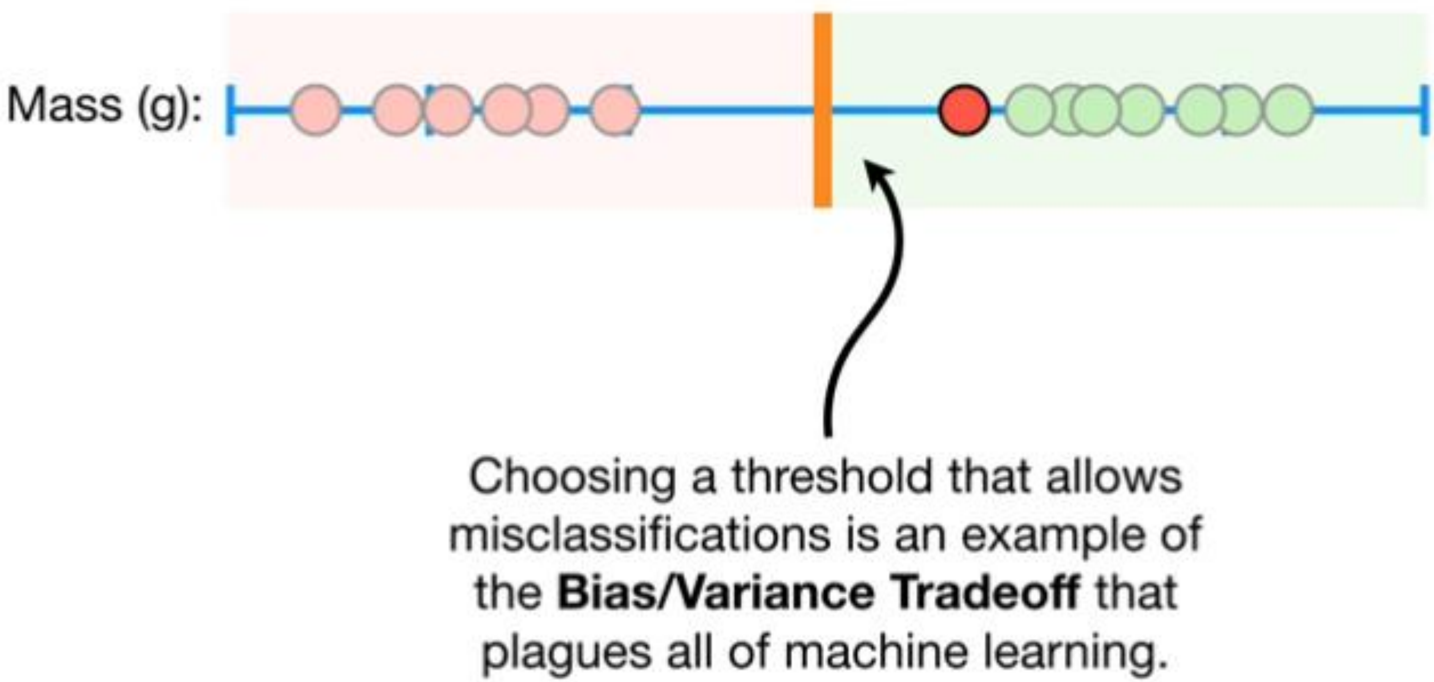
- Support Vector Machine --- 1D



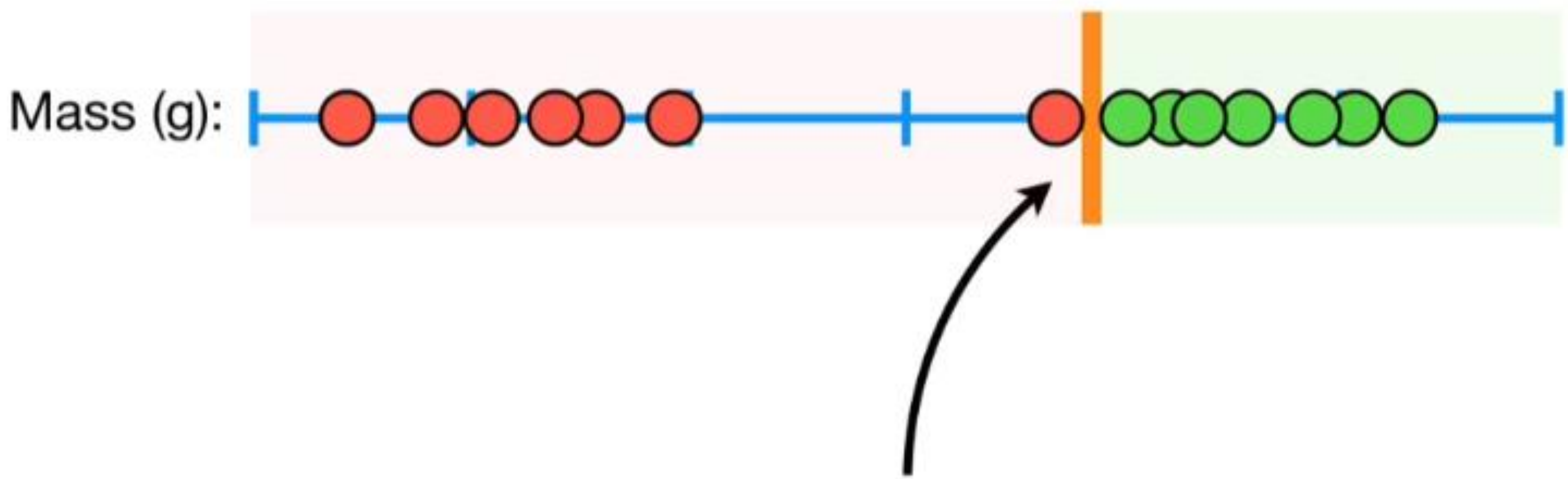
- Support Vector Machine --- 1D



- Support Vector Machine --- 1D

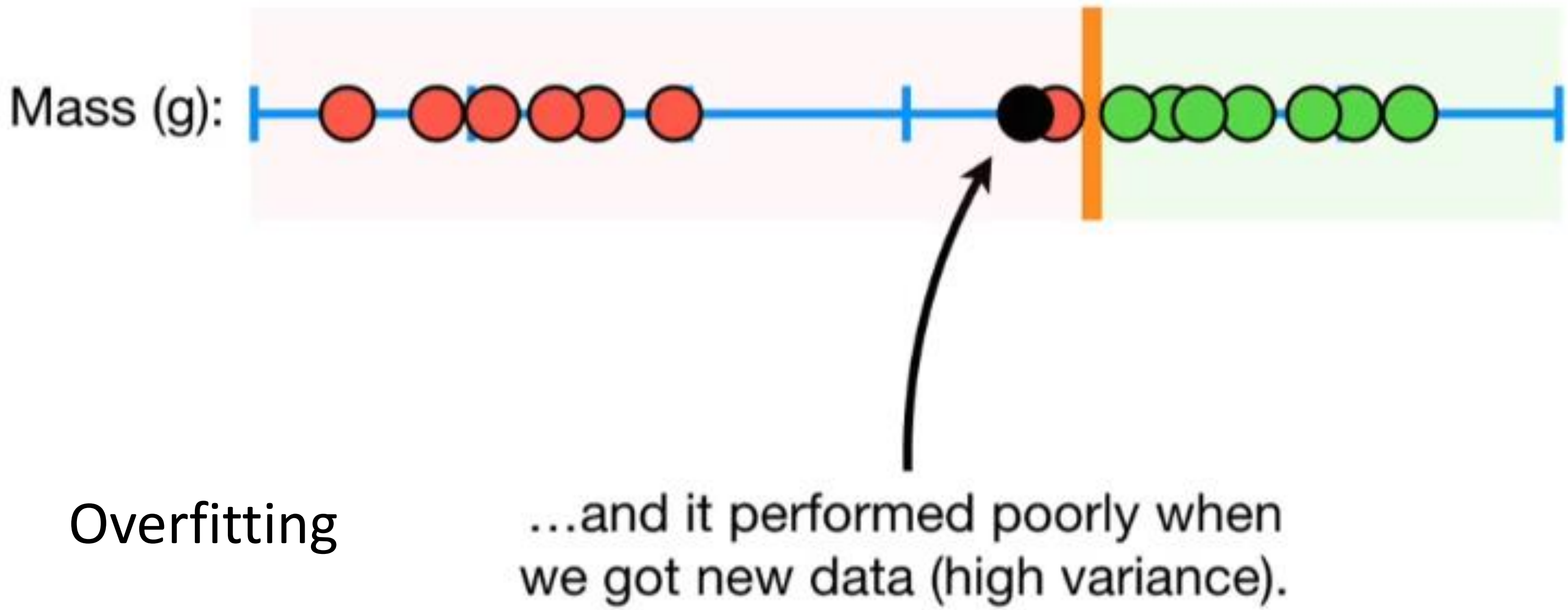


- Support Vector Machine --- 1D



In other words, before we allowed misclassifications, we picked a threshold that was very sensitive to the training data (low bias)…

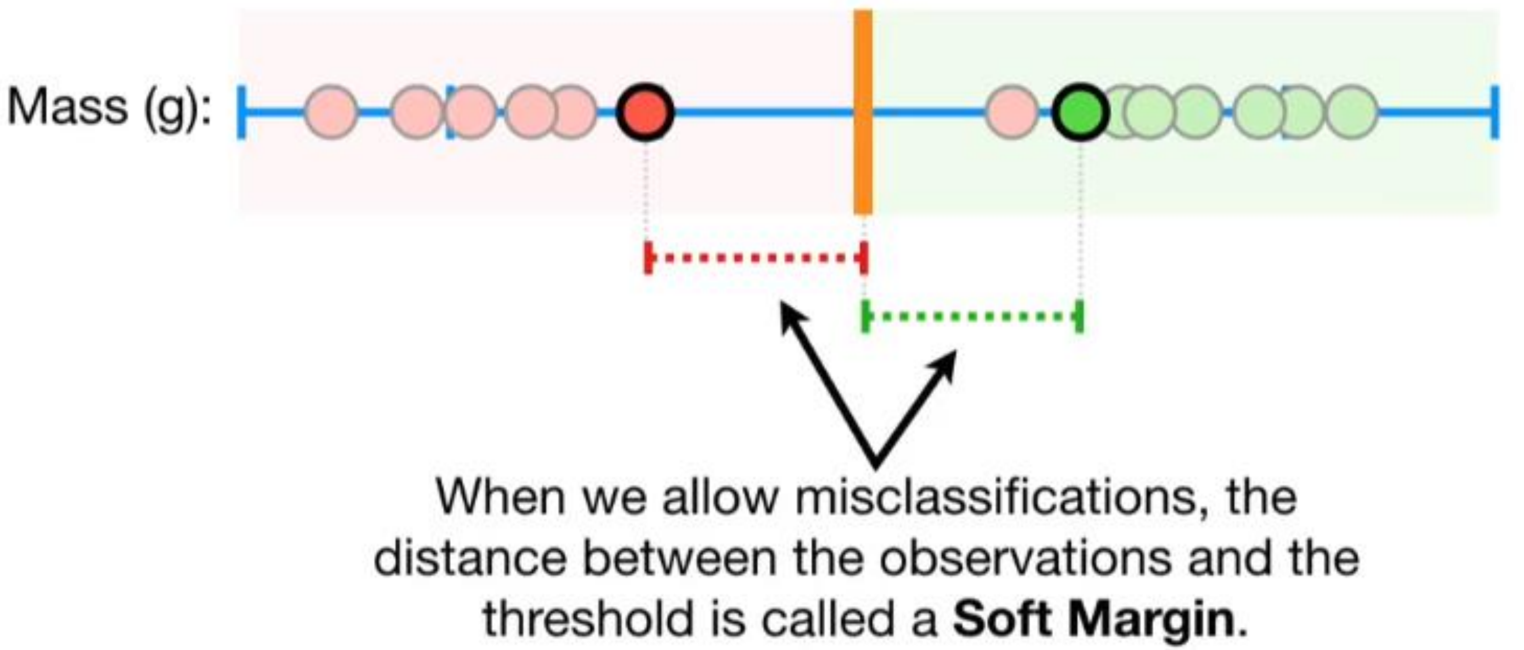
- Support Vector Machine --- 1D



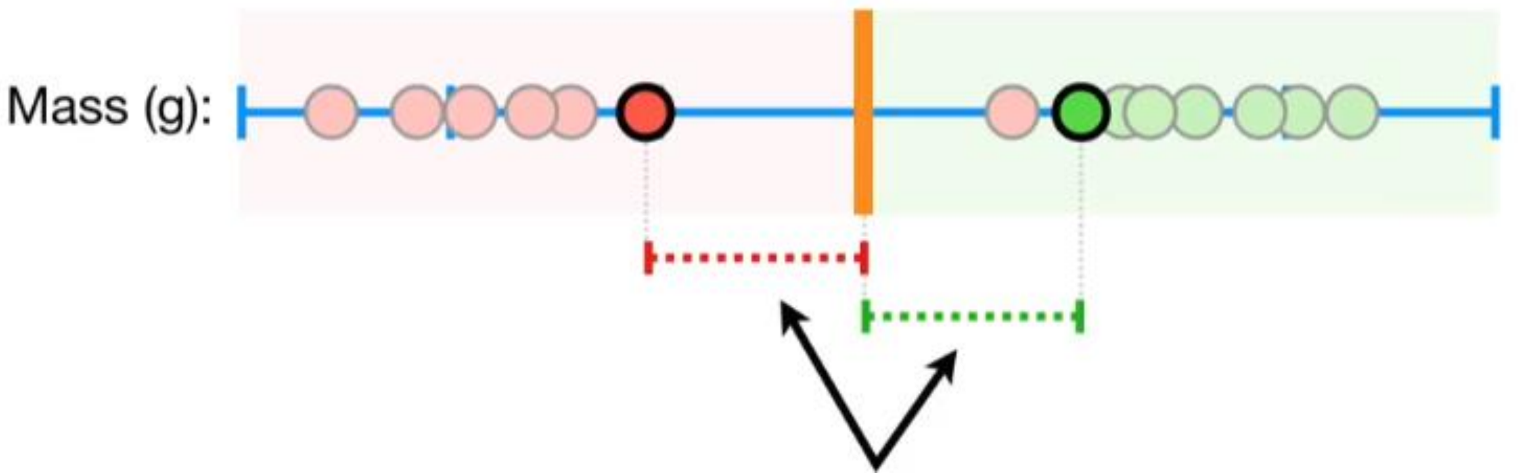
Overfitting

- Support Vector Machine --- 1D

Solution



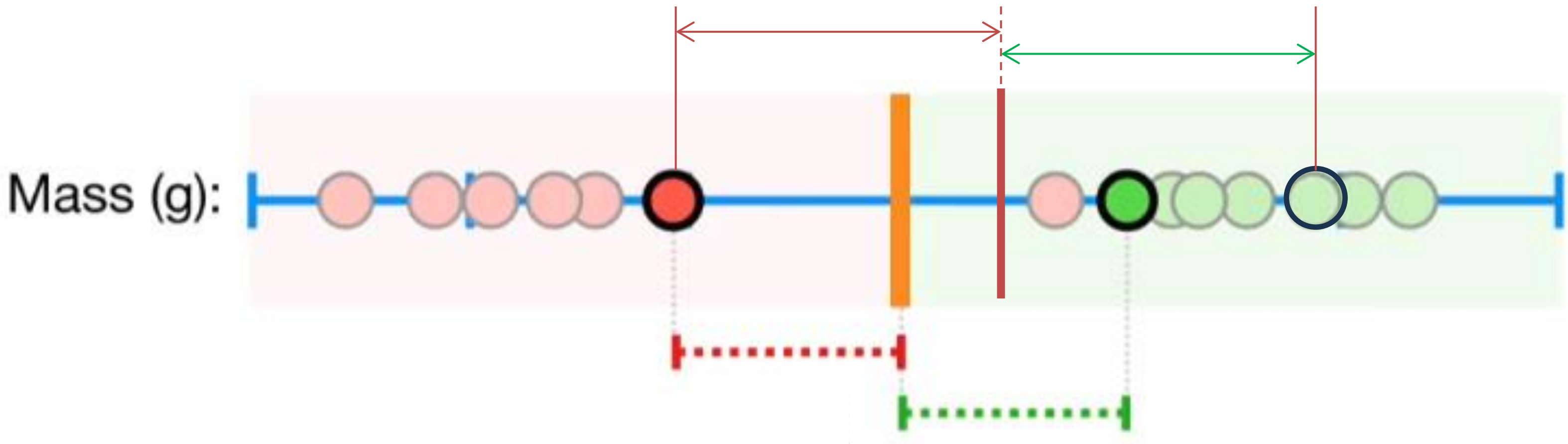
- Support Vector Machine --- 1D



When we use a **Soft Margin** to determine the location of a threshold

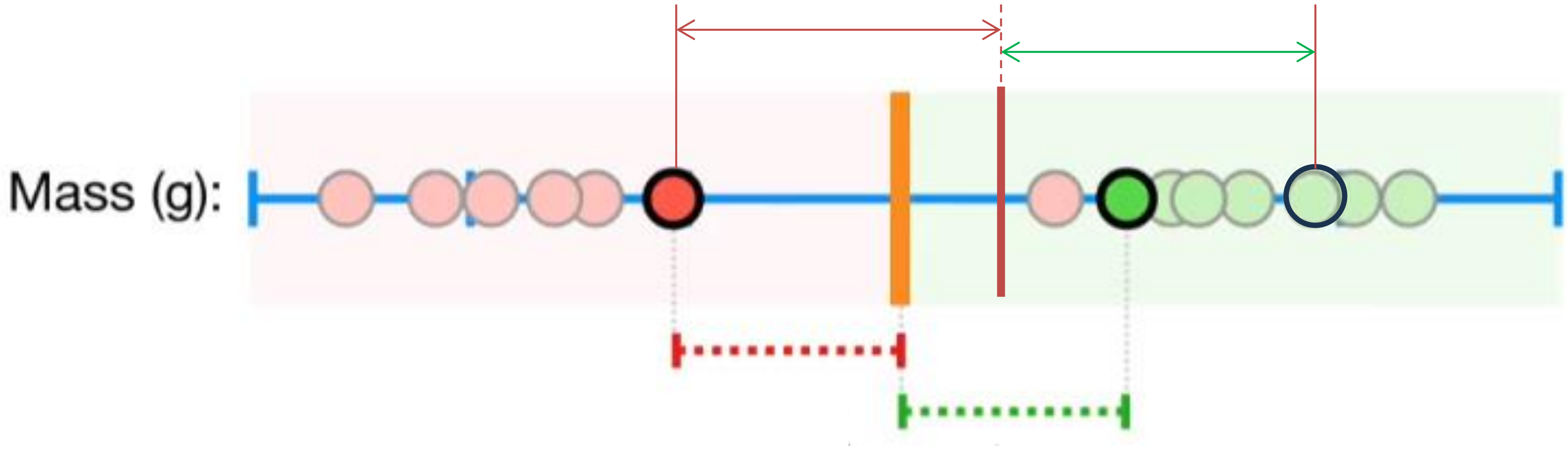
then we are using a **Soft Margin Classifier** aka a **Support Vector Classifier** to classify observations.

