ECS171: Machine Learning

L12: Support vector machines (SVM)

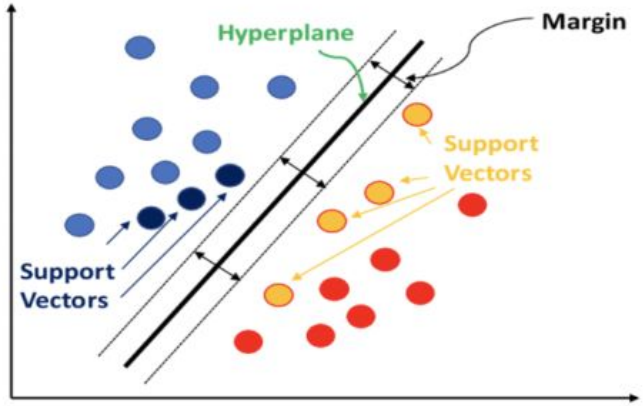
Instructor: Prof. Maike Sonnewald
TAs: Pu Sun & Devashree Kataria

Intended Learning Outcomes

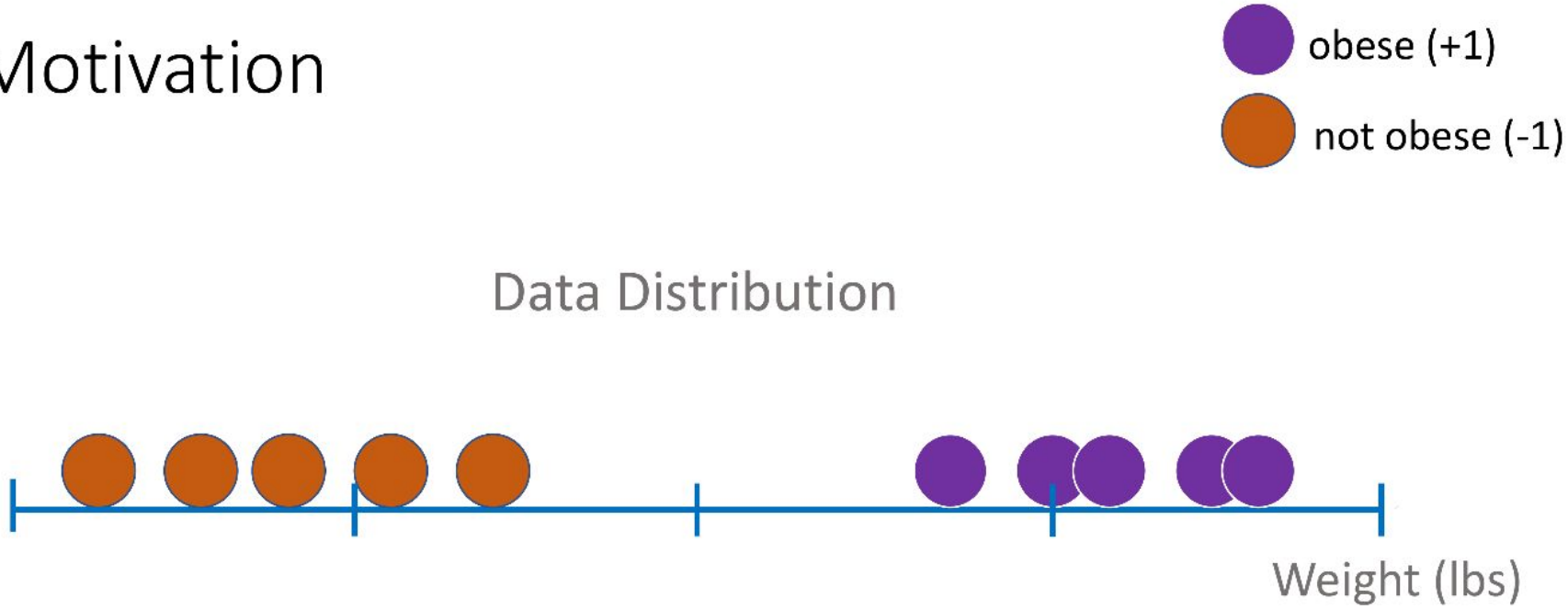
- Describe how a Support Vector Machine differs from previous methods like 1-layer or multi-layer neural nets
- Describe the role of the ‘Support Vectors’
- For a linear Support Vector Machine describe and apply how to calculate the margin between the bounding lines
    - Apply the Hinge cost function and weight update rule
- Describe the non-linear Support Vector Machine mappings to higher dimensions
    - Describe how the cost function is modified
- Describe different kernels and the kernel trick

Basic idea of support vector machines (SVM)

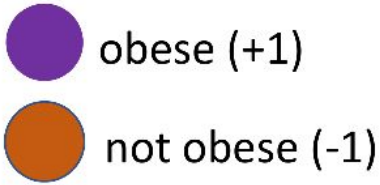
- Supervised learning model for complex classification, regression, and outlier detection problems
  - Uses optimal data transformations to determine boundaries between data points based on predefined classes, labels, or outputs
- Just like 1-layer or multi-layer neural nets
  - Optimal hyperplane for linearly separable patterns
- Extend to patterns that are not linearly separable by transformations of original data to map into new space – the **Kernel function**
- SVM algorithm for pattern recognition

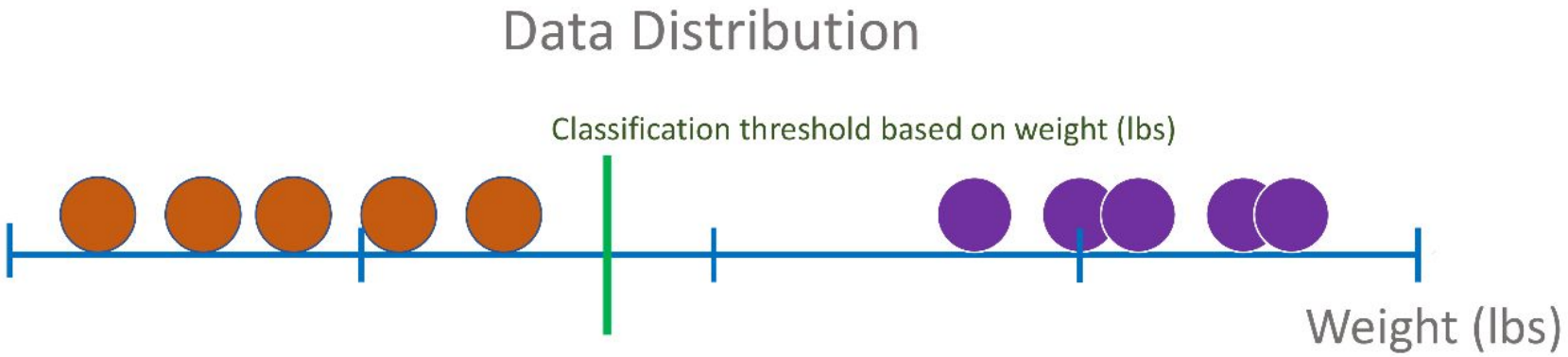


Motivation

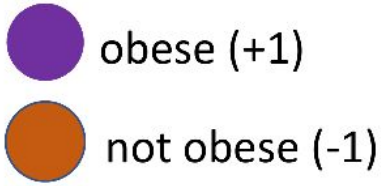


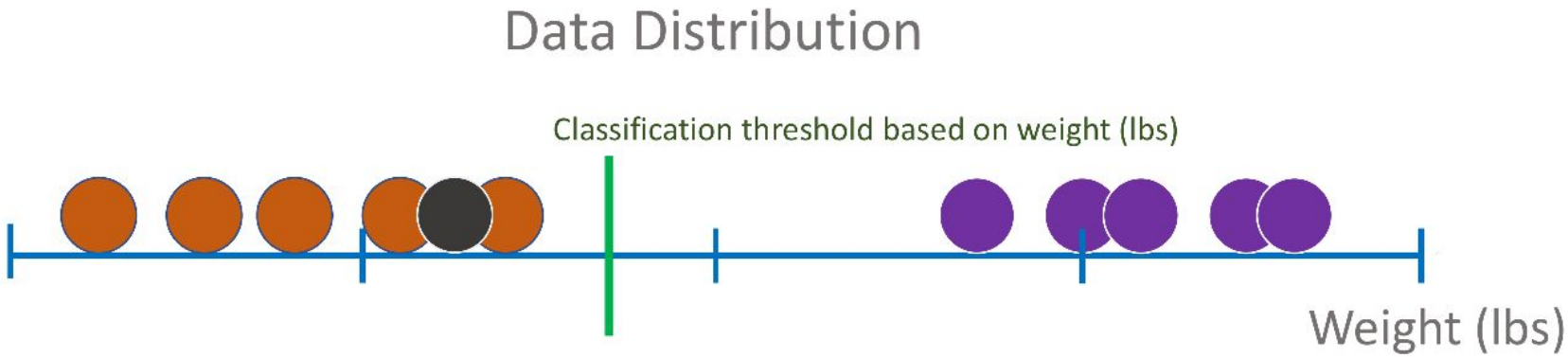
Motivation


obese (+1)
not obese (-1)

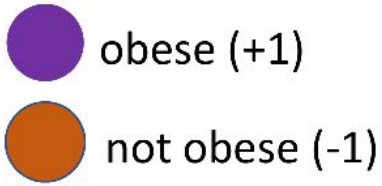

Data Distribution
Classification threshold based on weight (lbs)
Weight (lbs)

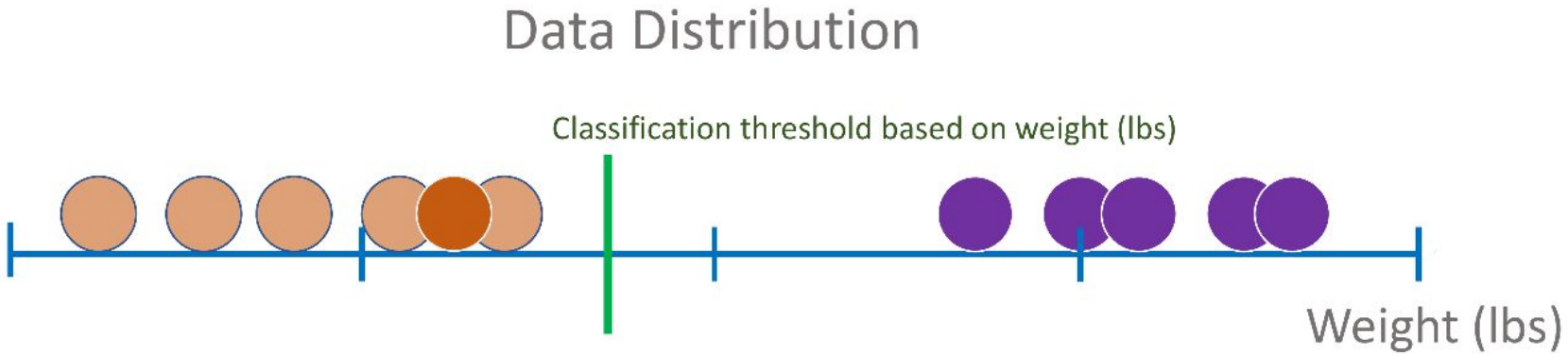
Motivation


obese (+1)
not obese (-1)

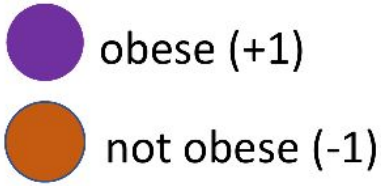

Data Distribution
Classification threshold based on weight (lbs)
Weight (lbs)

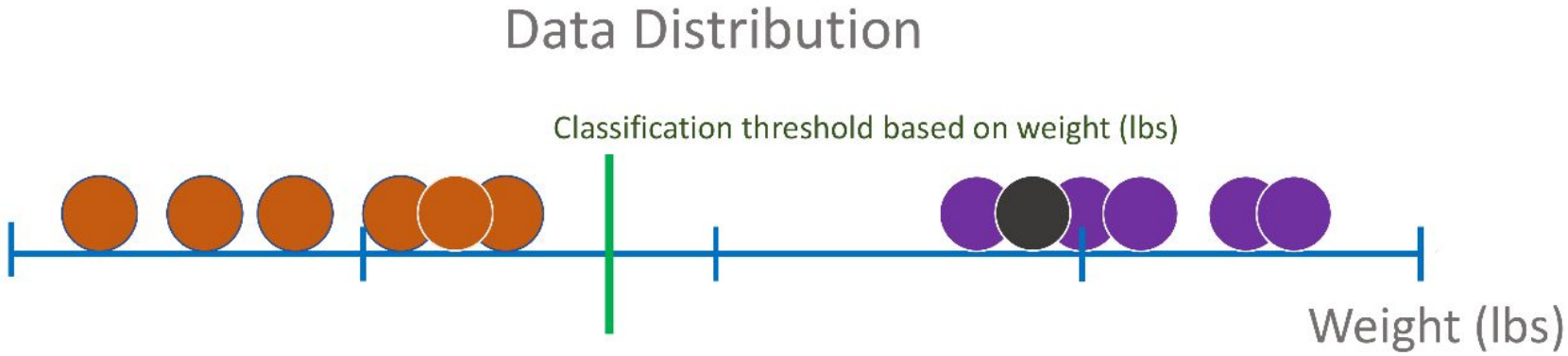
Motivation


obese (+1)
not obese (-1)

