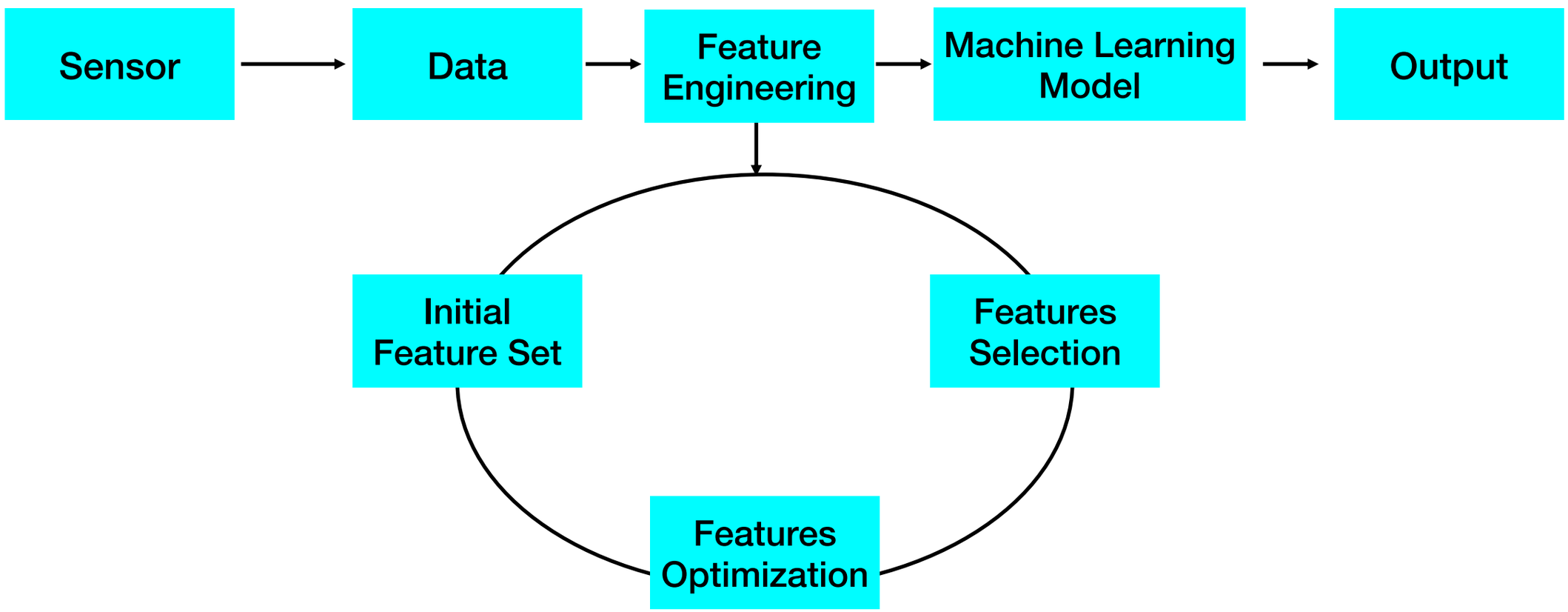
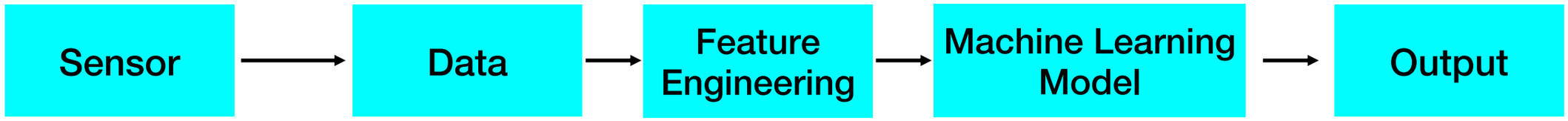
Classification – 1

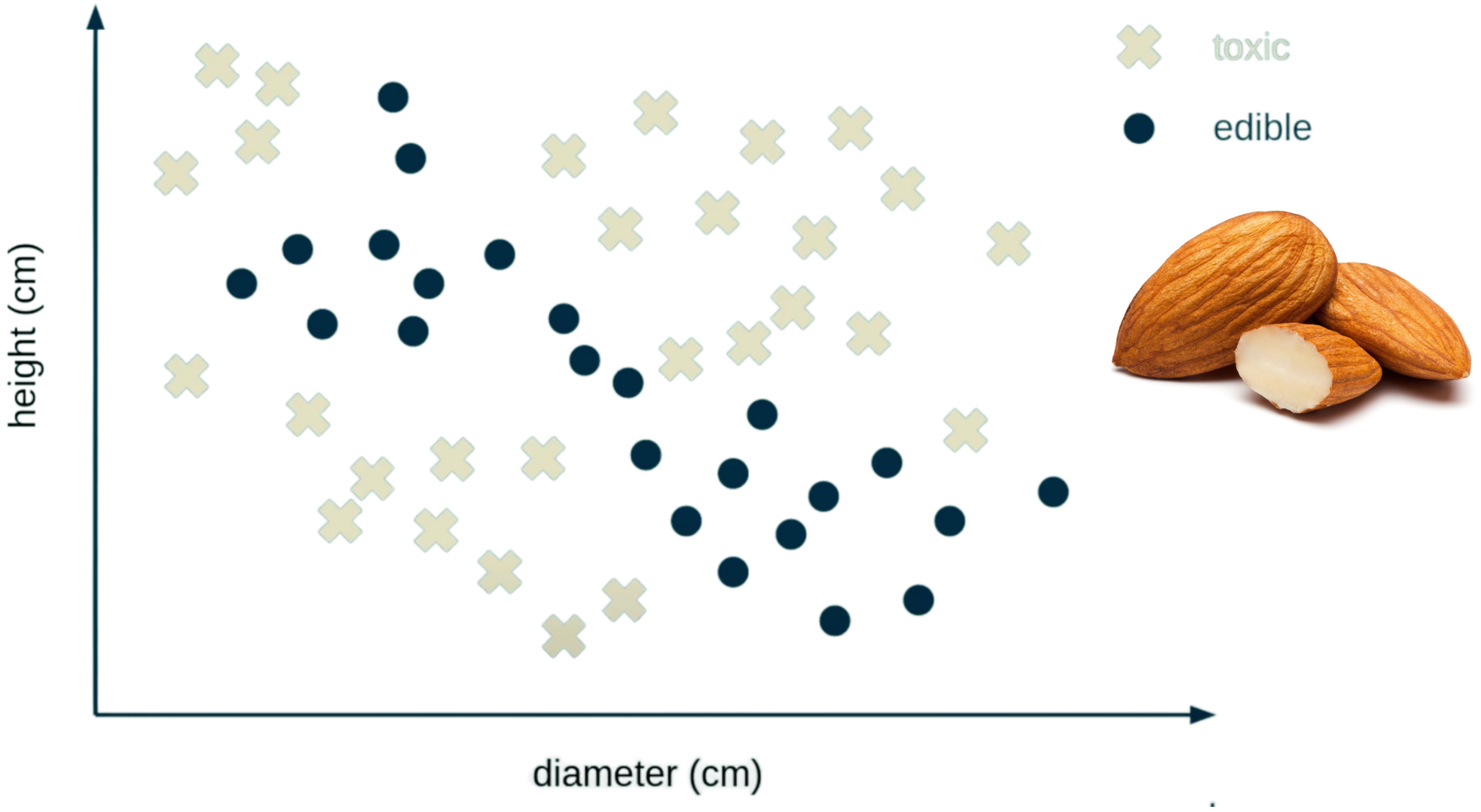
KNN, Linear Regression, SVM

Sai

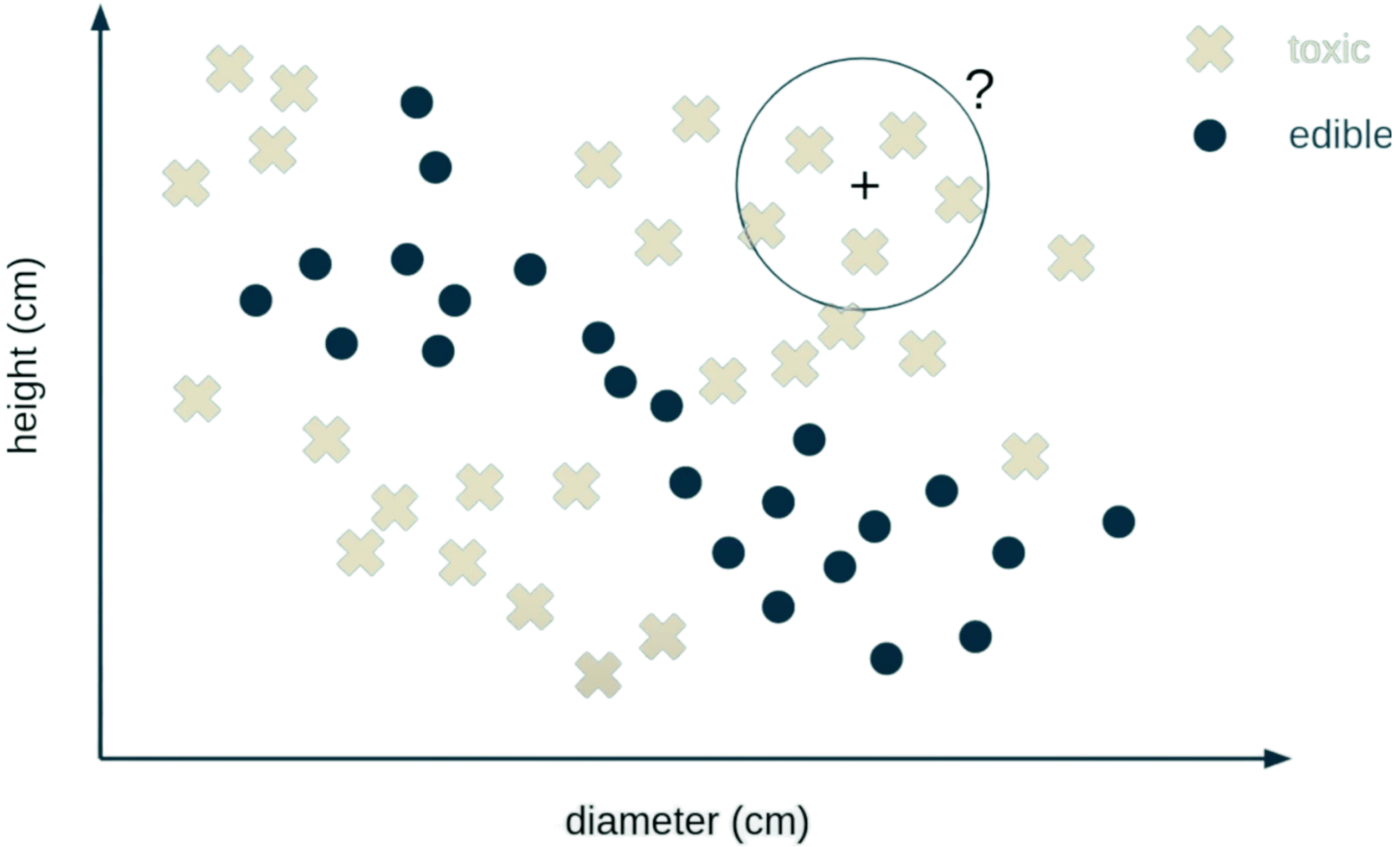

Sensor
Data
Feature Engineering
Machine Learning Model
Output
Initial Feature Set
Features Selection
Features Optimization