- Support Vector Machine --- 1D



How do we know which soft margin is better?

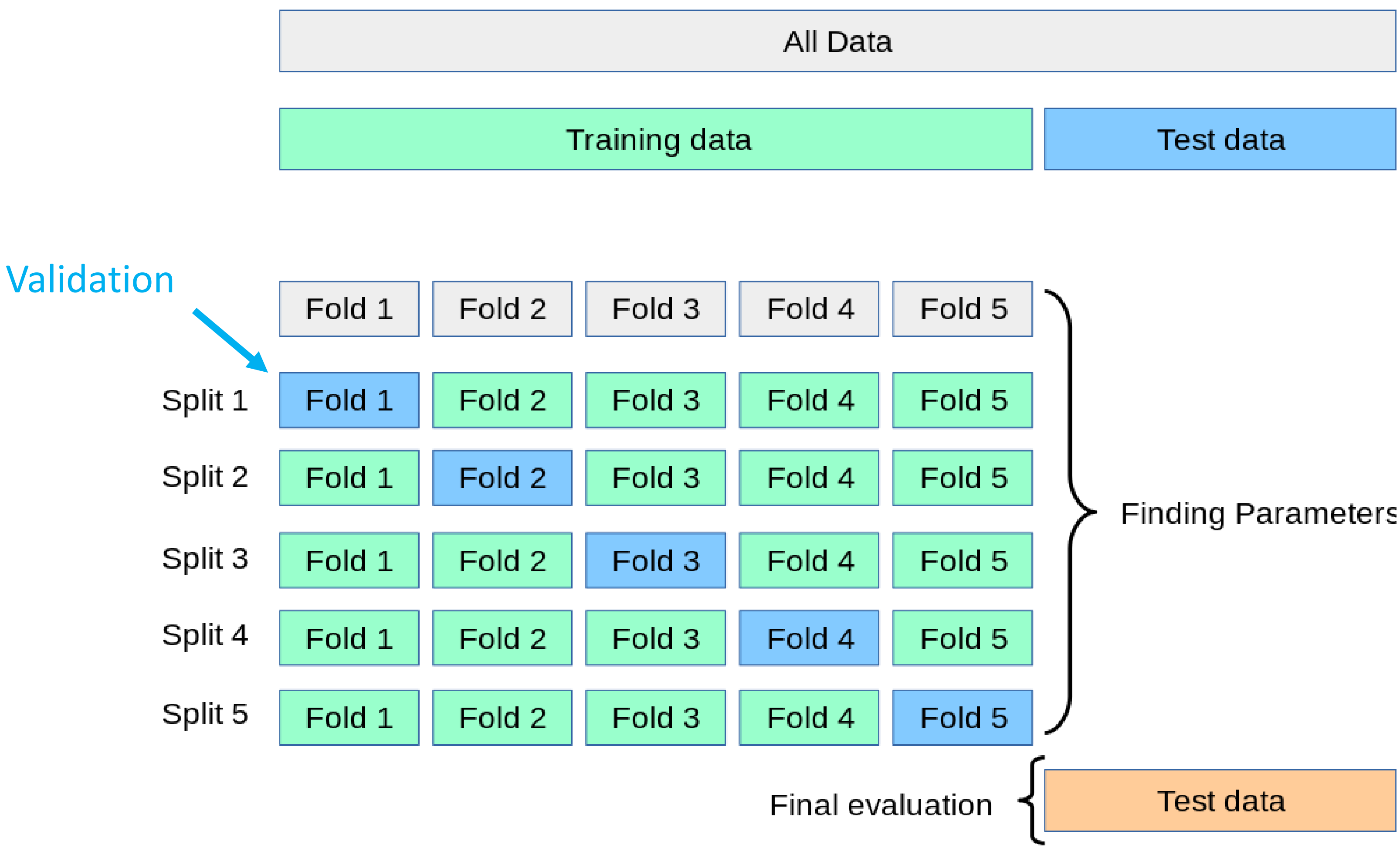
- Support Vector Machine --- 1D



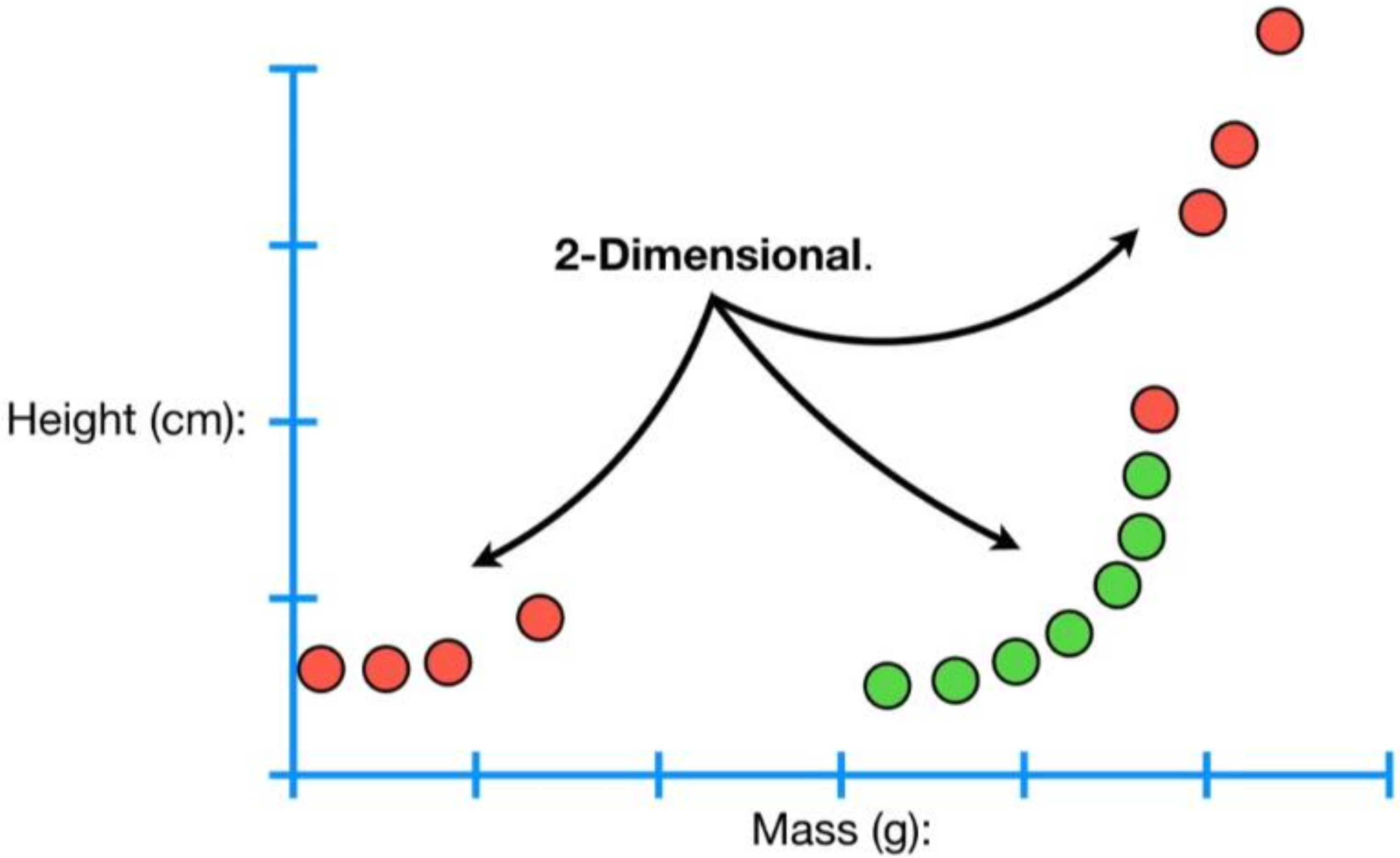
The answer is simple: We use **Cross Validation** to determine how many misclassifications and observations to allow inside of the **Soft Margin** to get the best classification.