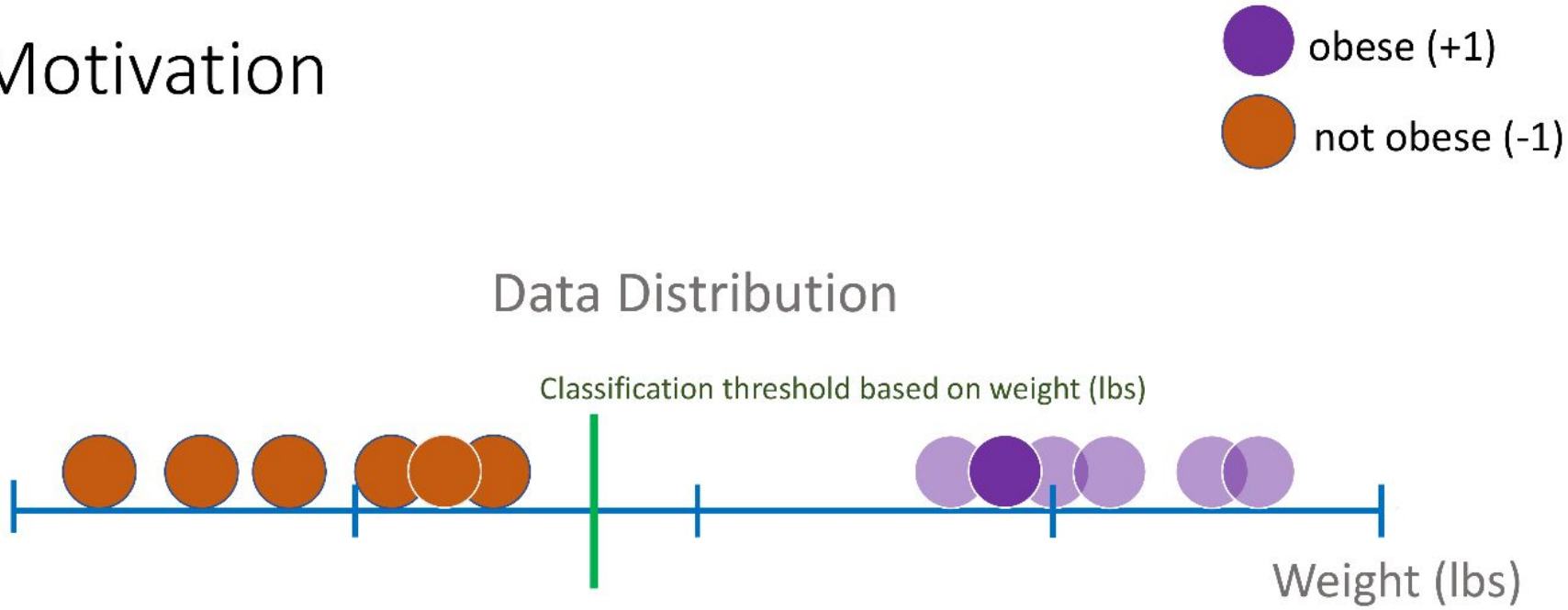

Data Distribution
Classification threshold based on weight (lbs)
Weight (lbs)

Motivation

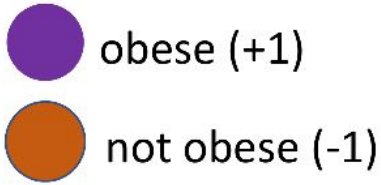

obese (+1)
not obese (-1)

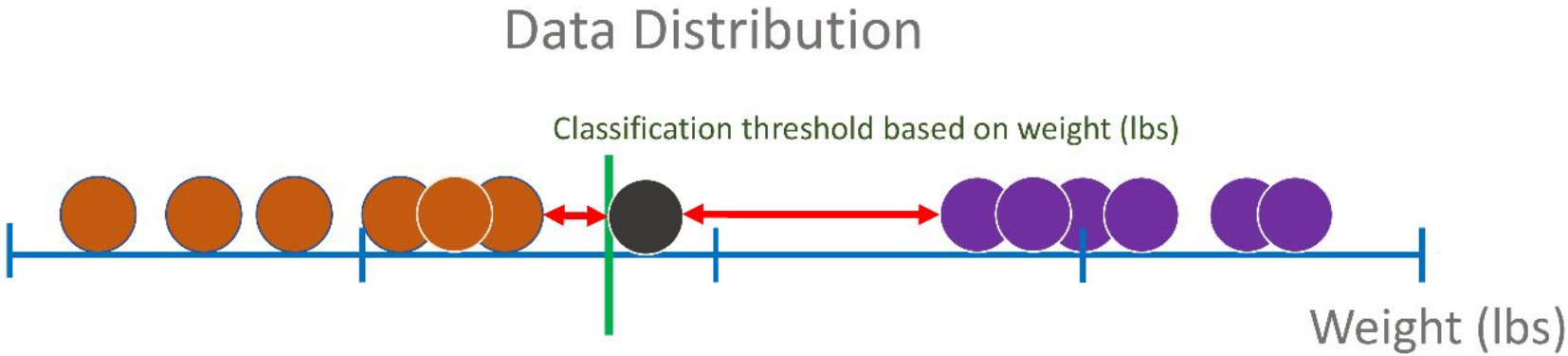

Data Distribution
Classification threshold based on weight (lbs)
Weight (lbs)

Motivation

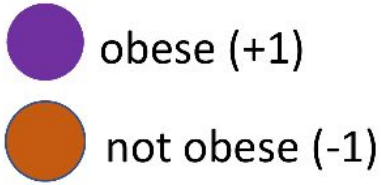

obese (+1)
not obese (-1)
Data Distribution
Classification threshold based on weight (lbs)
Weight (lbs)

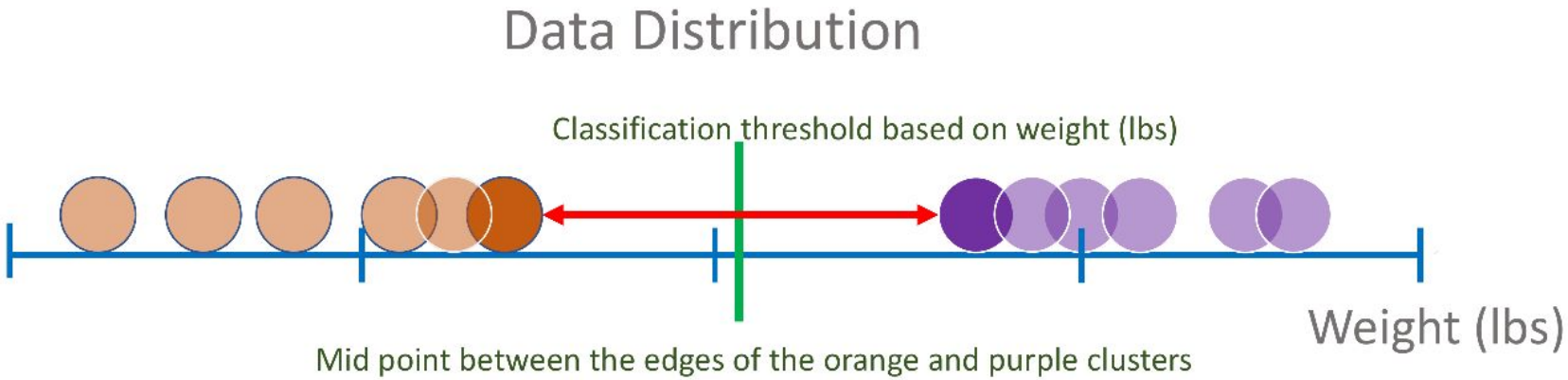
Motivation


obese (+1)
not obese (-1)

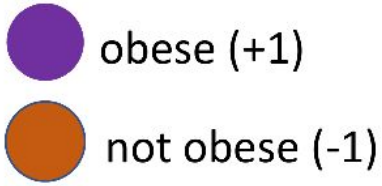

Data Distribution
Classification threshold based on weight (lbs)
Weight (lbs)

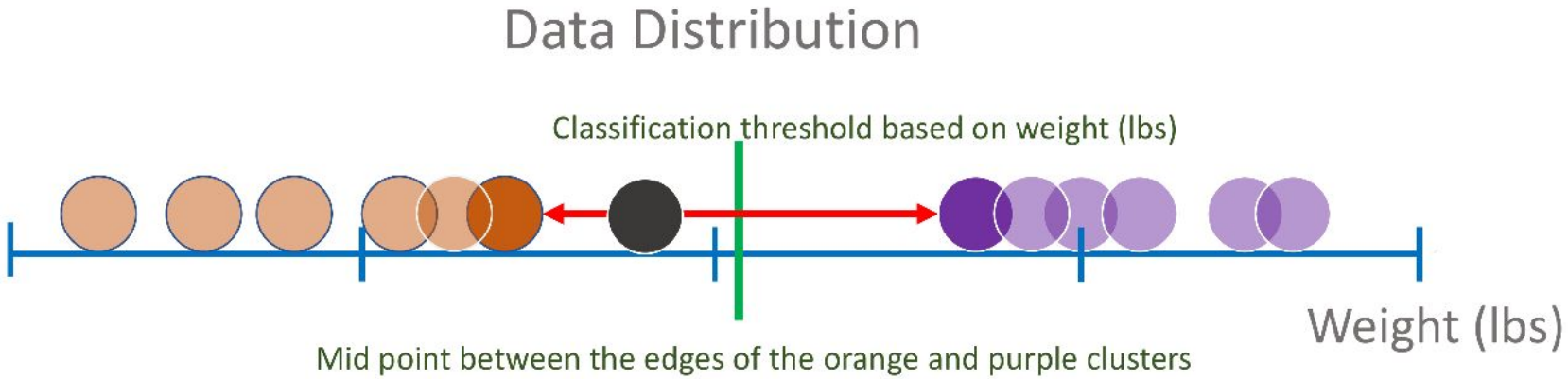
Motivation


obese (+1)
not obese (-1)

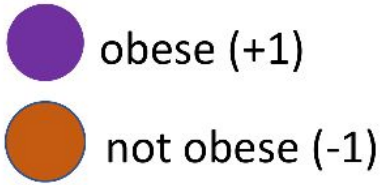

Data Distribution
Classification threshold based on weight (lbs)
Weight (lbs)
Mid point between the edges of the orange and purple clusters

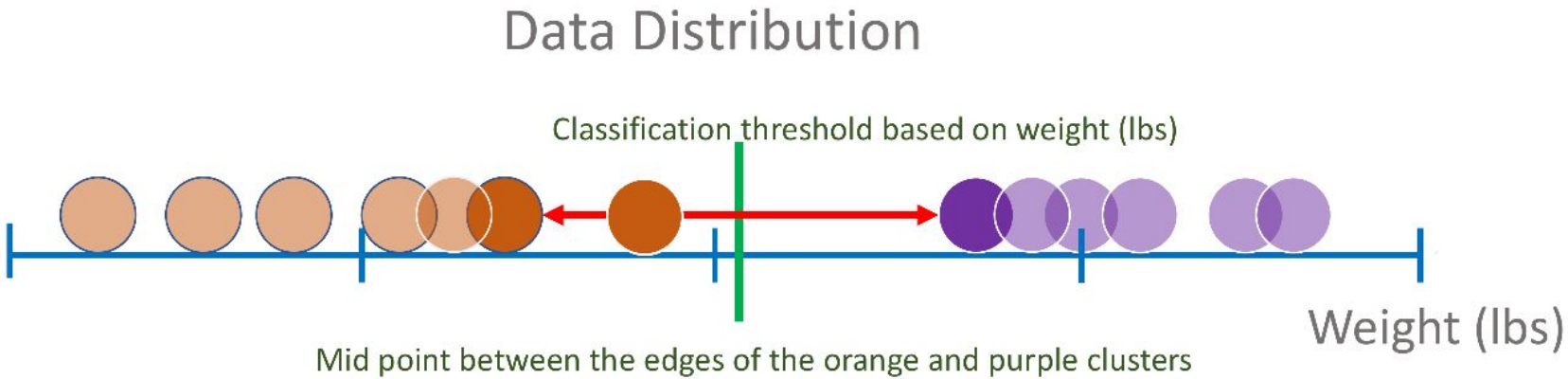
Motivation


obese (+1)
not obese (-1)