Sensor
Data
Feature Engineering
Machine Learning Model
Output

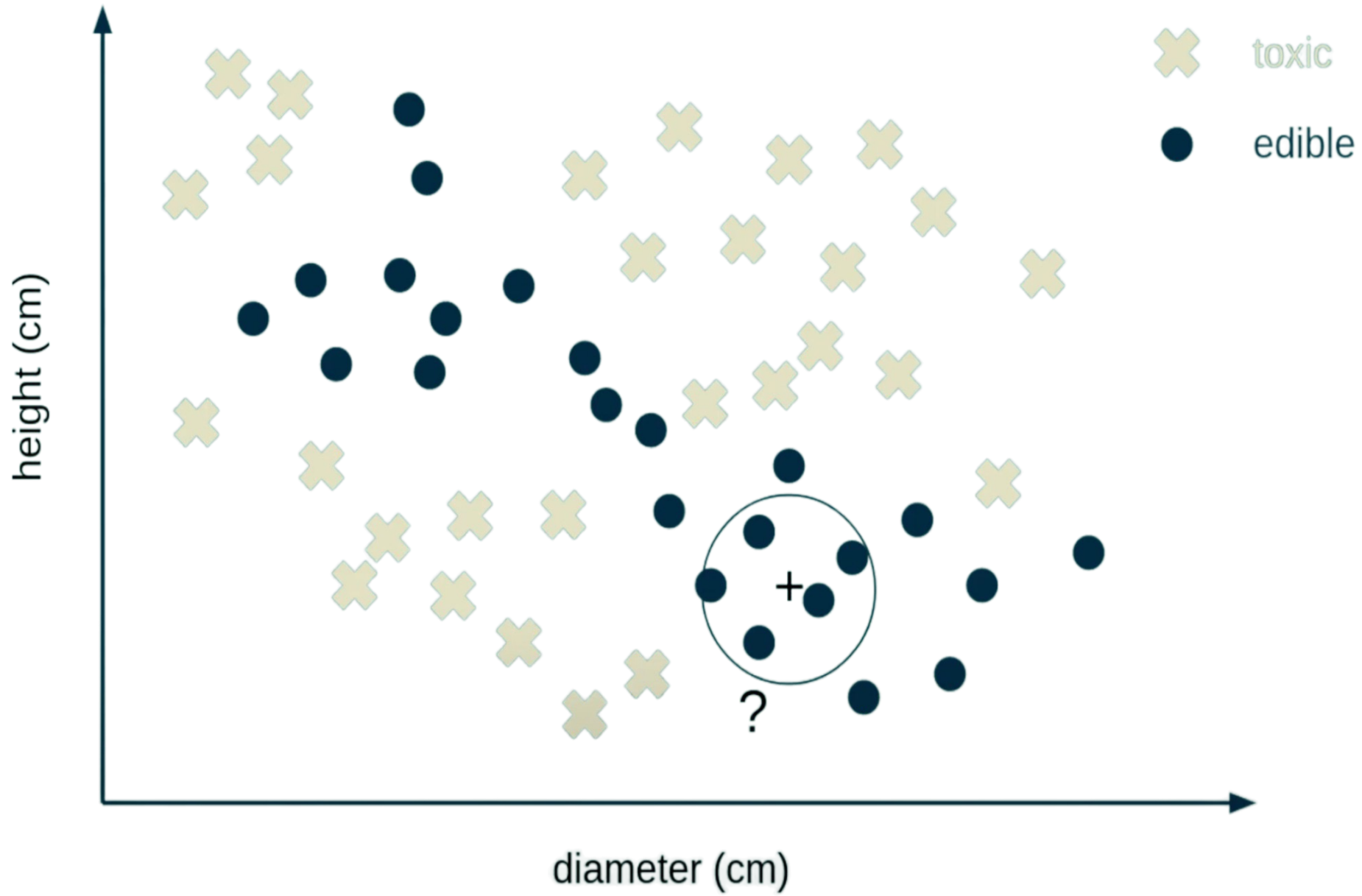
KNN Algorithm



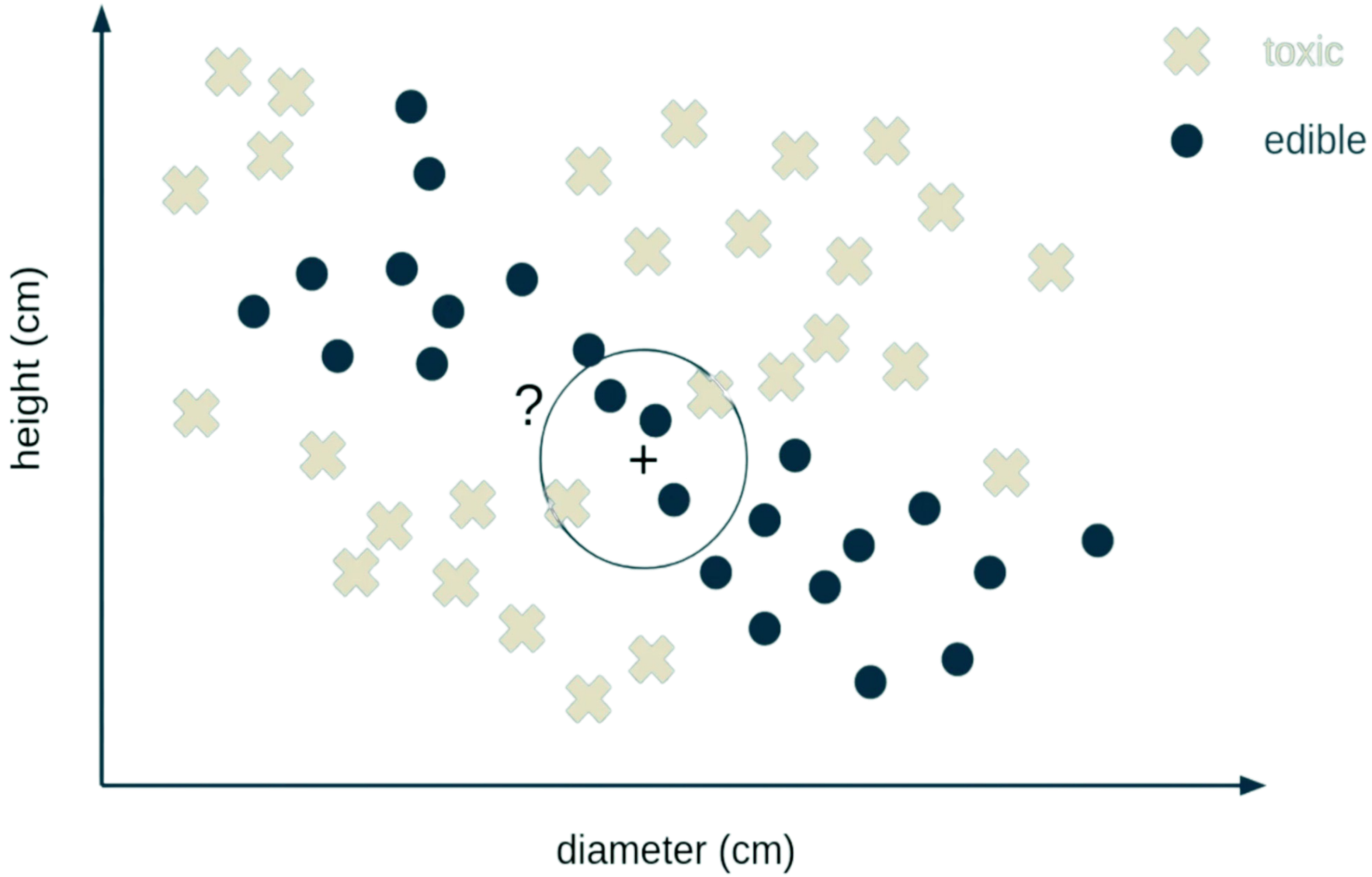
Graph courtesy: Brandon



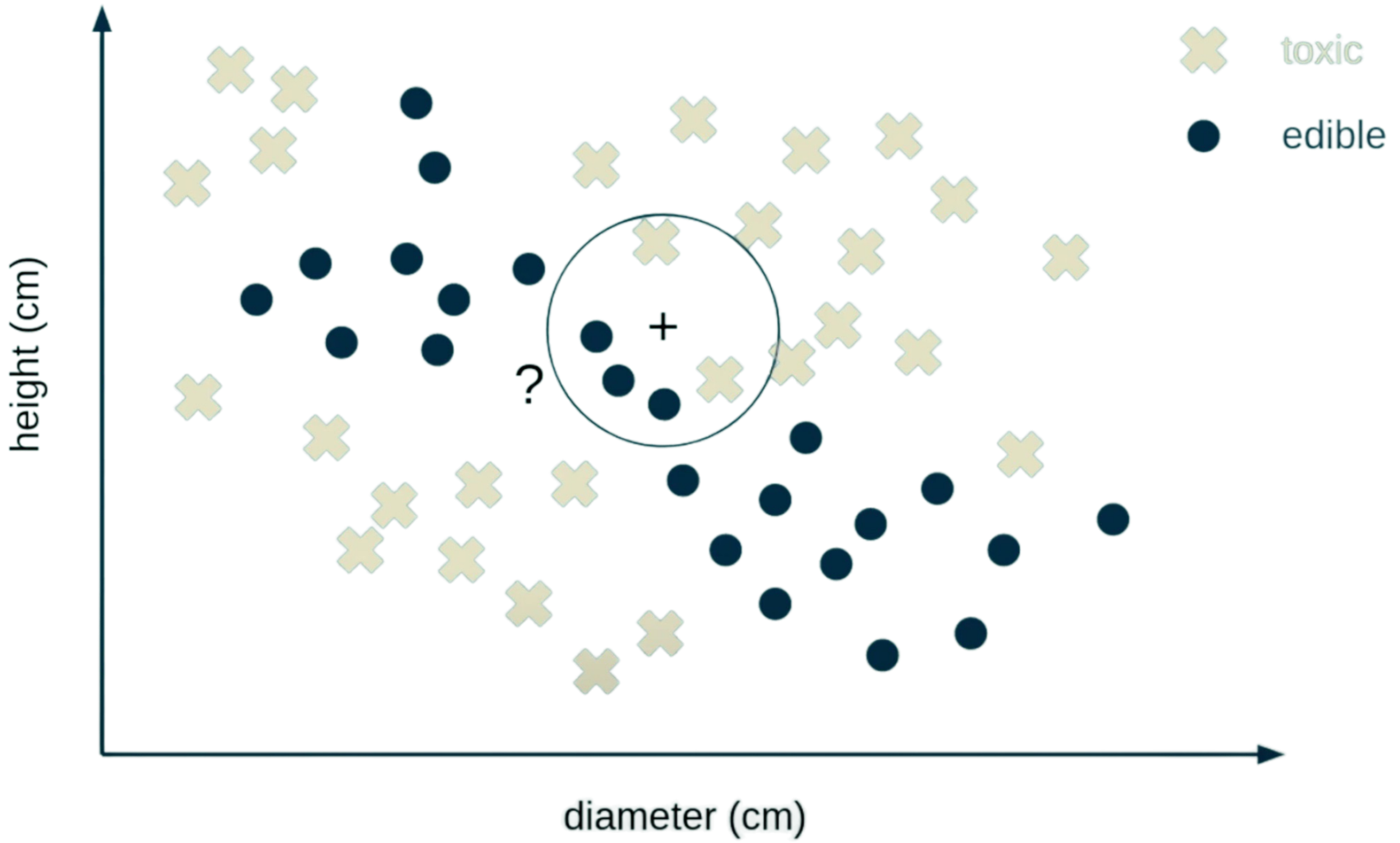
Graph courtesy: Brandon



Graph courtesy: Brandon

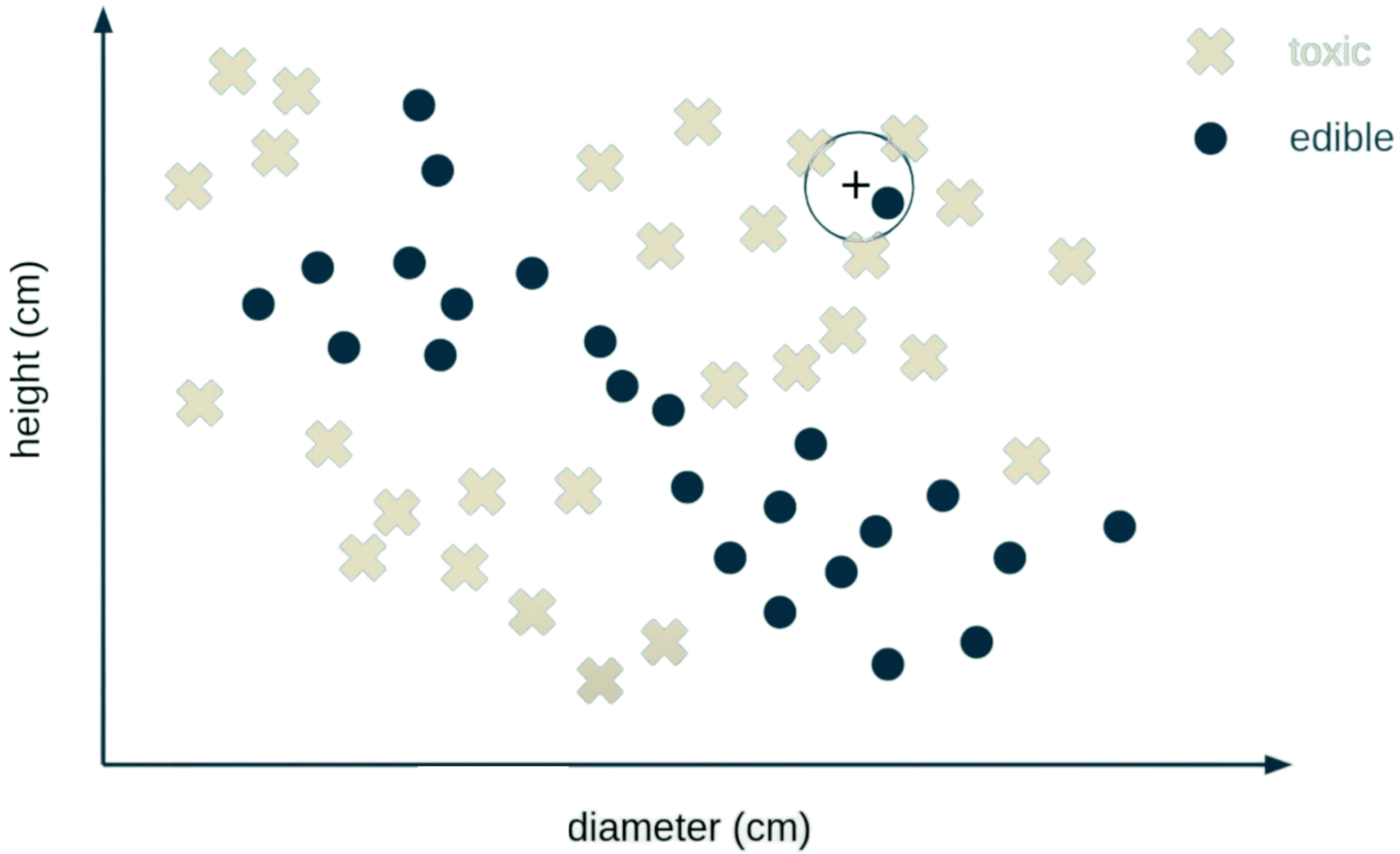


Graph courtesy: Brandon

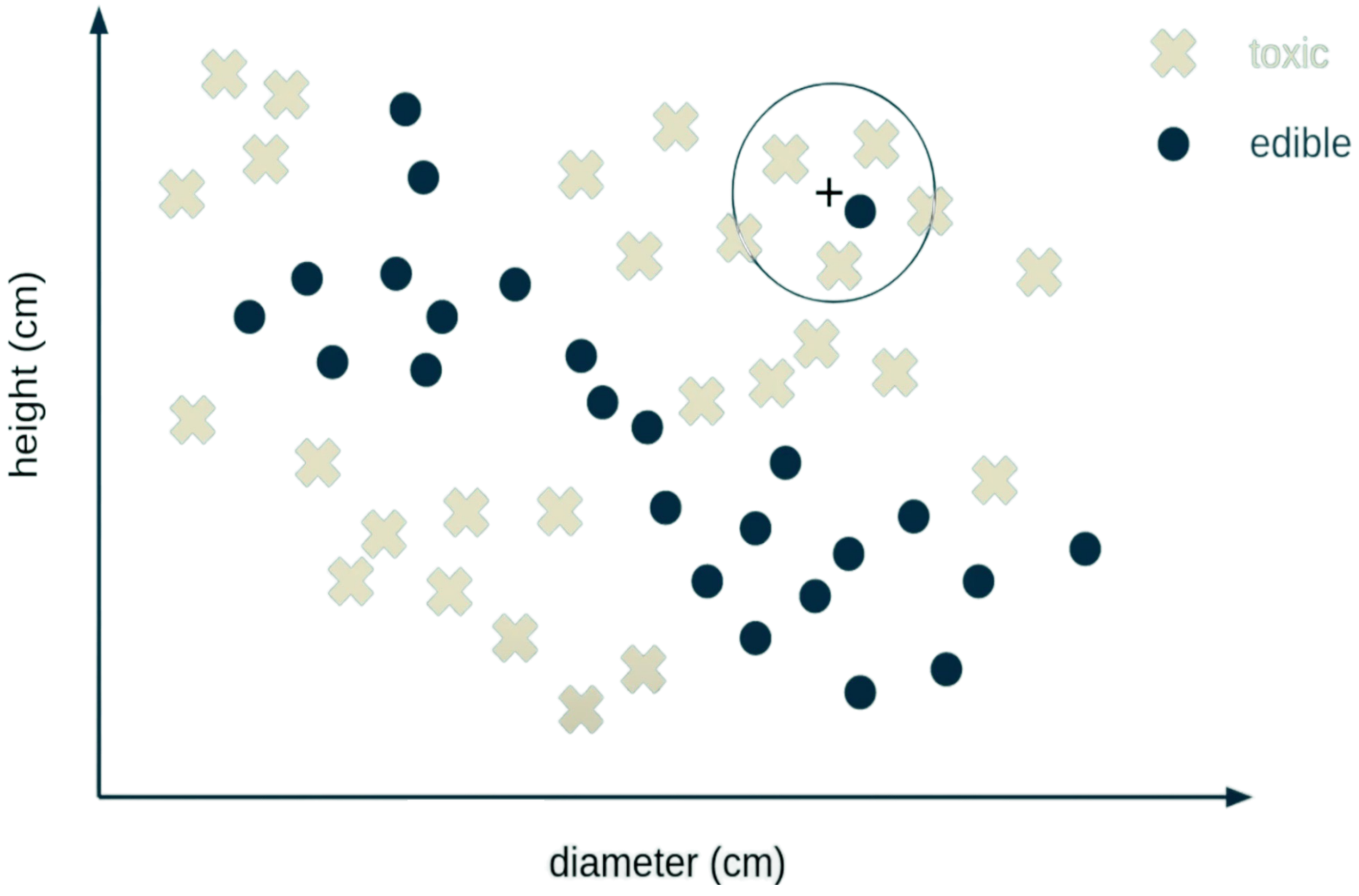


Graph courtesy: Brandon

Choice of K matters

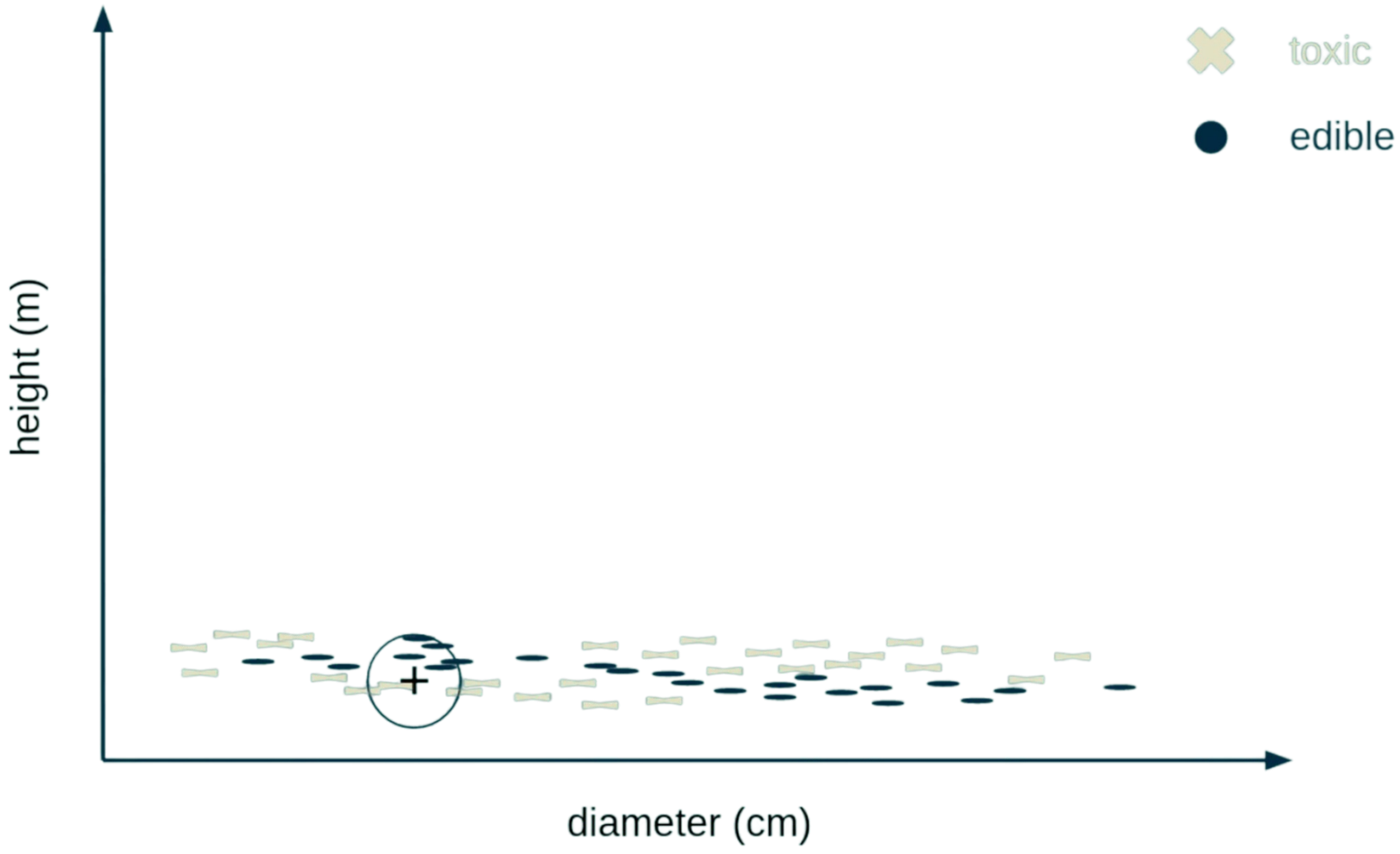


Graph courtesy: Brandon

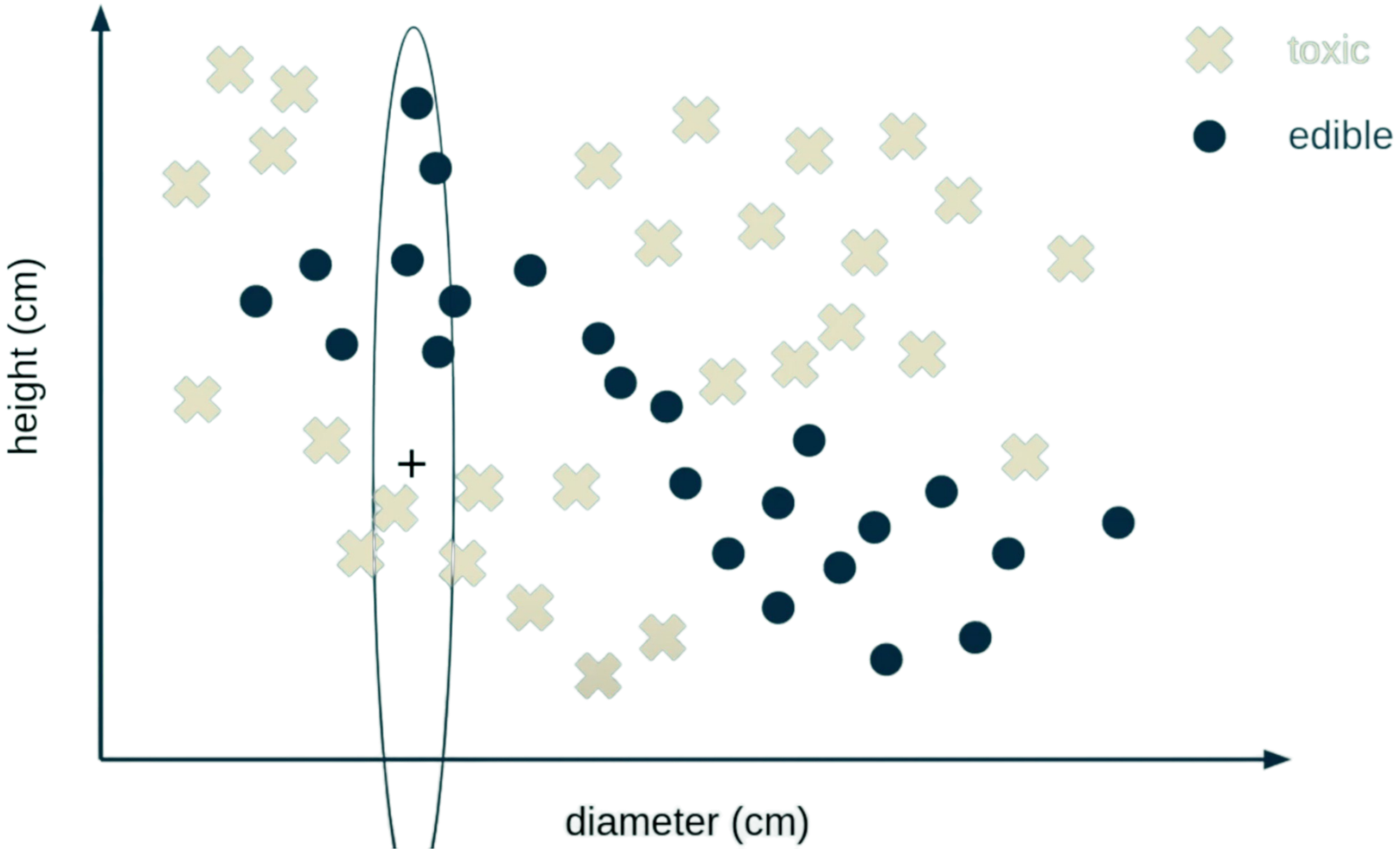


Graph courtesy: Brandon

Feature Scaling Matters

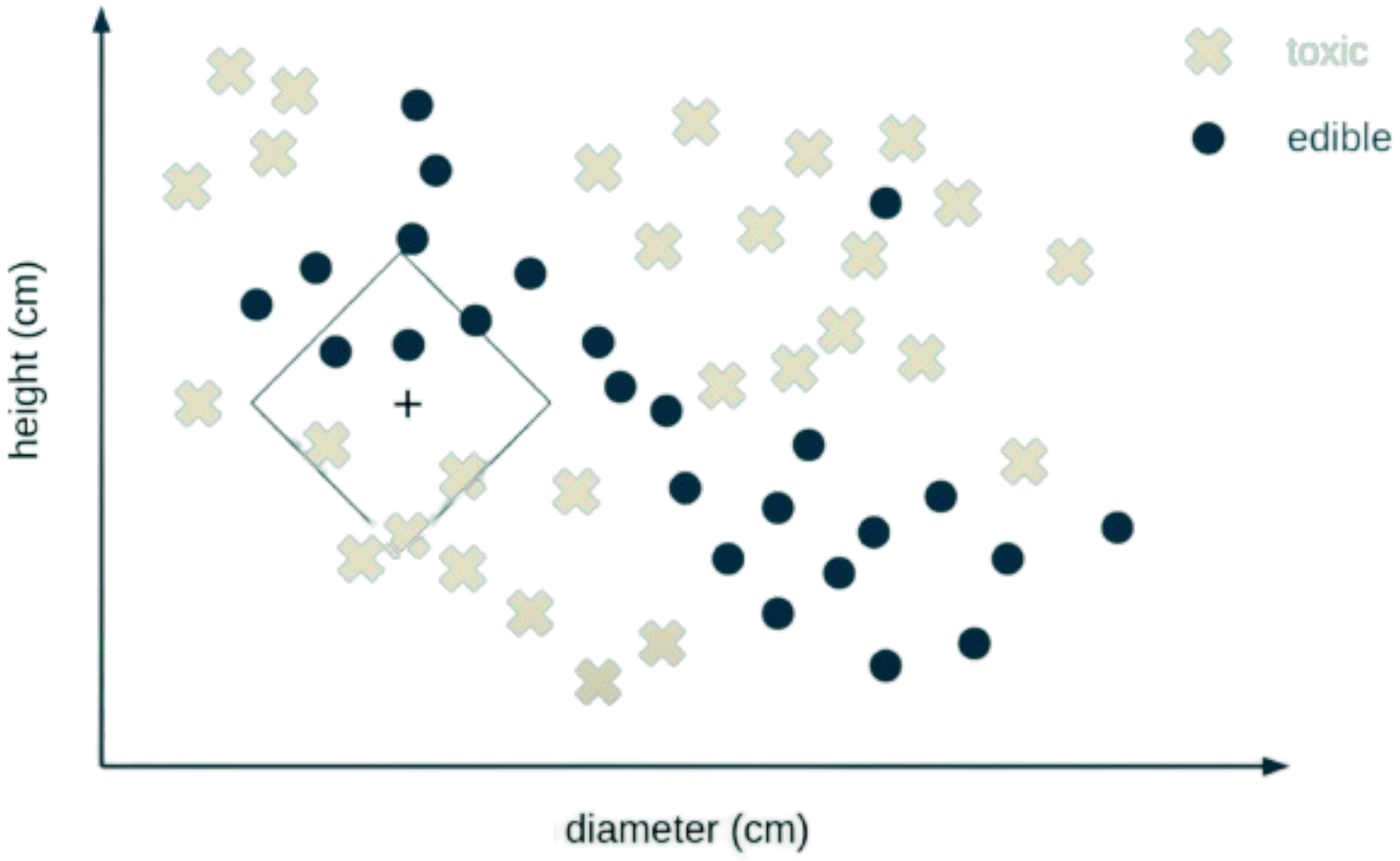


Graph courtesy: Brandon



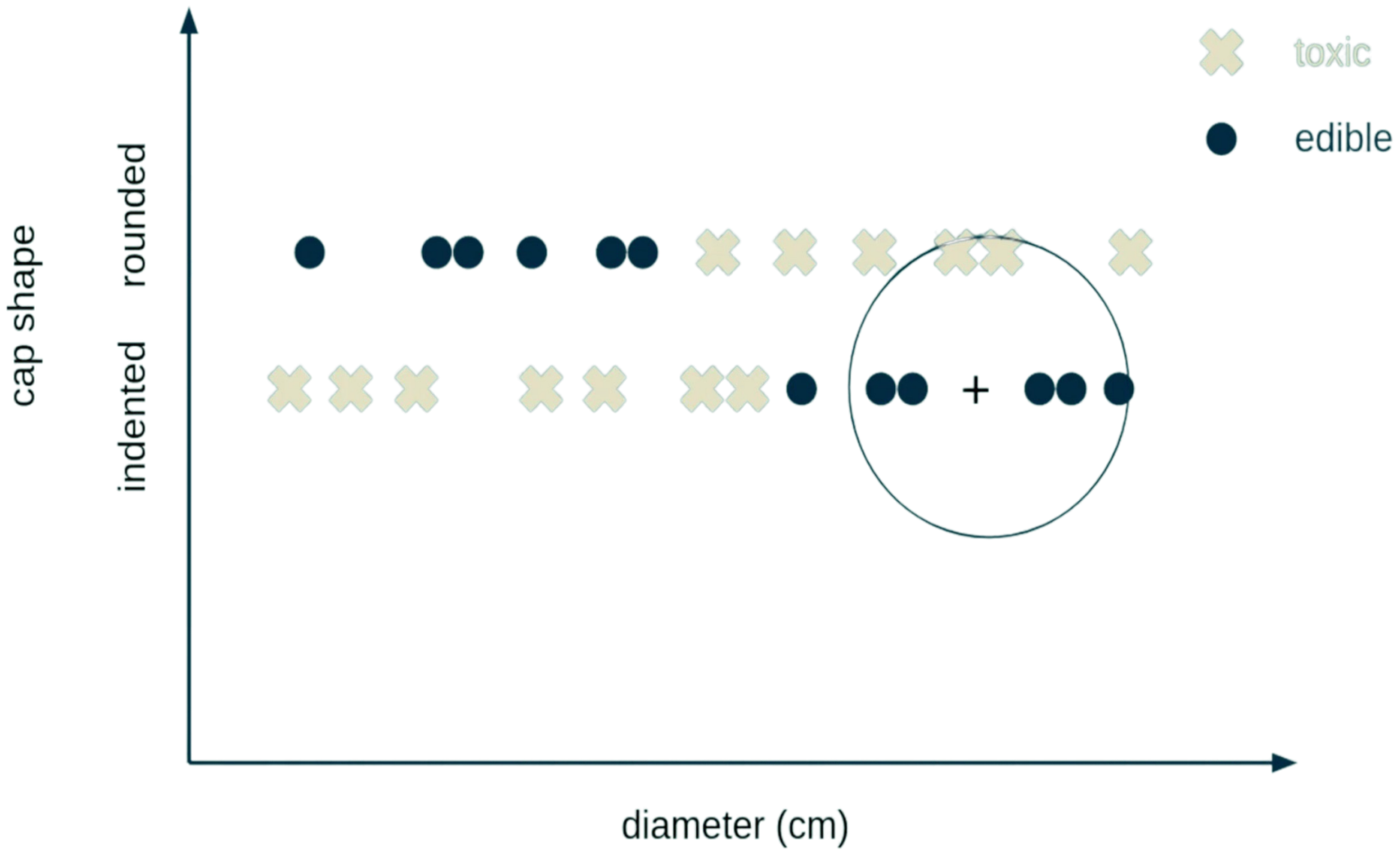
Graph courtesy: Brandon

Distance Metric Matters

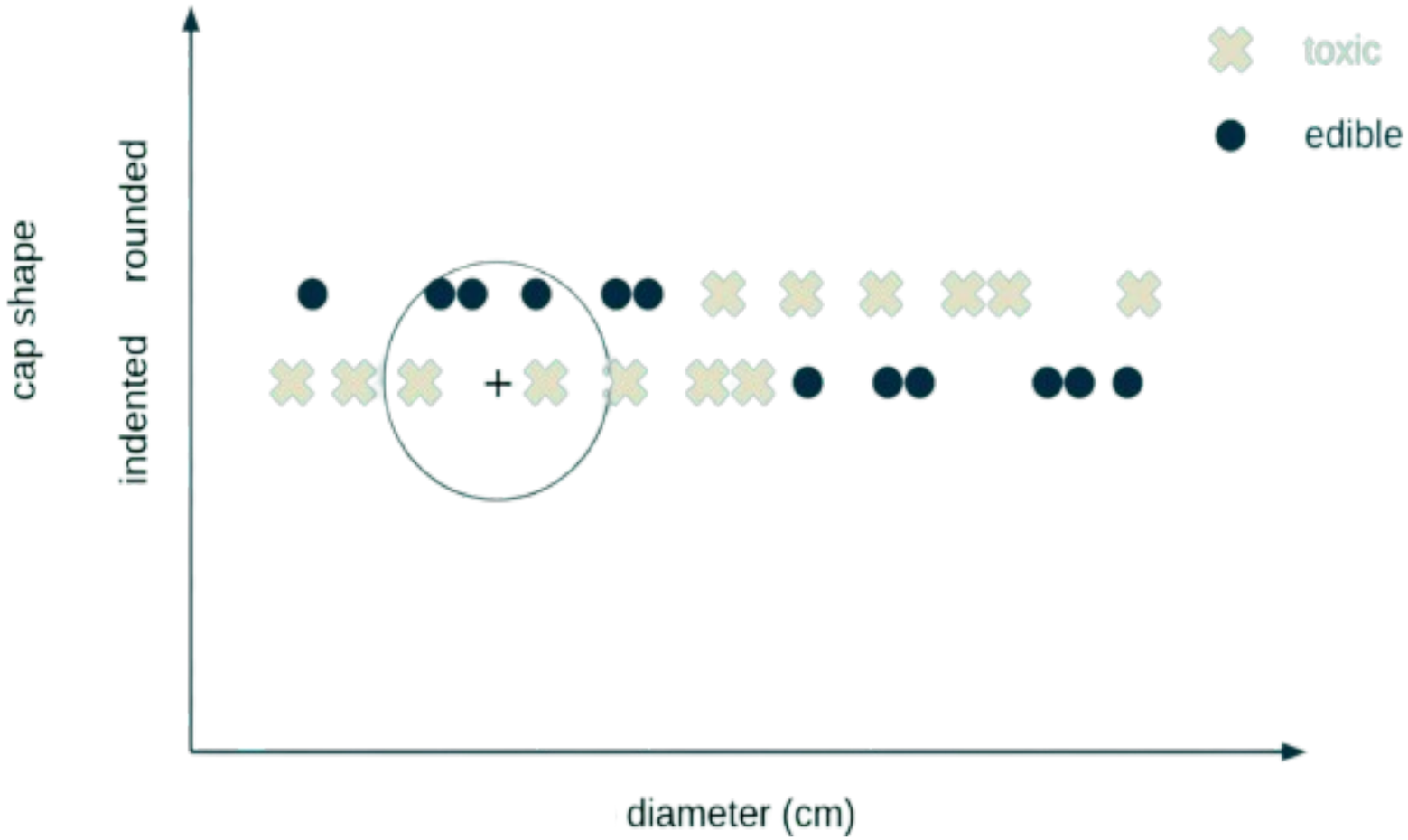


Graph courtesy: Brandon

Categorical Data

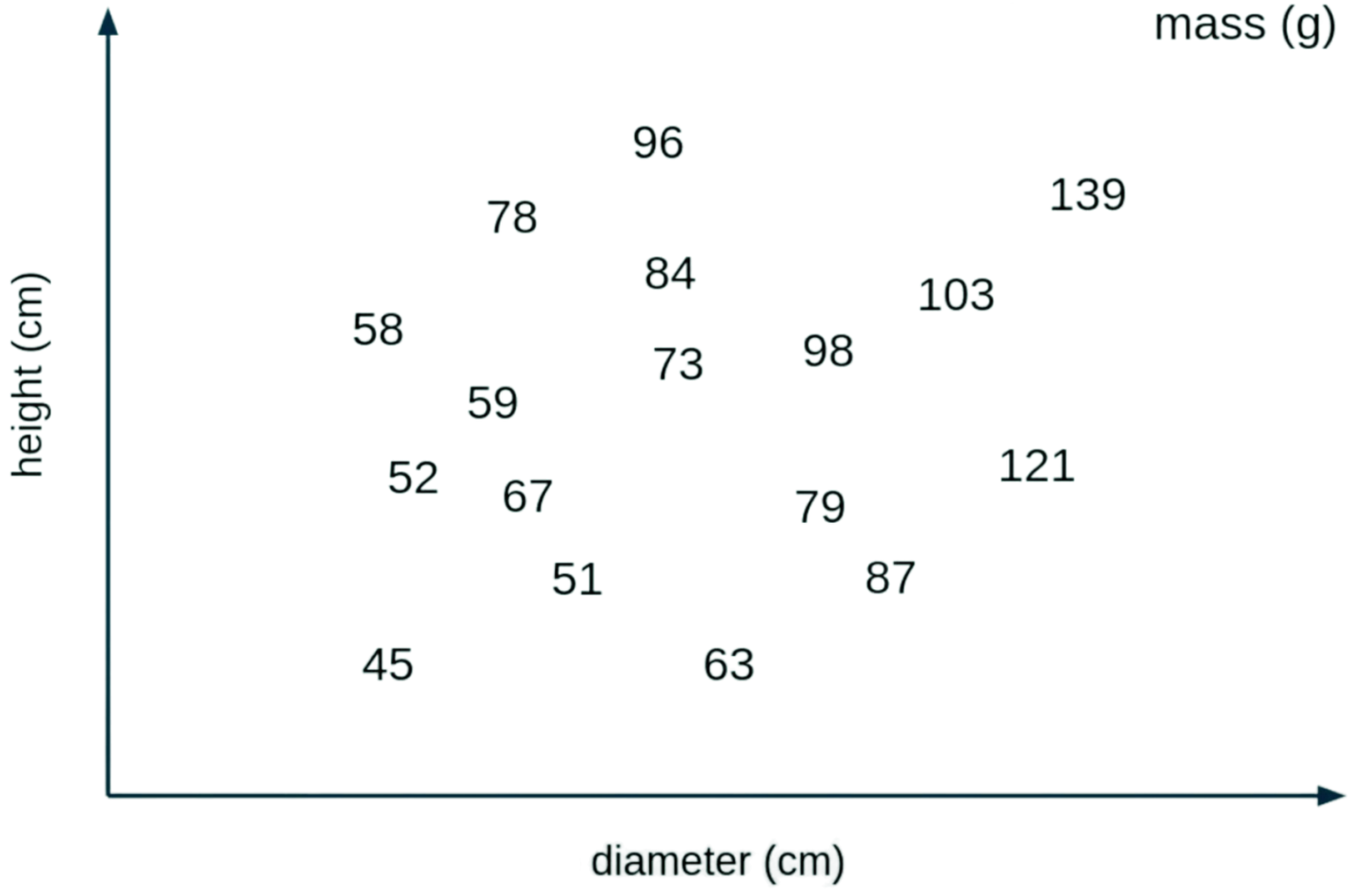


Graph courtesy: Brandon

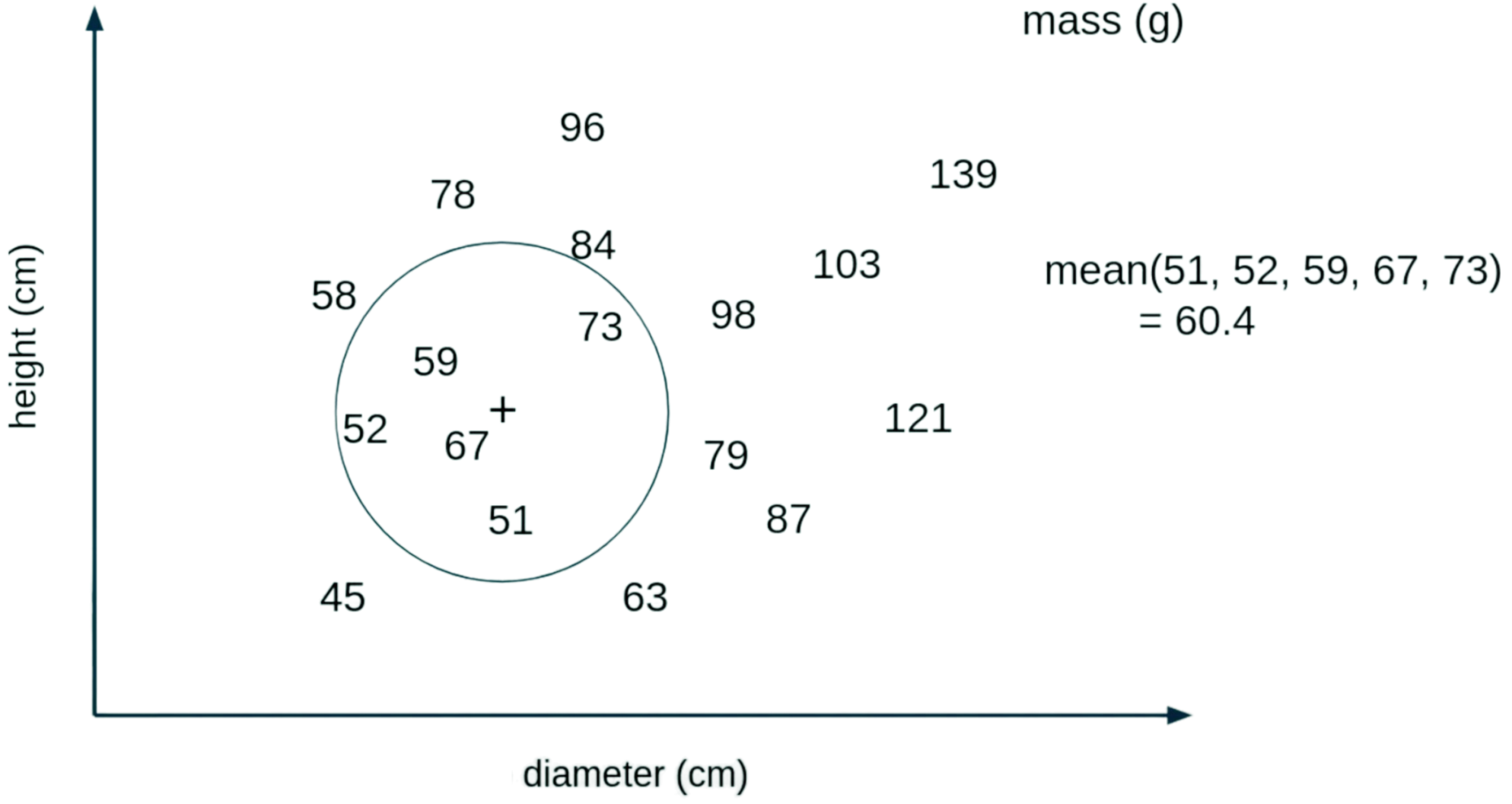


Graph courtesy: Brandon

Regression

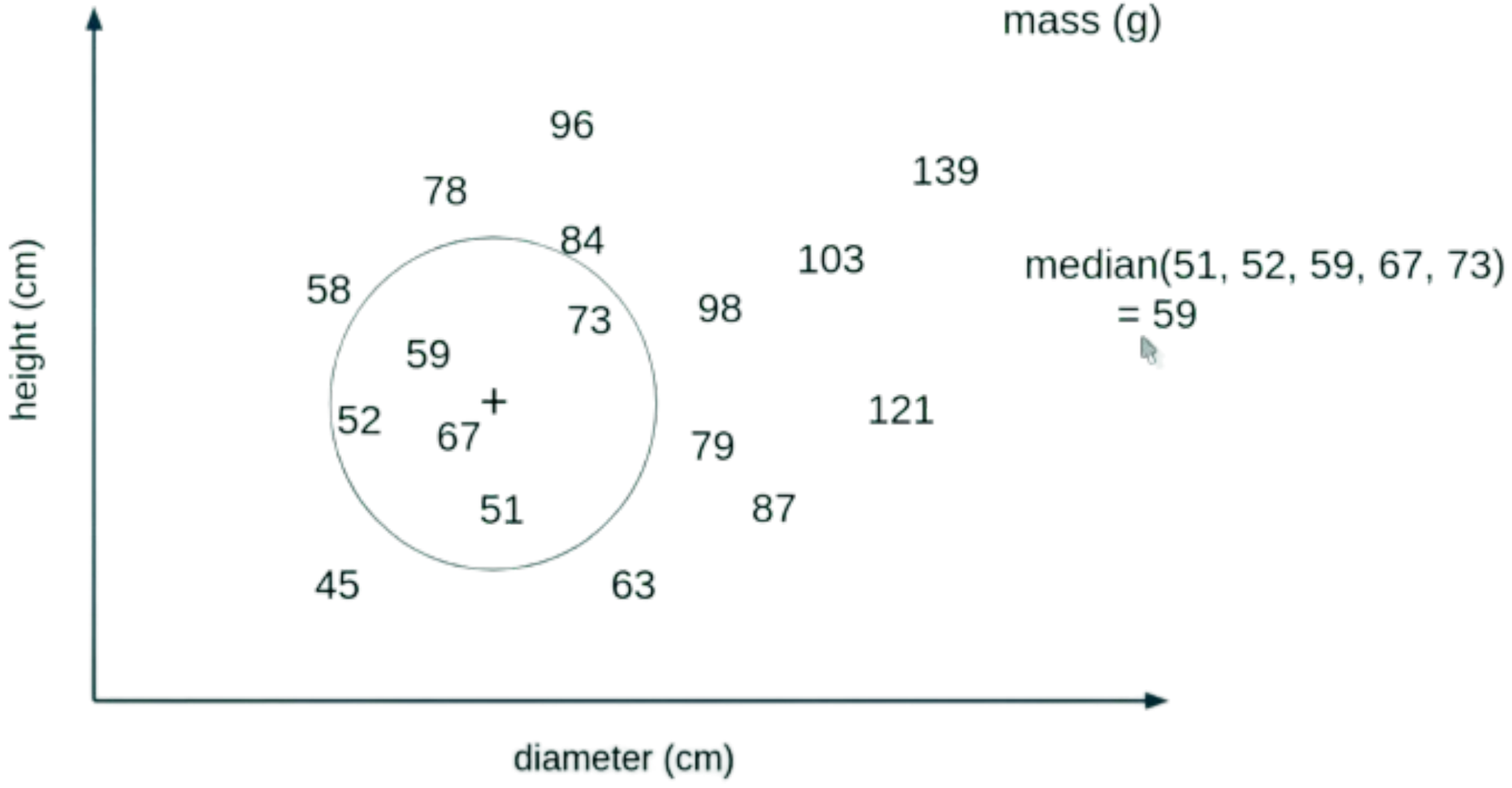