Support Vector Machine

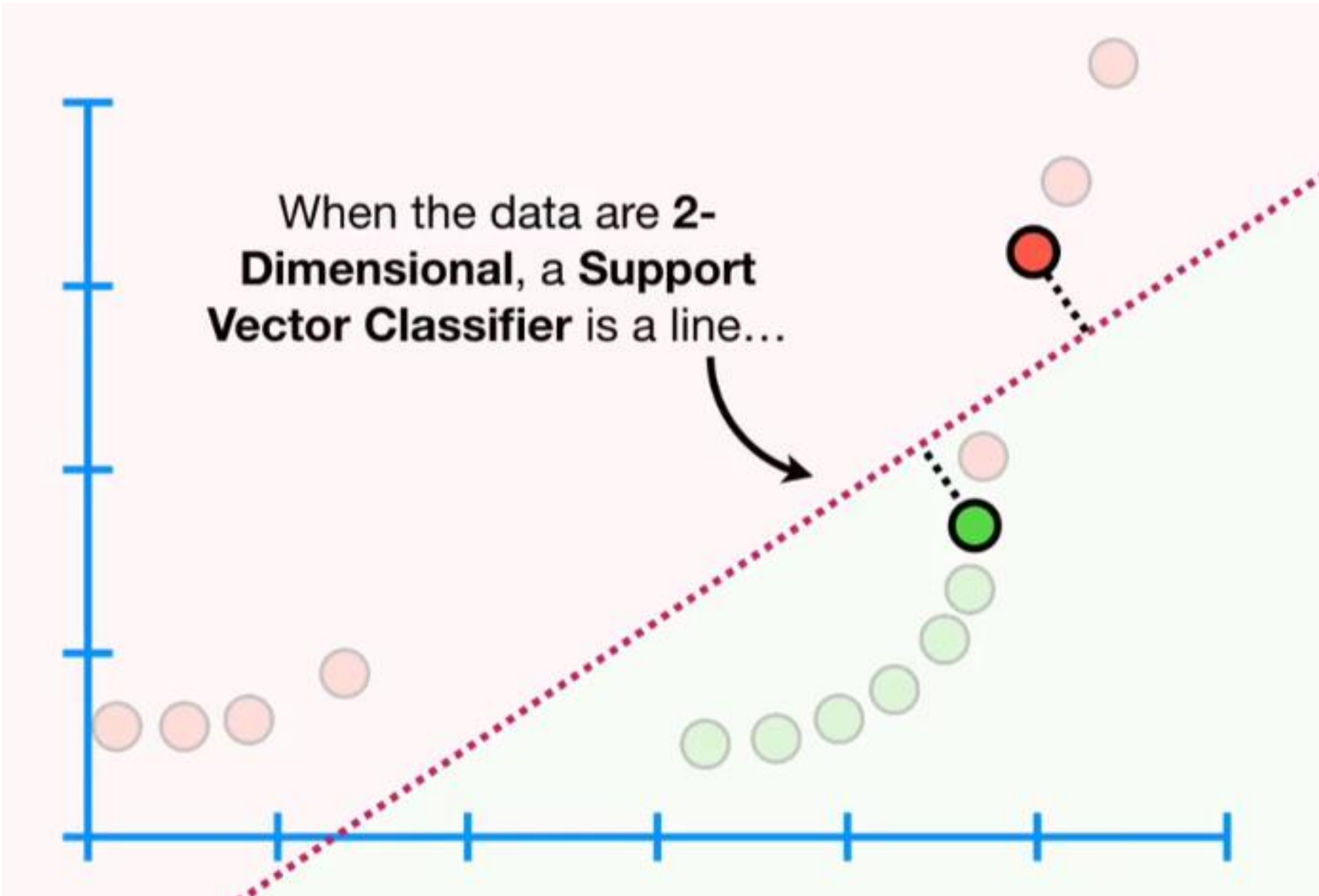
- K-fold cross validation
- k=5



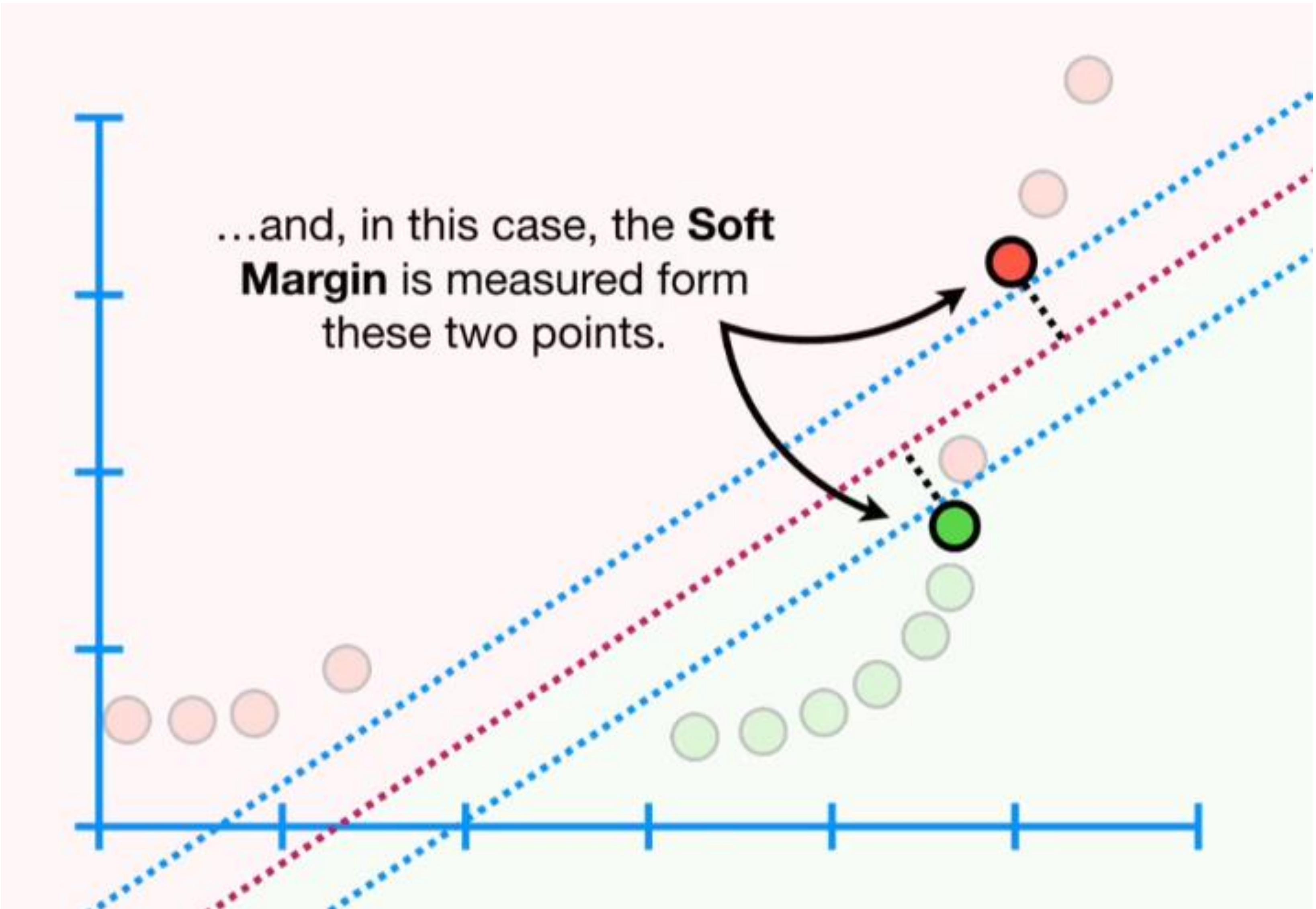
- Support Vector Machine --- 2D



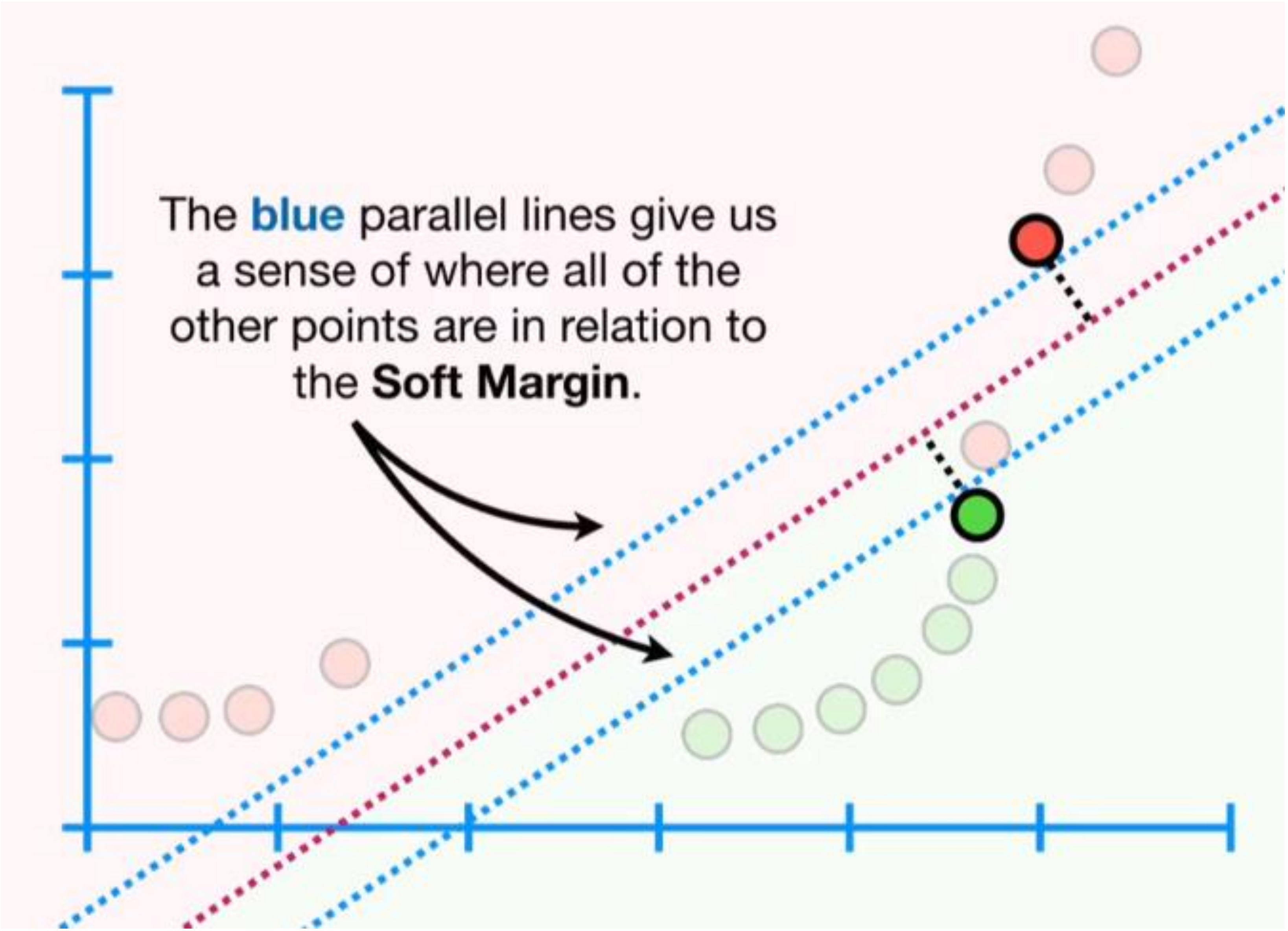
- Support Vector Machine --- 2D



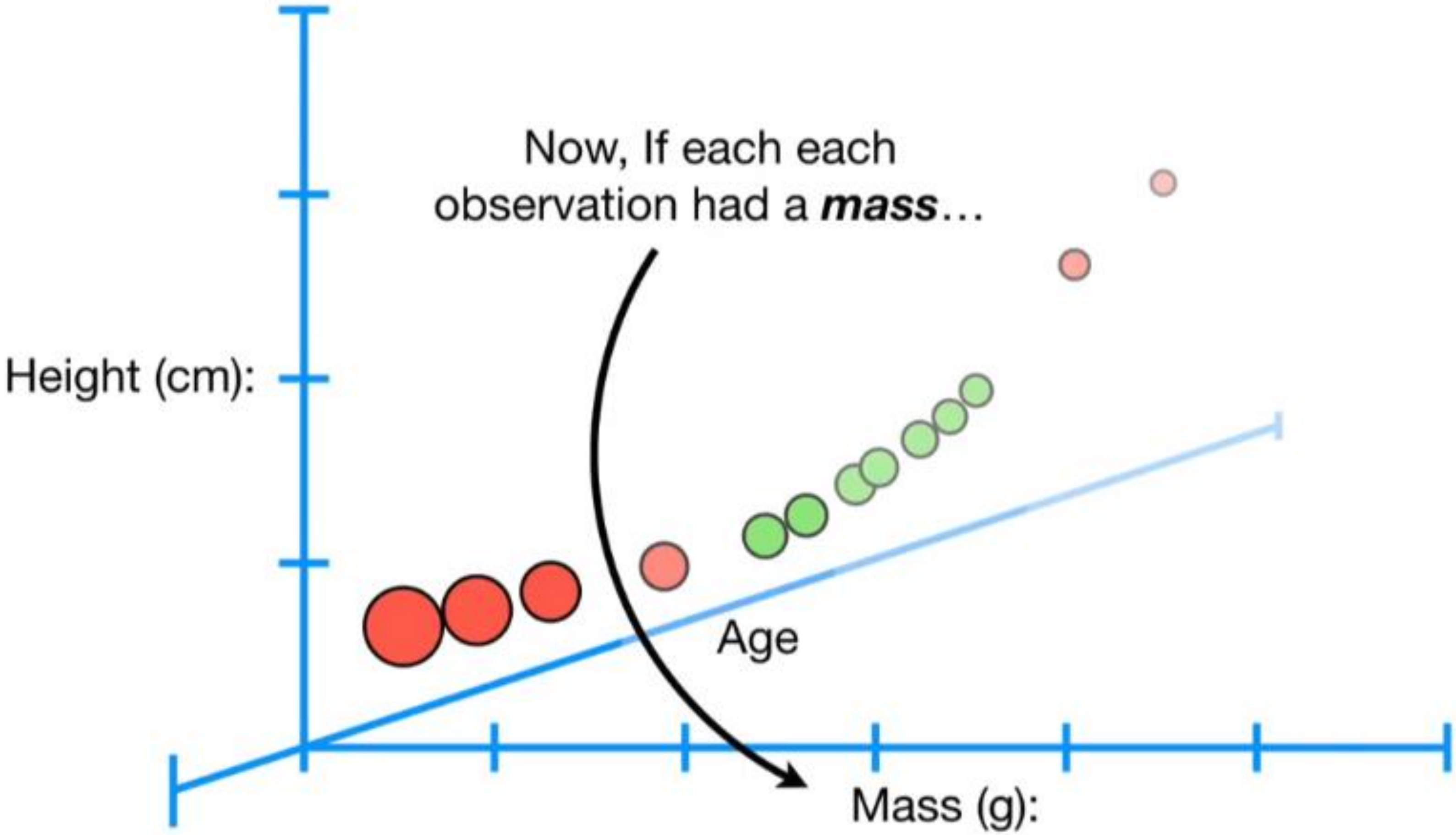
- Support Vector Machine --- 2D



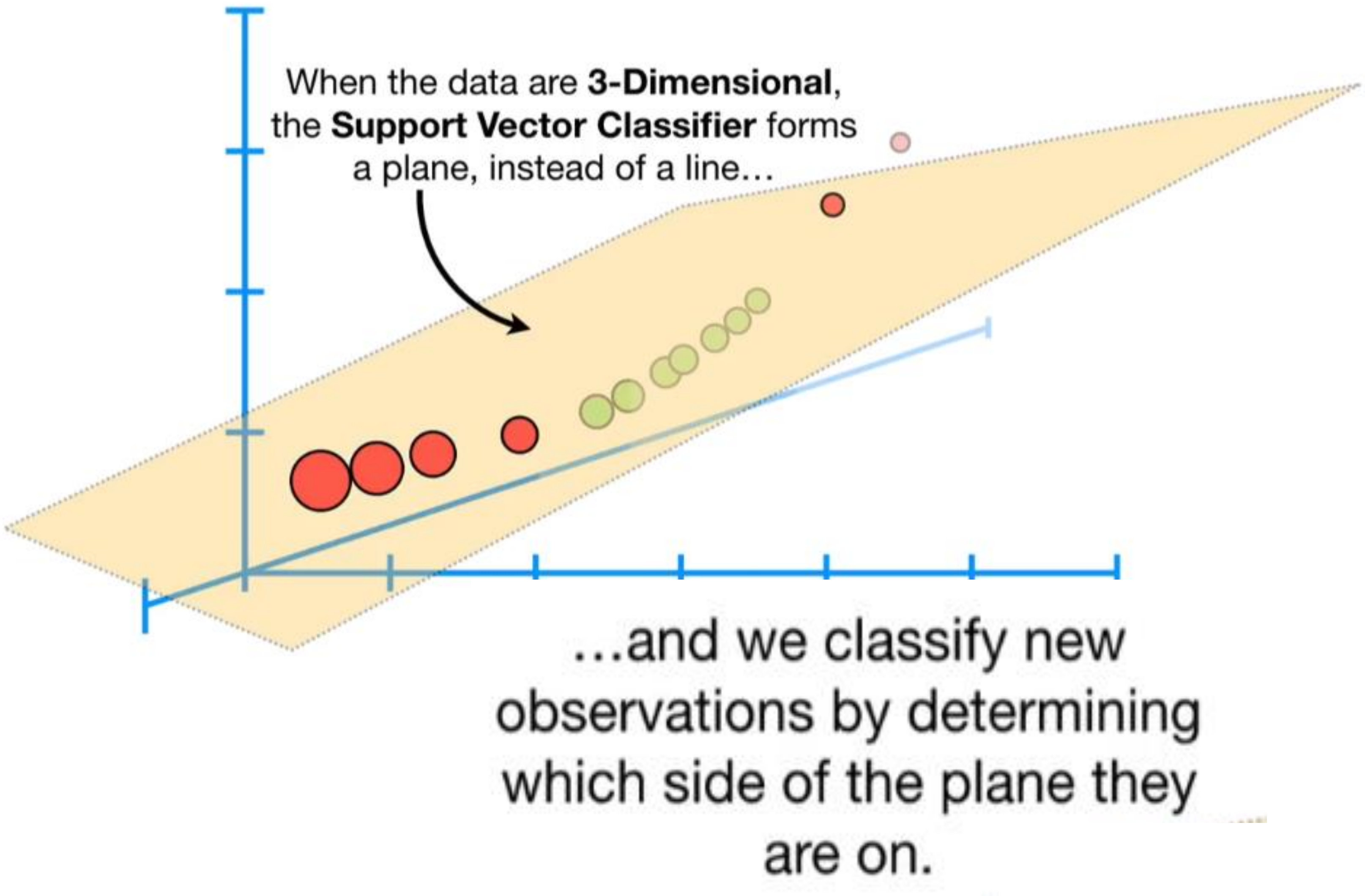
- Support Vector Machine --- 2D



- Support Vector Machine --- 3D



• Support Vector Machine --- 3D

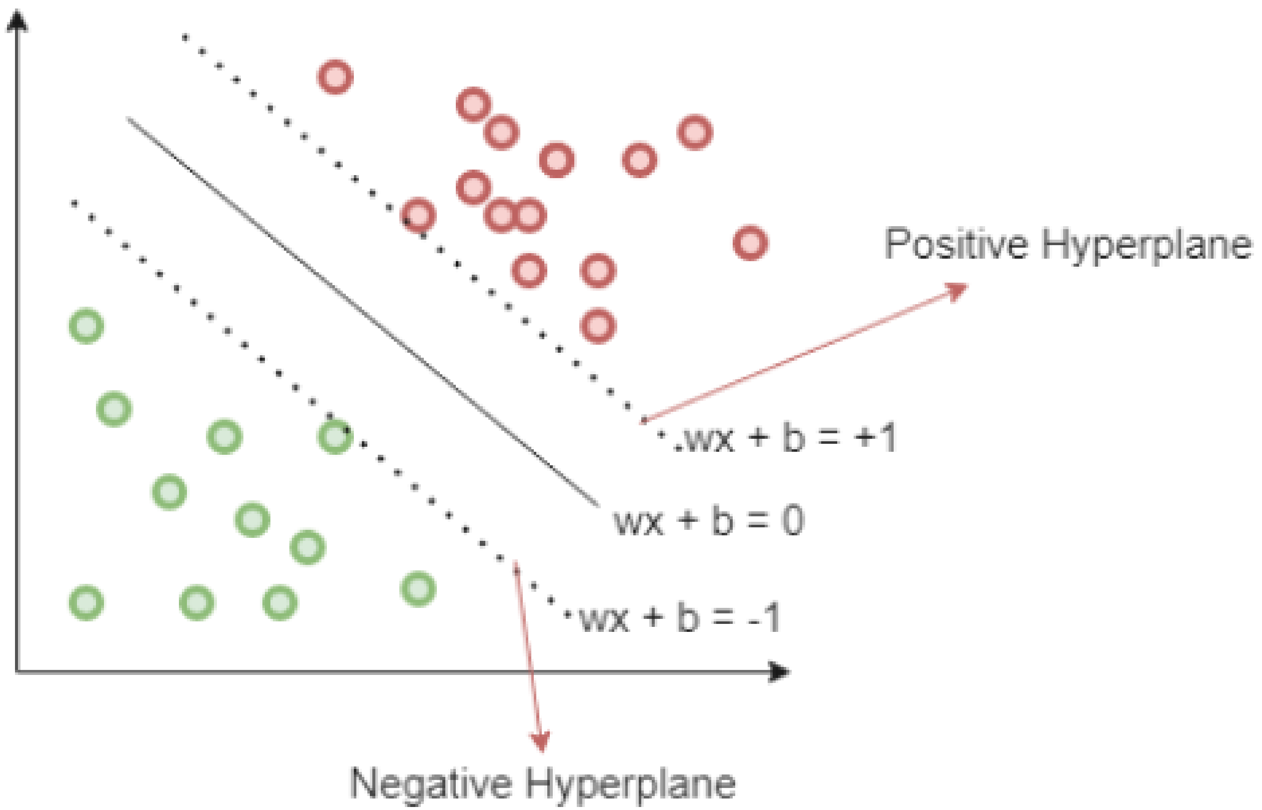


- Support Vector Machine --- >3D

NOTE: If we measured ***mass***, ***height***, ***age*** and ***blood pressure***, then the data would be in **4 Dimensions**…

The Support Vector Classifier is a **hyperplane**

How to find the optimal classifier?



https://www.pycodemates.com/2022/09/primal-formulation-of-svm-simplified.html
https://en.wikipedia.org/wiki/Support_vector_machine

Exercise/Homework: Optimal Maximal Margin Classifier

Labelled dataset $(x_1, y_1), (x_2, y_2), \dots, (x_d, y_d)$, where $x_i \in \mathbb{R}^n$, $y_i \in \{-1,1\}$ for all i.