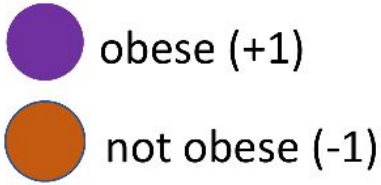

Data Distribution
Classification threshold based on weight (lbs)
Weight (lbs)
Mid point between the edges of the orange and purple clusters

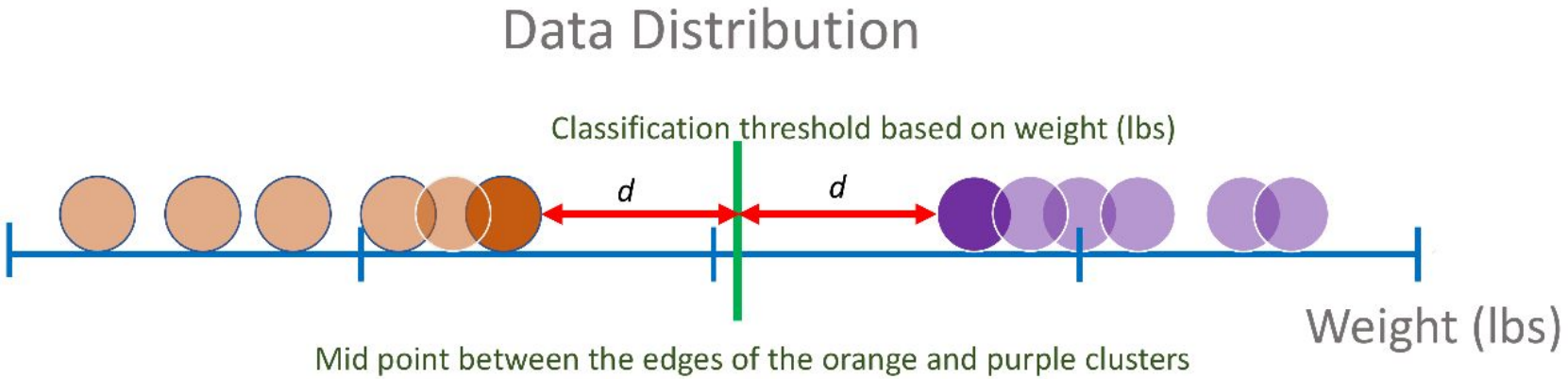
Motivation


obese (+1)
not obese (-1)


Data Distribution
Classification threshold based on weight (lbs)
Weight (lbs)
Mid point between the edges of the orange and purple clusters

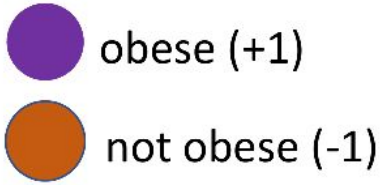
Margin

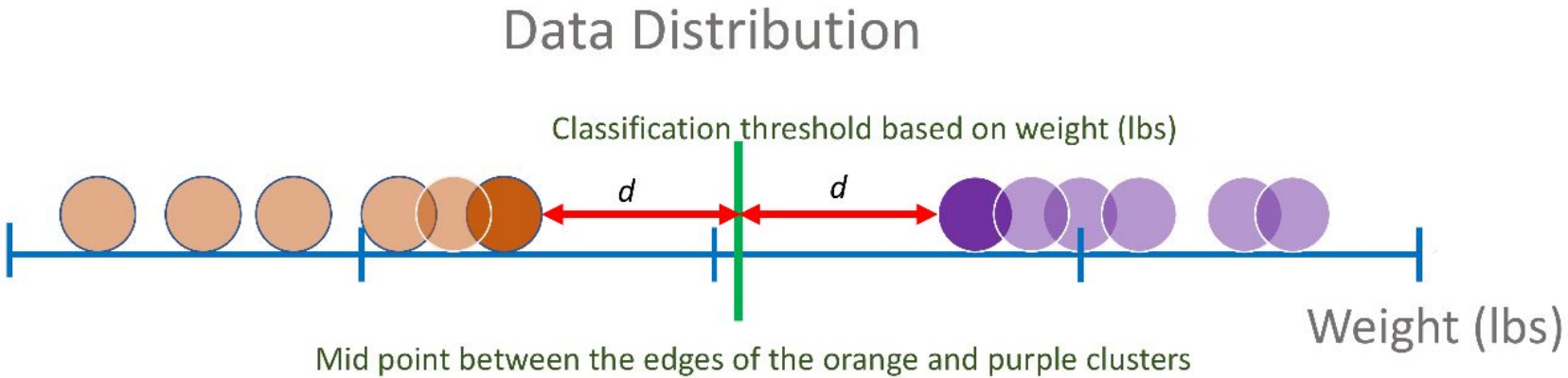




d: shortest distance between the observations in each cluster and the classification threshold.
d is called “margin”.

Maximal Margin Classifier





d is called “margin”.

The threshold that gives the largest margin to make classifications for both clusters, is called “**Maximal Margin Classifier**”.

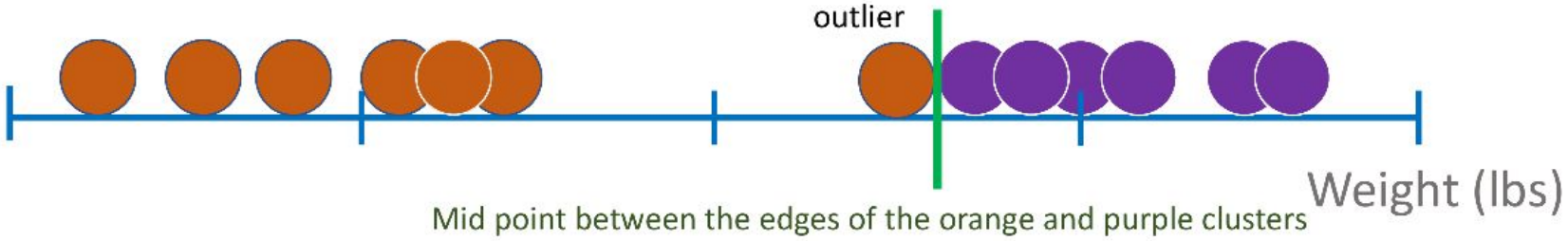
Maximal Margin Classifier : Outlier Sensitivity

obese (+1)

not obese (-1)

Training Data

Data Distribution



The maximal margin classifier is very close to the ”obese” observations and very far from the majority of “not obese” observations.

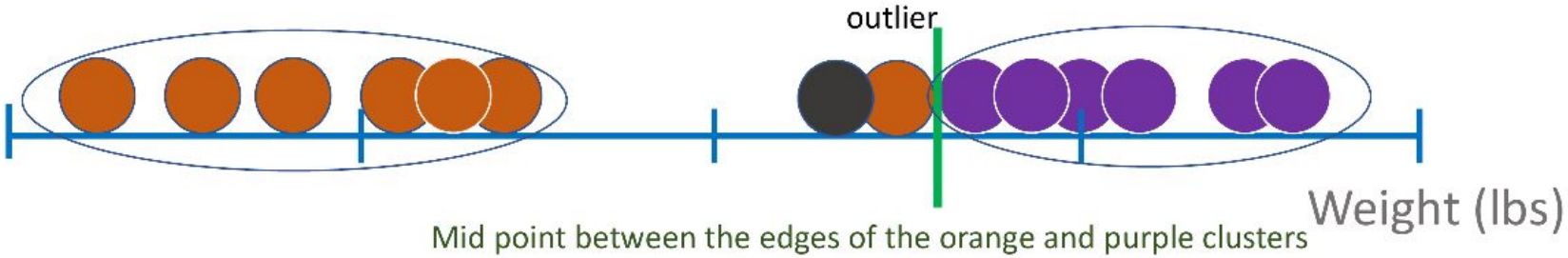
Maximum Margin Classifier : Outlier Sensitivity in Training Data

obese (+1)

not obese (-1)

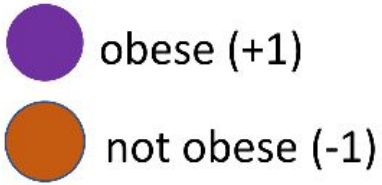
Training Data

Data Distribution



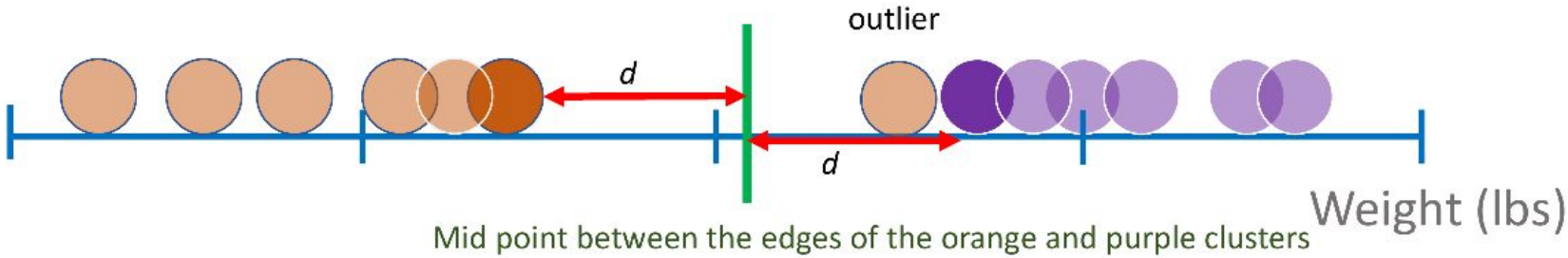
The maximal margin classifier is very close to the ”obese” observations and very far from the majority of “not obese” observations.
Here, the threshold is very sensitive to training data.

Making a threshold not very sensitive to outliers



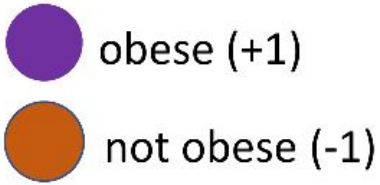
Training Data

Data Distribution



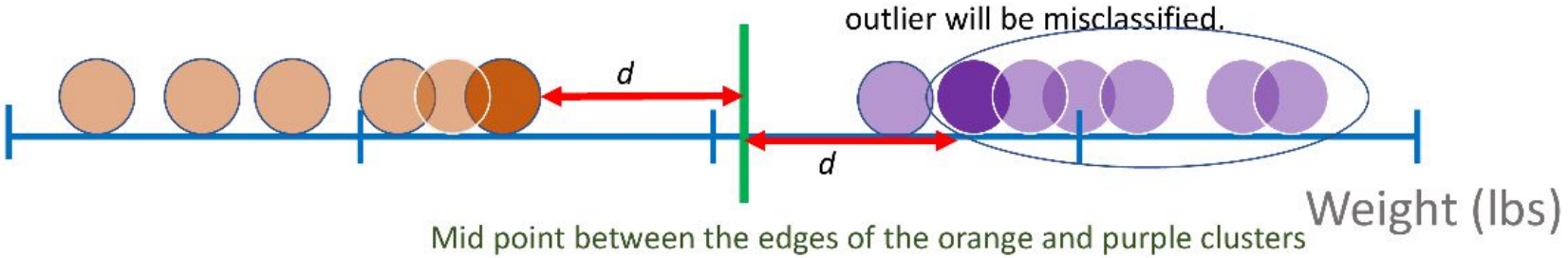
- We must allow misclassifications to find a threshold not very sensitive to outliers.
- Why is the outlier misclassified in this case?

Making a threshold not very sensitive to outliers: Bias-Variance Trade-off



Training Data

Data Distribution



- We must allow misclassifications to find a threshold not very sensitive to outliers. Here, the threshold is less sensitive to the training data.
- Why is the outlier misclassified in this case? Because it is closer to the "obese" observations.

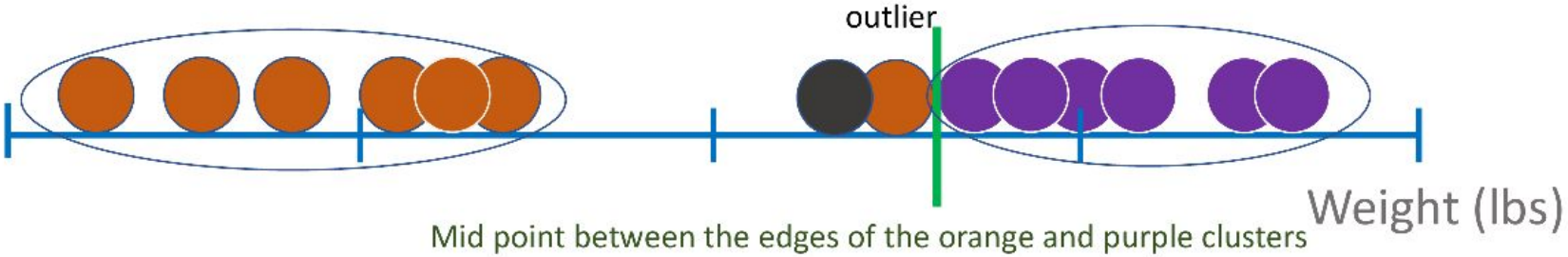
Making a threshold very sensitive to outliers (Hard Margin)

obese (+1)

not obese (-1)

Training Data

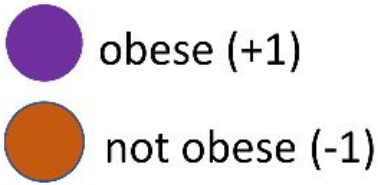
Data Distribution



The maximal margin classifier is very close to the ”obese” observations and very far from the majority of “not obese” observations.