Graph courtesy: Brandon

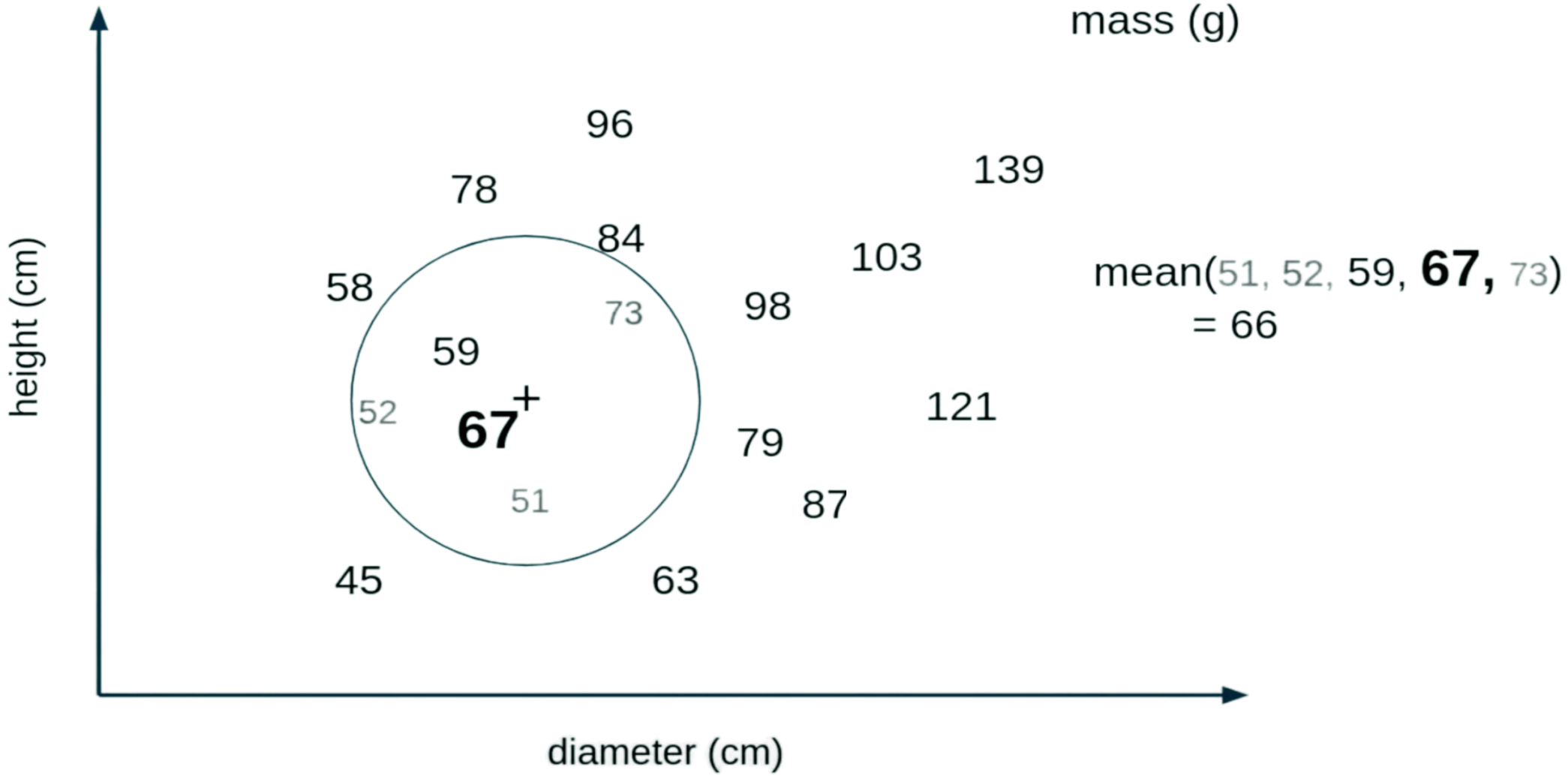


mean(51, 52, 59, 67, 73)
= 60.4

Graph courtesy: Brandon



Graph courtesy: Brandon



Graph courtesy: Brandon

Strengths

Zero training time.

Sample efficiency.

Explainable.

Easy to add and remove data.

Less sensitive to class imbalance.

Weaknesses

- Expensive to compute with large data sets.
- Sensitive to feature scaling.
- Sensitive to distance metric.

Workarounds

- Learned feature scaling.
- Clever data structures like k-d trees and ball trees.
- Data reduction.

Loss Function for Linear Regression and Classification

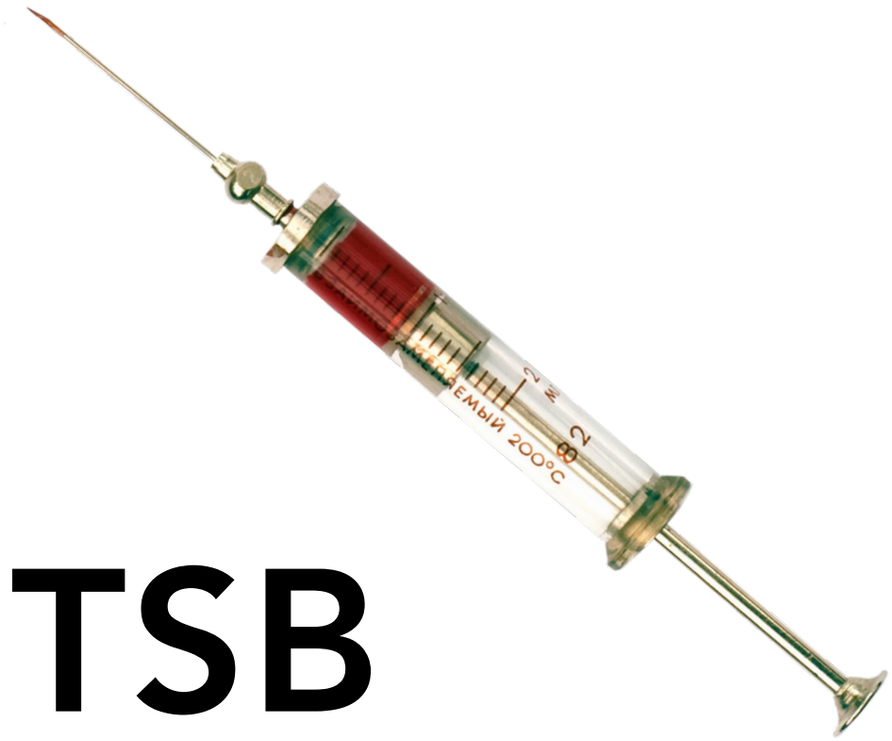
84% newborns are affected

First week of life

Devastating if not treated

BiliCam

Current Technologies



TSB

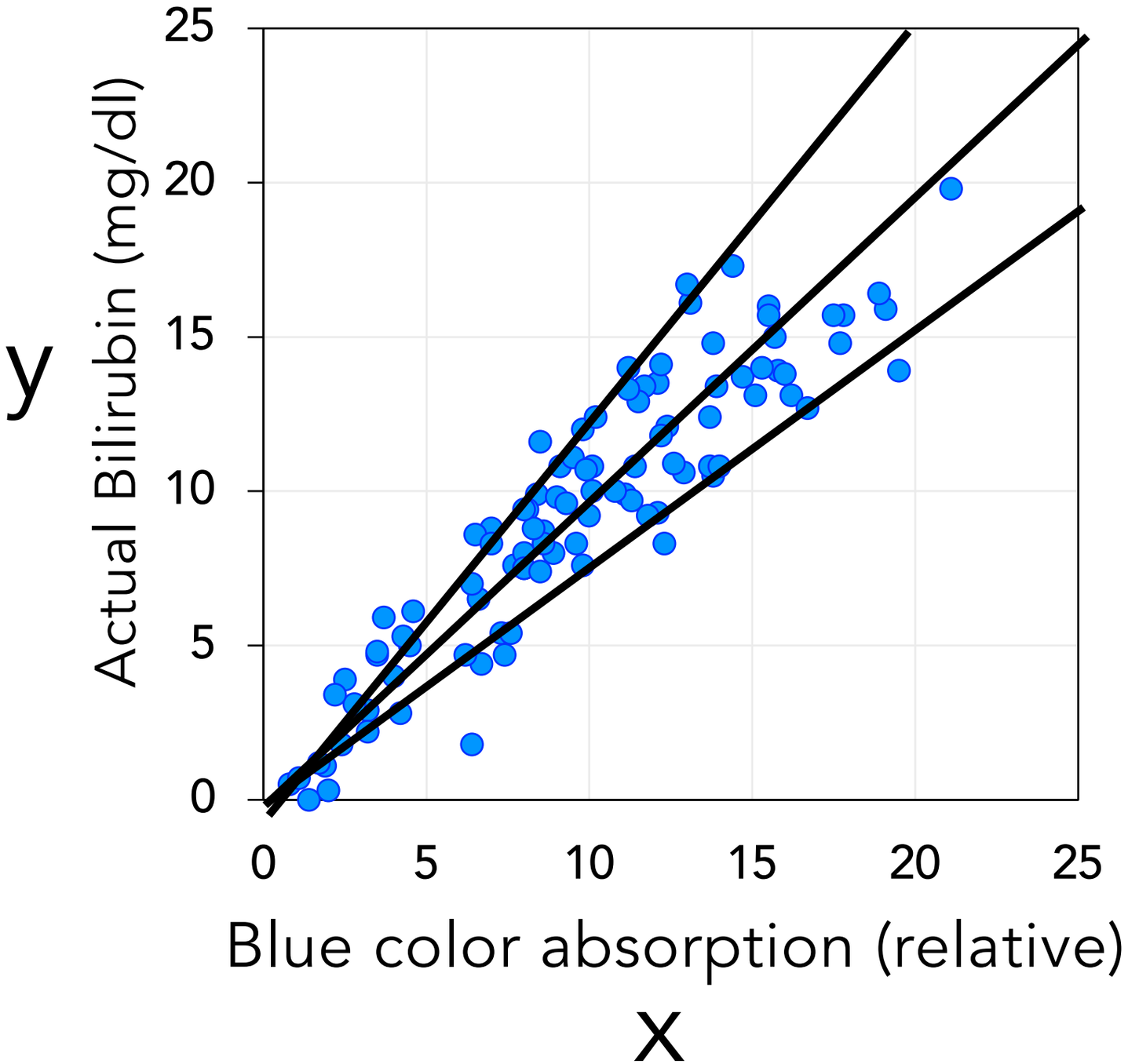
Medical Gold Standard

TcB

Screening Tool for TSB

Not as simple as measuring the yellowness of the skin

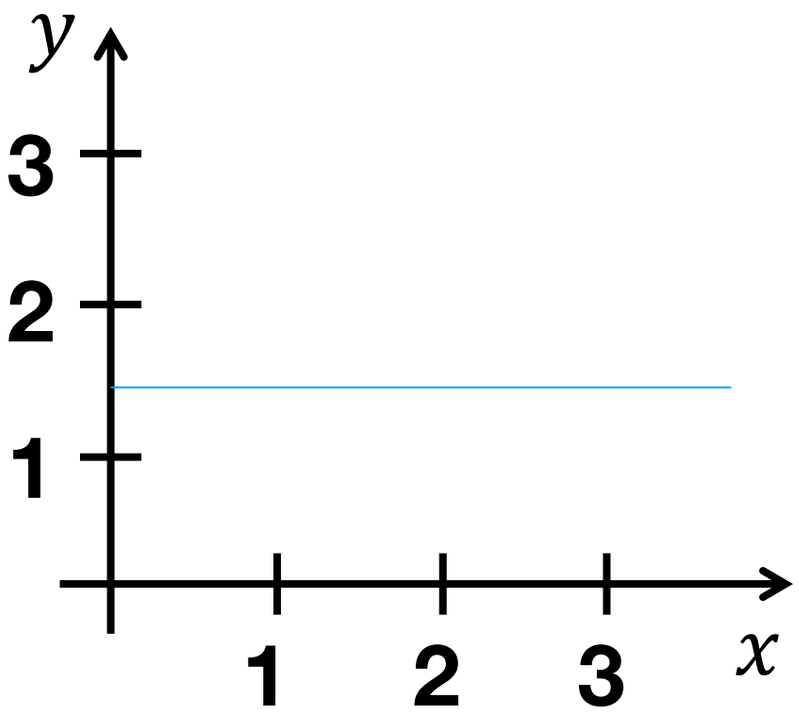
Linear Regression



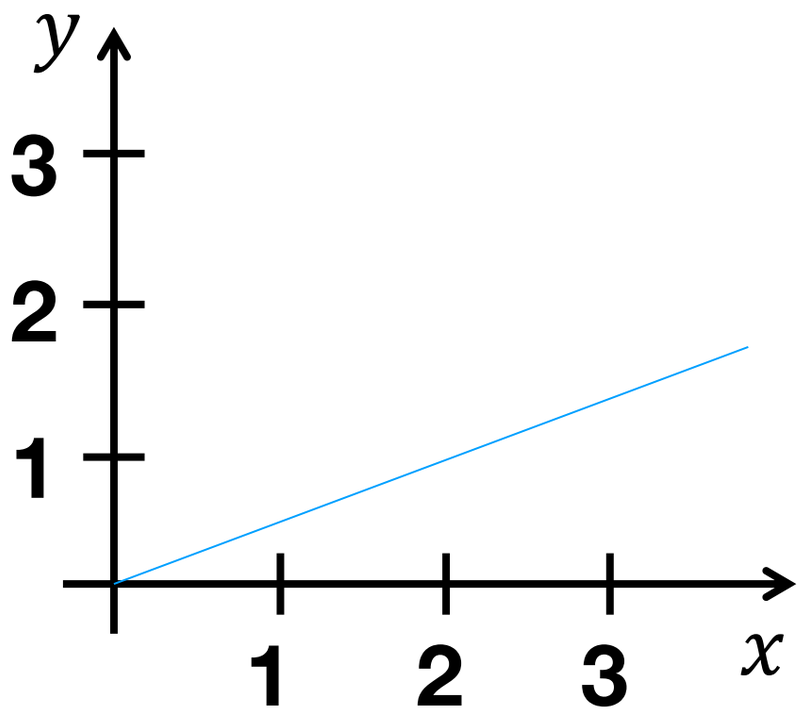
$$y = c + mx$$

But how well did the line fit?