We need to find parameters $w \in \mathbb{R}^n$, $b \in \mathbb{R}$, satisfying for all i:

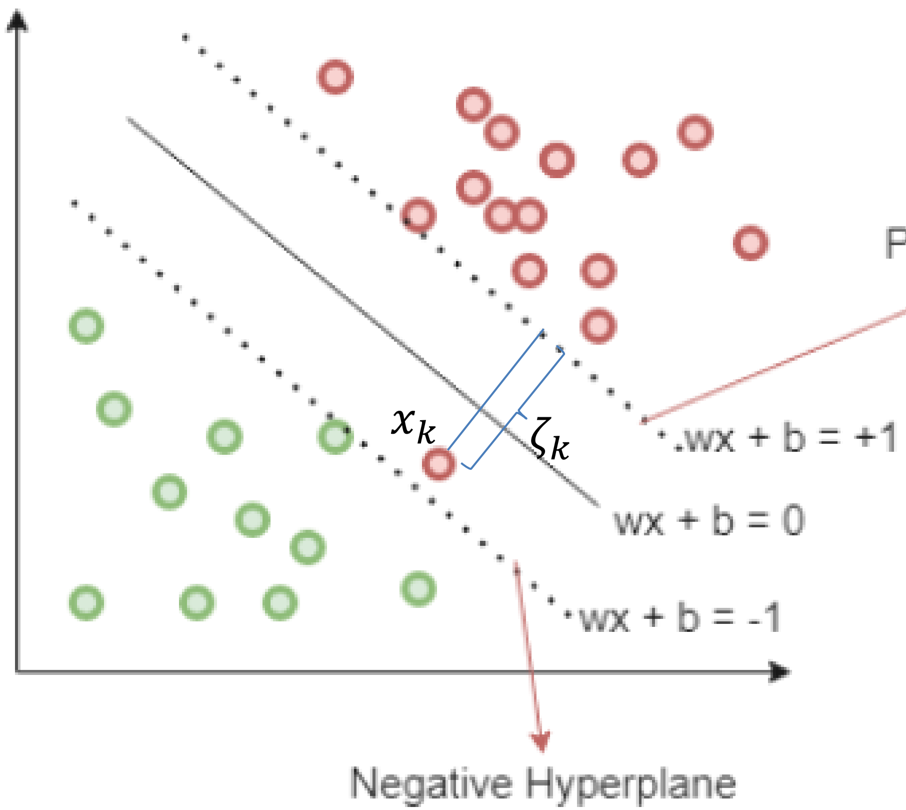
$$y_i = \begin{cases} 1 & w^T x_i + b \geq 1 \\ -1 & w^T x_i + b \leq -1 \end{cases}$$

That maximizes the distance between the two hyperplanes.

Question: Given any w and b, how do we compute the distance between the hyperplanes?

Hint: First prove that vector w is orthogonal to any vector in the hyperplane.

Optimal Soft Margin Classifier



$$\min_{w,b,\xi} \quad \frac{1}{2}\parallel w \parallel^2 + C\sum_{i=1}^{n}\xi_i$$

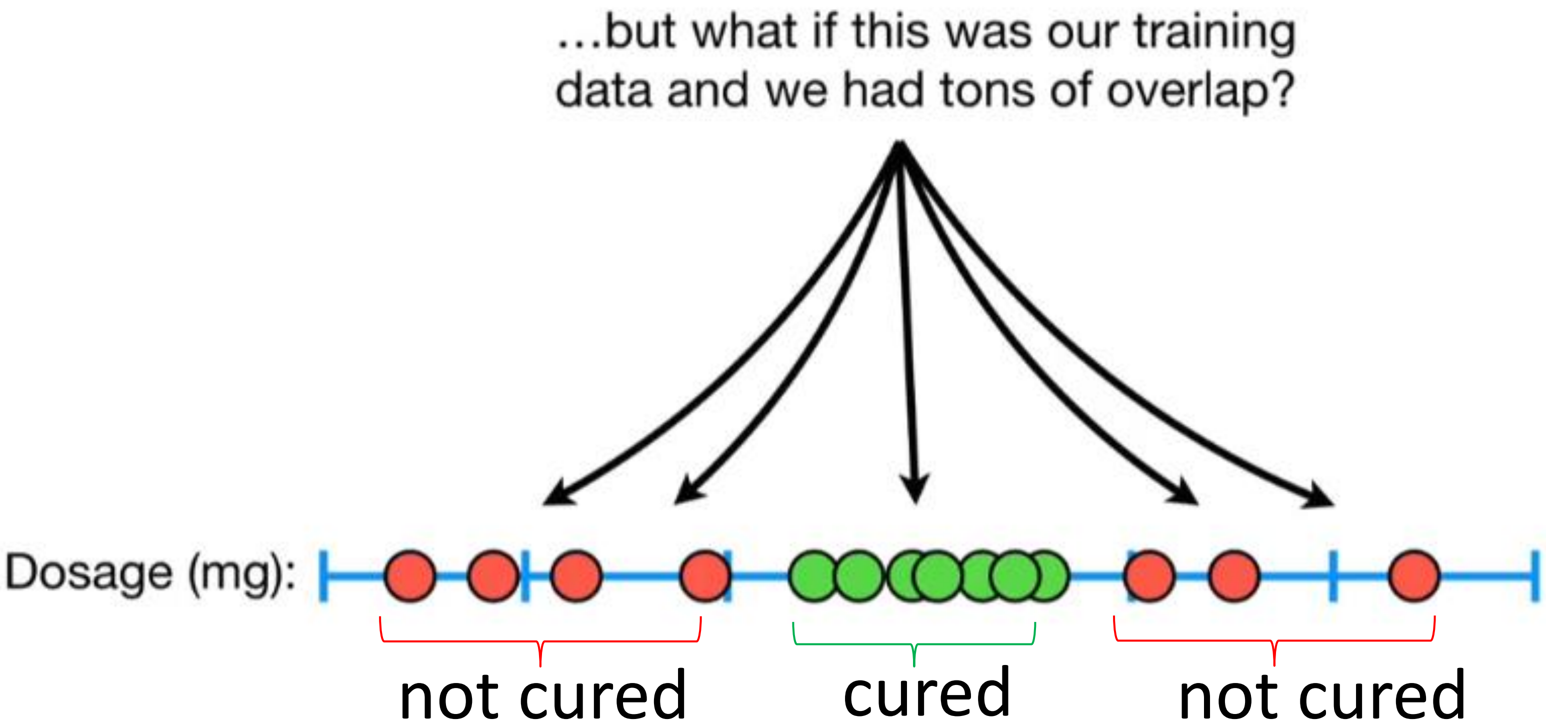
$$\text{s.t.} \quad y_i\left(w^T x_i + b\right) \geqslant 1 - \xi_i$$

$$\xi_i \geqslant 0$$

$$i = 1, 2, \cdots, N$$

https://www.pycodemates.com/2022/09/primal-formulation-of-svm-simplified.html
https://en.wikipedia.org/wiki/Support_vector_machine

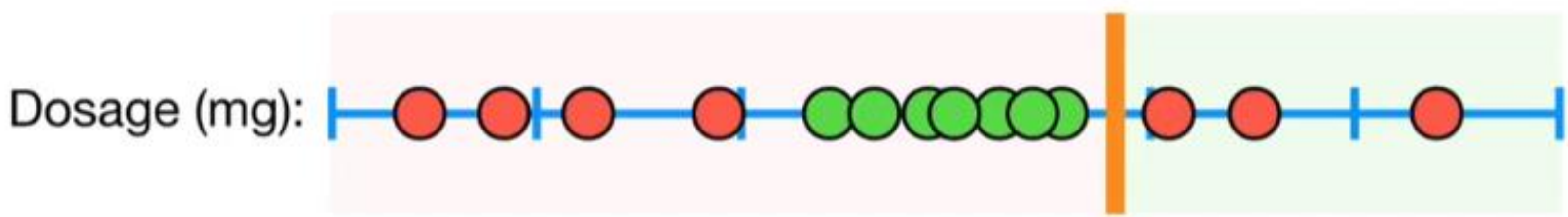
- Support Vector Machine



How can we find a classifier that can tell whether a new dosage can cure or not?

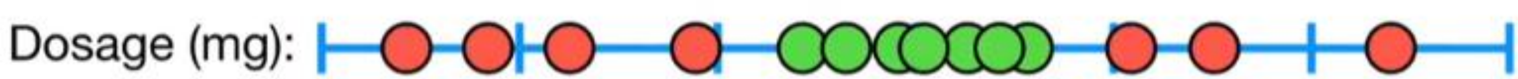
- Support Vector Machine

Can we do better than **Maximal Margin Classifiers** and **Support Vector Classifiers**?

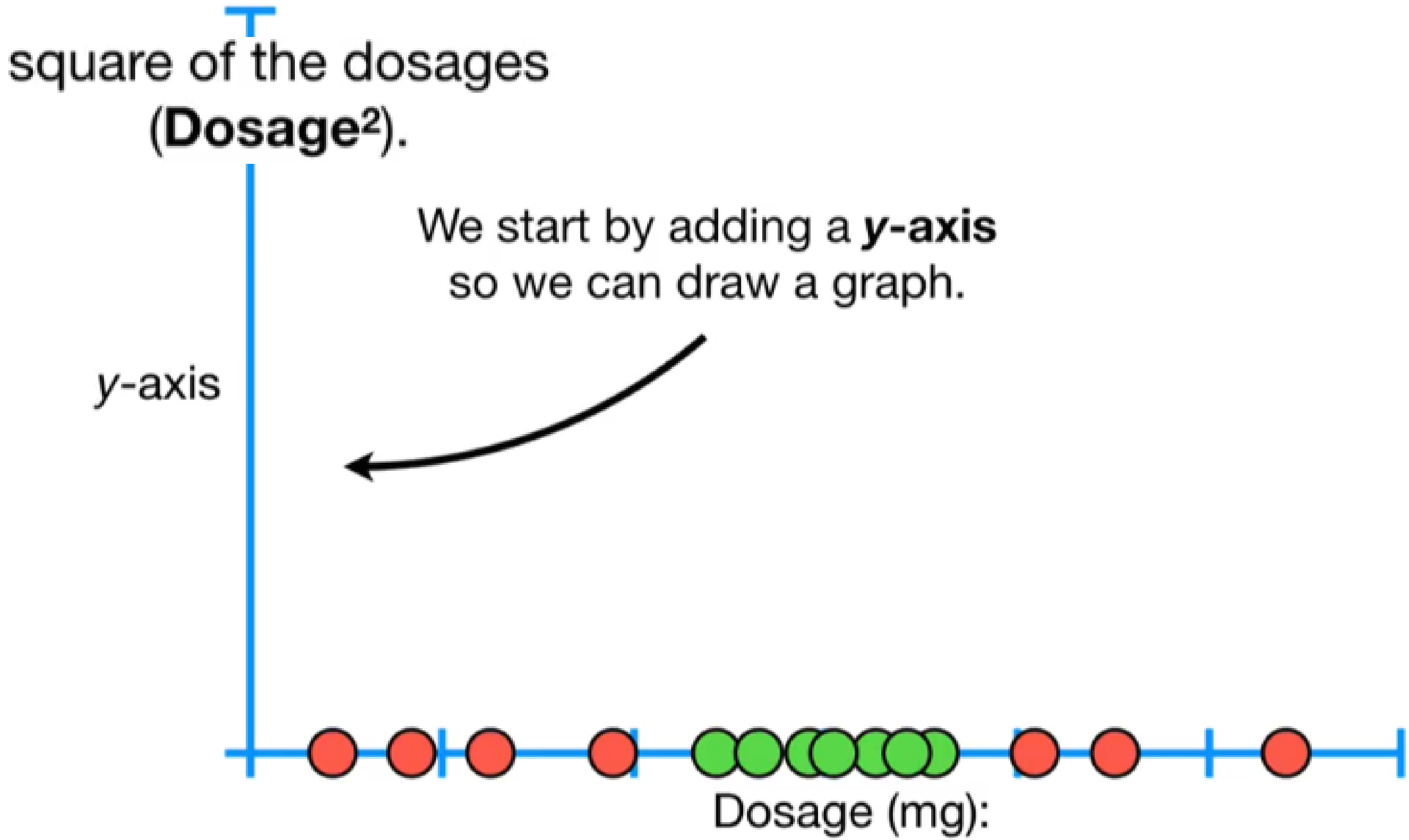


- Support Vector Machine

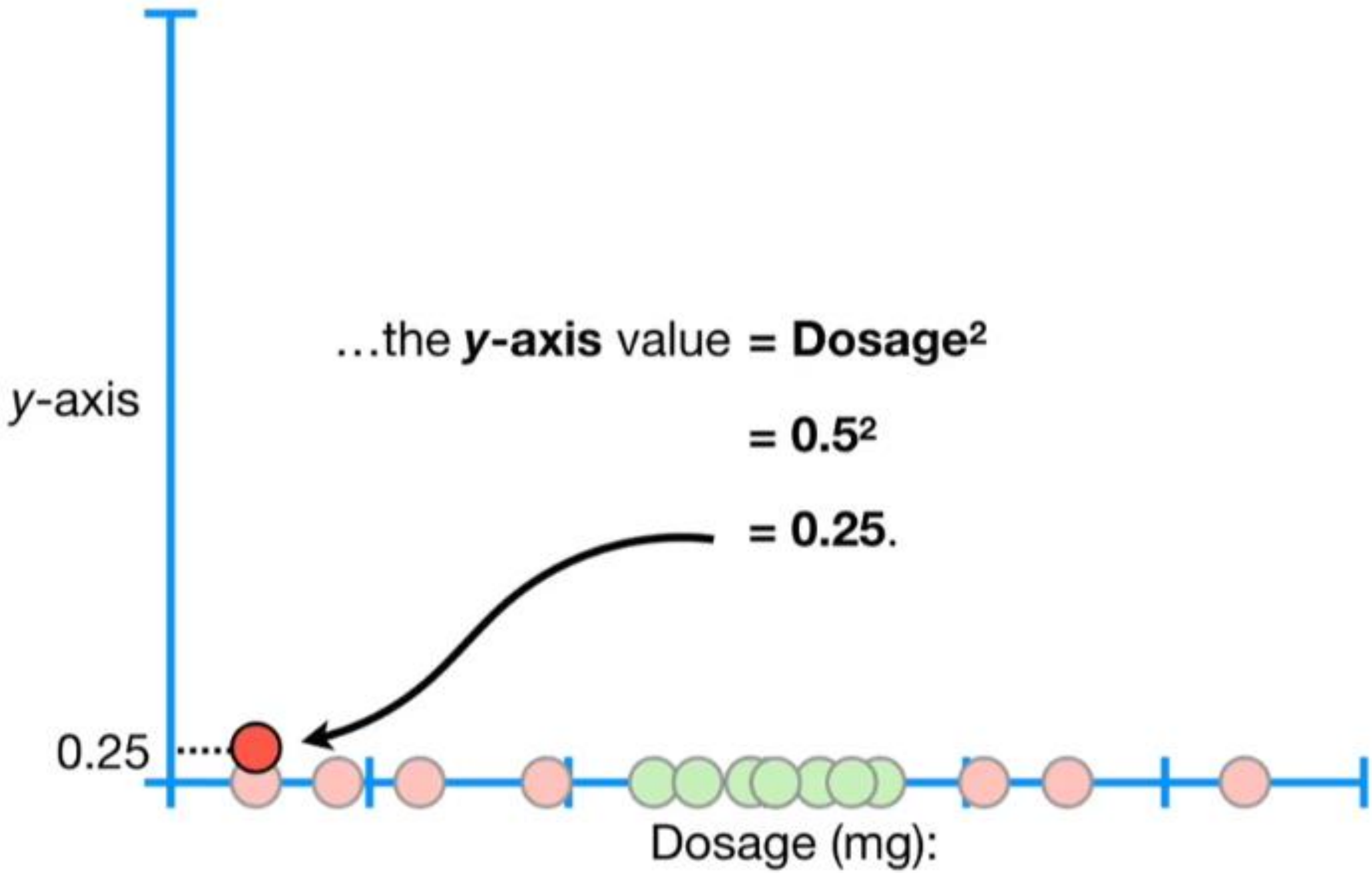
Support Vector Machines!!!