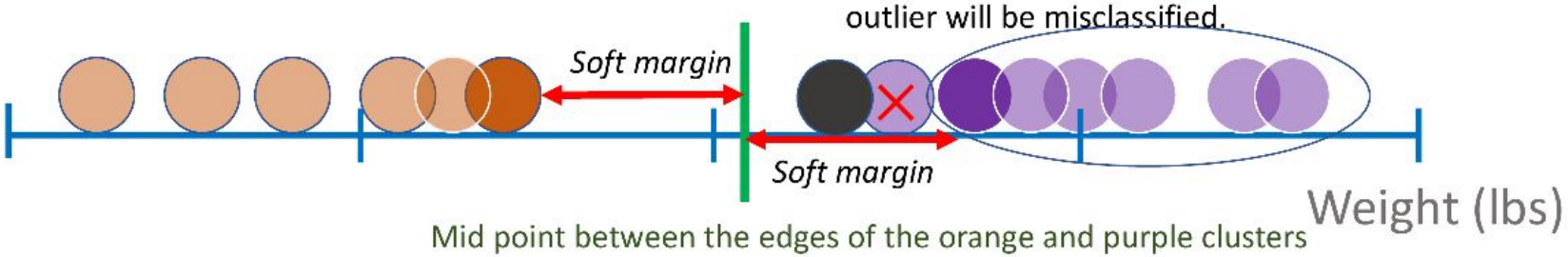
Here, the threshold is very sensitive to training data – Low Bias, High Variance.

Making a threshold less sensitive to outliers (Soft Margin)



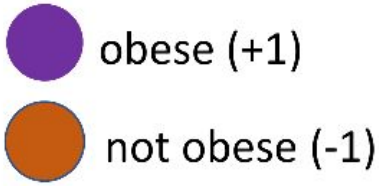
Training Data

Data Distribution



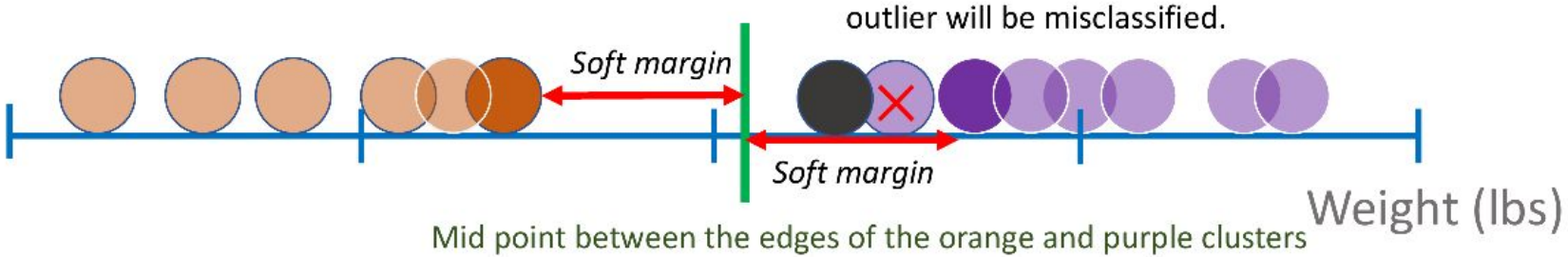
- We must allow misclassifications to find a threshold not very sensitive to outliers. Here, the threshold is less sensitive to the training data.
- Here, the threshold is less sensitive to the training data – higher bias, lower variance
- Here, the distance between the observations and the threshold is called "Soft Margin".

How to determine the soft margin?



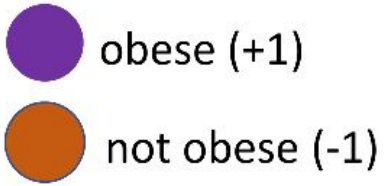
Training Data

Data Distribution

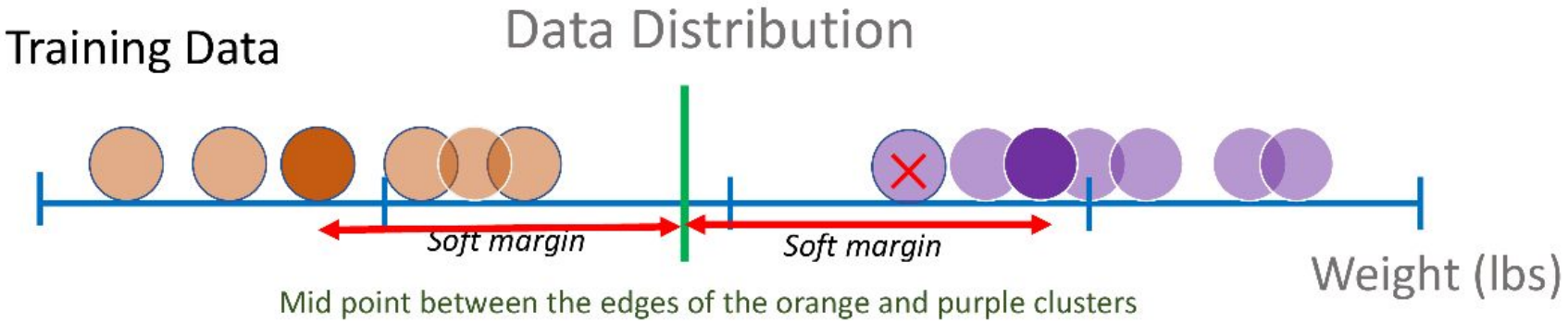


- We must allow misclassifications to find a threshold not very sensitive to outliers. But too many misclassifications is also bad!
- Here, the threshold is less sensitive to the training data – higher bias, lower variance
- Here, the distance between the observations and the threshold is called "Soft Margin".

How to determine the soft margin?



Soft margin is controlled by a parameter
denoted as 'C' that can be determined using CV.
Parameter 'C' is a regularization parameter.



- Which soft margin is better to achieve a generalized model?
  - Cross Validation is used to determine how many misclassifications and observations to allow inside of the Soft Margin to get the best Soft Margin Classifier. In other words, CV is used to tune the hyperparameter 'C'.

Soft Margin Classifier/Support Vector Classifier

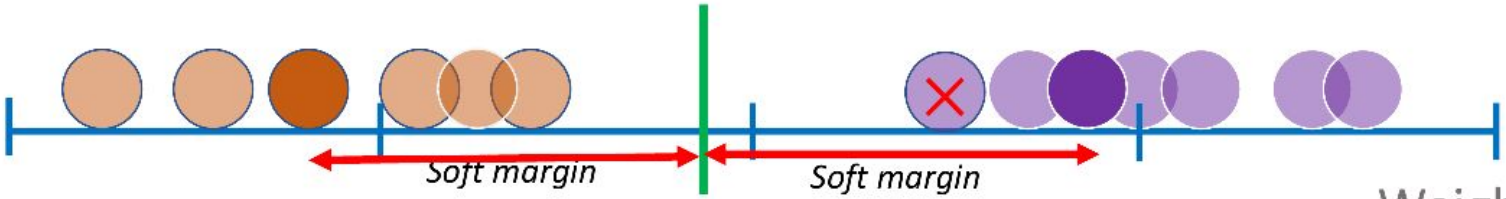
obese (+1)

not obese (-1)

Soft margin is controlled by hyperparameter ‘C’.
Parameter ‘C’ is a regularization parameter.

Training Data

Data Distribution



Weight (lbs)

Mid point between the edges of the orange and purple clusters

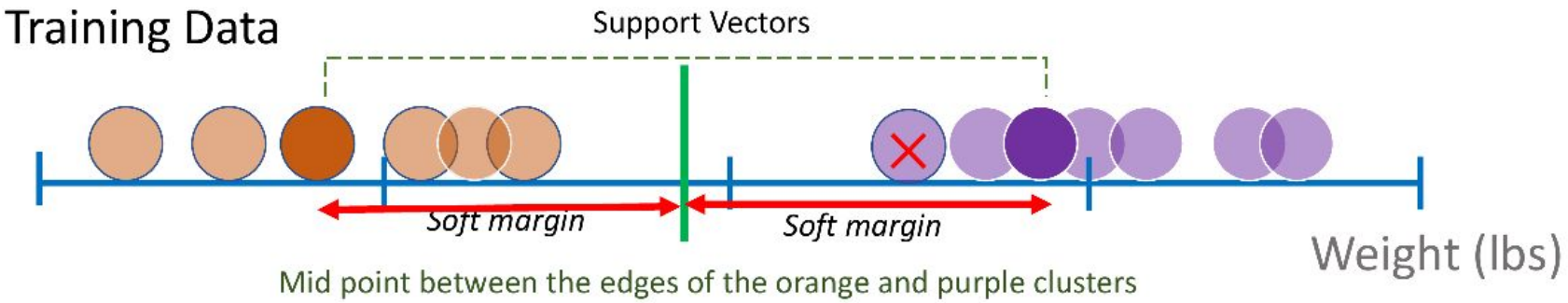
- When “Soft Margin” is used to determine the location of the decision boundary, this is called a “Soft Margin Classifier”.
- If this is the best soft margin determined by CV, then the model allows one misclassification and 4 observations that are correctly classified to be within the Soft Margin.

Soft Margin Classifier/Support Vector Classifier: Support Vectors

obese (+1)

not obese (-1)

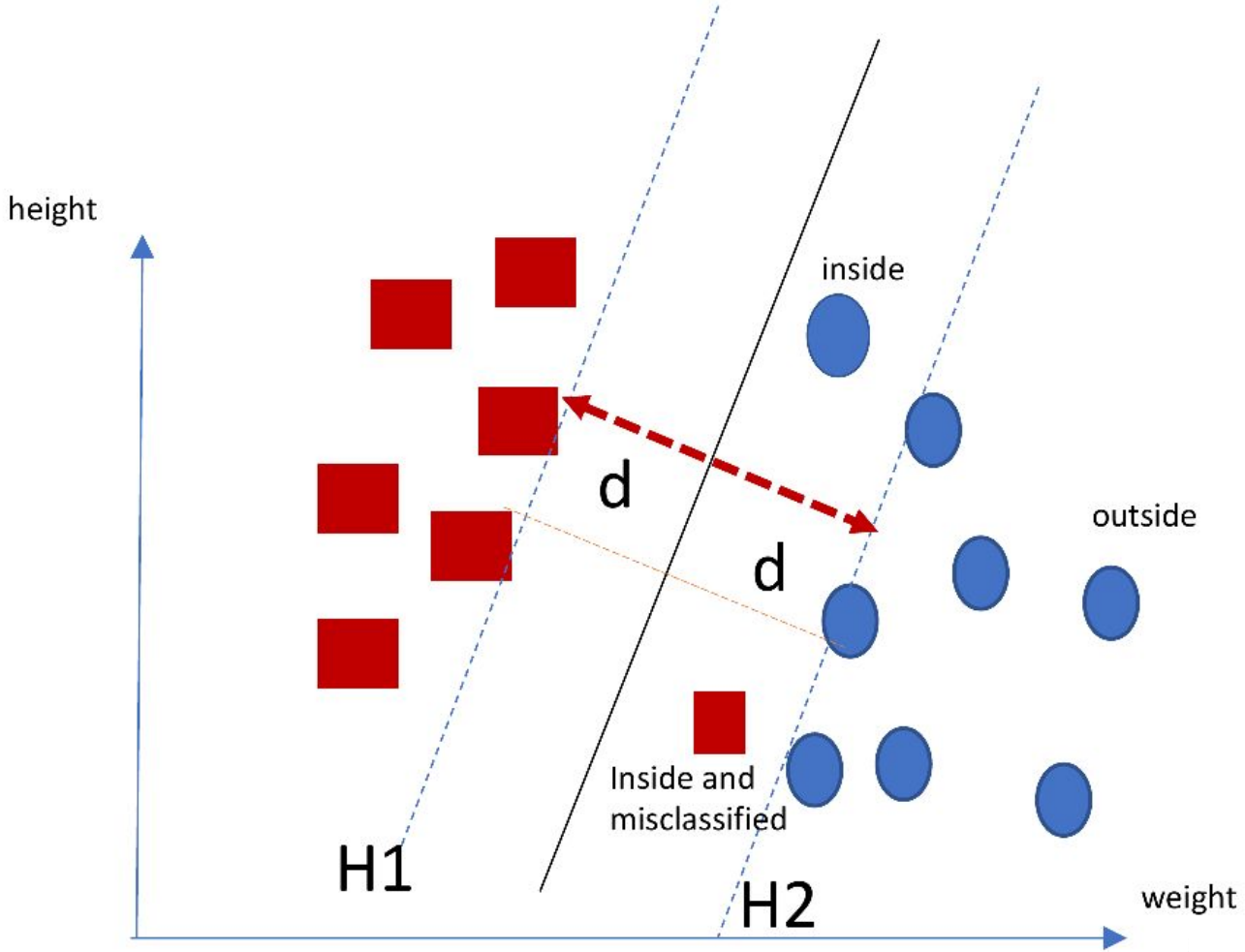
Soft margin is controlled by hyperparameter ‘C’.
Parameter ‘C’ is a regularization parameter.