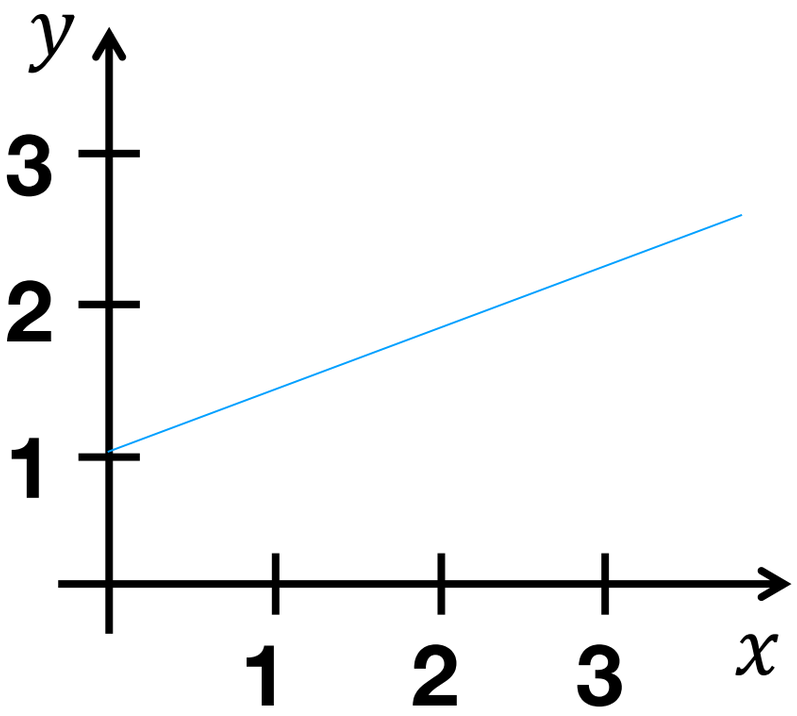
$$\mathbf{Y} = C + mx$$



C = 1.5
M = 0



c= 0
M = 0.5



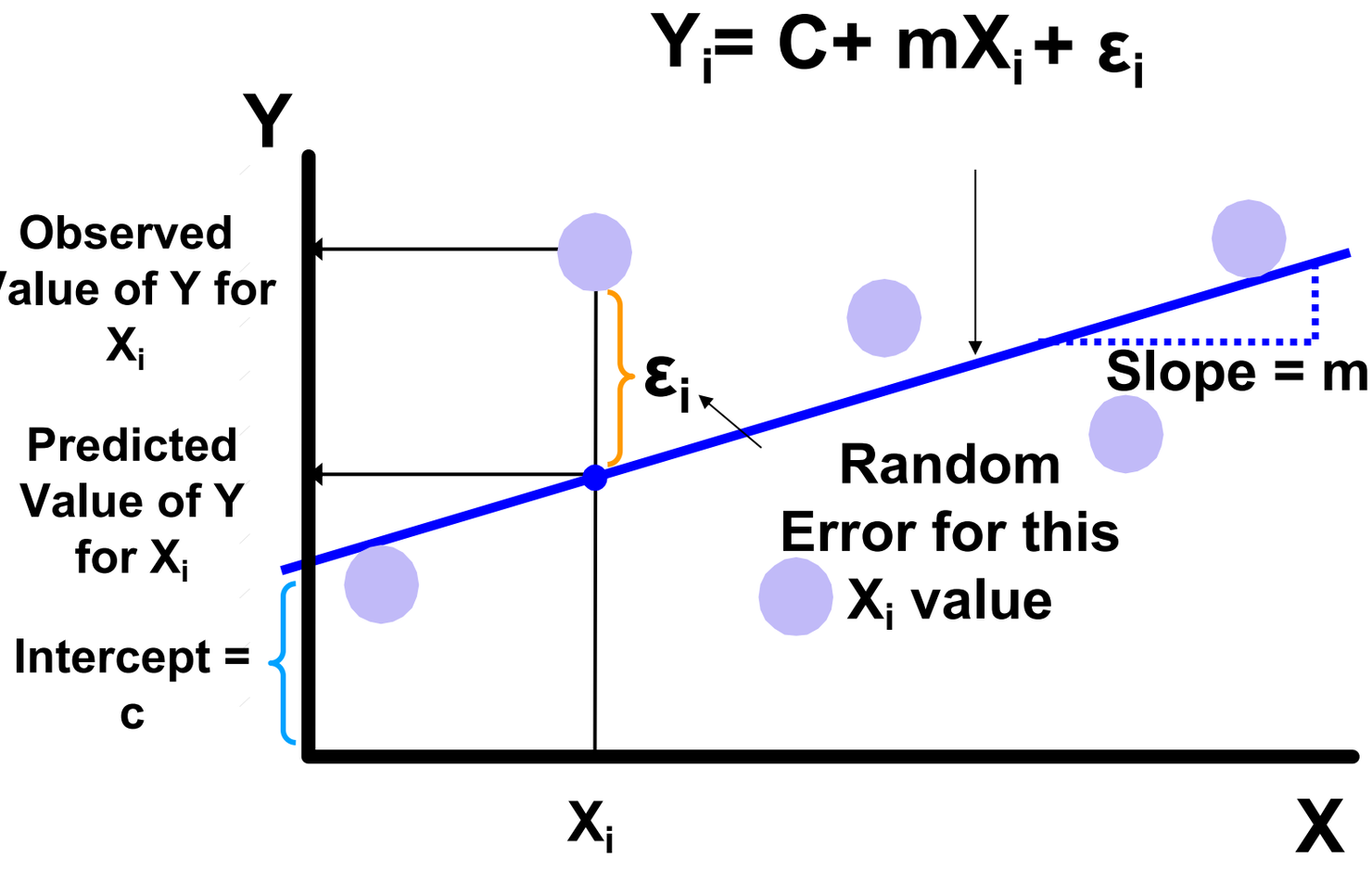
C = 1
M = 0.5

Simplest Possible Linear Regression Model

- This is the base model for <u>all</u> statistical machine learning
- *x* is a one feature data variable
- *y* is the value we are trying to predict
- ε is the unexplained, random, or error component.

Two parameters to estimate –

- the slope of the line m and the *y*-intercept c



Solving the regression problem

- We basically want to find {m, c} that minimize deviations from the predictor line

$$\text{Loss} = (y^i - mx^i - c)^2$$

Features

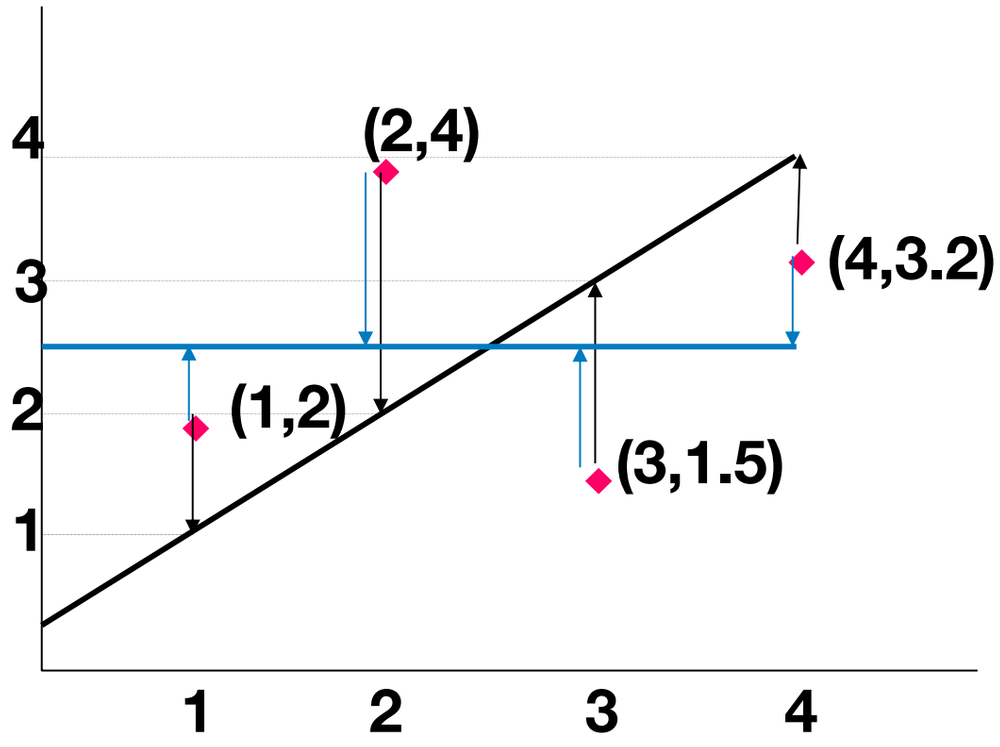
Observations

A good line is one that minimizes the sum of squared differences between the points and the line.

Example

Sum of squared differences = $(2 - 1)^2 + (4 - 2)^2 + (1.5 - 3)^2 + (3.2 - 4)^2 = 6.89$

Sum of squared differences = $(2 - 2.5)^2 + (4 - 2.5)^2 + (1.5 - 2.5)^2 + (3.2 - 2.5)^2 = 3.99$



Let us compare two lines

The second line is horizontal

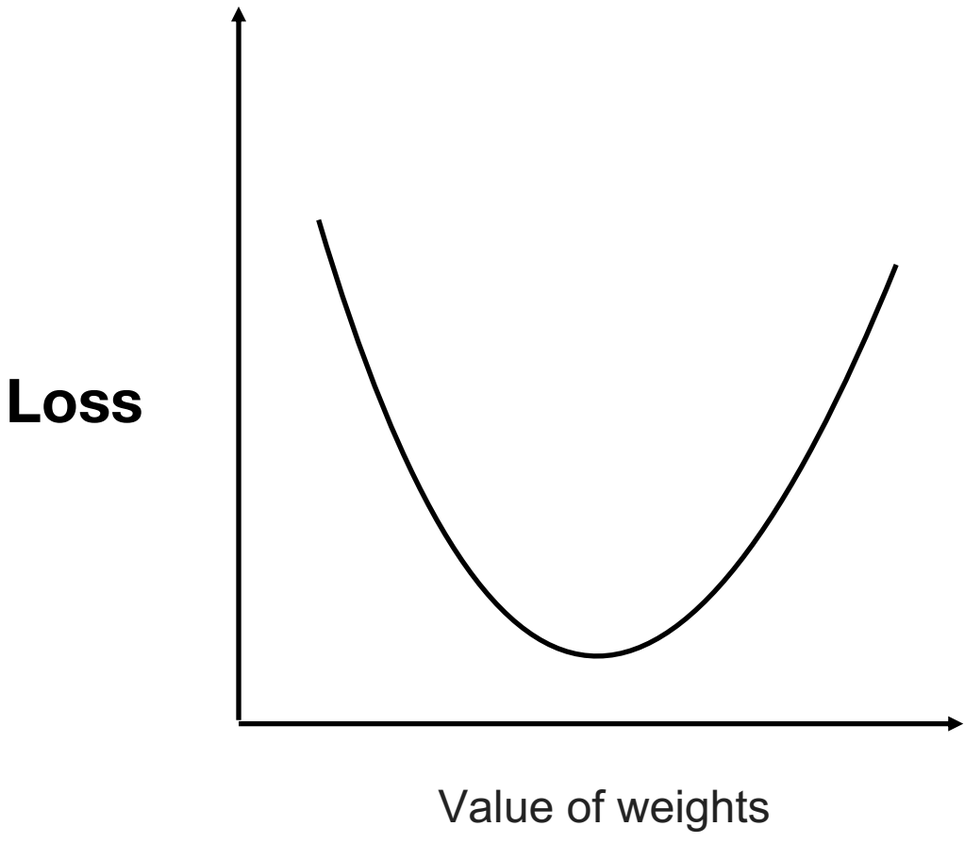
The smaller the sum of squared differences the better the fit of the line to the data.

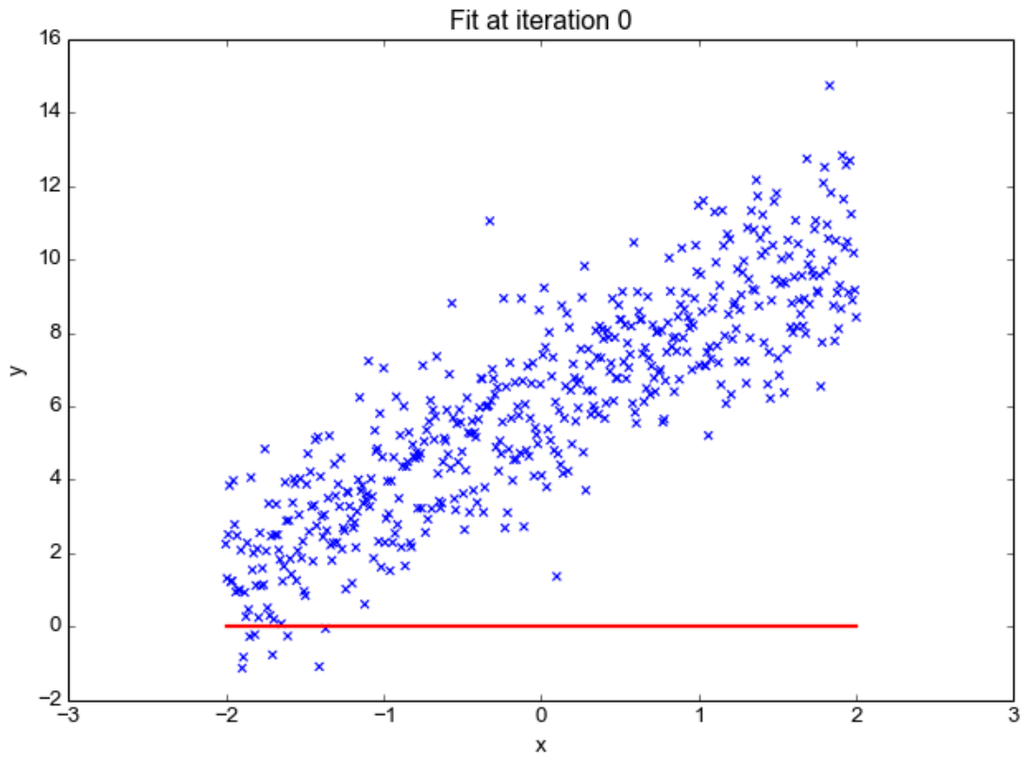
- How do we do it automatically?
  - **Iterate over all possible m and values along the two dimensions?**
  - **Something more smarter – Gradient Descent**

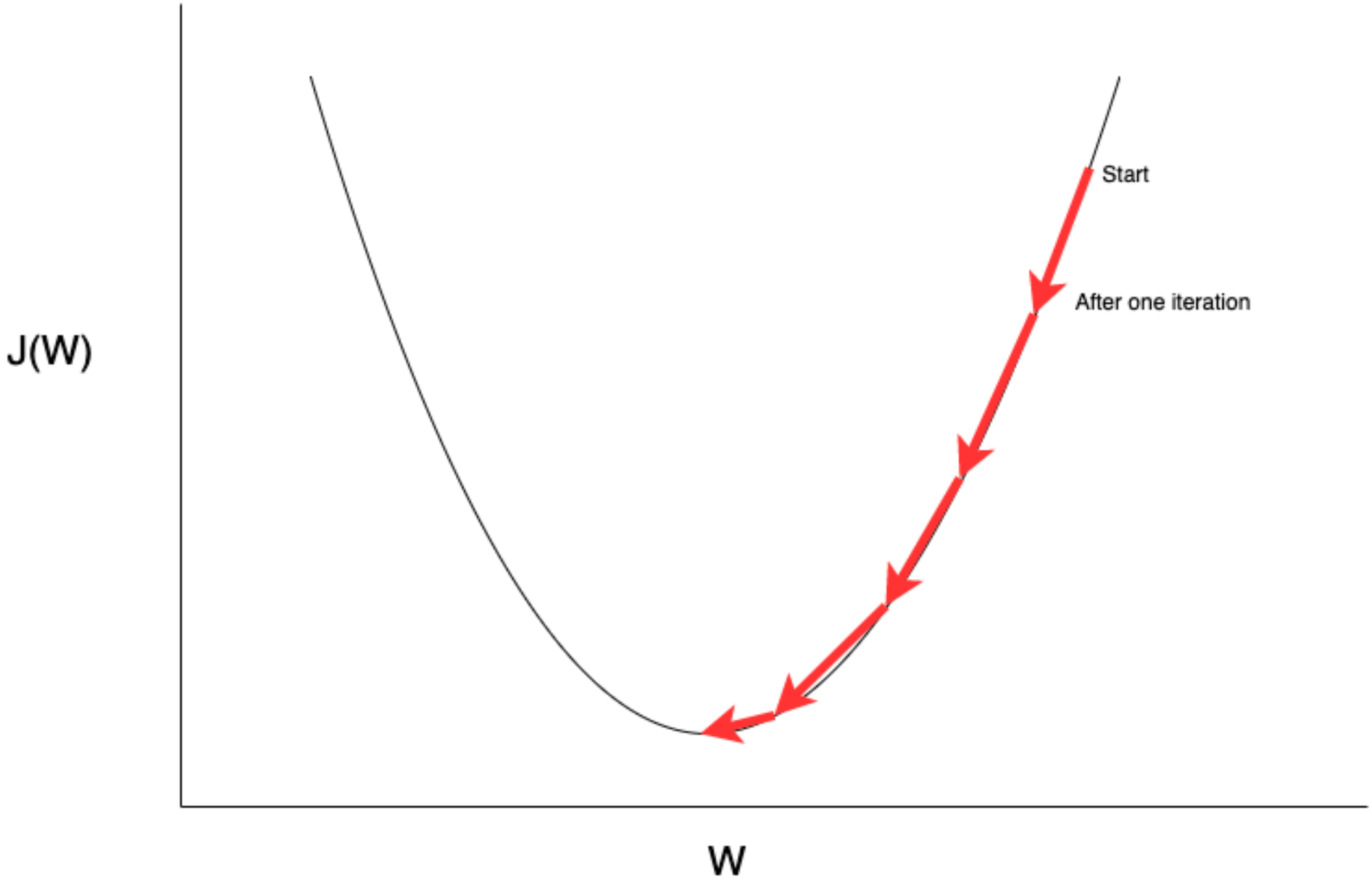
$$\text{Loss} = \frac{1}{n}\sum_{i=1}^{n} (y^i - mx^i - c)^2$$

Features

Observations

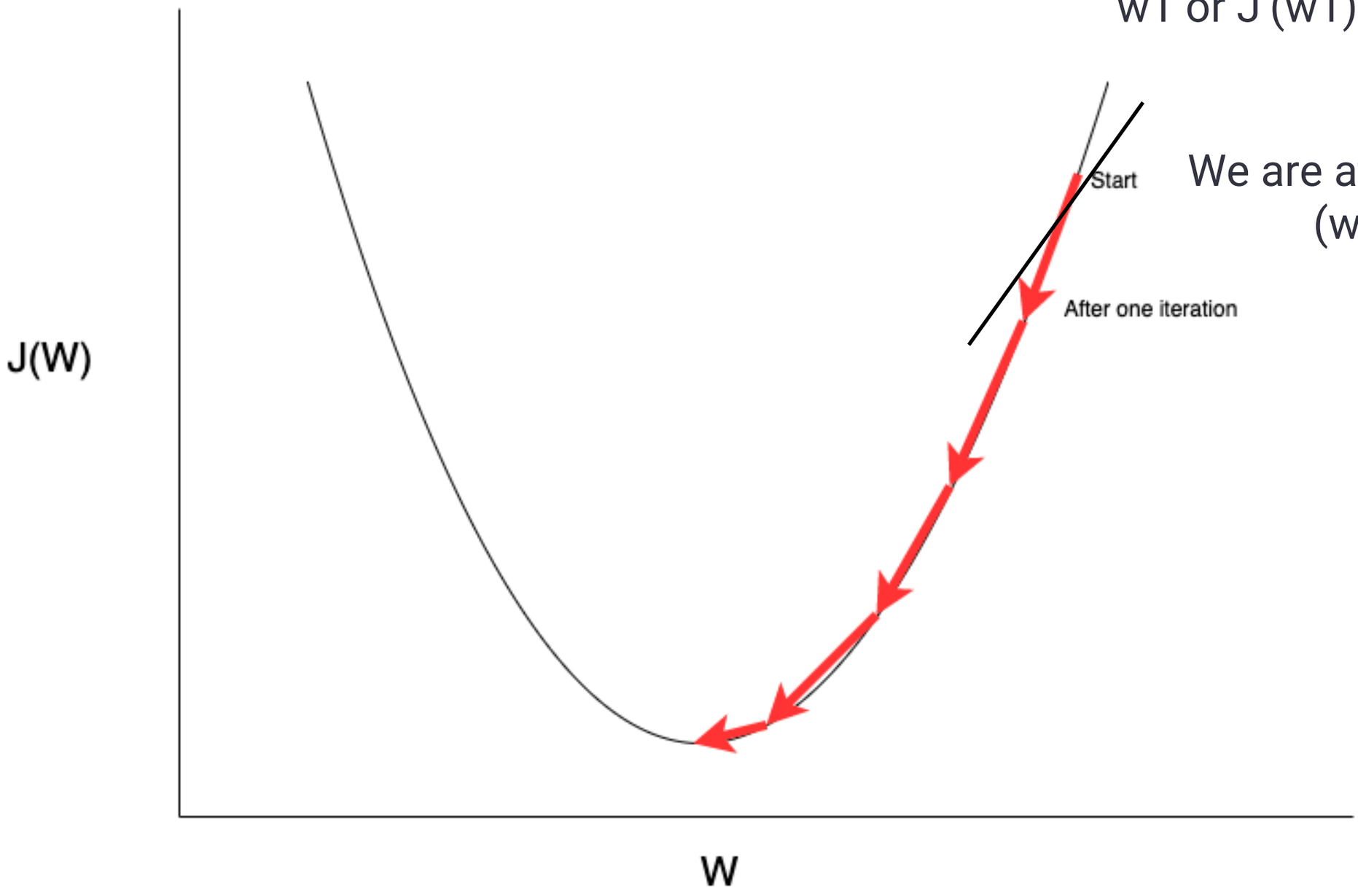






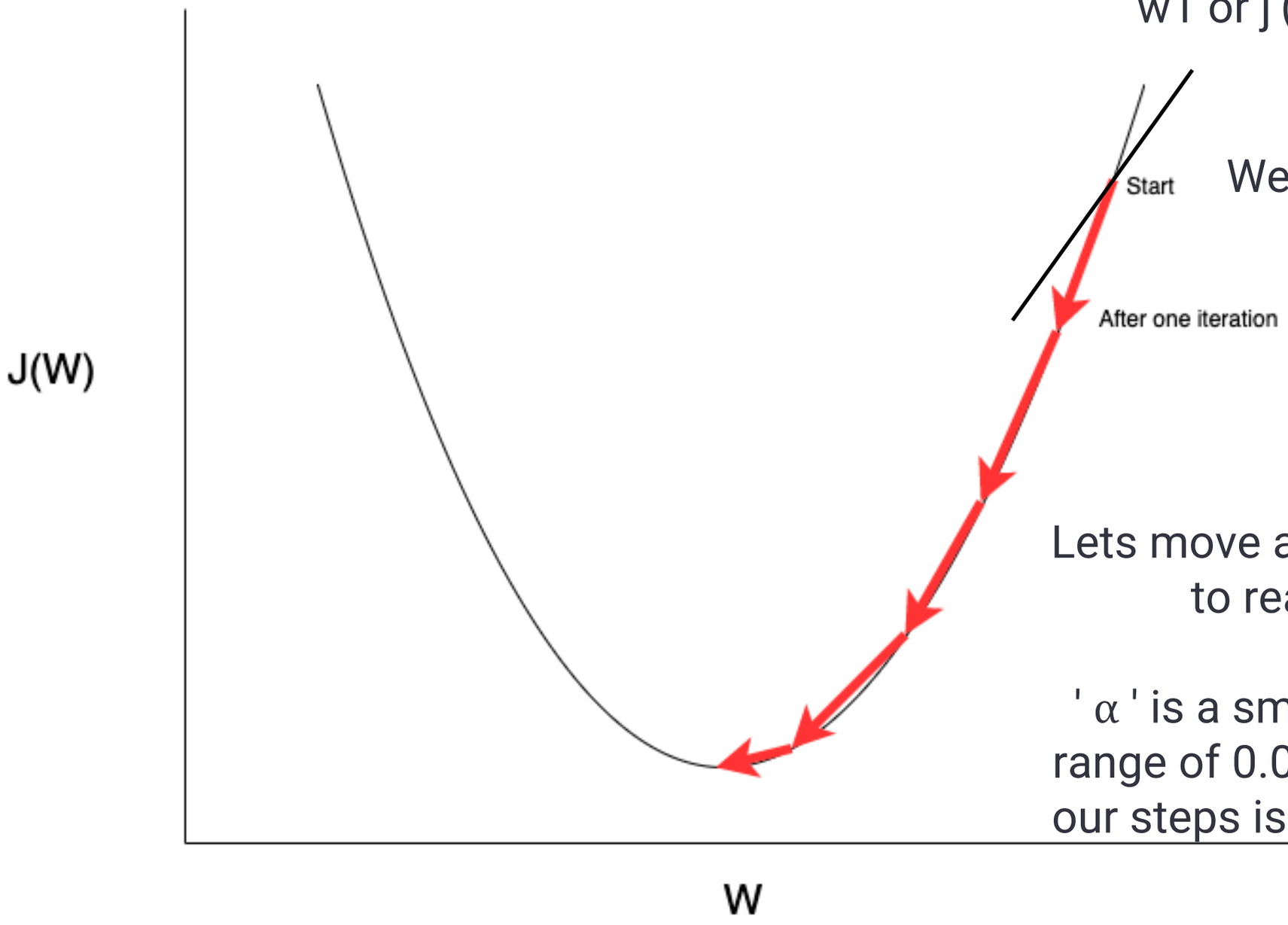
This curve J (w) represents the values I will assume for different values of ‘w’ , which represent the weights