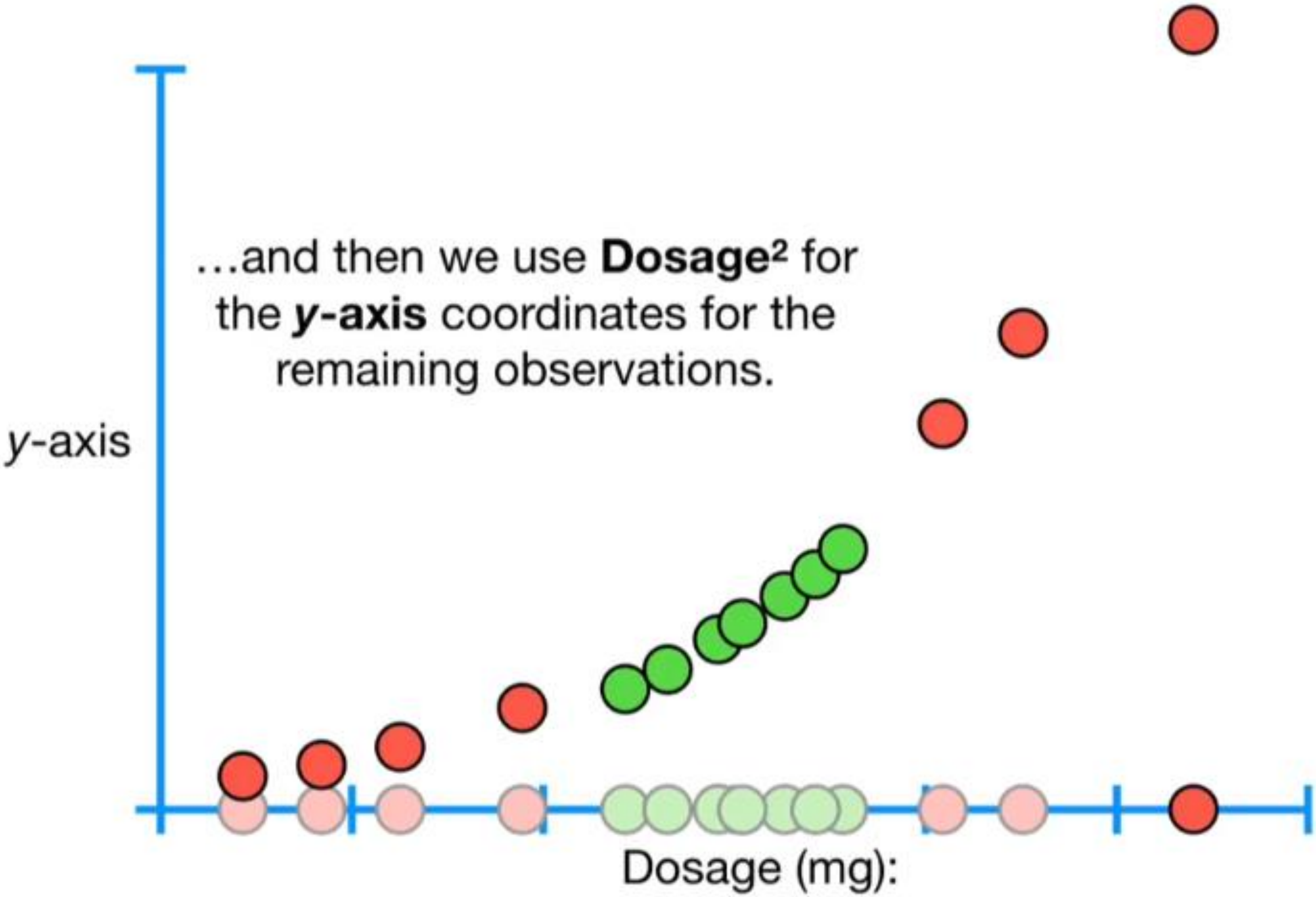
- Support Vector Machine



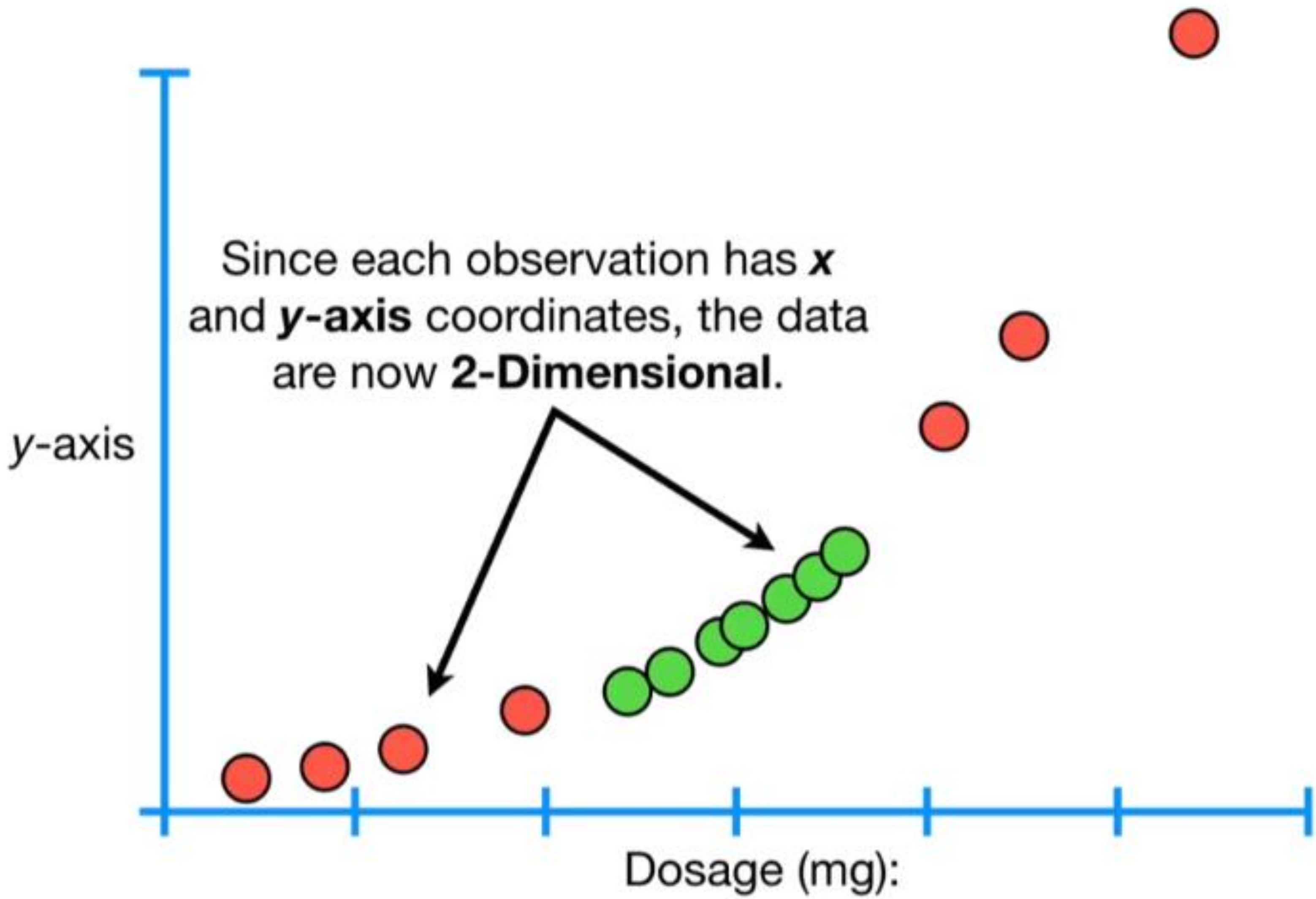
- Support Vector Machine



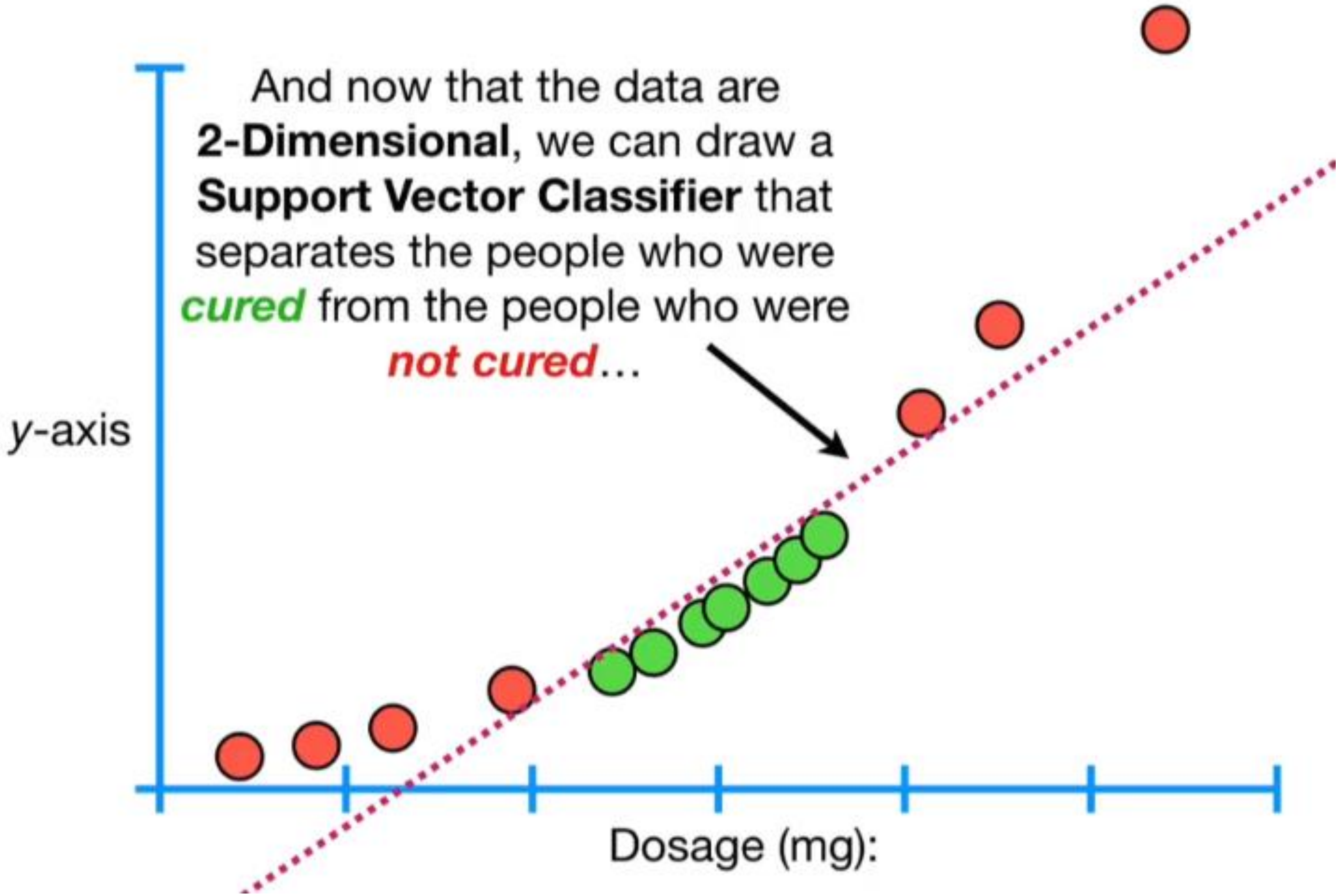
- Support Vector Machine



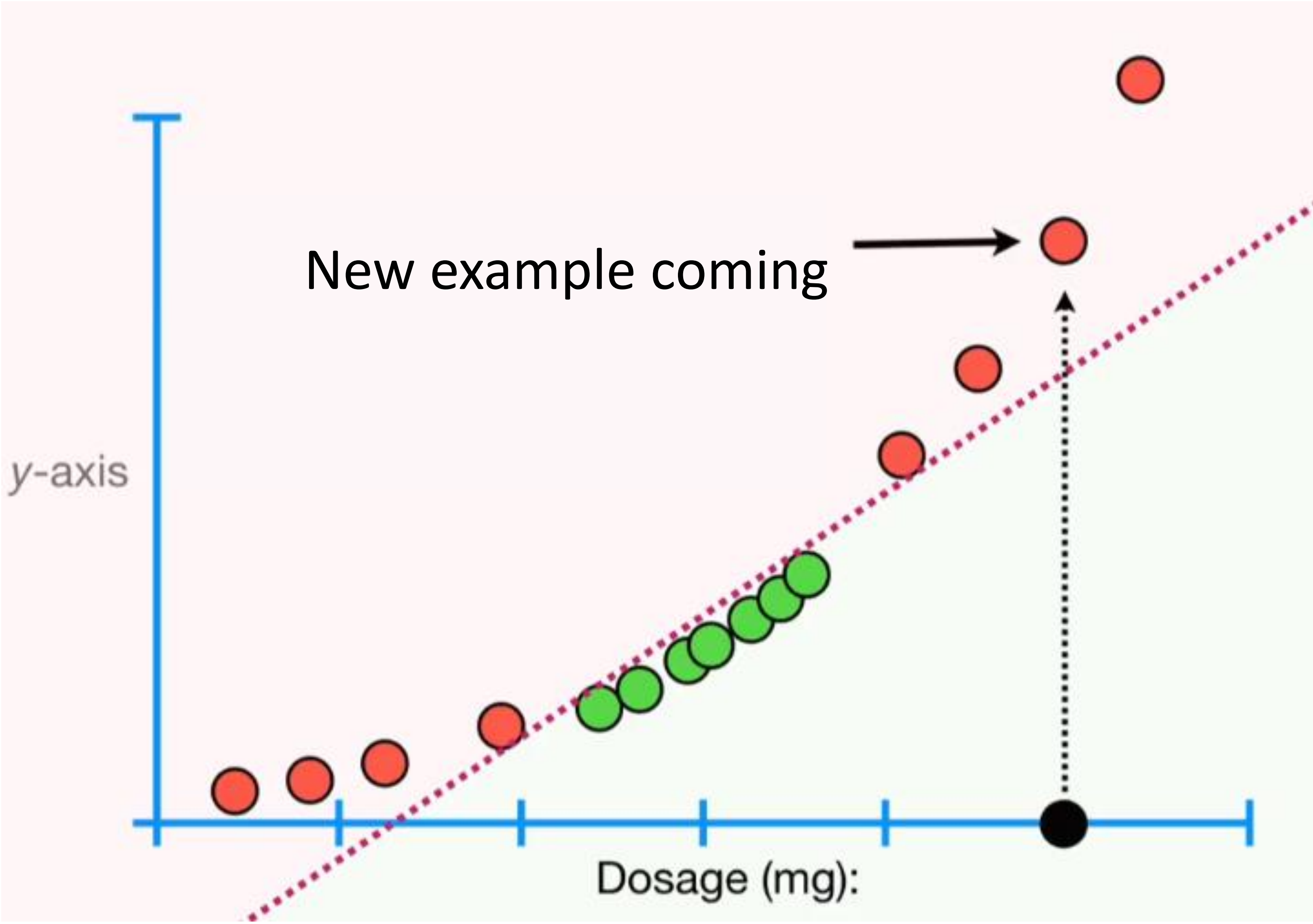
- Support Vector Machine



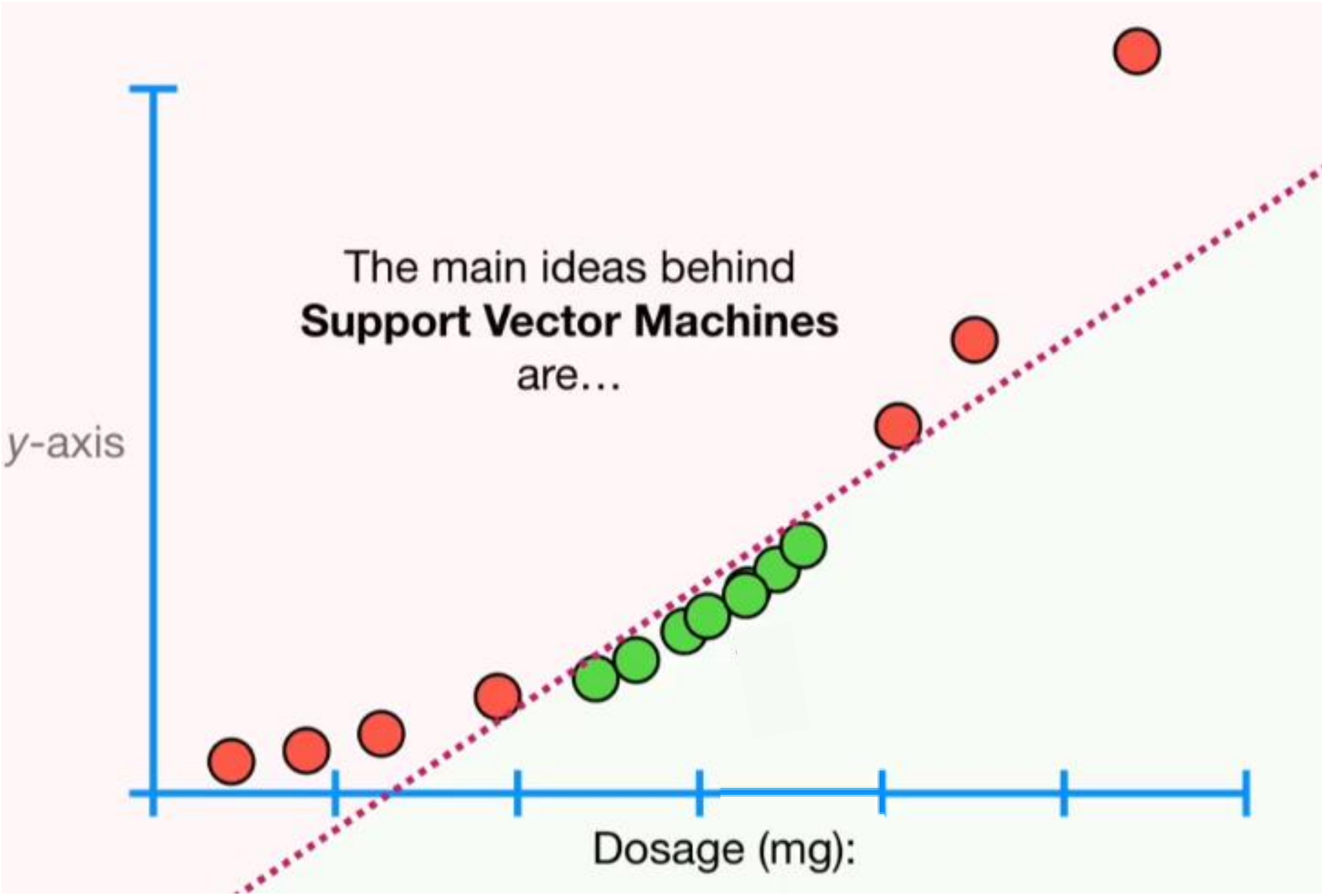
- Support Vector Machine



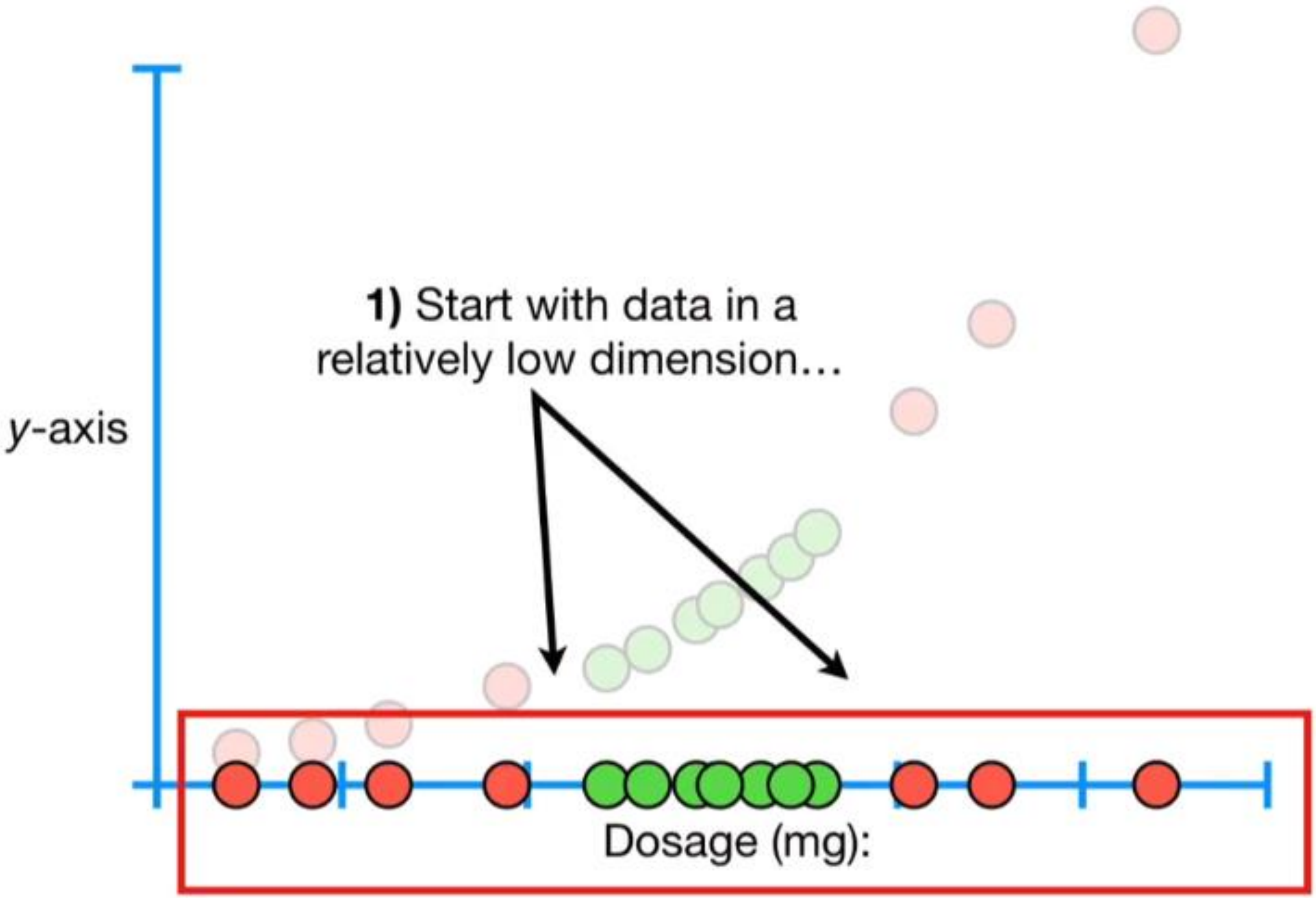
- Support Vector Machine