- Observations on the edge and within the Soft margin are called “Support Vectors”.
- Support Vector Classifier for a 1D data is a point.

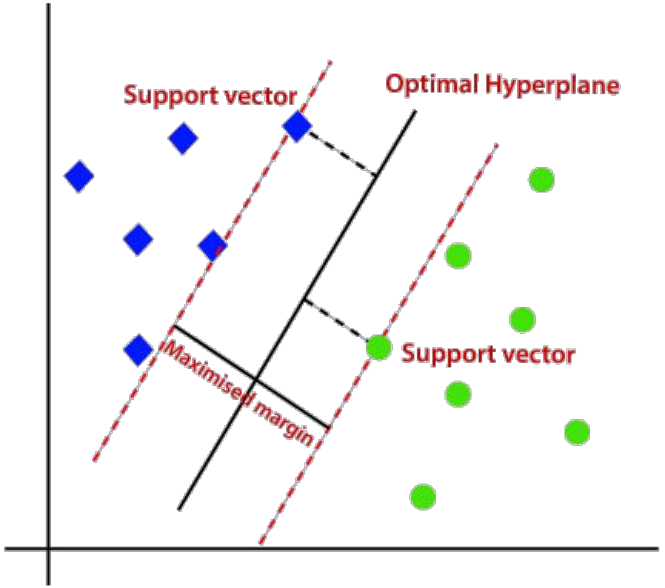
2D Data

Support Vector Classifier for a 2D data is a line.



Support Vector Classifier or Linear SVM

- Only ‘support vectors’ have a significant impact on model training
- We can remove the non-support vectors without impacting the model



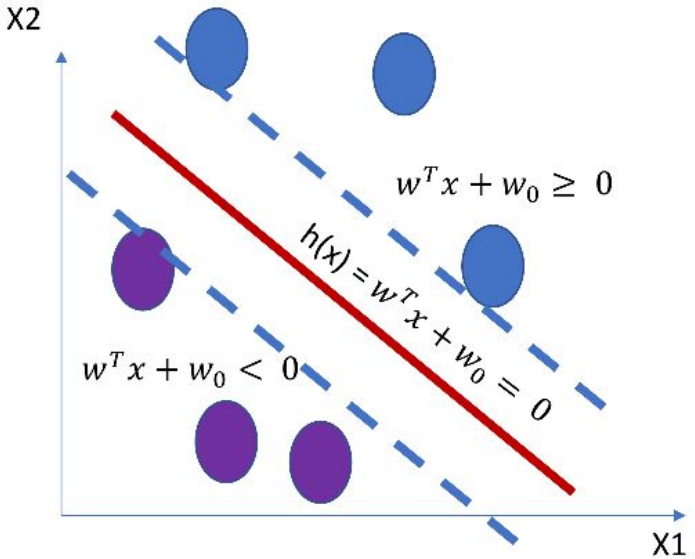
Support Vector Classifier or Linear SVM

Given a learned weight vector w and bias w_0 and an input vector x, the SVM 'hypothesis function' h for our line is:

$$h(x) = w^T x + w_0 : \quad \begin{matrix} +1 & \text{if } h(x) \geq 0 \\ -1 & \text{if } h(x) < 0 \end{matrix}$$

Where the lower and upper bounds are shown in the stippled blue lines (right)

Maximizing the margin means a more generalized model

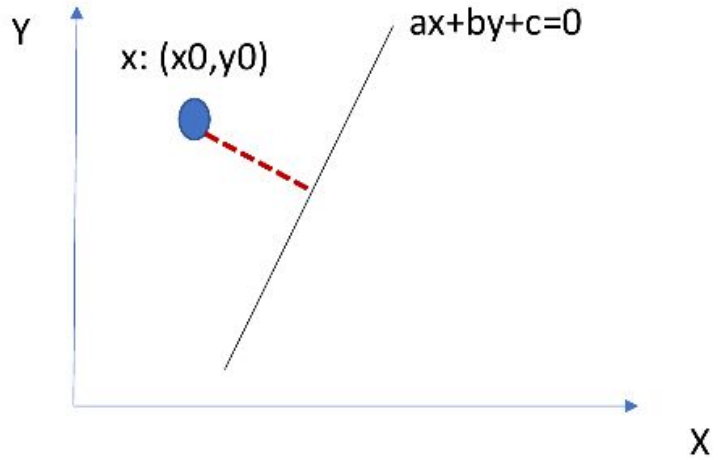


Determining the distance of a point to the line

The numerator $|ax_0+by_0+c|$ is the absolute value of the line equation evaluated at the point (x_0, y_0)

Distance of a point x_0 from the line:

$$\frac{|\, ax_0 + by_0 + c \,|}{\sqrt{a^2 + b^2}}$$



Determining the margin between the bounding lines

The decision boundary H is:

$$\mathcal{H} : \mathrm{h}(x) = w^T \mathrm{x} + w_0 = 0$$

The distance (d) or margin from the line for a given set of points can be written as:

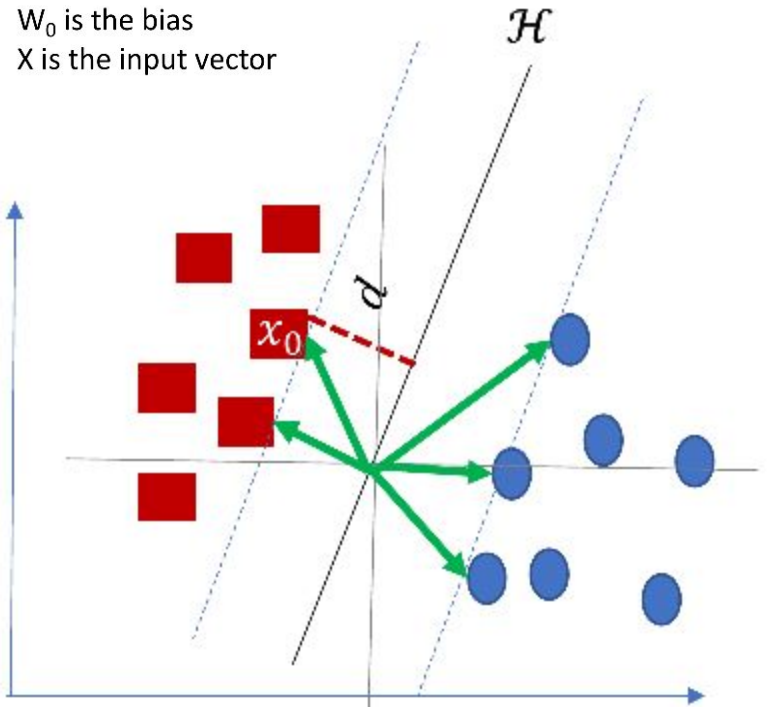
$$d_{\mathcal{H}}(x_0) = \frac{|w^T x_0 + w_0|}{||w||^2} = \frac{|h(x)|}{||w||^2}$$

Where we have, as for the single point:

$||w||$ = sqrt(w1^2 + w2^2 + ... + wn^2)

$||w||^2$ = w1^2 + w2^2 + ... + wn^2

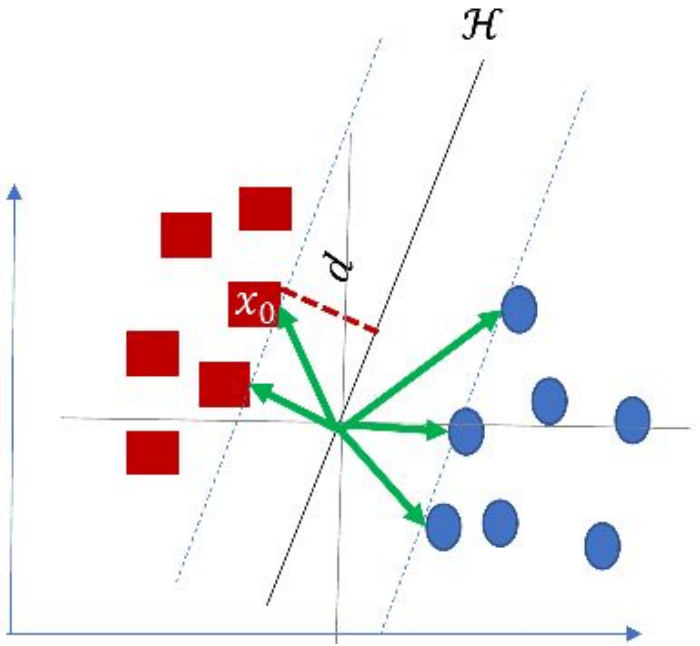
W is the weight vector
W_0 is the bias
X is the input vector



Determining the distance to the line

- We want a classifier (linear separator) with as big a margin (d) as possible
- In order to maximize the margin, we need to minimize ||w||
- With the condition that there are no data-points between the bounding lines

$$d_{\mathcal{H}}(x_0) = \frac{|w^T x_0 + w_0|}{||w||^2} = \frac{|h(x)|}{||w||^2}$$



Maximising the distance to the line

$$d_{\mathcal{H}}(x_0) = \frac{|w^T x_0 + w_0|}{||w||^2} = \frac{|h(x_0)|}{||w||^2} = \frac{1}{||w||^2}$$

$$\xrightarrow{\text{yields}} \textit{distance between clusters' edges} = \frac{2}{||w||^2}$$

The task is to determine how to modify ||w||. Note that:

$$\text{Maximizing } \frac{2}{||w||^2} \equiv \text{Minimizing } \frac{1}{2}||w||^2$$

Margins are often defined as having a distance equal to 1 from the data-separating-hyperplane

Maximising the distance to the line: Cost function

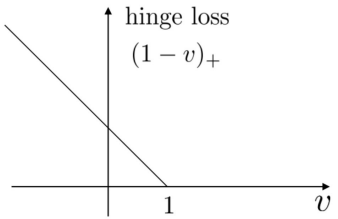
For our cost function, a desirable property is that not all points need to satisfy the requirement

We can use a ‘Soft margin classifier’ : This change allows some points in the training data to violate the separating line

Here we use the ‘hinge loss’:

$$c(x, y, f(x)) = \begin{cases} 0, & \text{if } y * f(x) \geq 1 \\ 1 - y * f(x), & \text{else} \end{cases}$$

y: actual label
f(x): predicted label



With number of observations (n) and the number of features (m) we have:

$$SVM\ Cost = min_w \frac{1}{2} \sum_{i=1}^{m} w_i^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i * f(x_i))$$

Here, the C term is a regularization parameter where C >=0

- Determining C helps to identify the support vectors which determine the maximal margin

Gradient of the Cost Function

$$SVM\ Cost = min_w \frac{1}{2}\sum_{i=1}^{m} w_i^2 + C\sum_{i=1}^{n} \max\left(0, 1 - y_i * f(x_i)\right)$$

$$SVM\ Cost = min_w \frac{1}{2}\sum_{i=1}^{m} w_i^2 + C\sum_{i=1}^{n} \xi(i) \qquad \xi(i) \geq 0 \text{ for } i = 1,\ldots,n$$

The hinge loss here is represented by the slack variables ξ(i).

$$SVM\ Cost = min_w \frac{1}{2}\sum_{i=1}^{m} w_i^2 + C\sum_{i=1}^{n} \max(0, 1 - y_i * (w^T x_i + w_0))$$

Next, we take partial derivatives wrt the weights to find the gradients

Using the gradients, we update the weights or parameters of the model.

We minimize the Cost with respect to w and w_0 and maximize it with respect to C, to lower misclassifications.