Lets calculate slope at this point. It is dj/dw at w1 or J’(w1)



We are at this point now (w1, J(w1))

Lets calculate slope at this point. It is dj/dw at w1 or j’(w1) al

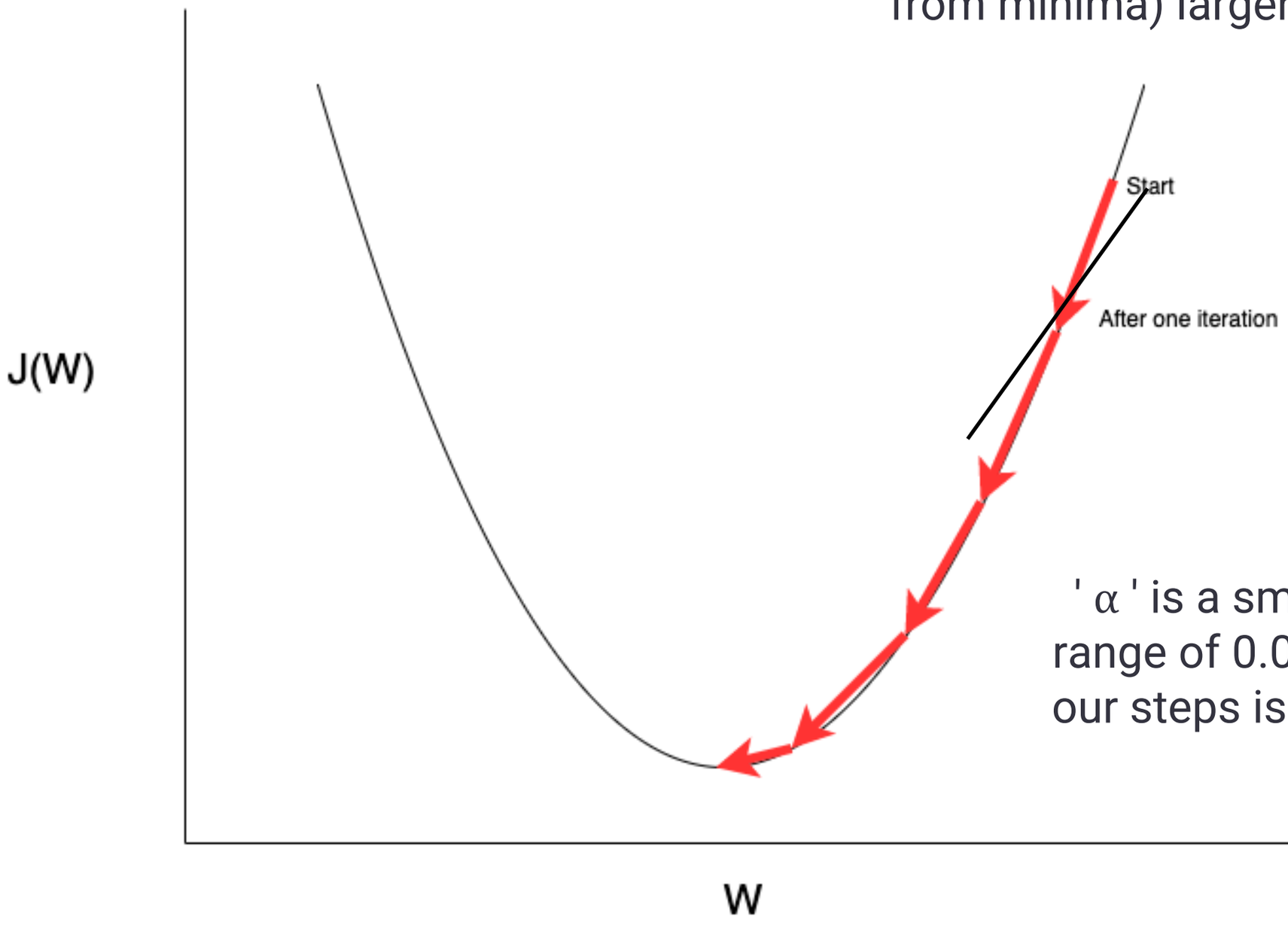


We are at this point now (w1, j(w1))

Lets move a step ‘α' in this direction to reach w1- α *J’(w1)

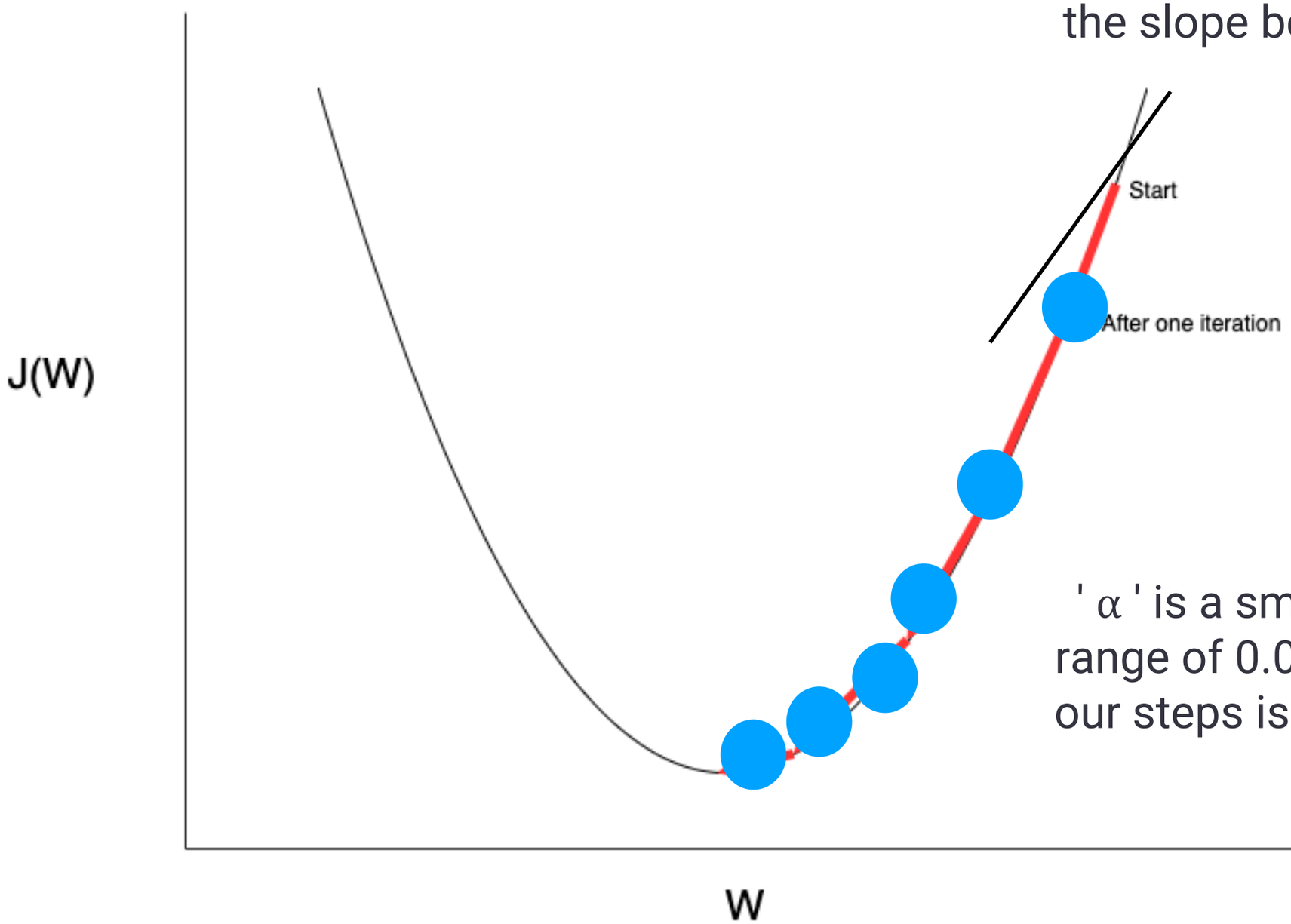
' α ' is a small fixed quantity in the range of 0.01. Therefore, the size of our steps is dependent on the slope J’(w1).

Higher the value of slope (which occurs away from minima) larger the steps and vice versa



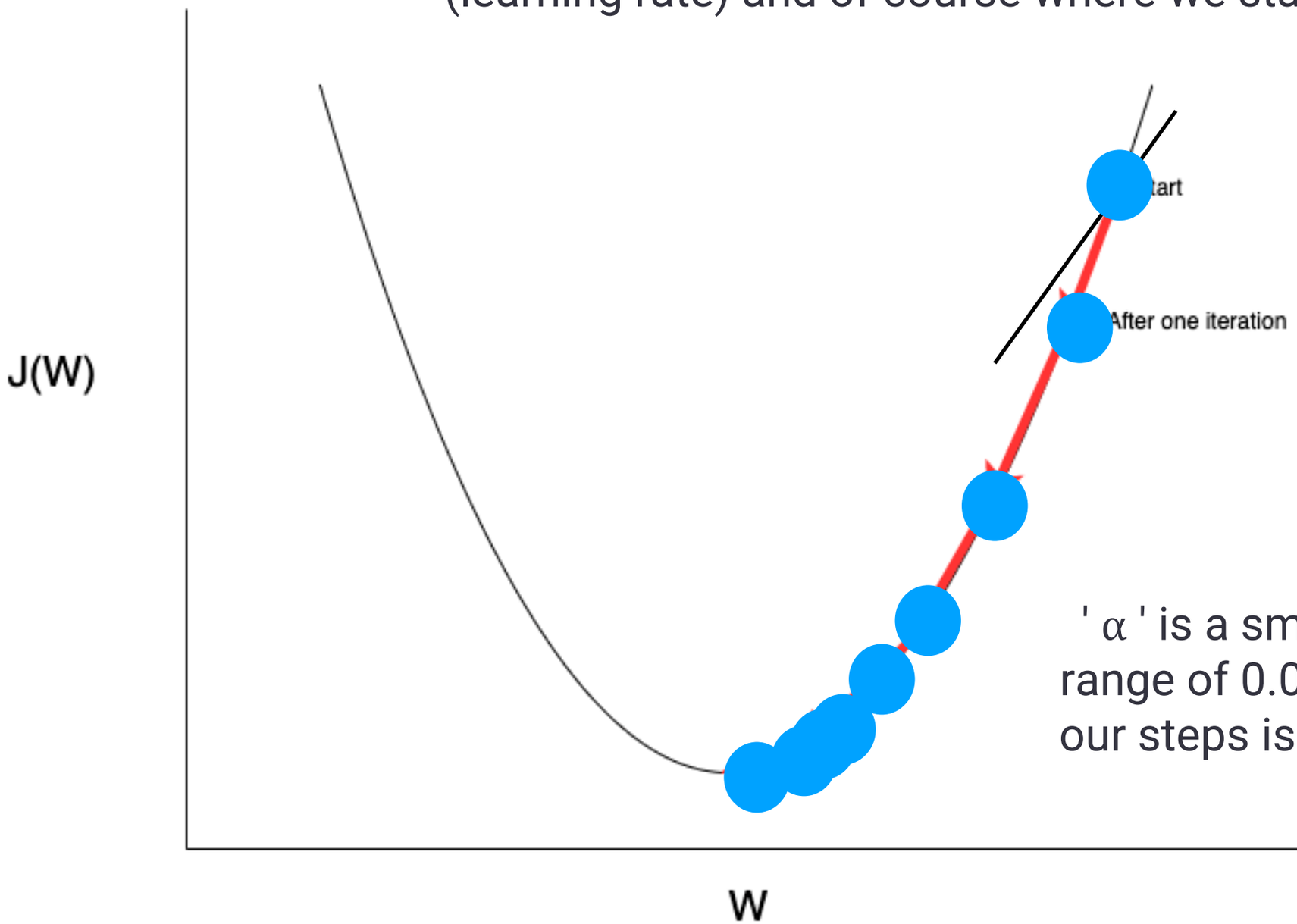
' α ' is a small fixed quantity in the range of 0.01. Therefore, the size of our steps is dependent on the slope J’(w1).

As we move closer to the minimum, the slope decreases and hence our steps towards minimum keeps getting smaller. At minimum the slope becomes zero



' α ' is a small fixed quantity in the range of 0.01. Therefore, the size of our steps is dependent on the slope J’(w1).

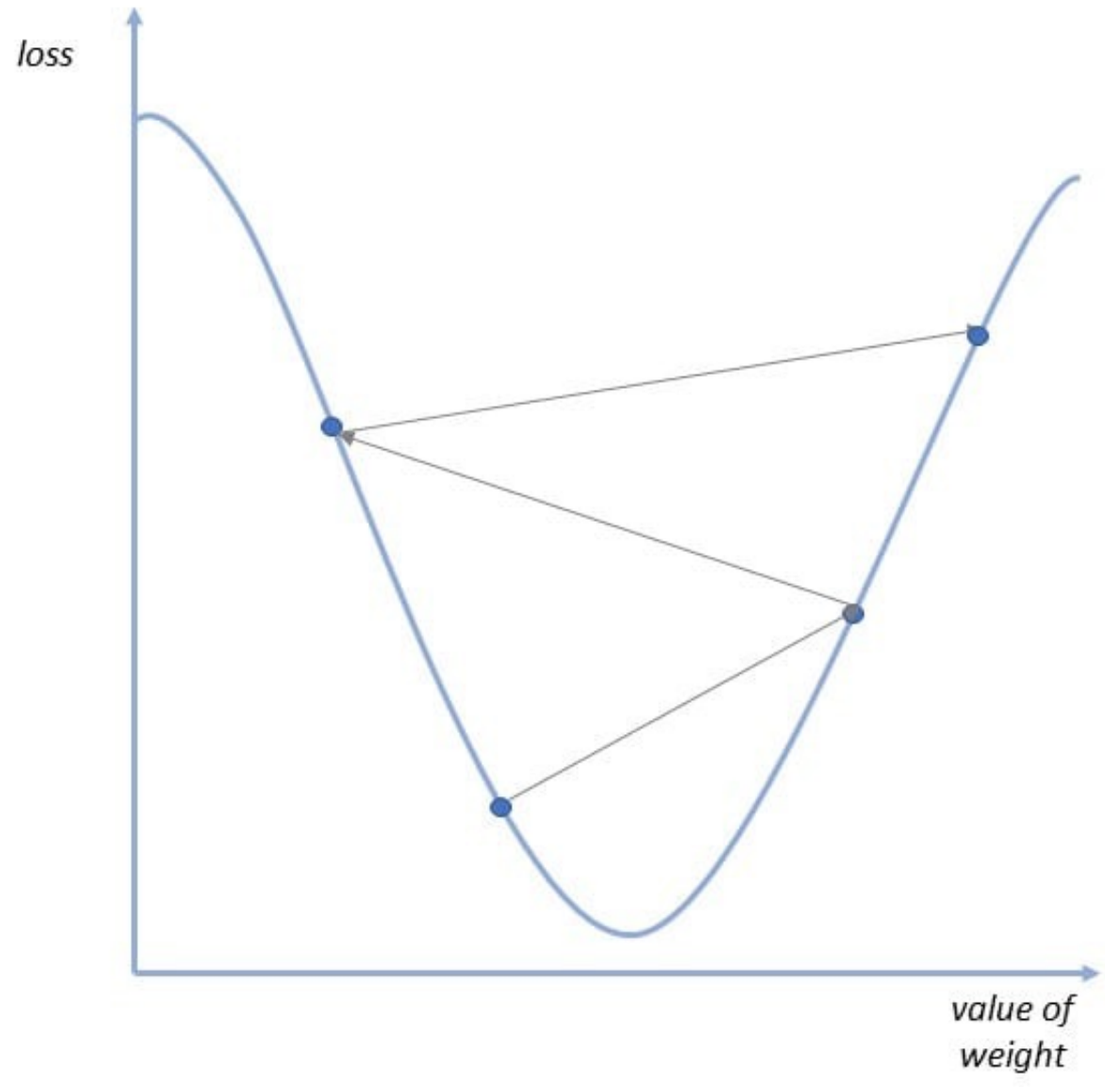
These iteration of Gradient Descent algorithm can run in 1000s or 10000s of steps depending on nature of function and our ' α ' (learning rate) and of course where we start from, ‘w1' in this case



' α ' is a small fixed quantity in the range of 0.01. Therefore, the size of our steps is dependent on the slope J’(w1).

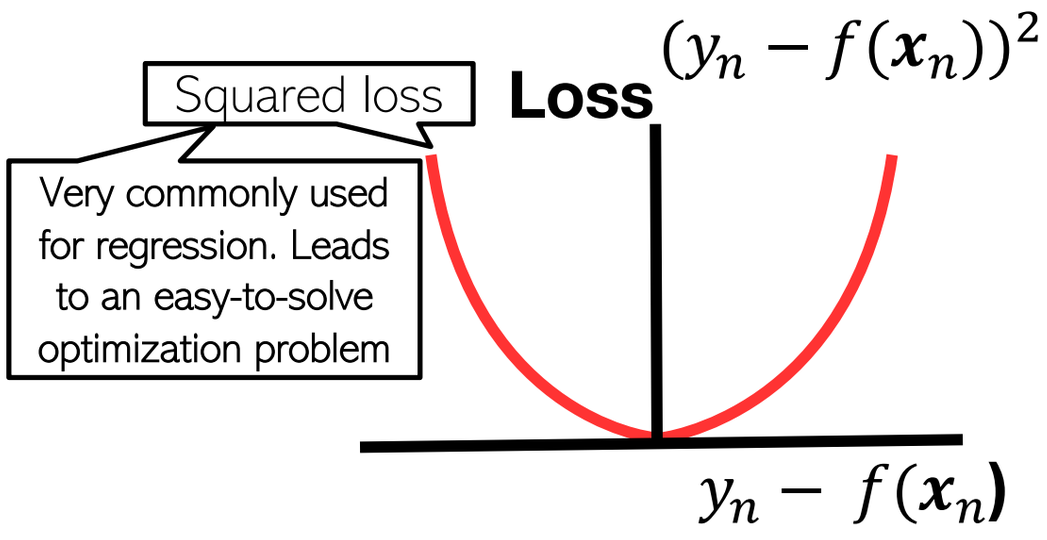
What happens if α is large, say 100?

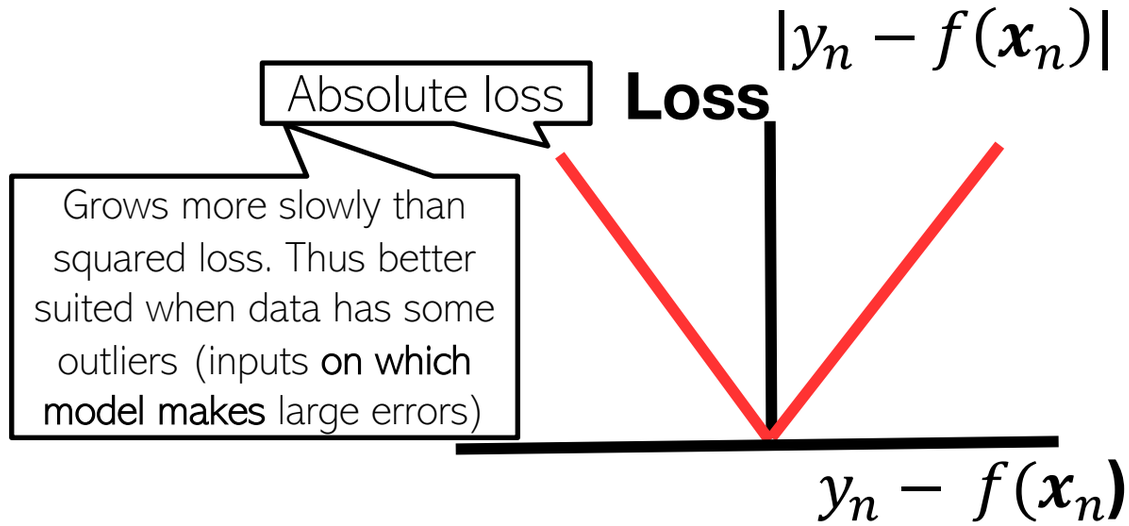
Large Learning Rate



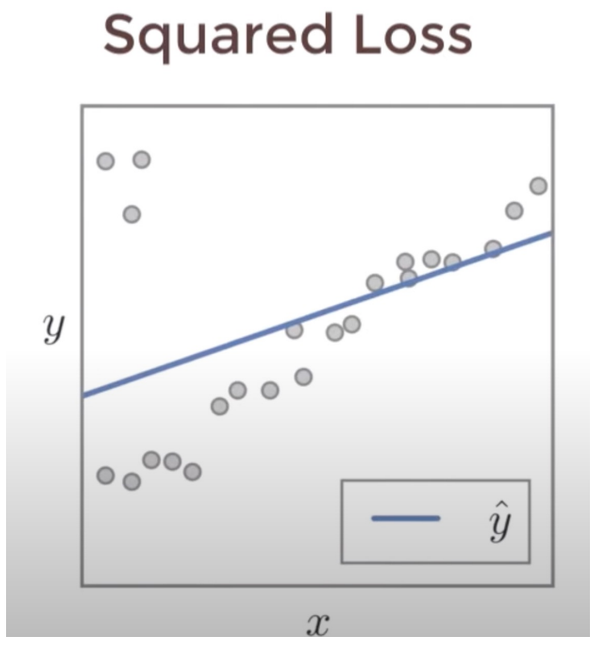
Good News: There are inbuilt packages that implement all of this in an optimized manner
so we don’t have to worry about the math behind it

Loss functions





Many possible loss functions for regression problems



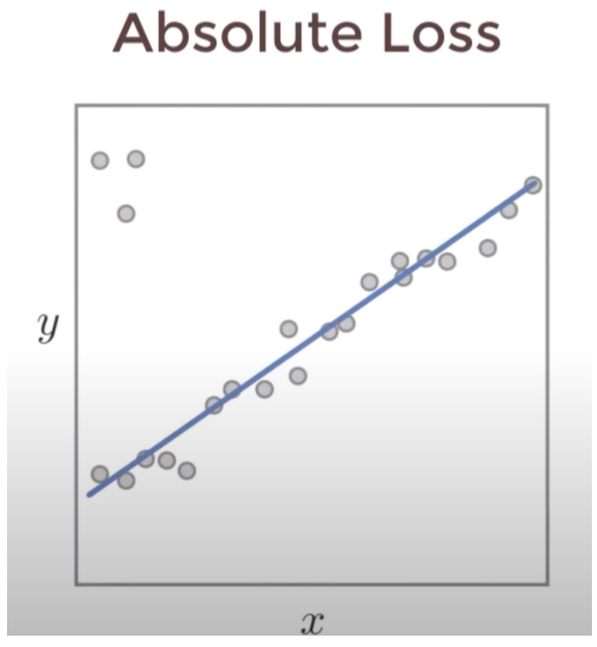
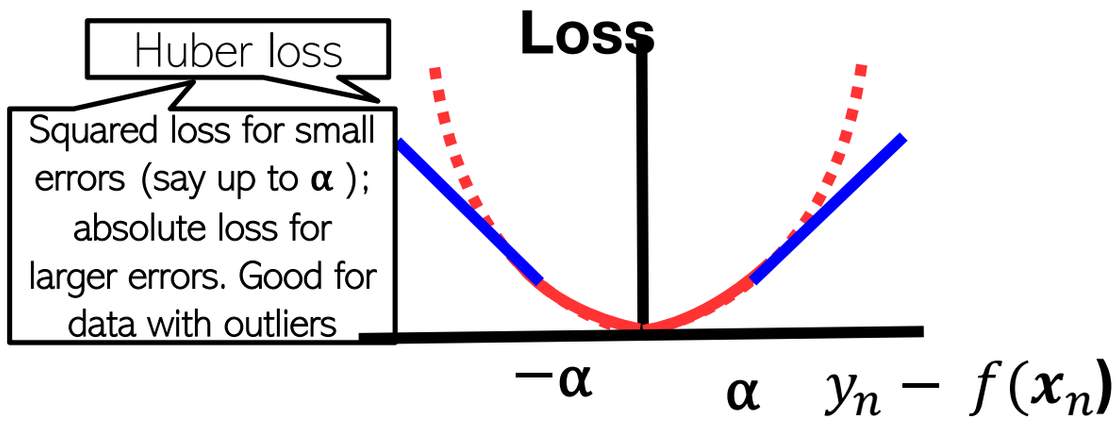


Image Credit: CodeEmporium

Alternative loss functions



Pseudo-Huber Loss $= \begin{cases} (y-\hat{y})^2 & ; |y-\hat{y}| \leq \alpha \\ |y-\hat{y}| & ; otherwise \end{cases}$