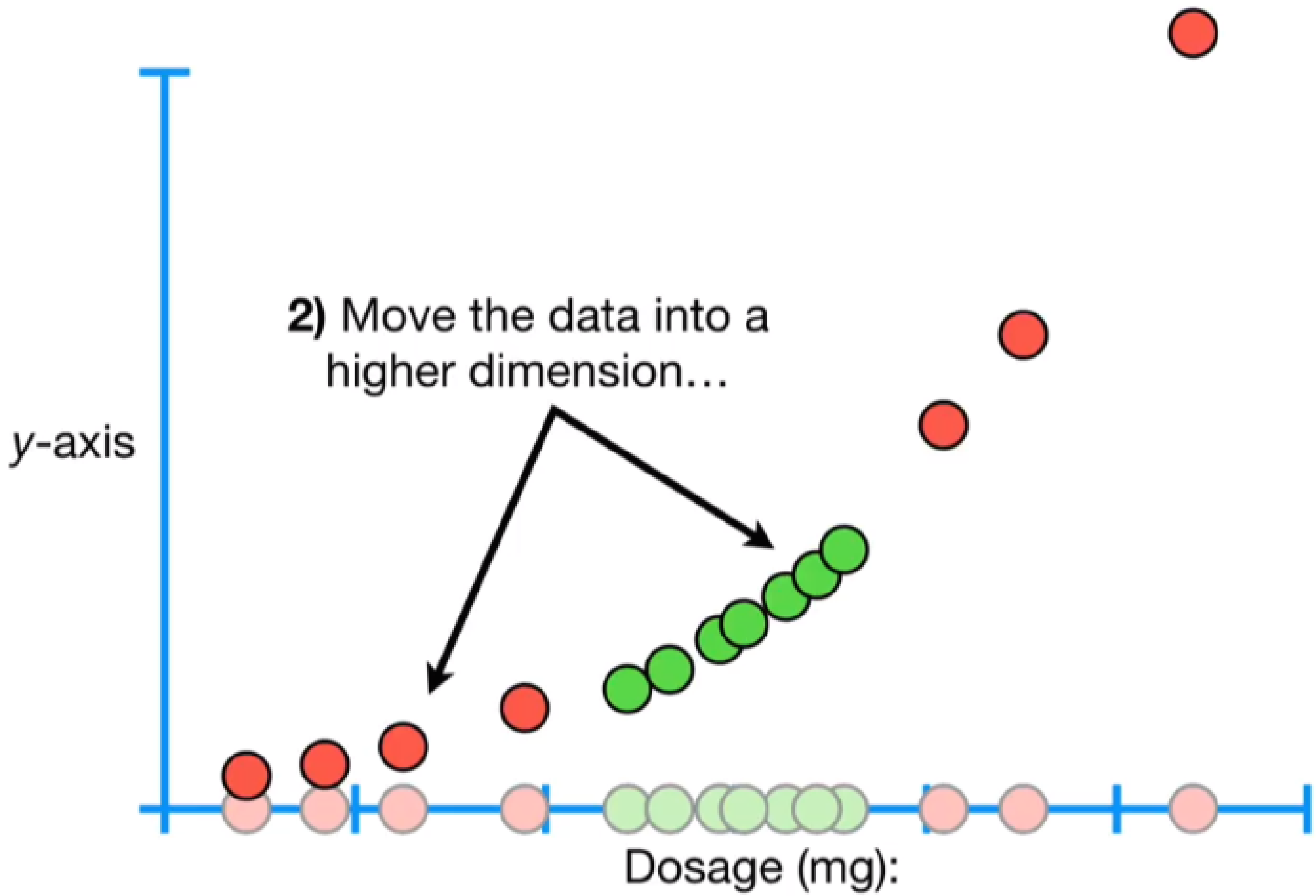
- Support Vector Machine



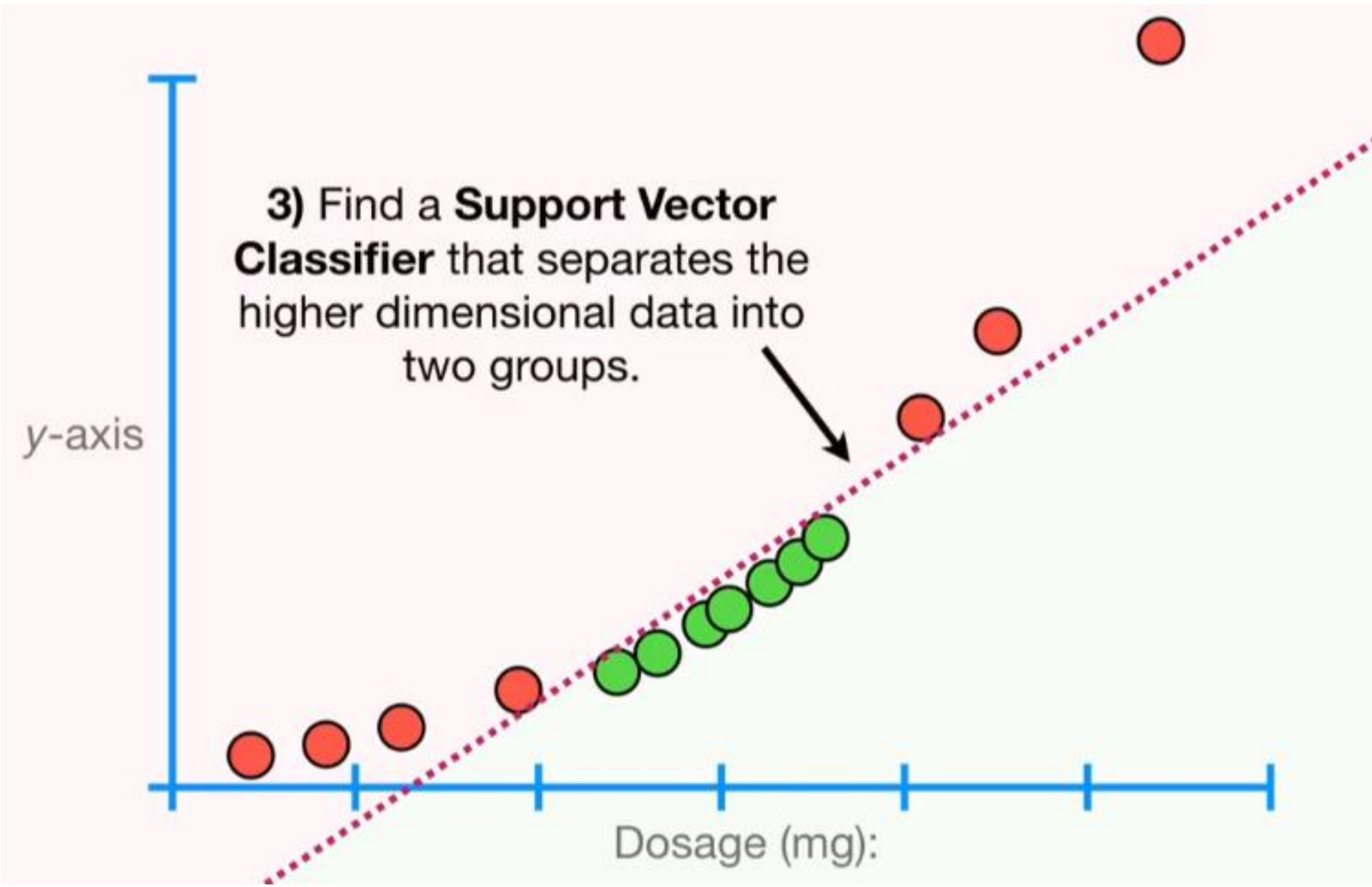
- Support Vector Machine



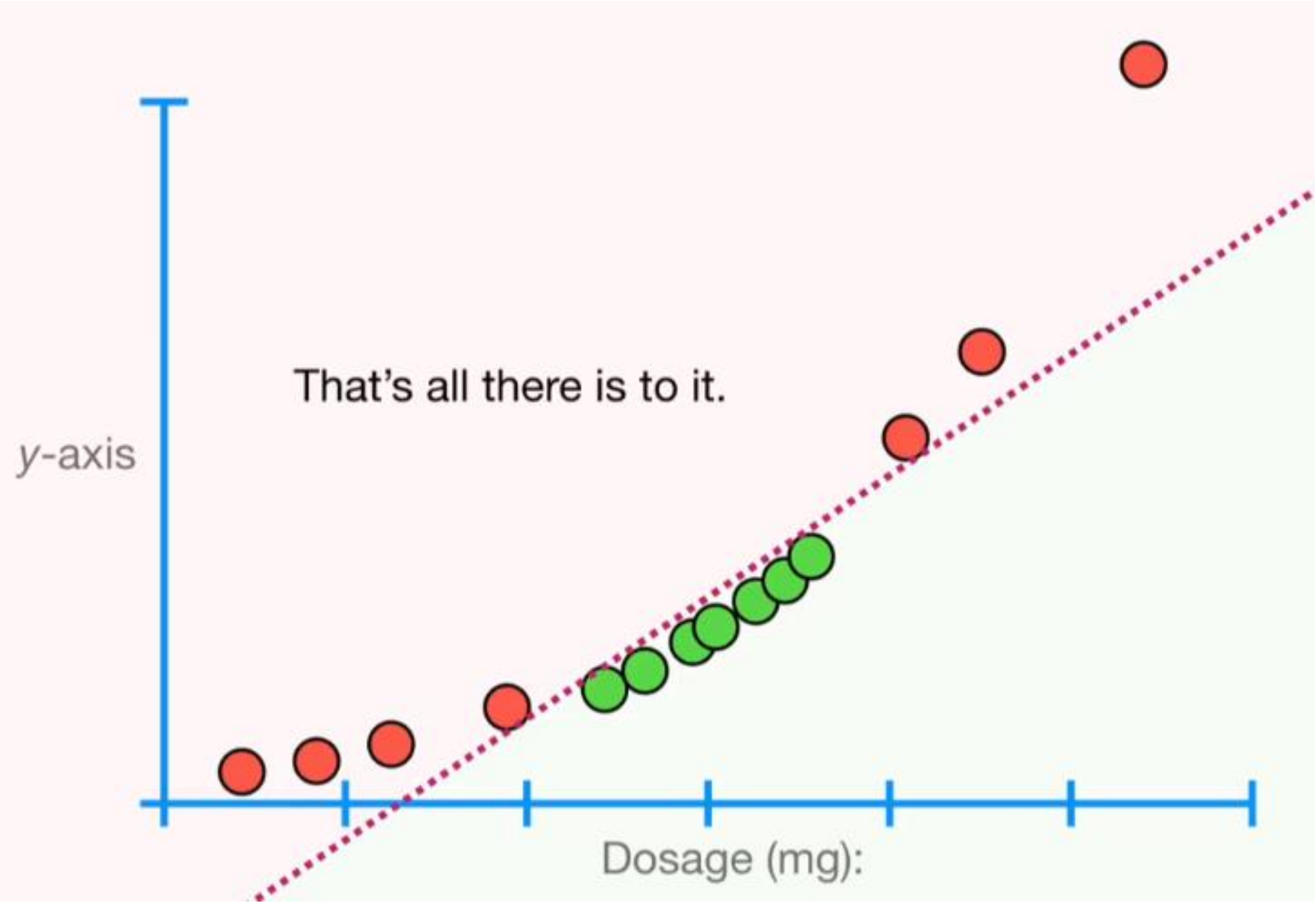
- Support Vector Machine



- Support Vector Machine



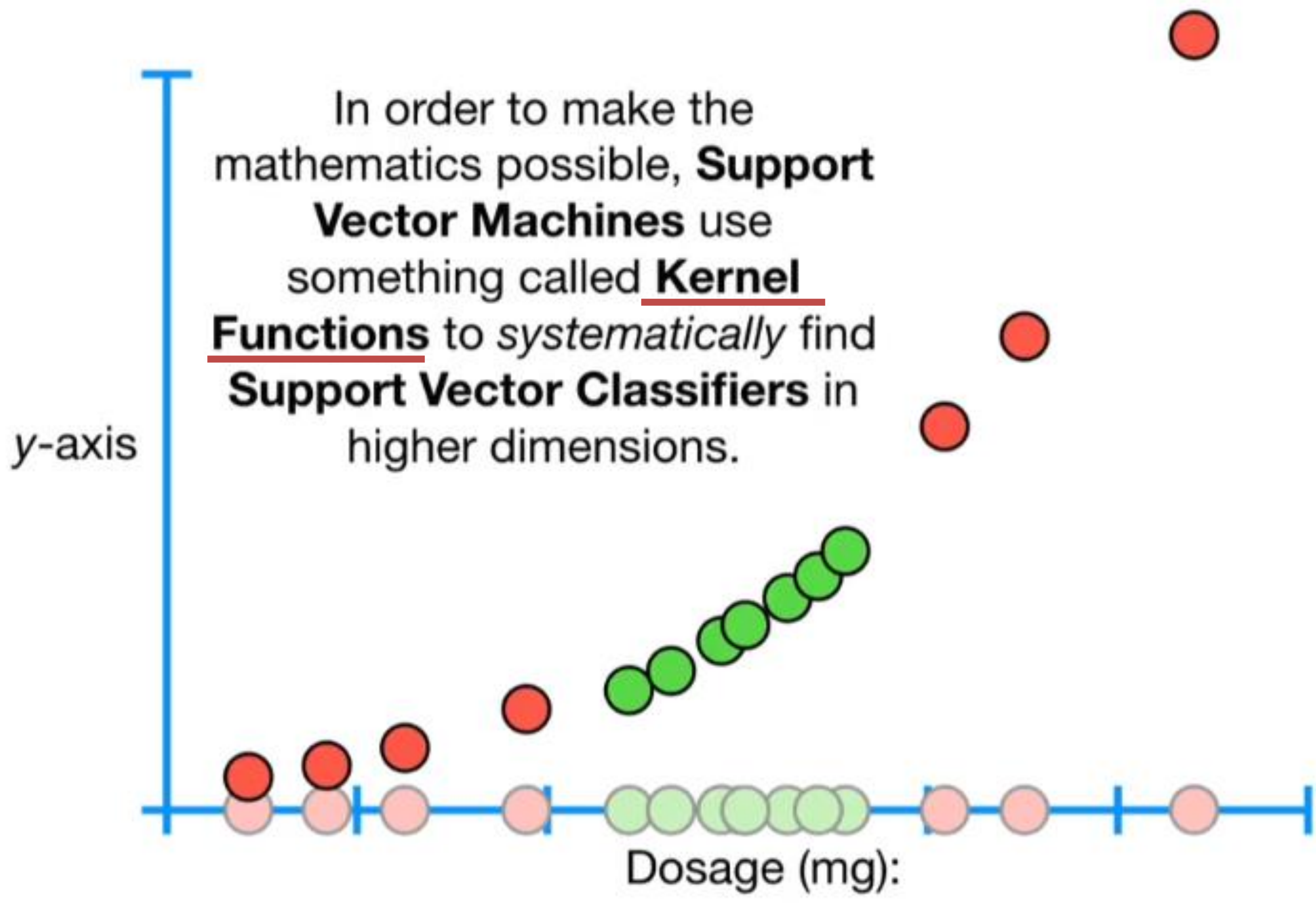
- Support Vector Machine



- Support Vector Machine

How do we know what kind of higher dimensions should be used?