Weight Update Rule

$$c\big(x, y, f(x)\big) = \max(0, 1 - y * f(x))$$

$$c(x, y, f(x)) = \begin{cases} 0, & \text{if } y * f(x) \geq 1 \\ 1 - y * f(x), & \text{else} \end{cases}$$

For **correctly** classified samples given the following and α as the learning rate:

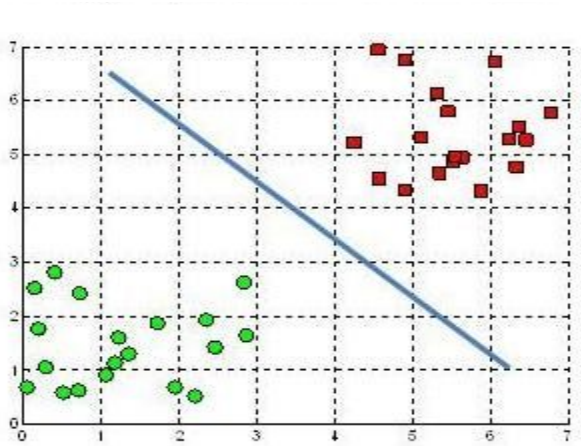
$$SVM\ Cost = \ min_w \frac{1}{2} \sum_{i=1}^{m} w_i^2 + \mathrm{C} \sum_{i=1}^{n} \max(0, 1 - y_i * (w^T x_i + w_0))$$

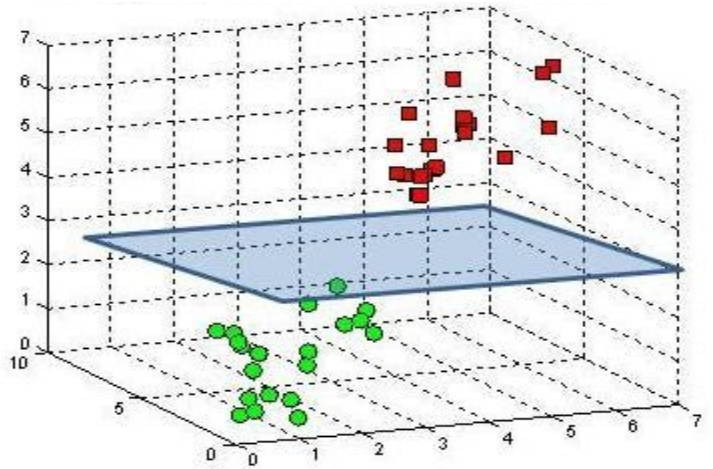
$$w = w - \alpha \ * \ w$$

For **incorrectly** classified samples:

$$w = w - \alpha \ * (w - \ \mathrm{C} * y_i * x_i) = w + \alpha * (\ \mathrm{C} y_i x_i \ - \mathrm{w})$$

Higher dimensional SVMs



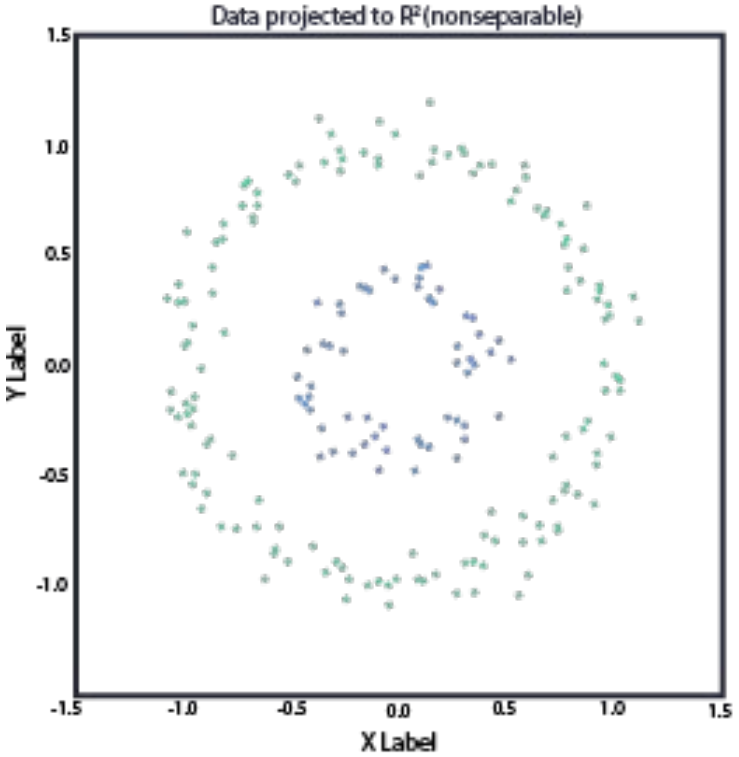


What if our data is not linearly separable?

Often there is no straight line we can draw to separate the data

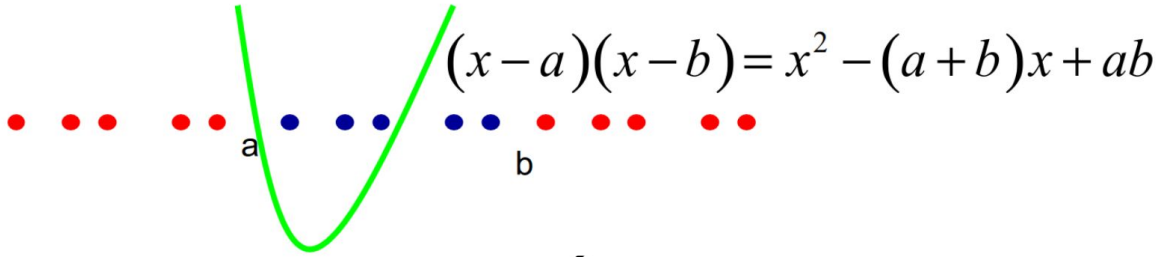
The idea is to gain linearly separation by *mapping* the data to a higher dimensional space

We apply a function to the data to transform it into a space that is easier to work in

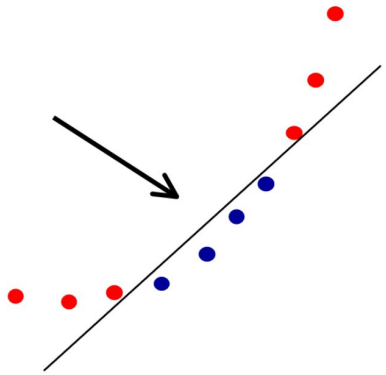


Non–Linear SVMs: Mappings

The following set can't be separated by a linear function, but can be separated by a quadratic one



However, if we map it as $x \mapsto \{x^2, x\}$ we can separate:



Non–Linear SVMs: Mappings

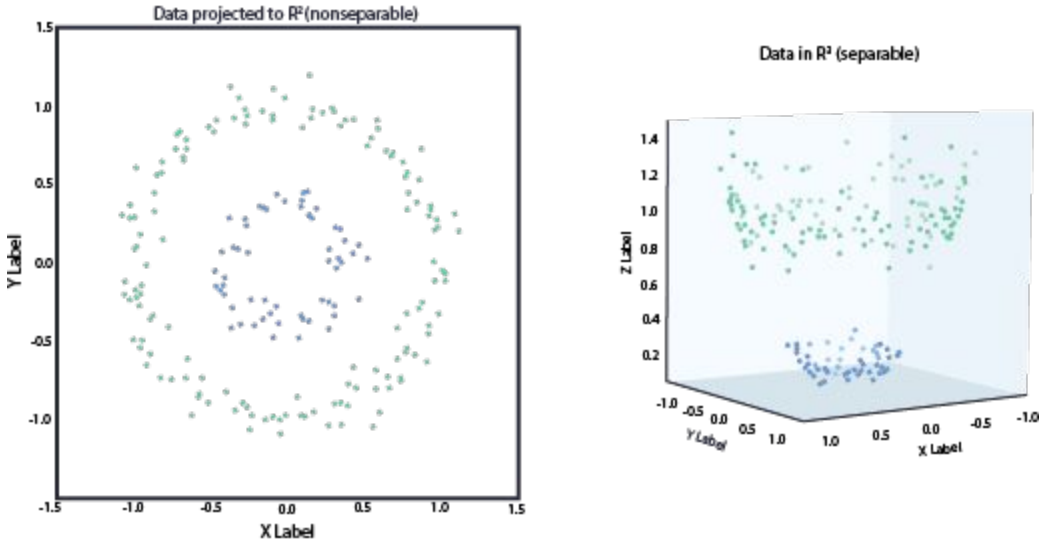
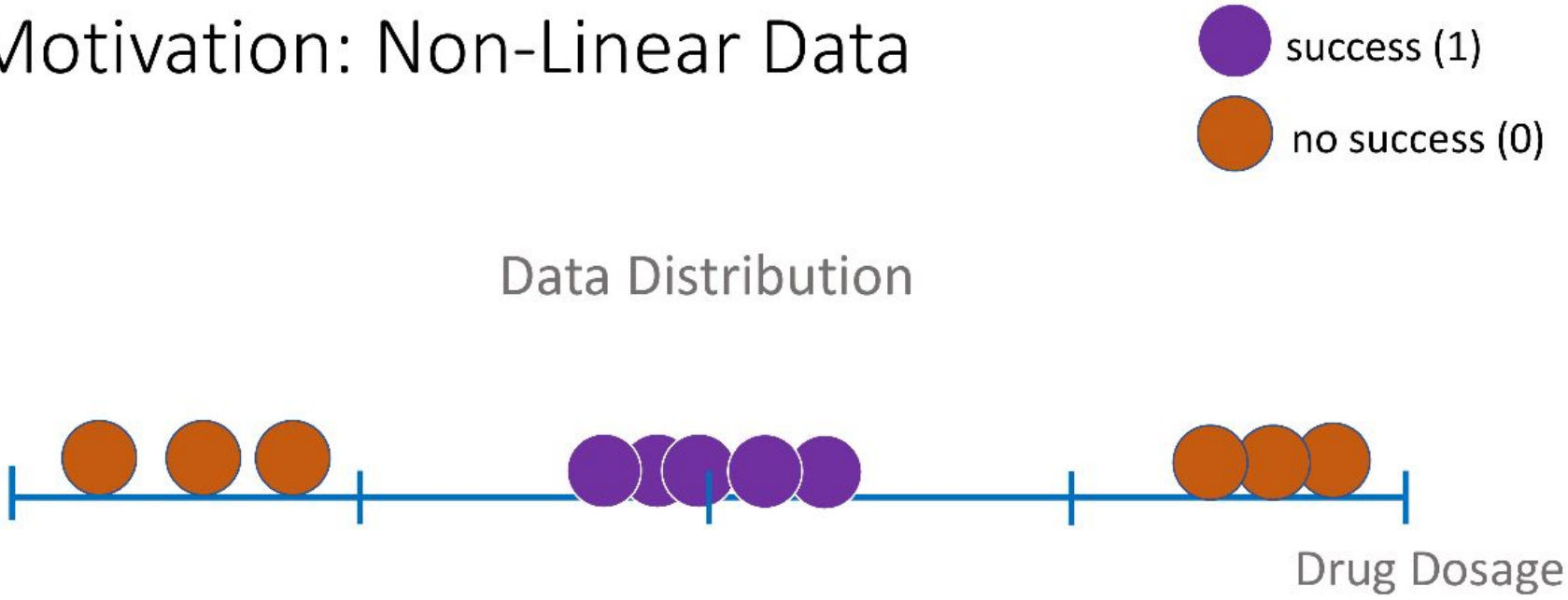
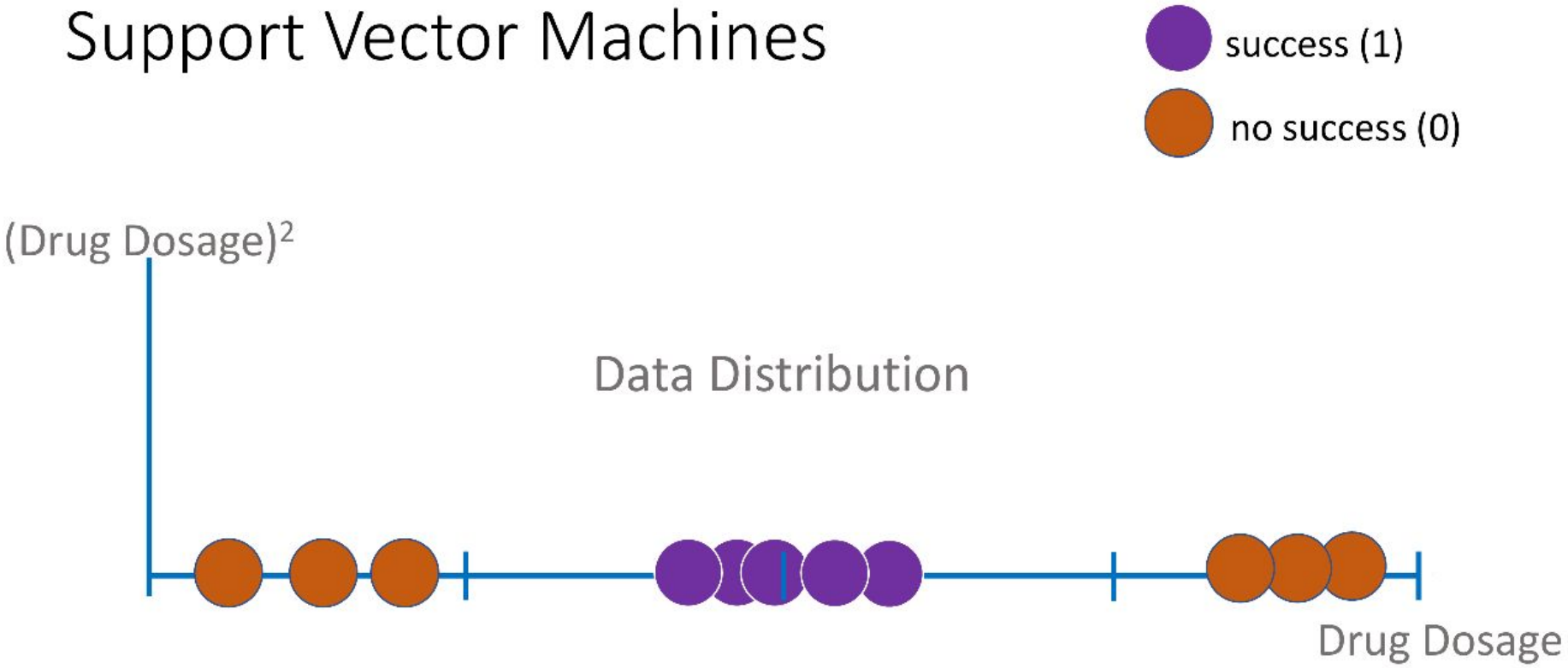