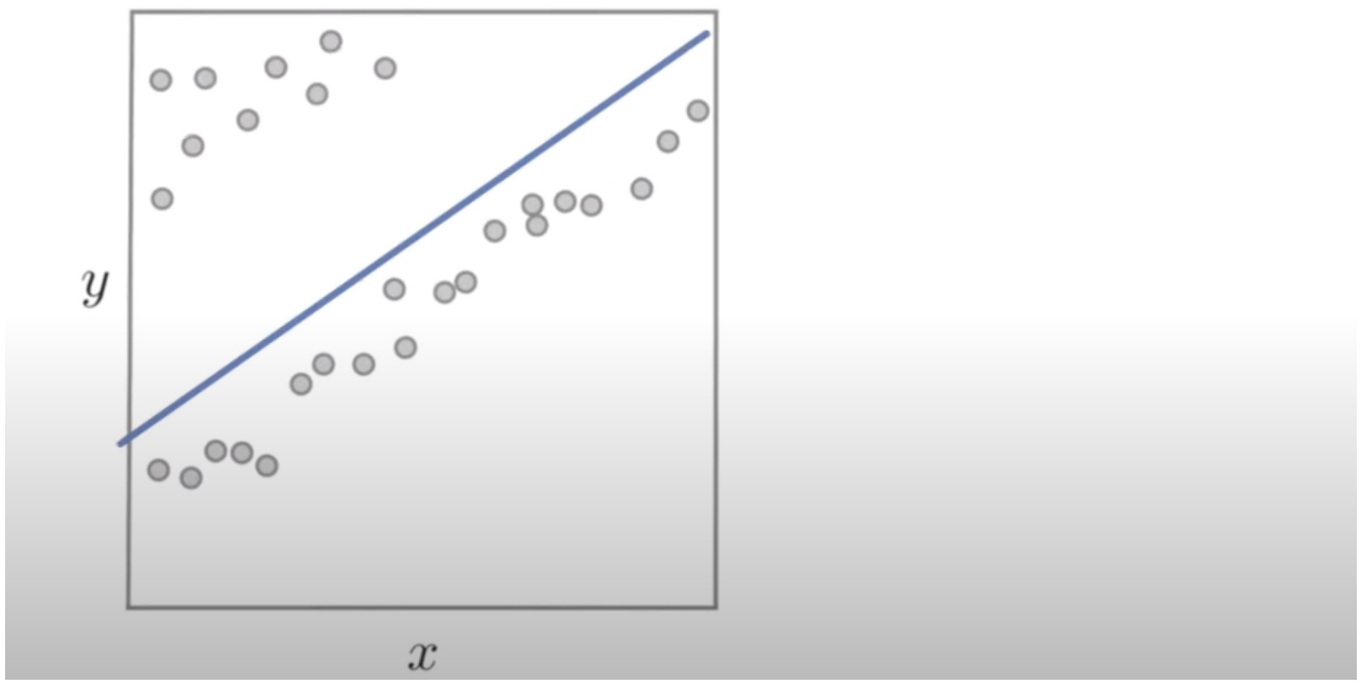
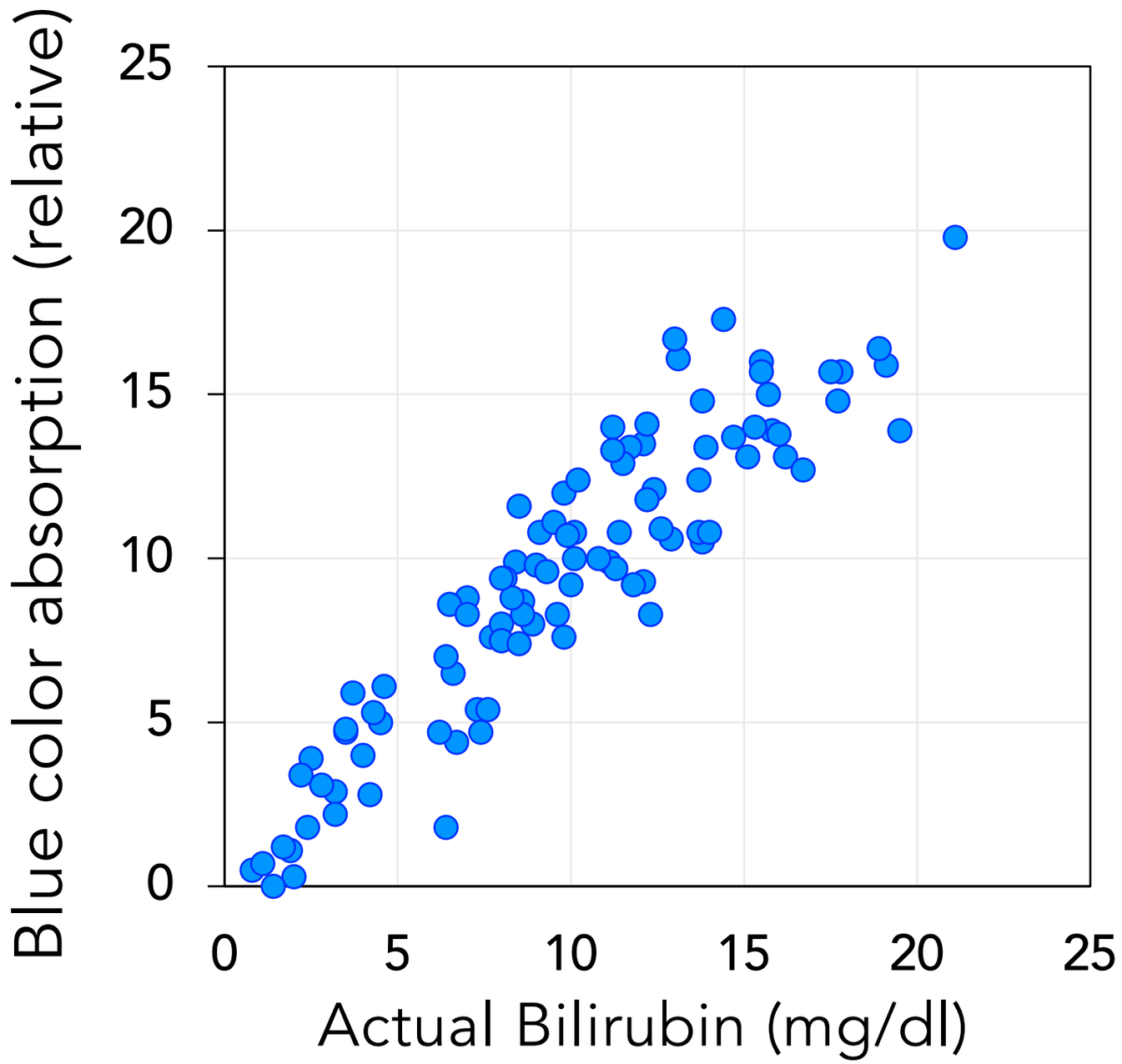
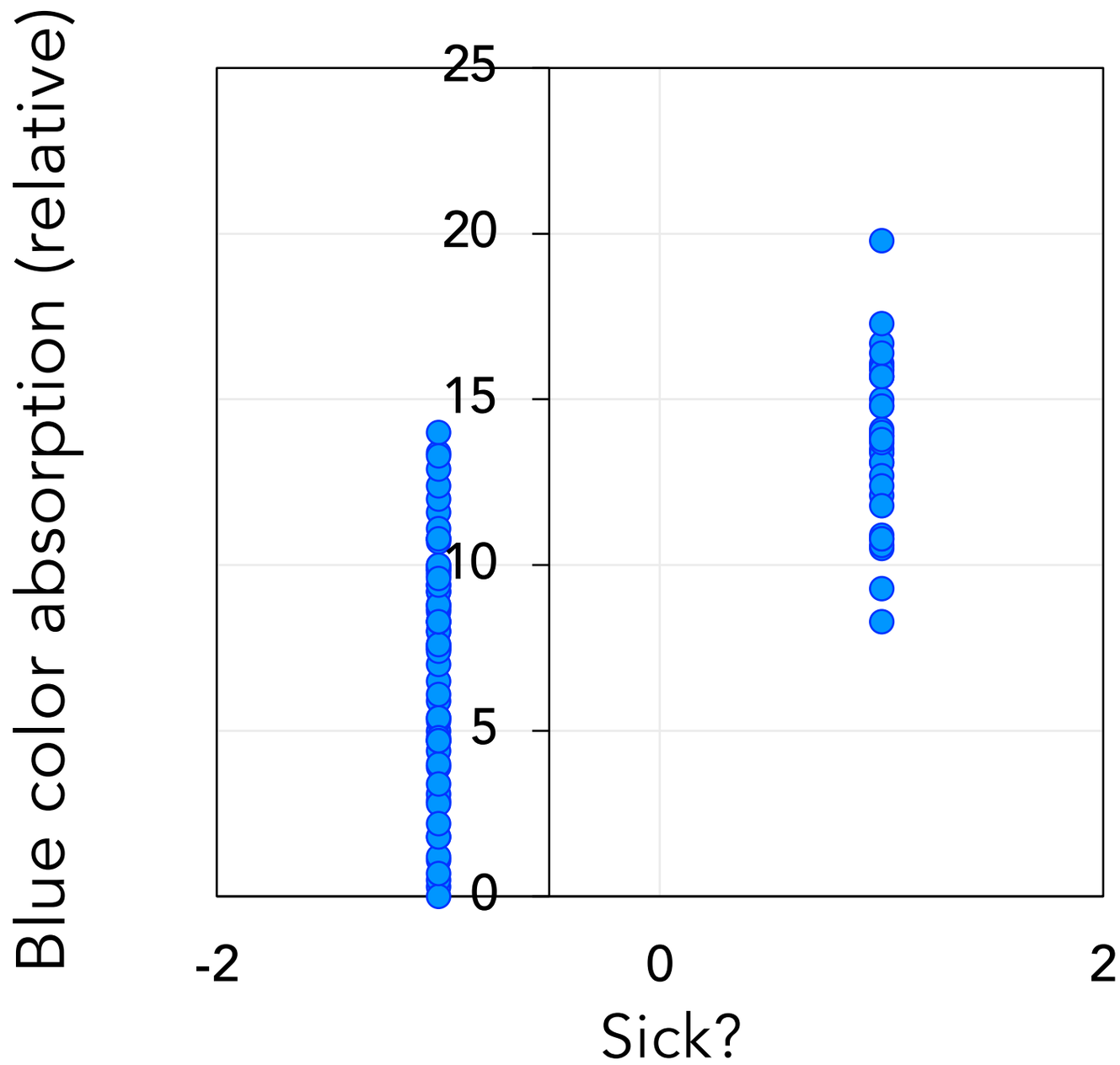


Image Credit: CodeEmporium

What about classification?

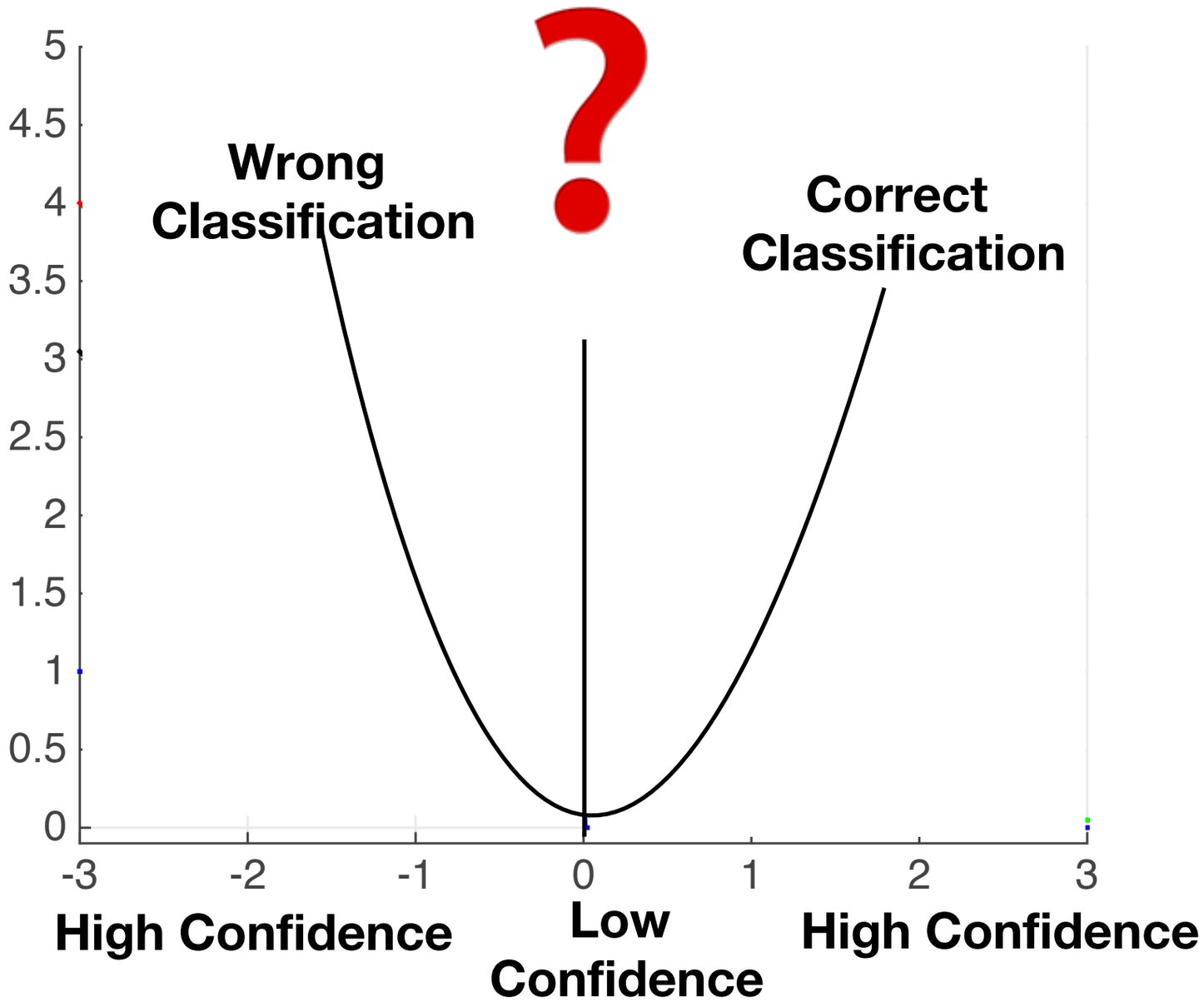


What about classification?

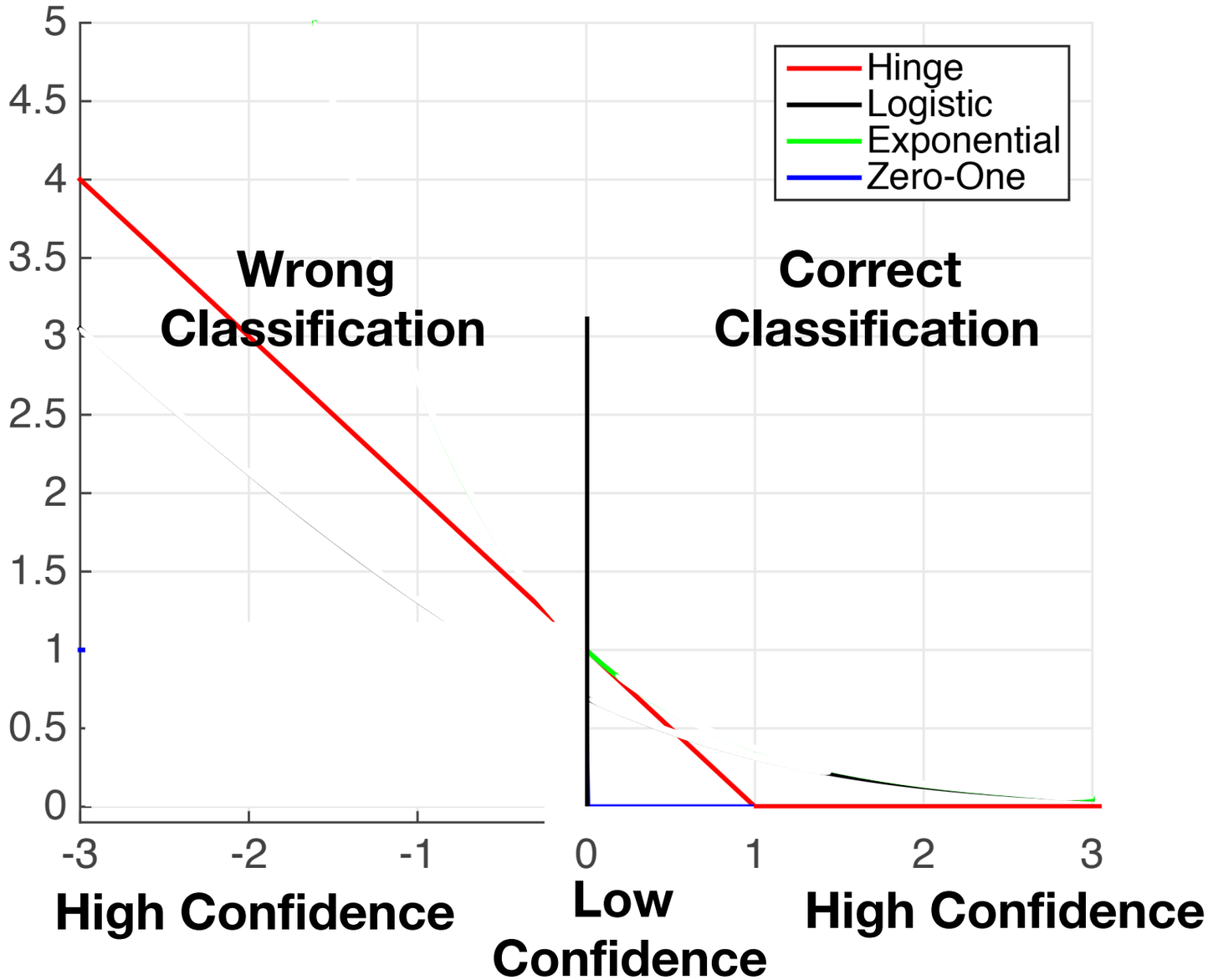


Can we use the same loss function as regression for classification?

Loss Functions for Classification Tasks

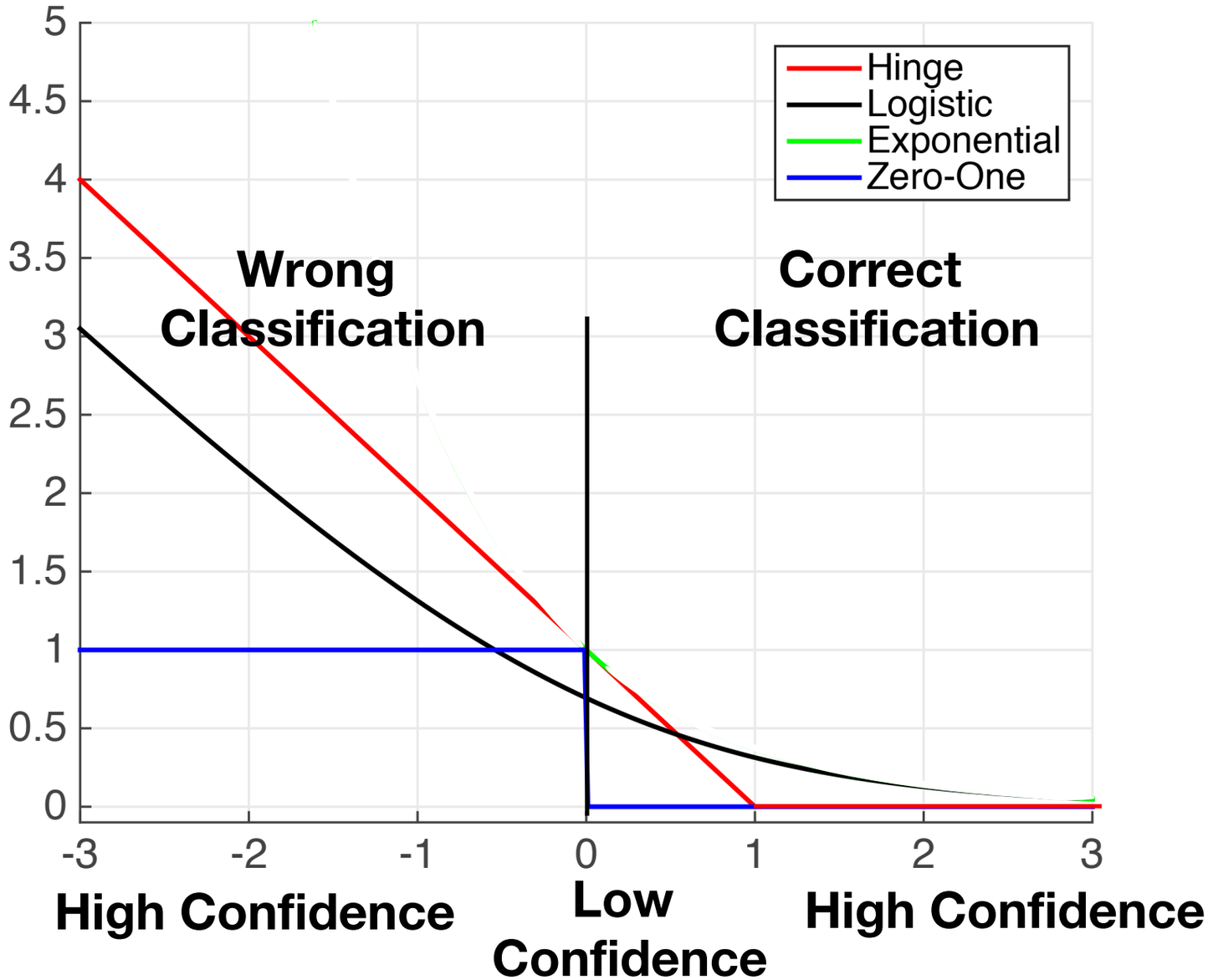


Loss Functions for Classification Tasks



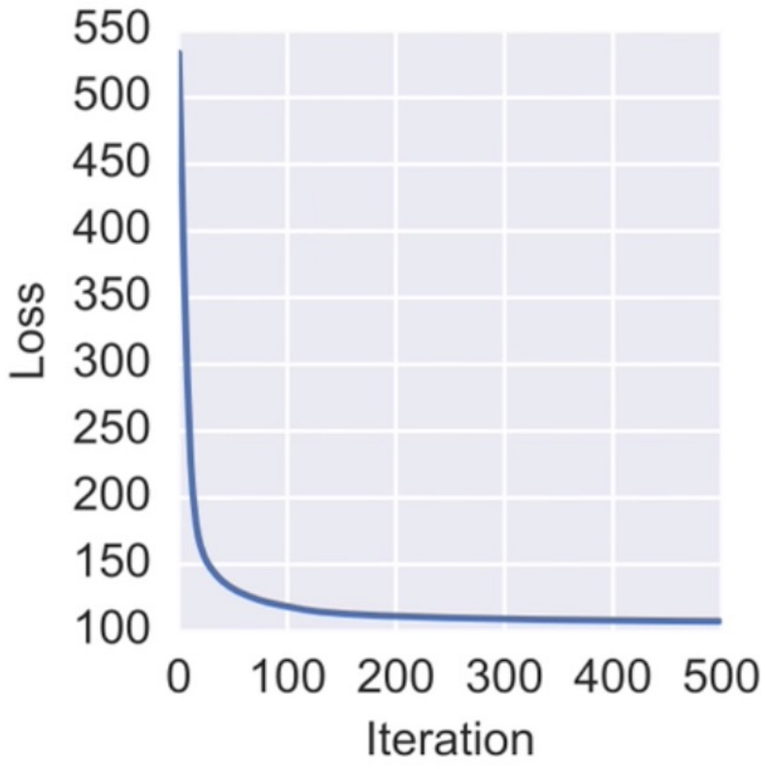
Graph courtesy: Weinberger, Cornell University