- Support Vector Machine



- Support Vector Machine

Polynomial kernel

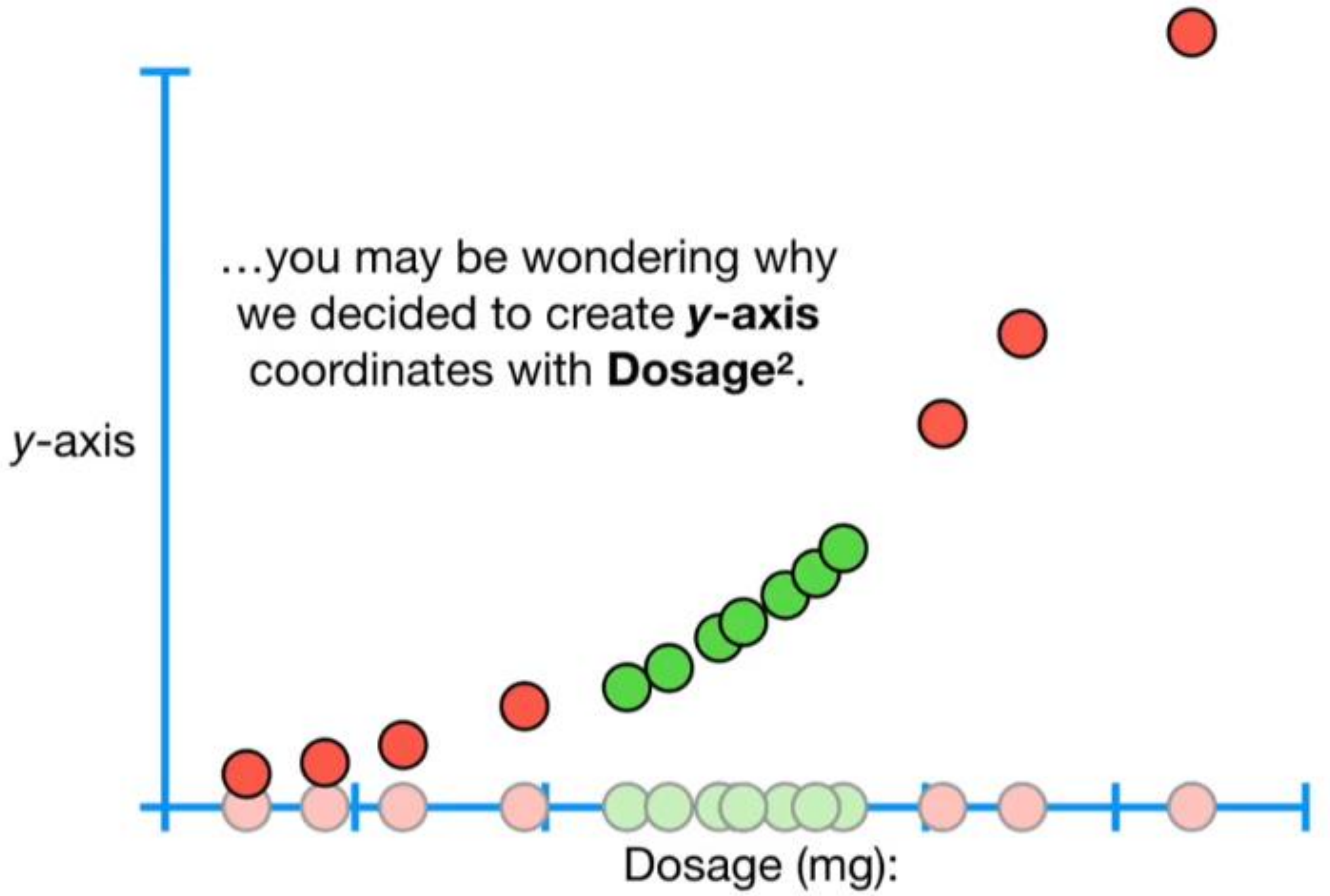
$$k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^d$$

Gaussian kernel

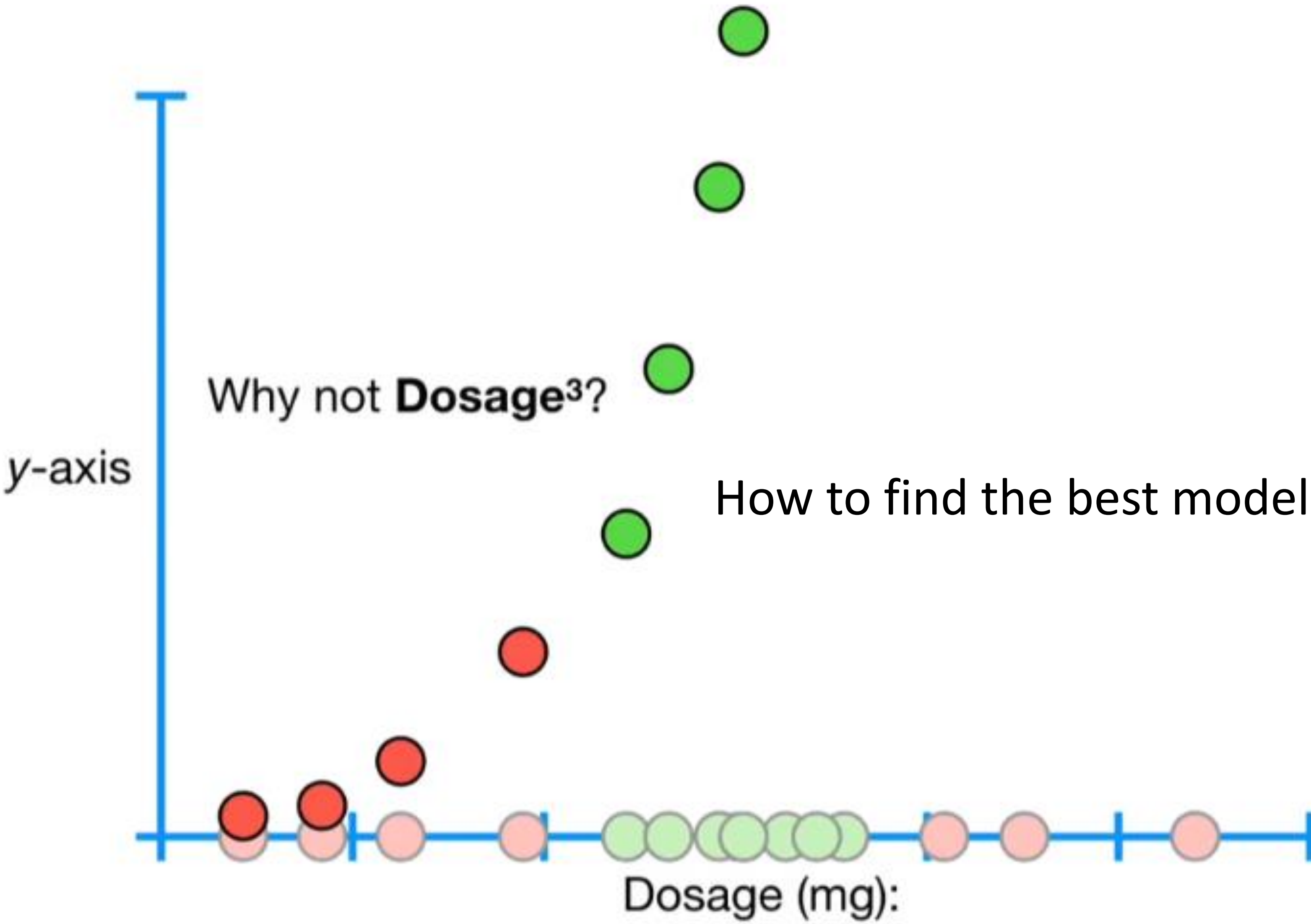
$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right)$$

Using these kernel techniques, we can get classifiers that work in higher dimensions

- Support Vector Machine



- Support Vector Machine



How to find the best model?

- Support Vector Machine

How do we find the best hyper-parameters?

- Support Vector Machine

1. We train models with different hyper-parameters
2. Compare their performance

Model Evaluation

$$\min_{w,b,\xi} \quad \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{n} \xi_i$$

Example: SVC with Polynominal kernel

$$k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^d$$

Setting A, d=2, C=1 $\hat{y} = f_A(x),$

Setting B, d=2, C=10 $\hat{y} = f_B(x),$

Say we have 145 mice for testing, we get two groups of predictions

Model A: $f_A(x_1), f_A(x_2), \ldots, f_A(x_{145})$

Model B: $f_B(x_1), f_B(x_2), \ldots, f_B(x_{145})$

- # Evaluation

Model A: $f_A(x_1), f_A(x_2), \ldots, f_A(x_{145})$

Ground truth

Prediction

	Cured	Not Cured
Cured	True Positive	False Positive
Not Cured	False Negative	True Negative

$$\text{Accuracy} = \frac{\text{Correct Prediction}}{\text{Total Prediction}}$$

$$= \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{False Positive} + \text{True Negative} + \text{False Negative}}$$

- # Evaluation

Results of model A, under split1 setting

		Ground truth: Cured	Ground truth: Not Cured
Prediction	Cured	True Positive 34	False Positive 23
	Not Cured	False Negative 12	True Negative 76

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{False Positive} + \text{True Negative} + \text{False Negative}}$$

$$= \frac{34+76}{34+23+76+12}$$

- # Evaluation

How do we know the performance of a model is good or bad?

Ground truth

Prediction

	Cured	Not Cured
Cured	True Positive 34	False Positive 23
Not Cured	False Negative 12	True Negative 76

Sensitivity or
True Positive Rate or
Recall

$$= \frac{\text{True Positive}}{\text{Actual Positive}} = \frac{\text{True Positive}}{\text{True Positive + False Negative}}$$

$$= \frac{34}{34+12}$$

• Evaluation

How do we know the performance of a model is good or bad?

		Ground truth	
		Cured	Not Cured
Prediction	Cured	True Positive 34	False Positive 23
	Not Cured	False Negative 12	True Negative 76

$$\text{Precision} = \frac{\text{True Positive}}{\text{Predicted Positive}} = \frac{\text{True Positive}}{\text{True Positive + False Positive}}$$

$$= \frac{34}{34+23}$$

Evaluation

How do we know the performance of a model is good or bad?

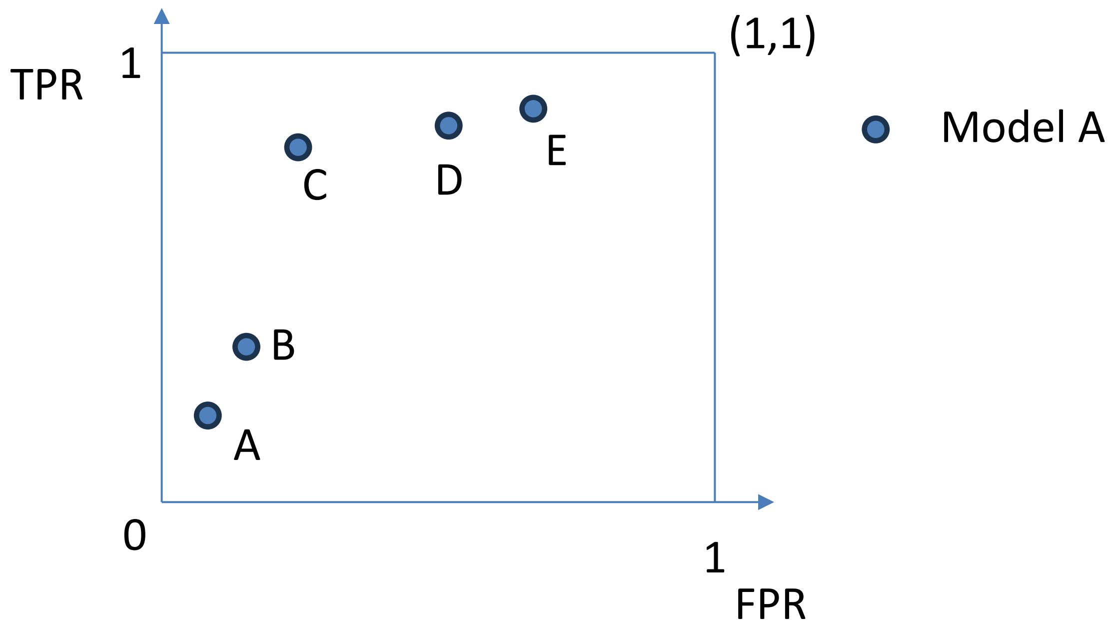
	Ground truth: Cured	Ground truth: Not Cured
Prediction: Cured	True Positive 34	False Positive 23
Prediction: Not Cured	False Negative 12	True Negative 76

$$\text{False Positive Rate} = \frac{\text{False Positive}}{\text{Actual Negative}} = \frac{\text{False Positive}}{\text{False Positive + True Negative}}$$

$$= \frac{23}{23+76}$$

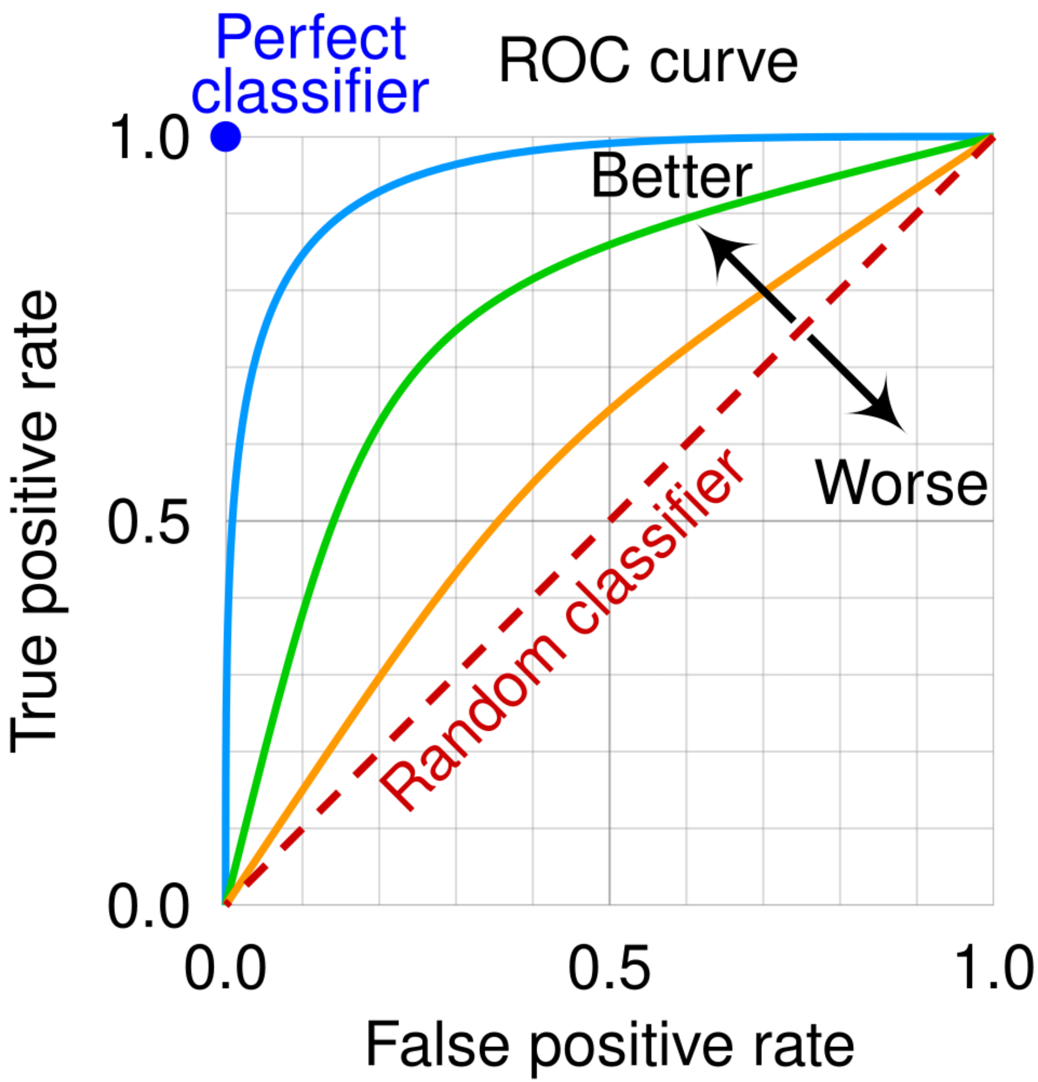
- **Evaluation**

Receiver Operating Characteristic curve (ROC curve)



Closer to left upper corner, better performance

ROC curve


Perfect classifier
ROC curve
1.0
Better
Worse
0.5
0.0
Random classifier
True positive rate
0.0
0.5
1.0
False positive rate

Theorem (Random classifier)

A random classifier lies anywhere on the line TPR = FPR.