Figure 5: (Left) A data set in R^2, not linearly separable
(Right) The same dataset transformed by the transformation
$(x - M\mu_x)^2 + (y - M\mu_y)^2$

Motivation: Non-Linear Data



Support Vector Machines



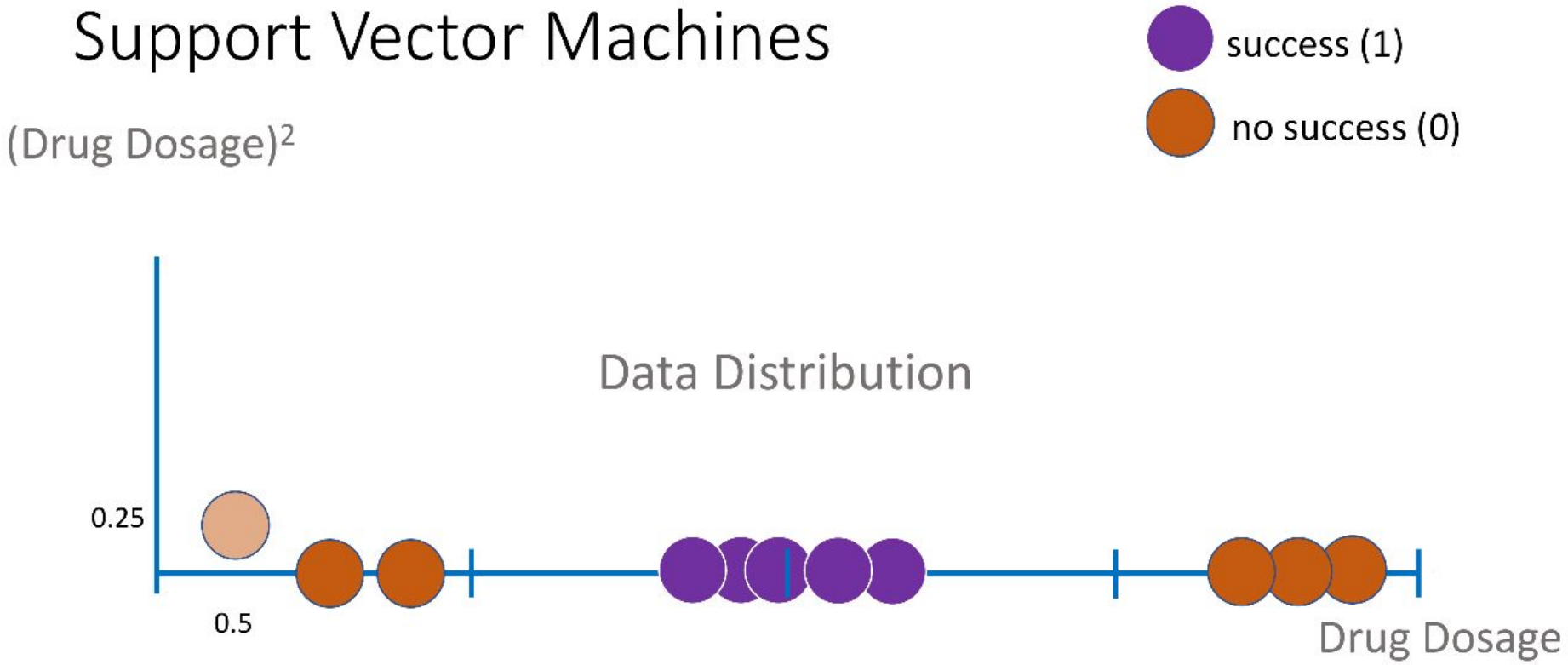
Support Vector Machines

success (1)

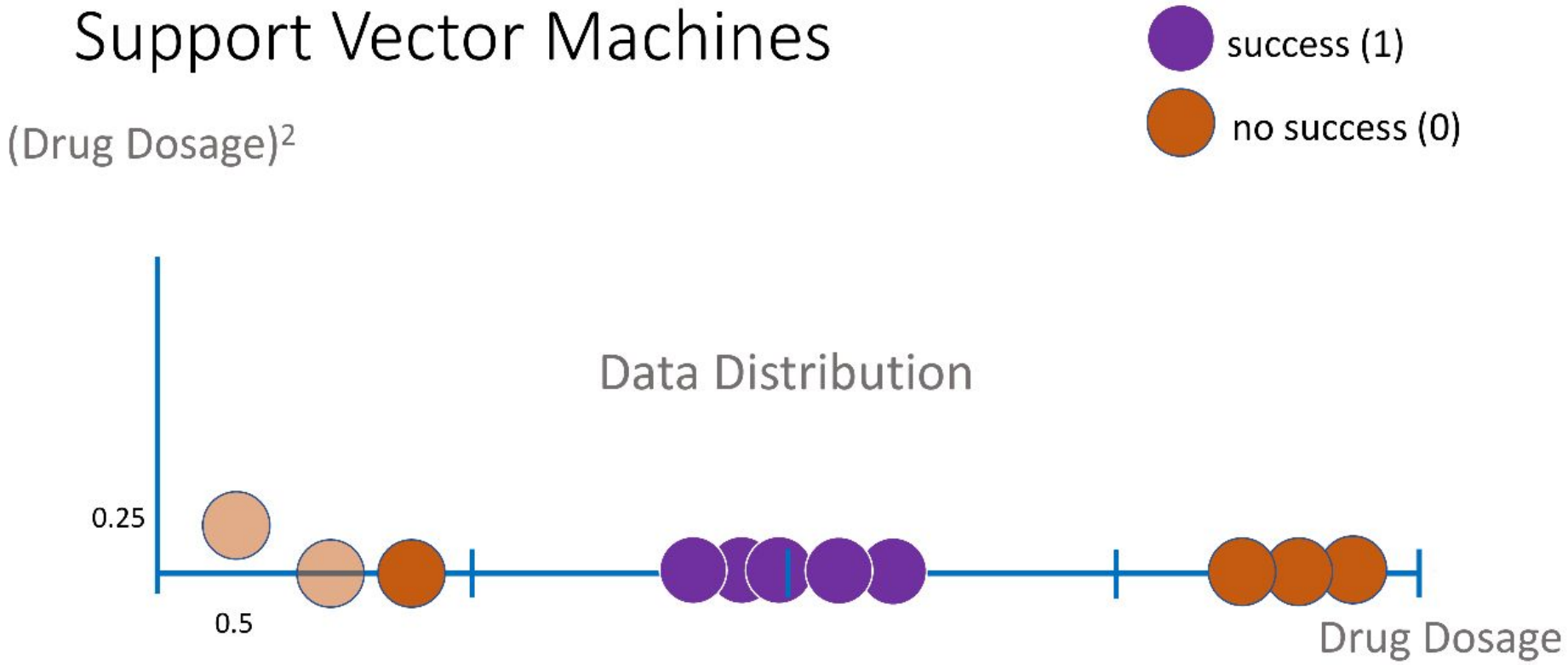
no success (0)