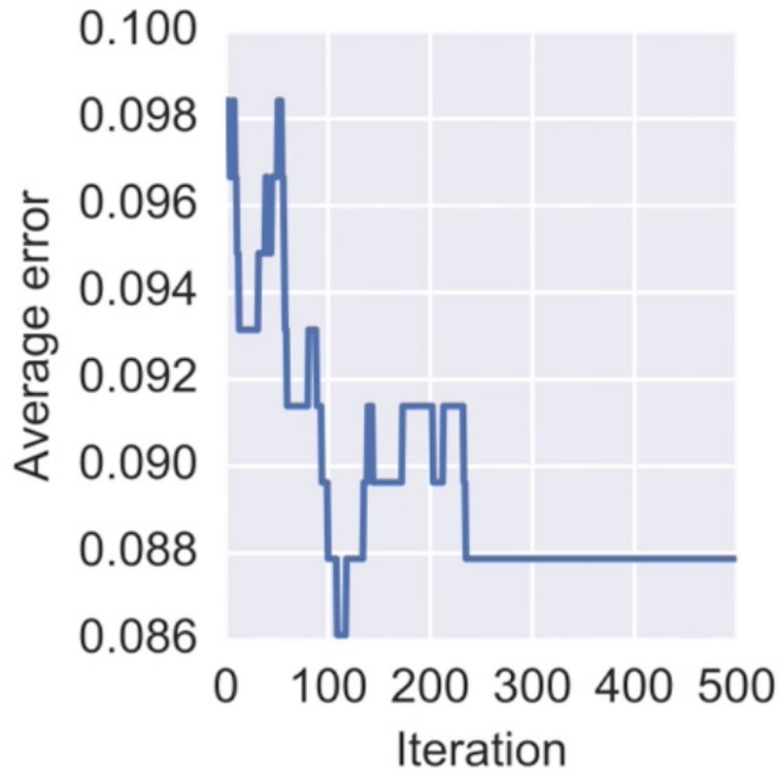
Loss Functions for Classification Tasks



Graph courtesy: Weinberger, Cornell University

Loss *vs*. Error

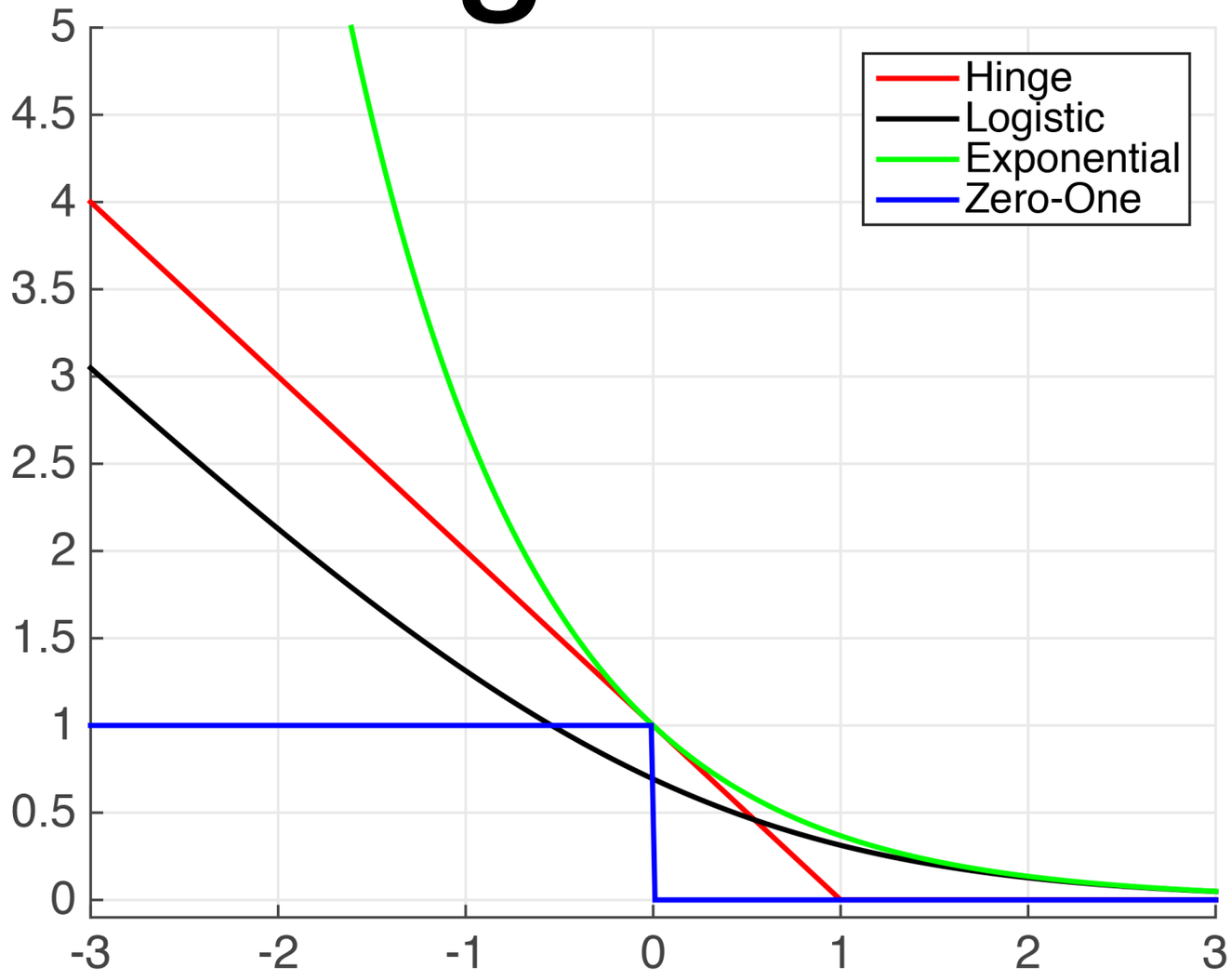




When do you stop iterating?

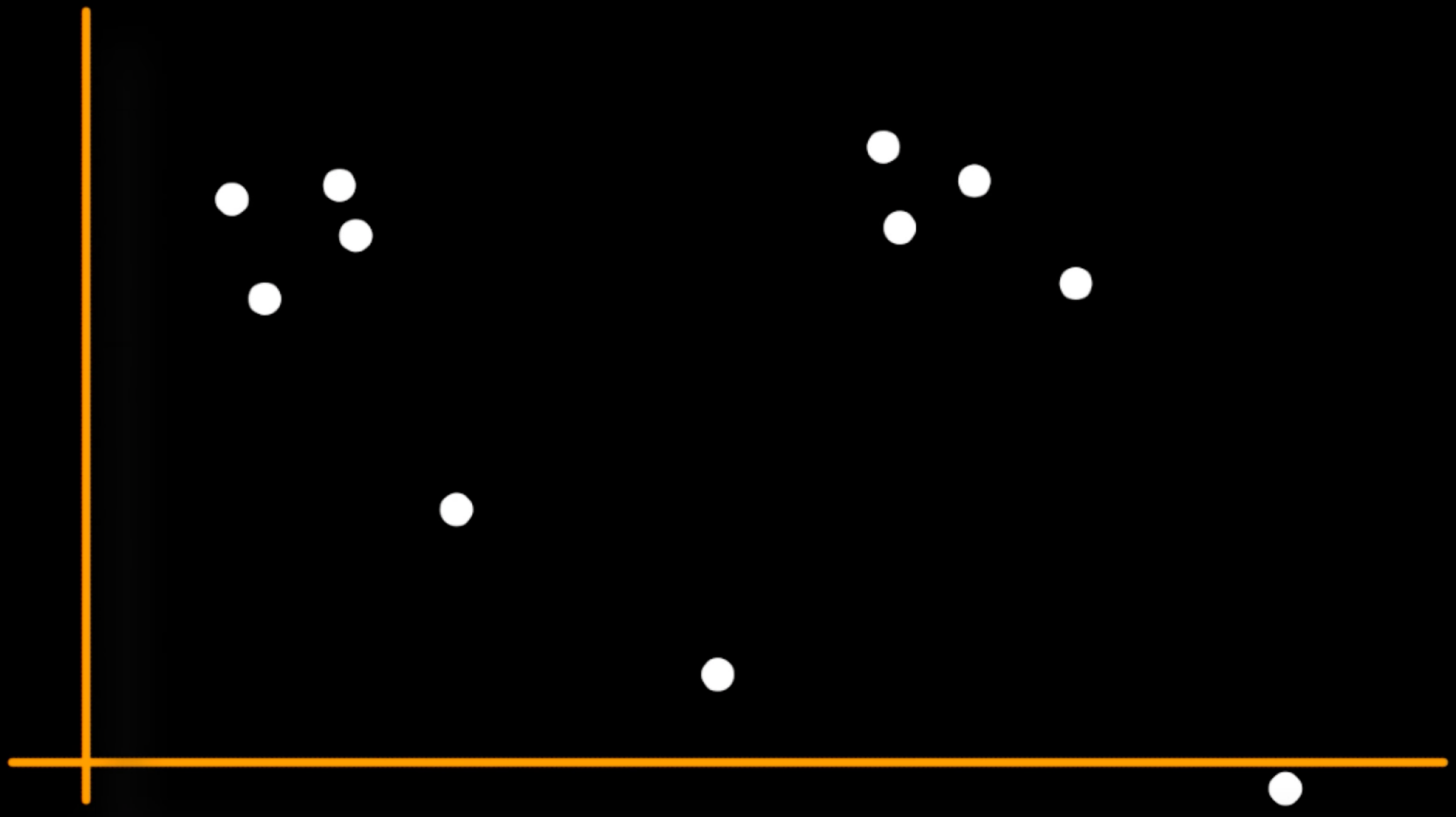
Graph courtesy: Zico Kolter, CMU

Common Classification Algorithms



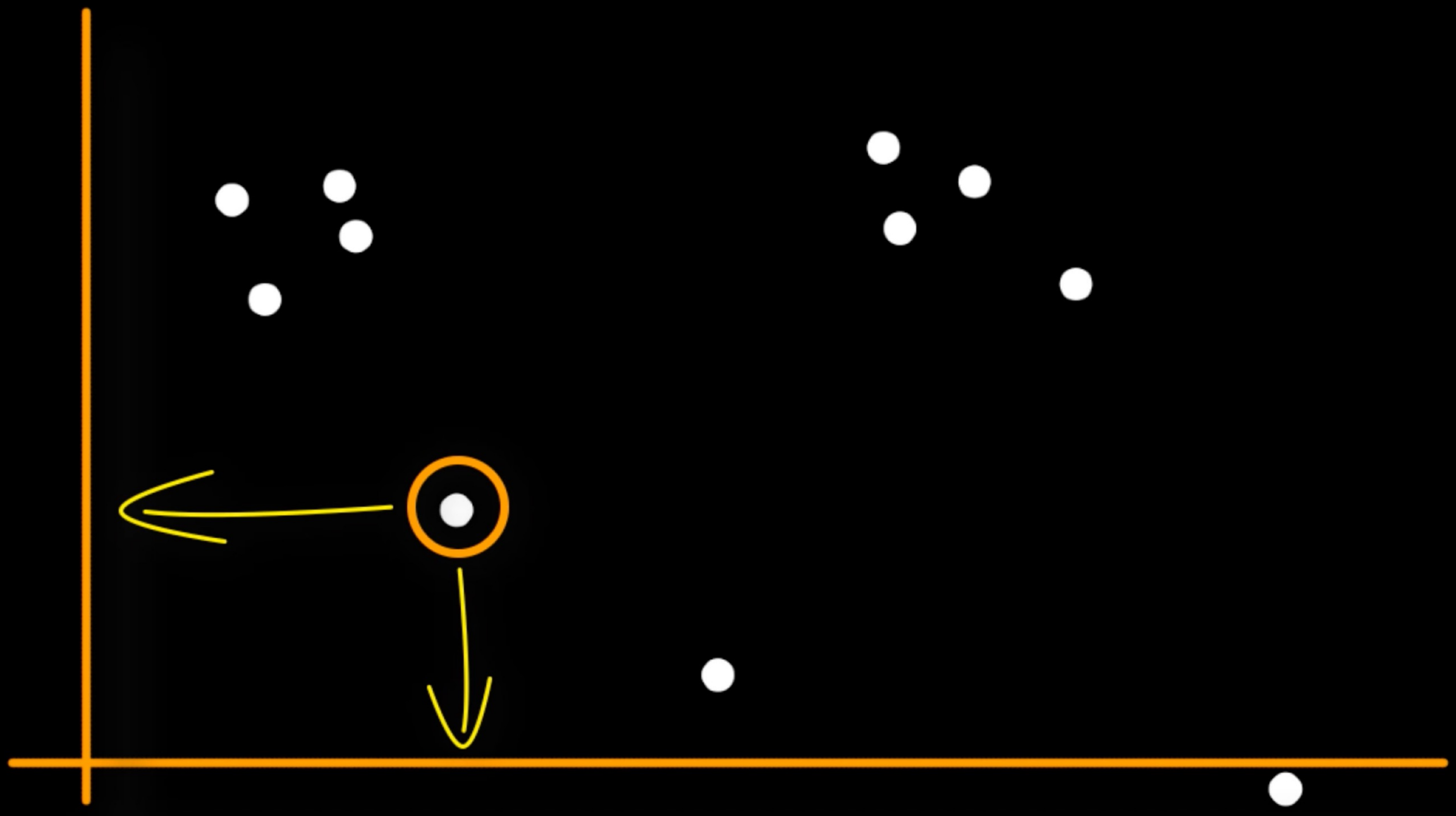
- Support Vector Machines - Hinge Loss
- Logistic Regression - Logistics Loss