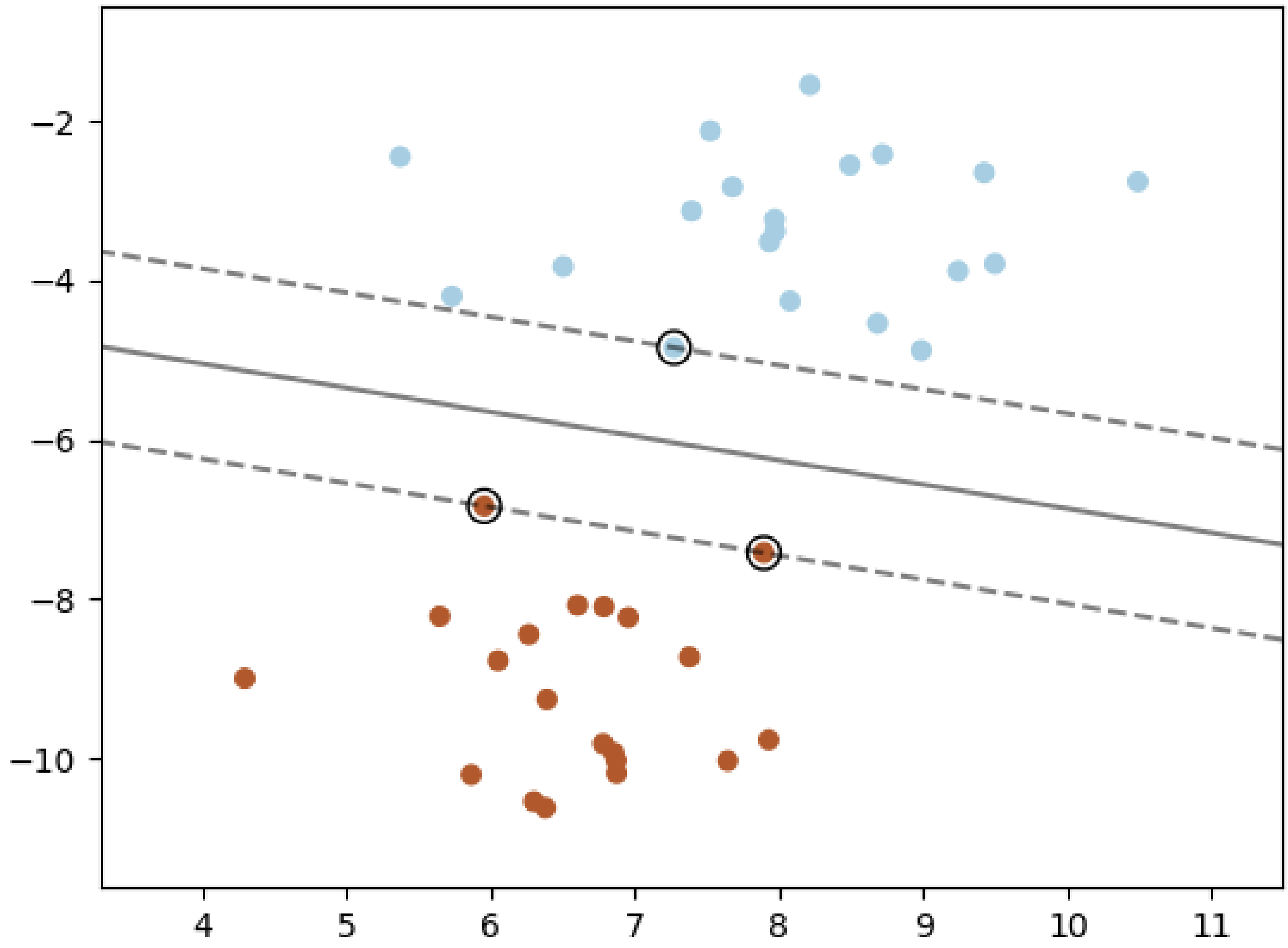
Proof: Let n be the number of participants/test subjects that our classifier is tasked with predicting whether they are cured. Let p denote the probability that a person is cured.

Given a random classifier, that randomly predicts a person is cured with probability q and predicts a person is not cured with probability $1 - q$.

Prove that $E[TPR] = E[FPR]$ holds.

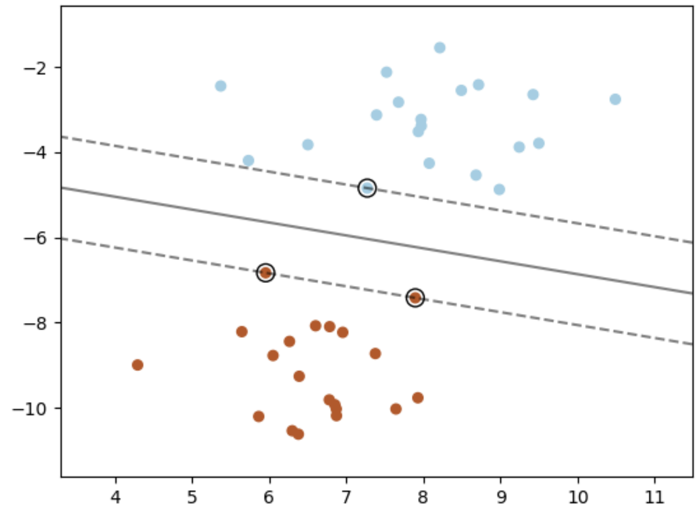
Hint: First find the values for $E[TP], E[FP], E[FN], E[TN]$; then use the formulas $\text{FPR} = \frac{FP}{FP+TN}$, $\text{TPR} = \frac{TP}{TP+FN}$.

SVM Example



SVM Example

conda install python=3.10
conda install matplotlib
conda install scikit-learn



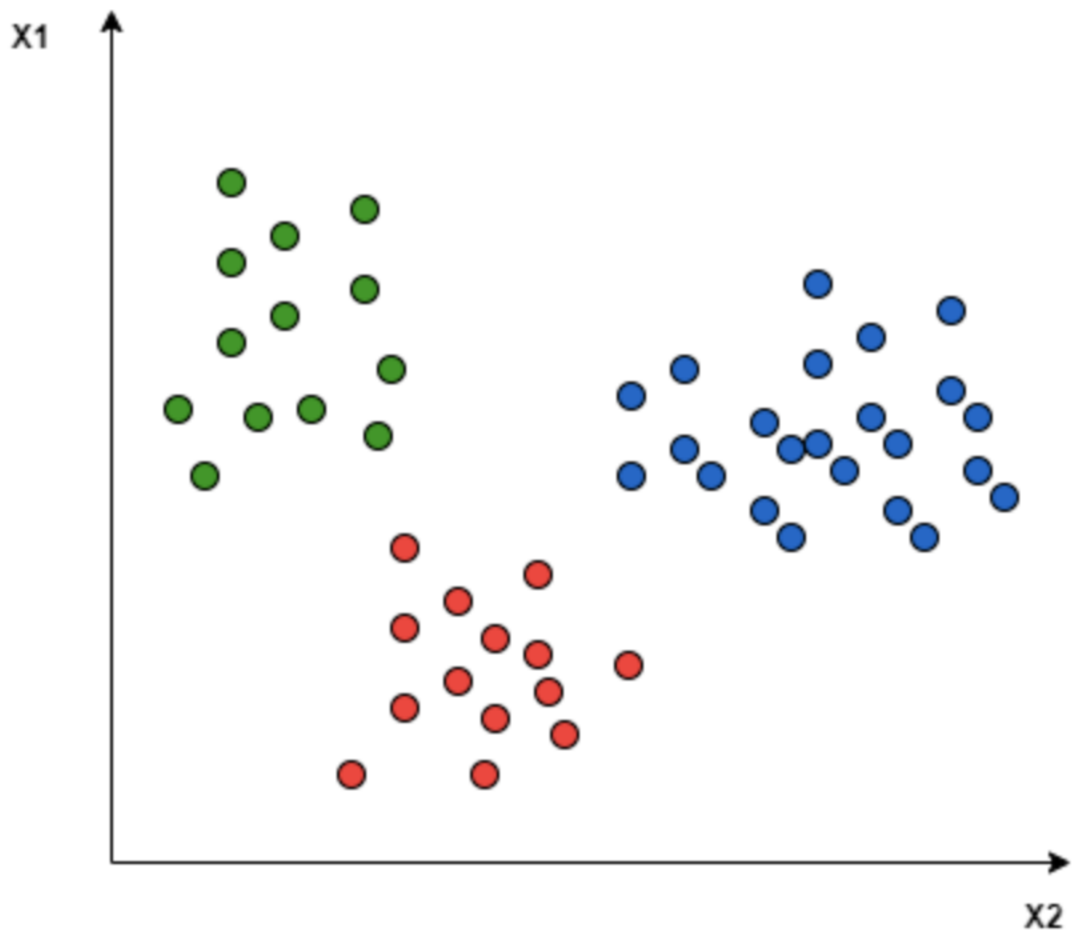

```python
import matplotlib.pyplot as plt

from sklearn import svm
from sklearn.datasets import make_blobs
from sklearn.inspection import DecisionBoundaryDisplay

# we create 40 separable points
X, y = make_blobs(n_samples=40, centers=2, random_state=6)
plt.scatter(X[:, 0], X[:, 1], c=y, s=30, cmap=plt.cm.Paired)

# fit the model, don't regularize for illustration purposes
clf = svm.SVC(kernel="linear", C=1000)
clf.fit(X, y)
```

https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC

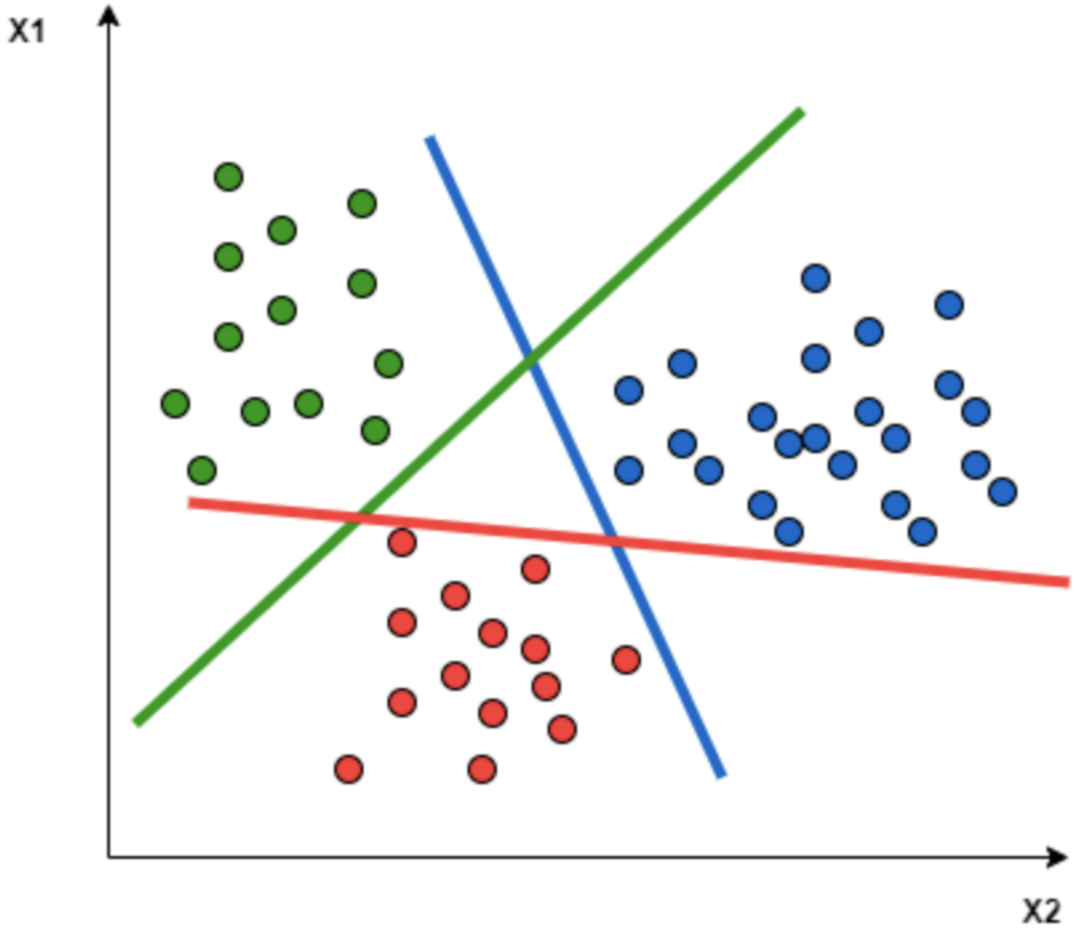
Example

```python
# plot the decision function
ax = plt.gca()
DecisionBoundaryDisplay.from_estimator(
    clf,
    X,
    plot_method="contour",
    colors="k",
    levels=[-1, 0, 1],
    alpha=0.5,
    linestyles=["--", "-", "--"],
    ax=ax,
)
# plot support vectors
ax.scatter(
    clf.support_vectors_[:, 0],
    clf.support_vectors_[:, 1],
    s=100,
    linewidth=1,
    facecolors="none",
    edgecolors="k",
)
plt.show()
```

Can SVM do multiclass classification?

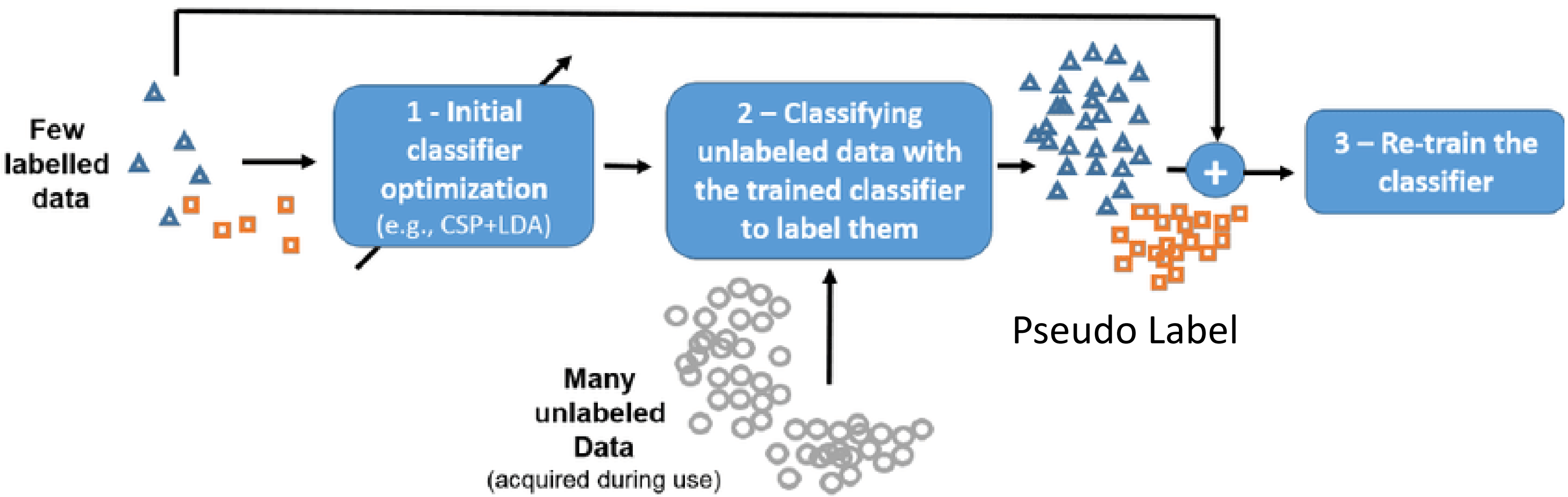


Can SVM do multiclass classification?



Semi-supervised Learning

Semi-supervised Learning



- Acknowledgement

Part of the material are from Josh Starmer

https://www.youtube.com/watch?v=efR1C6CvhmE