$(\text{Drug Dosage})^2$

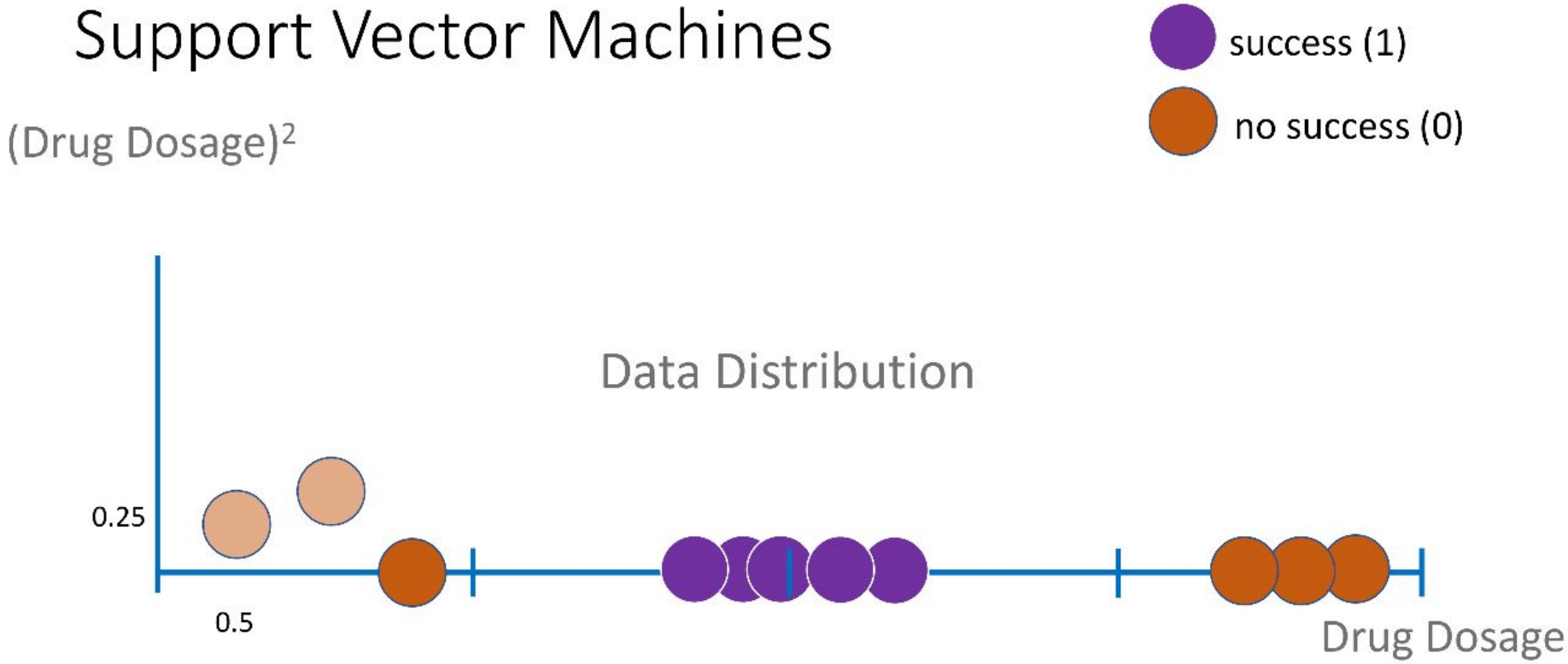
Support Vector Machines



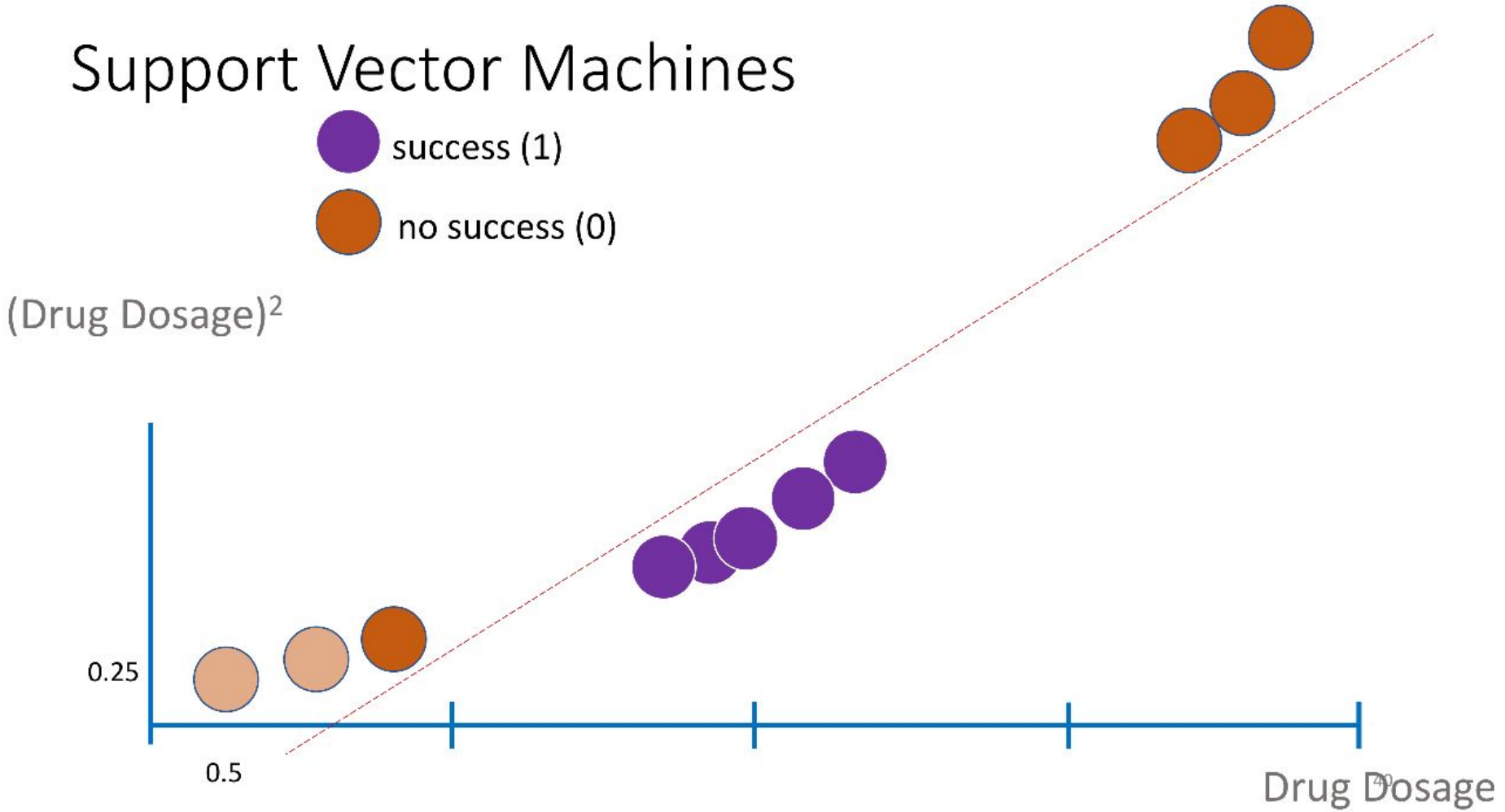
Support Vector Machines



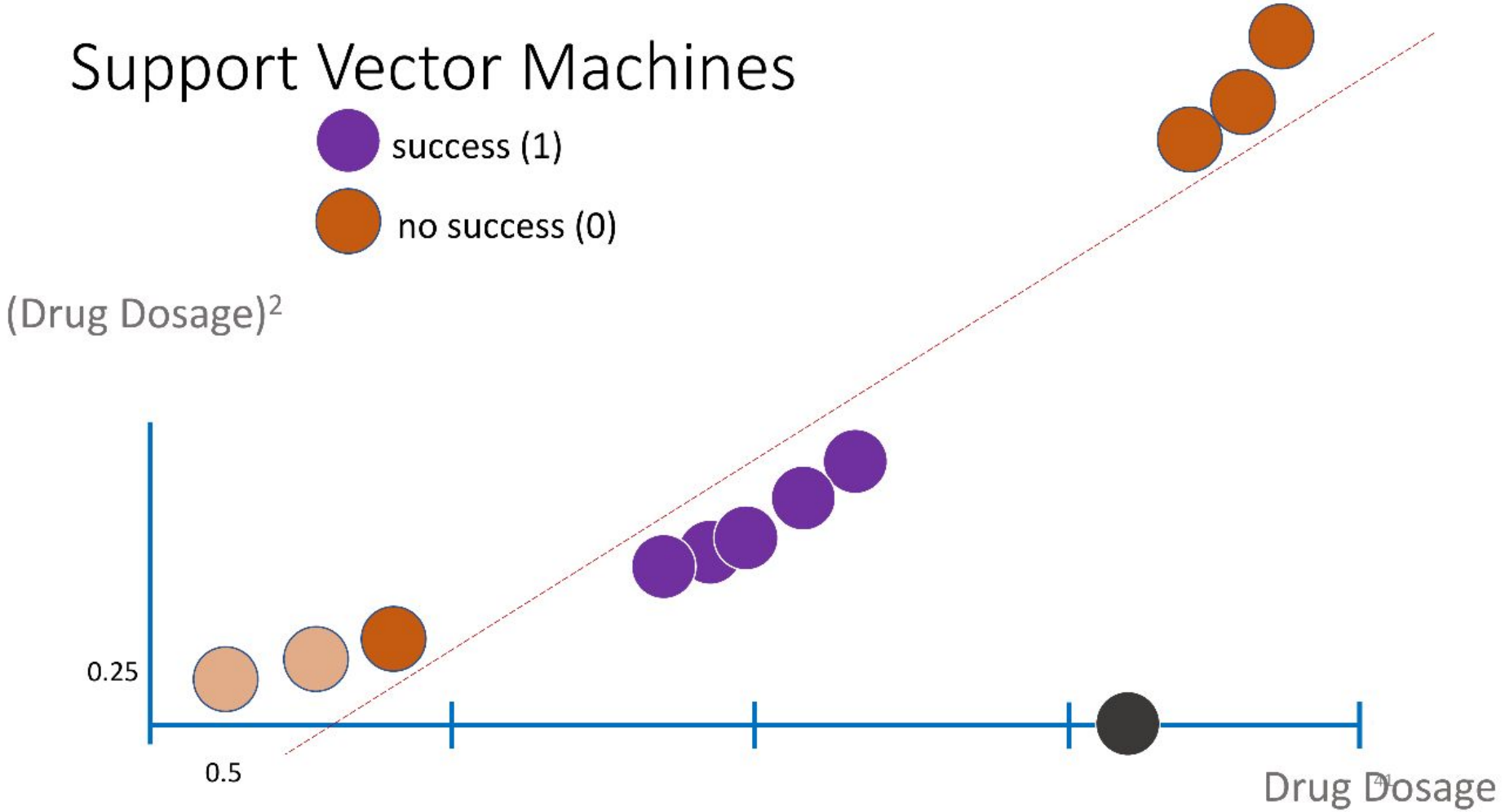
Support Vector Machines



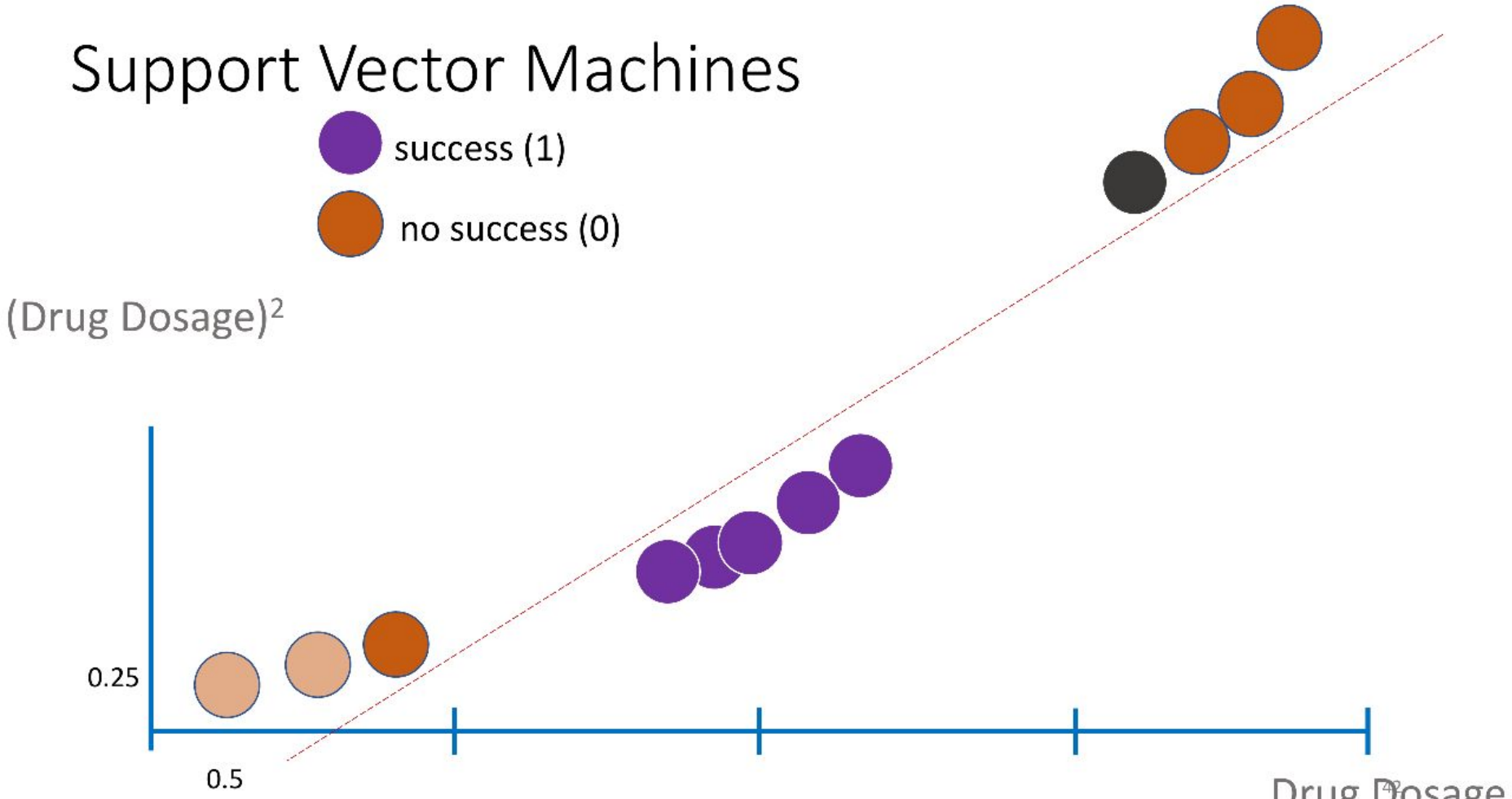
Support Vector Machines



Support Vector Machines



Support Vector Machines



Gradient of the Cost Function with higher dim transform

$$SVM\ Cost = min_w \frac{1}{2}\sum_{i=1}^{m} w_i^2 + C\sum_{i=1}^{n} \max\left(0, 1 - y_i * f(\phi(x_i))\right)$$

$$SVM\ Cost = min_w \frac{1}{2}\sum_{i=1}^{m} w_i^2 + C\sum_{i=1}^{n} \xi(i) \qquad \xi(i) \geq 0 \text{ for } i = 1,\ldots,n$$

The hinge loss here is represented by the slack variables ξ(i).

$$SVM\ Cost = min_w \frac{1}{2}\sum_{i=1}^{m} w_i^2 + C\sum_{i=1}^{n} \max(0, 1 - y_i * (w^T \phi(x_i) + w_0))$$

As before, we take partial derivatives wrt the weights to find the gradients

- Using the gradients, we update the weights or parameters of the model

We minimize the Cost with respect to w and w_0 and maximize it with respect to C, to lower misclassifications.

We add the Φ to represent the higher dimensional transform of the input vectors

Support Vector Machines and kernel functions

- Move the data from a low dimension to another dimension
- Find a support vector classifier that classifies the higher dimensional data
- There may be several ways to move the data to higher dimension
  $X \rightarrow X2$, $X \rightarrow X3$, ... which one to use?
- Kernel functions systematically find Support Vector Classifiers in higher dimensions.
- Kernel Functions calculate the relationships between every pair of observations as if they are in a higher dimension; they don't actually do the transformation.
- Examples: Polynomial Kernel, Radial Basis Function (RBF) Kernel

Polynomial Kernel

- Has a parameter d that indicates the degree of polynomial
- For a value of d, the kernel function computes the relationship between each pair of observations after they are moved to the d-Dimensional space. These relationships are used to find a Support Vector Classifier.
- Polynomial Kernel increases dimensions by setting d.
- Cross Validation can be used to find a good value for d.
- Example:
  - d=2 for the drug dosage example. The polynomial kernel computes the 2D relationships between each pair of observations.
  - These relationships are used to find a Support Vector Classifier.

Radial Basis Function

- Finds Support Vector Machines in infinite dimensions
- Behaves like a weighted nearest neighbor model
- It looks at the closest observations (neighbors) and gives them a higher weight (or influence) to classify the new observation

Kernel trick

- Kernel functions systematically find Support Vector Classifiers in higher dimensions.
- Kernel Functions calculate the relationships between every pair of observations as if they are in a higher dimension; they don’t actually do the transformation.
- Calculating the higher dimensional relationships without transforming the data to a higher dimension is called “The Kernel Trick”
- Benefits:
  - Reduces the amount of computation required for SVMs by avoiding the math that transforms the data
  - Makes calculating the relationships in the infinite dimensions used by the Radial Kernel possible.