Support Vector Machines



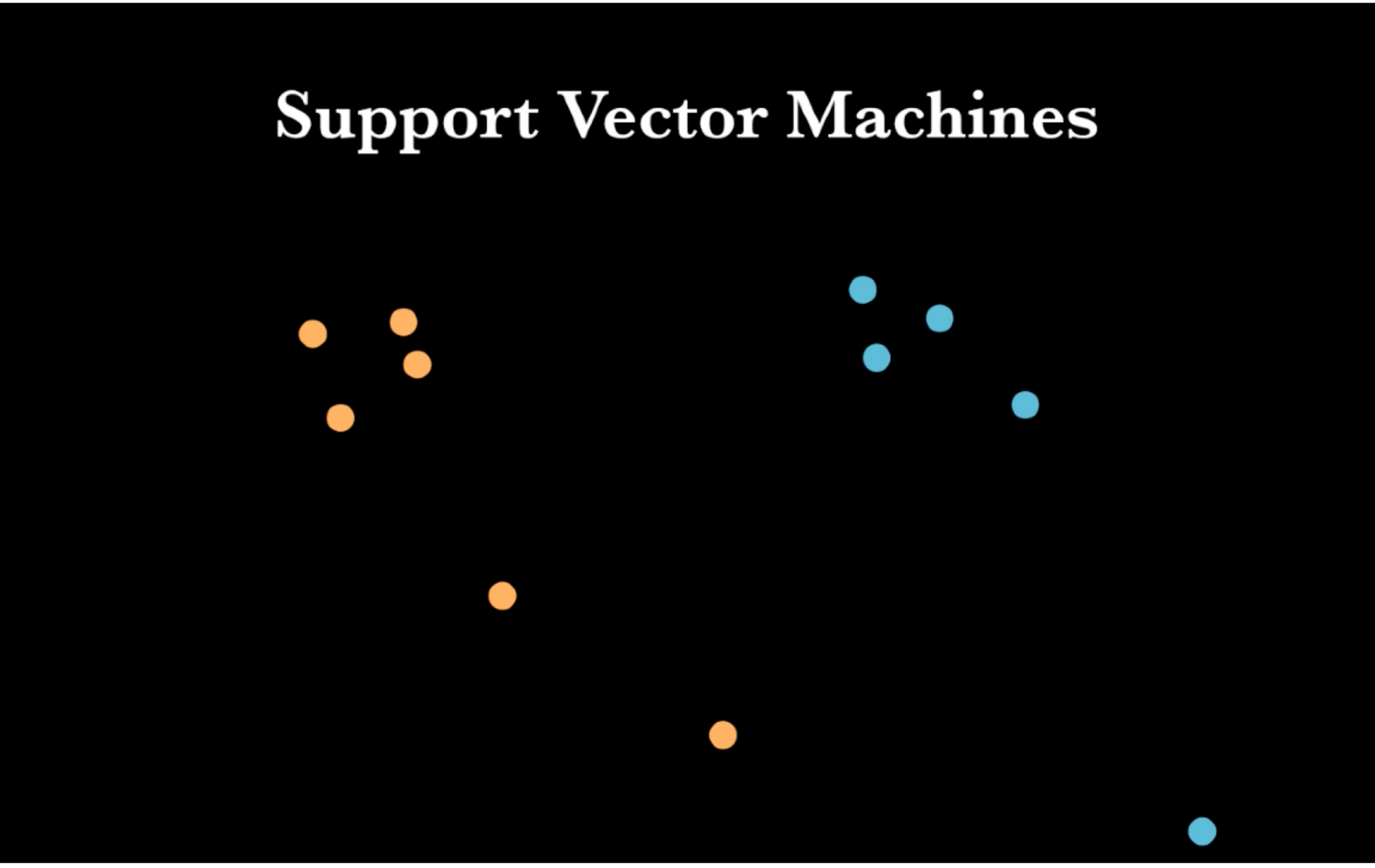
Credit: Greene, CU Boulder

Support Vector Machines



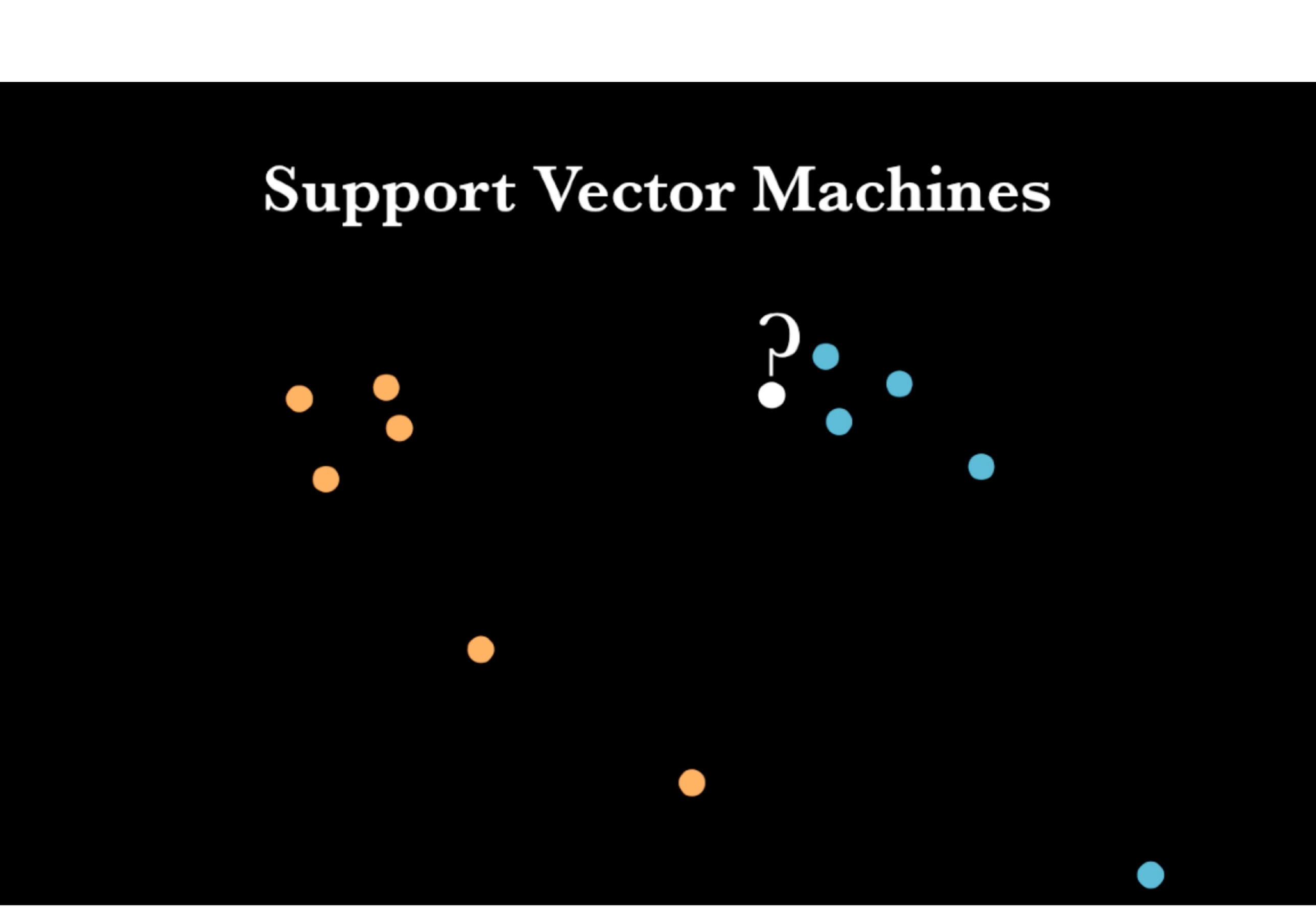
Credit: Greene, CU Boulder

Support Vector Machines



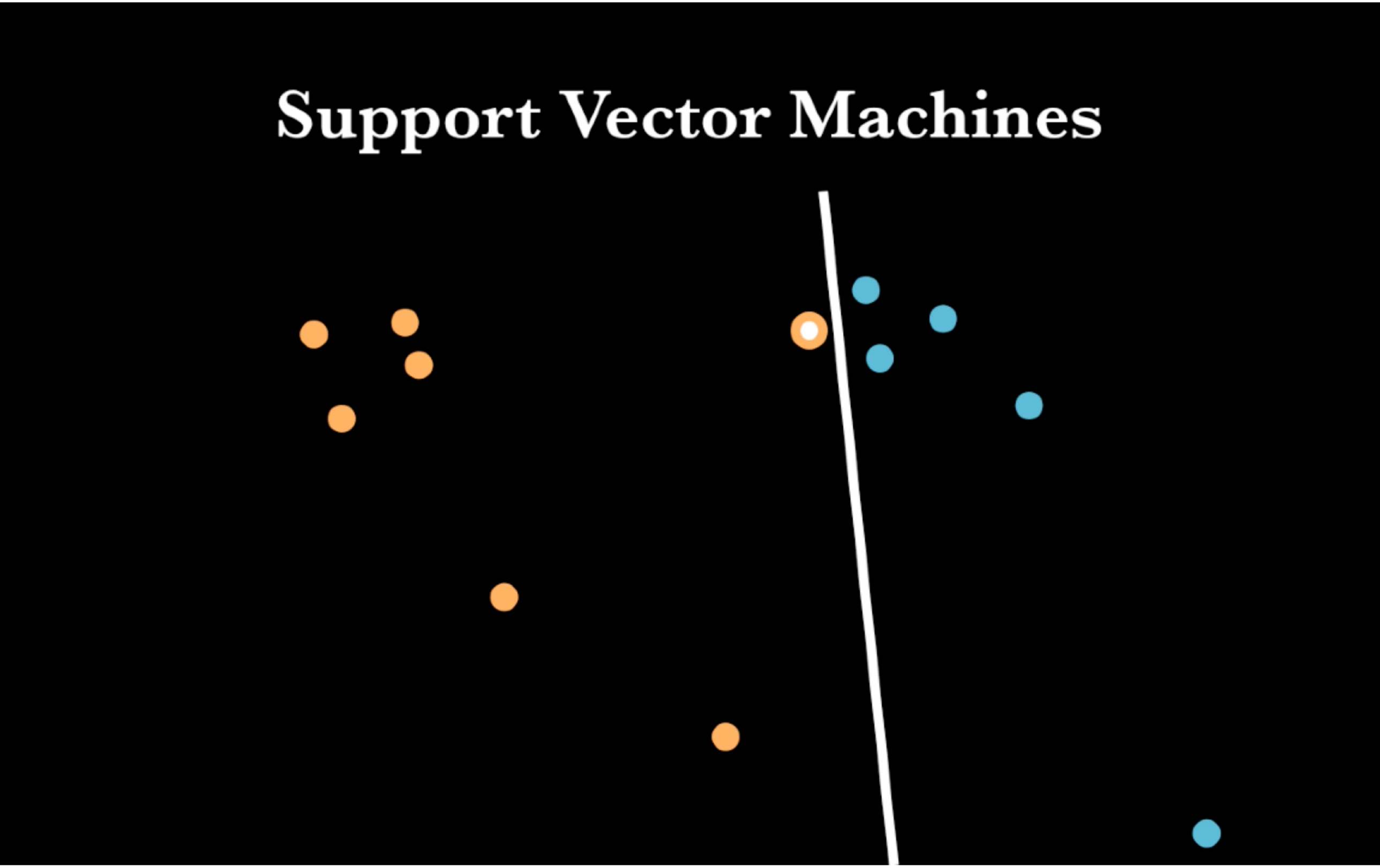
Credit: Greene, CU Boulder

Support Vector Machines



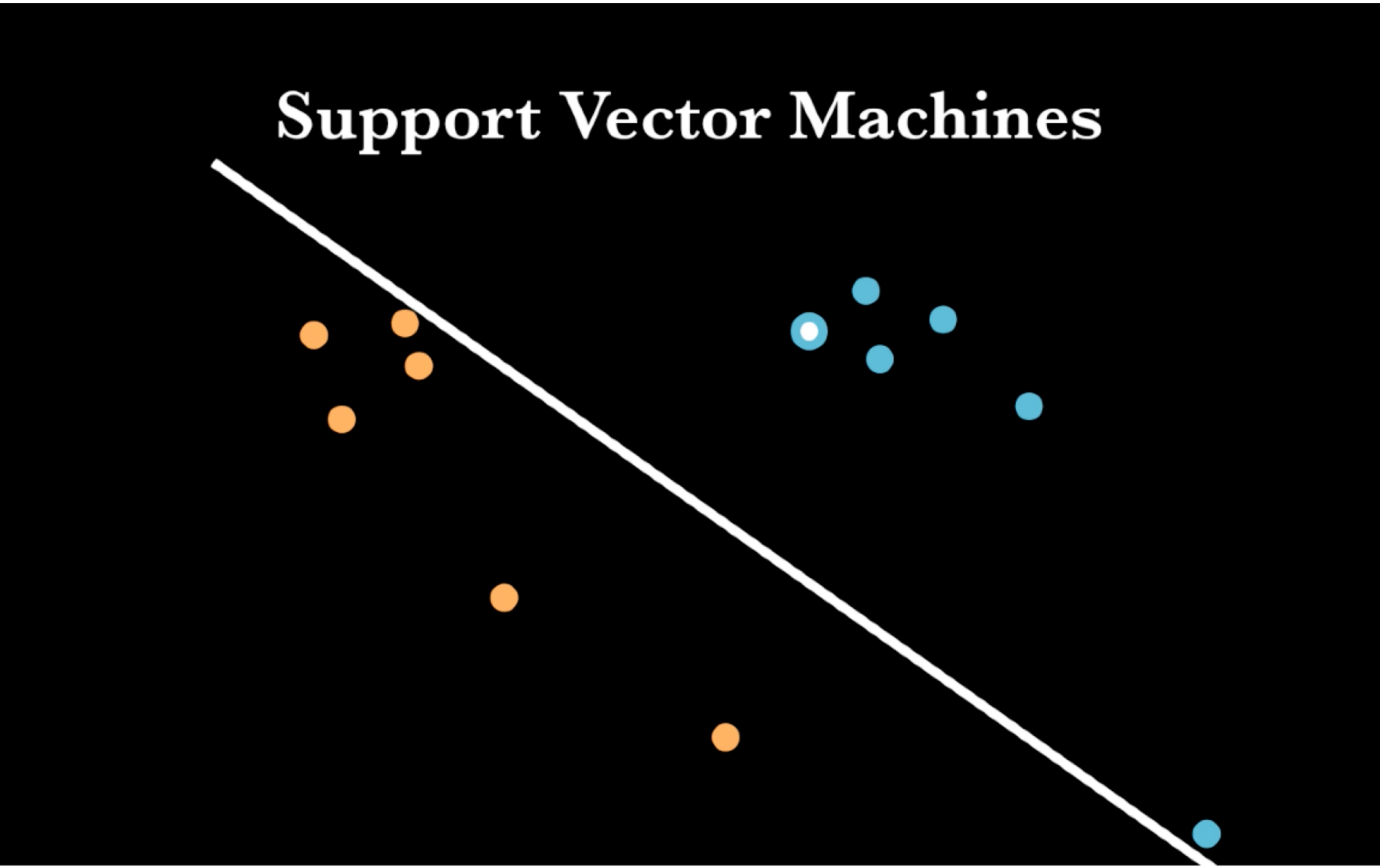
Credit: Greene, CU Boulder

Support Vector Machines



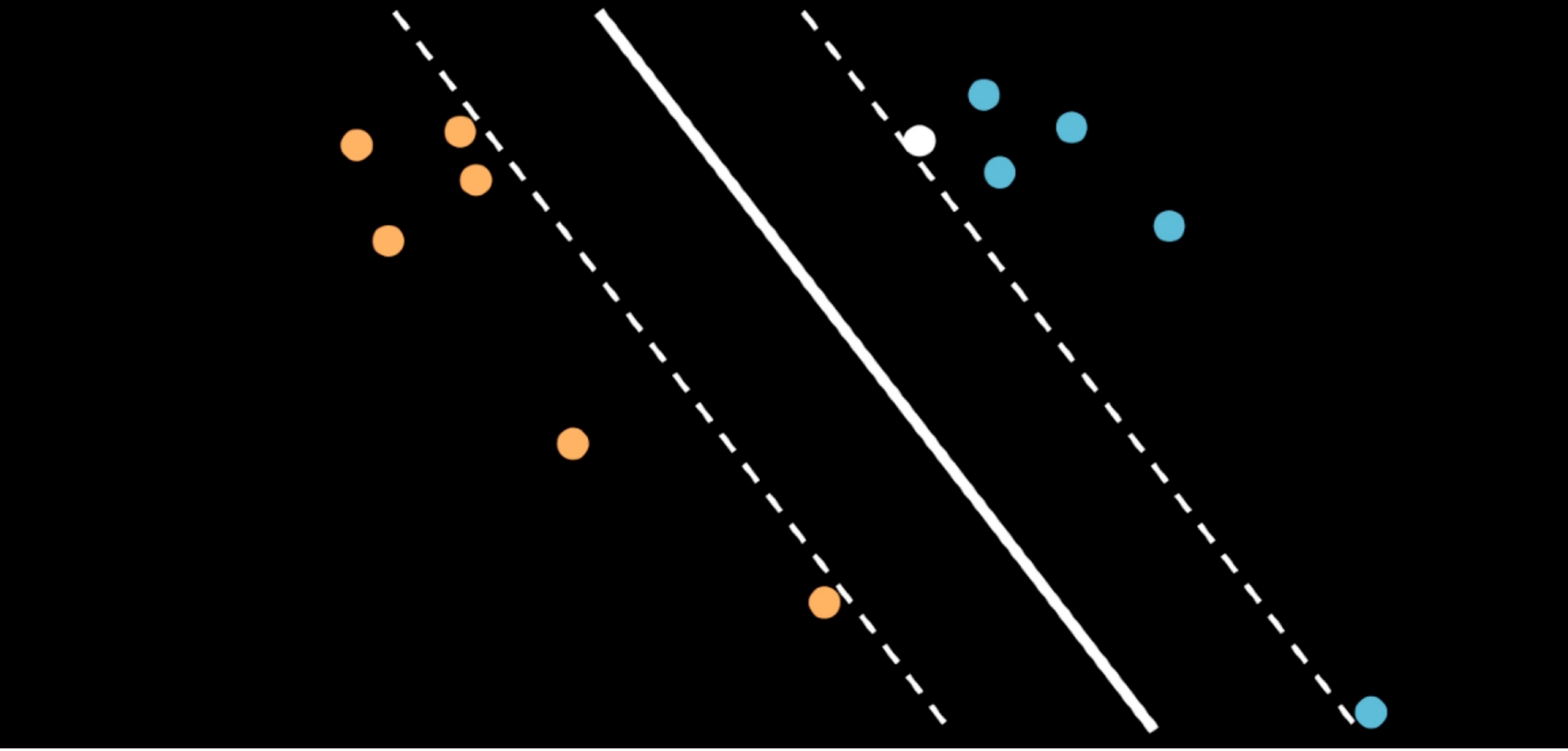
Credit: Greene, CU Boulder

Support Vector Machines



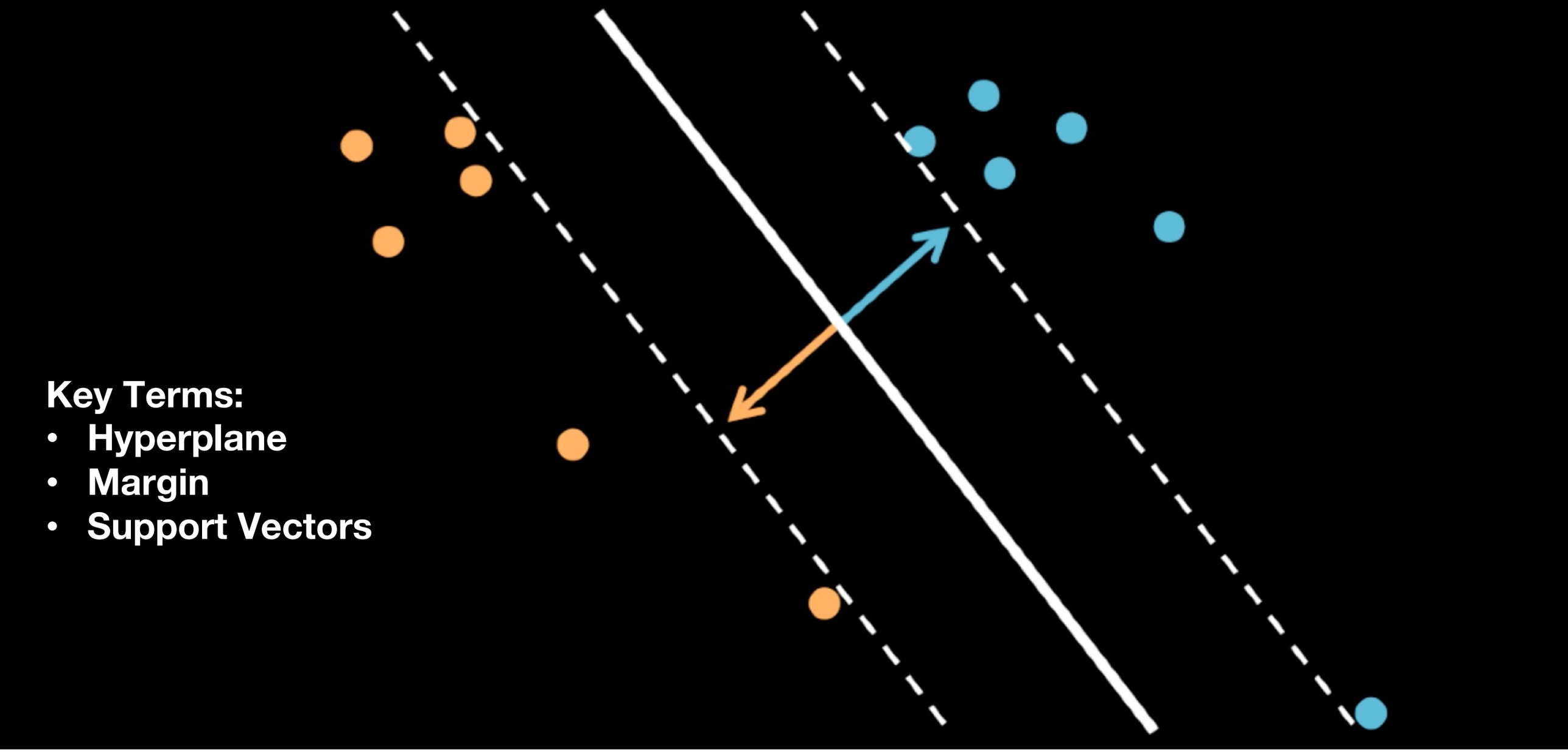
Credit: Casey Greene

Support Vector Machines



Credit: Greene, CU Boulder

Support Vector Machines

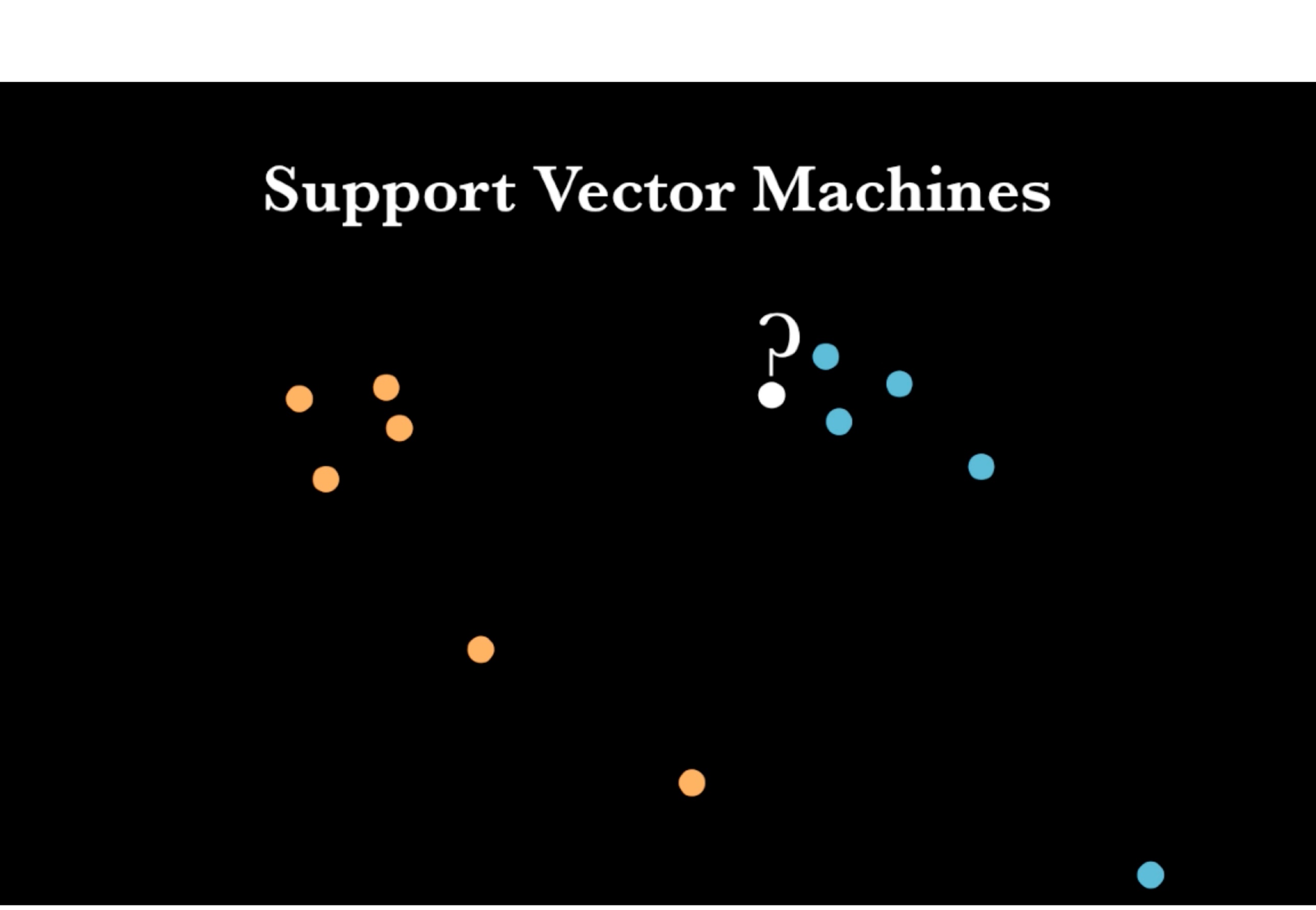


Key Terms:

- **Hyperplane**
- **Margin**
- **Support Vectors**

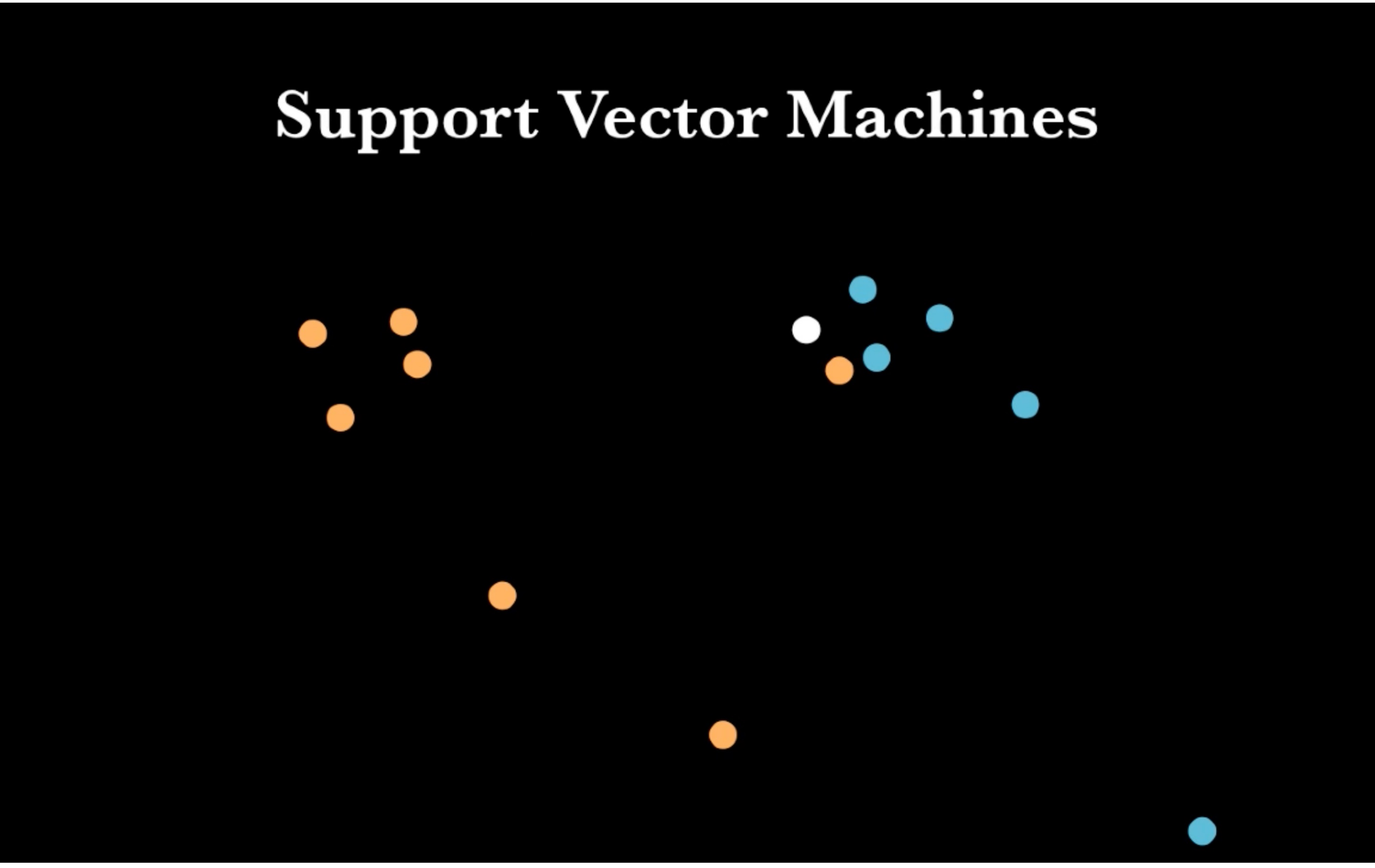
Credit: Greene, CU Boulder

Support Vector Machines



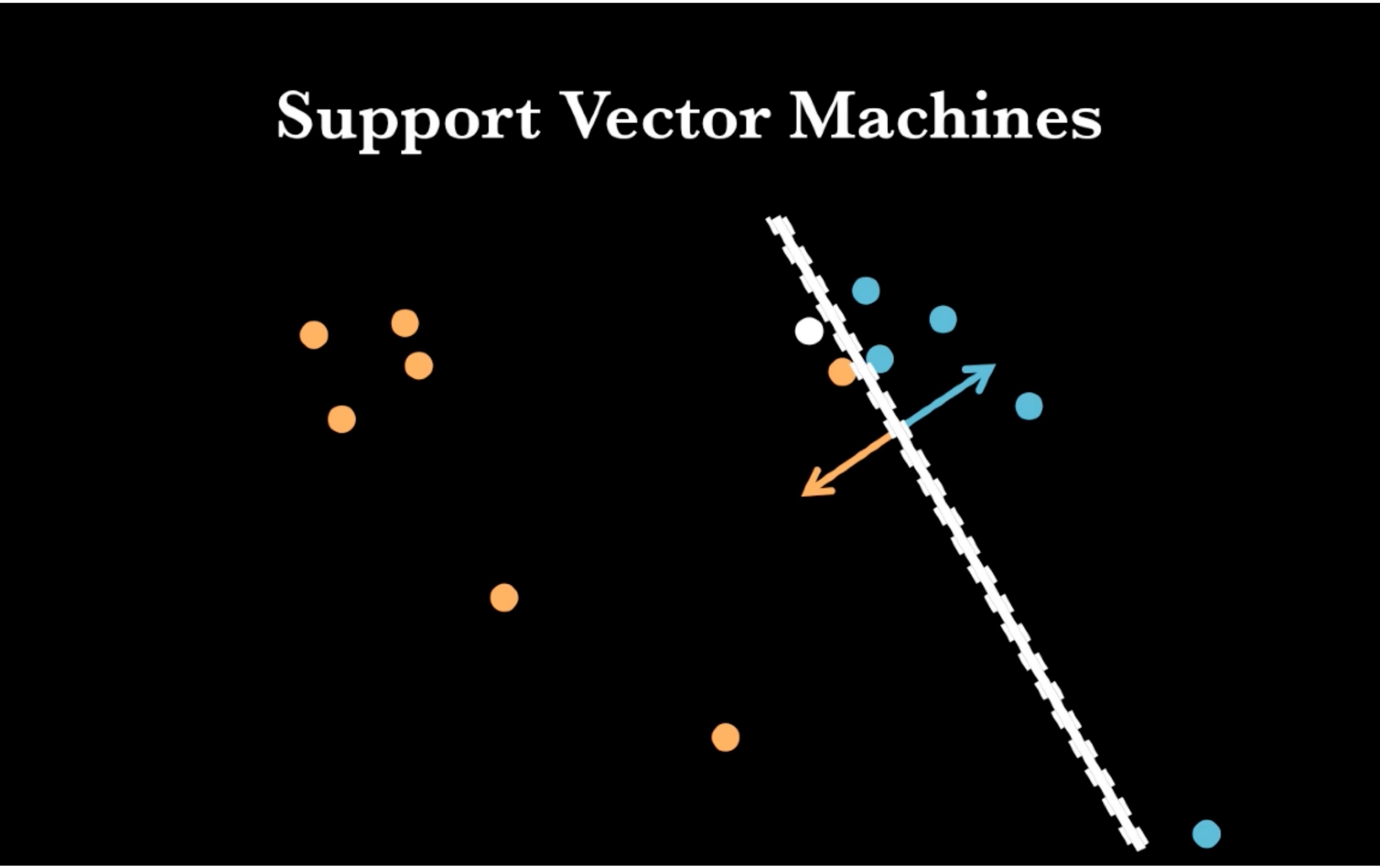
Credit: Greene, CU Boulder

Support Vector Machines



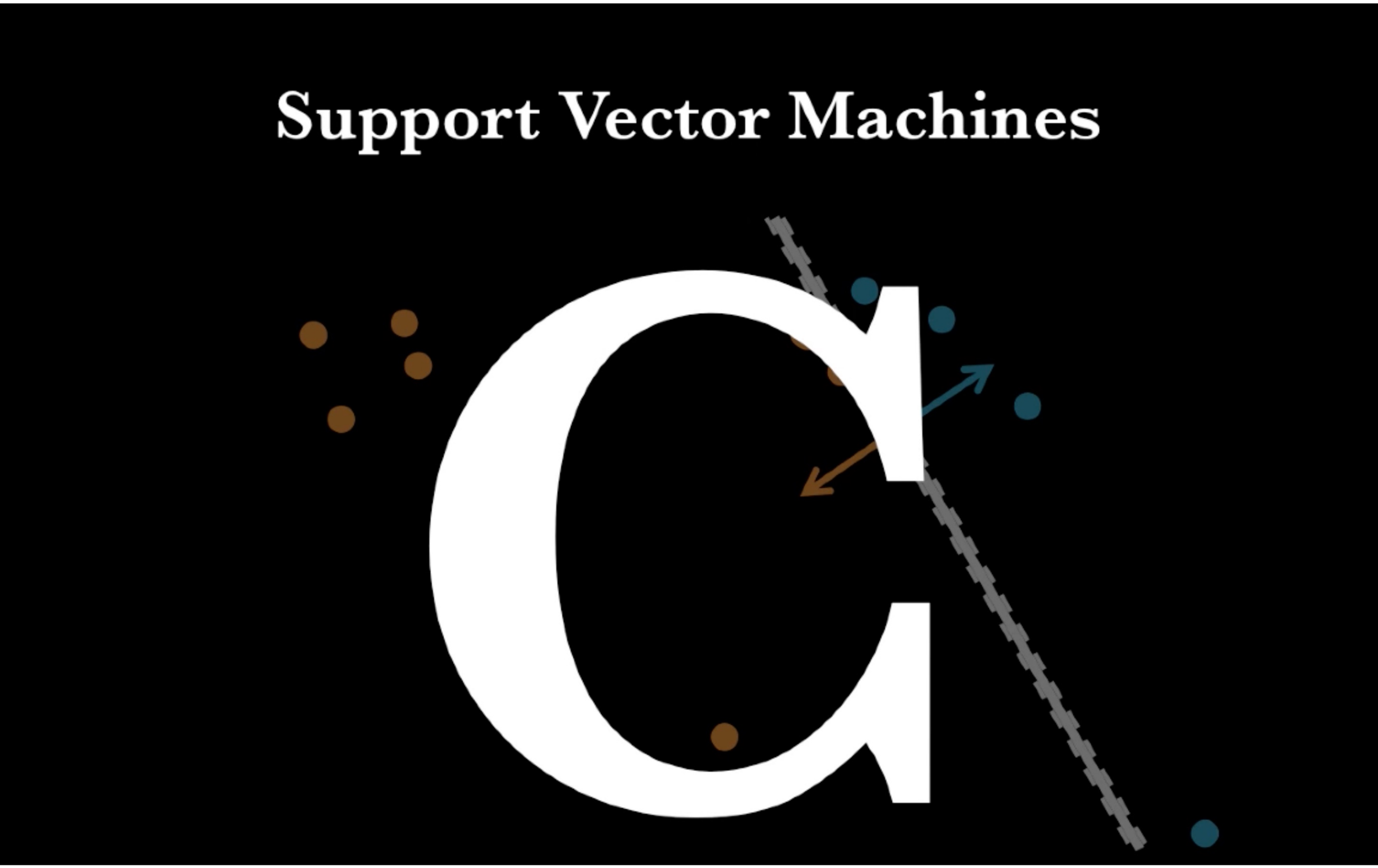
Credit: Greene, CU Boulder

Support Vector Machines



Credit: Greene, CU Boulder

Support Vector Machines



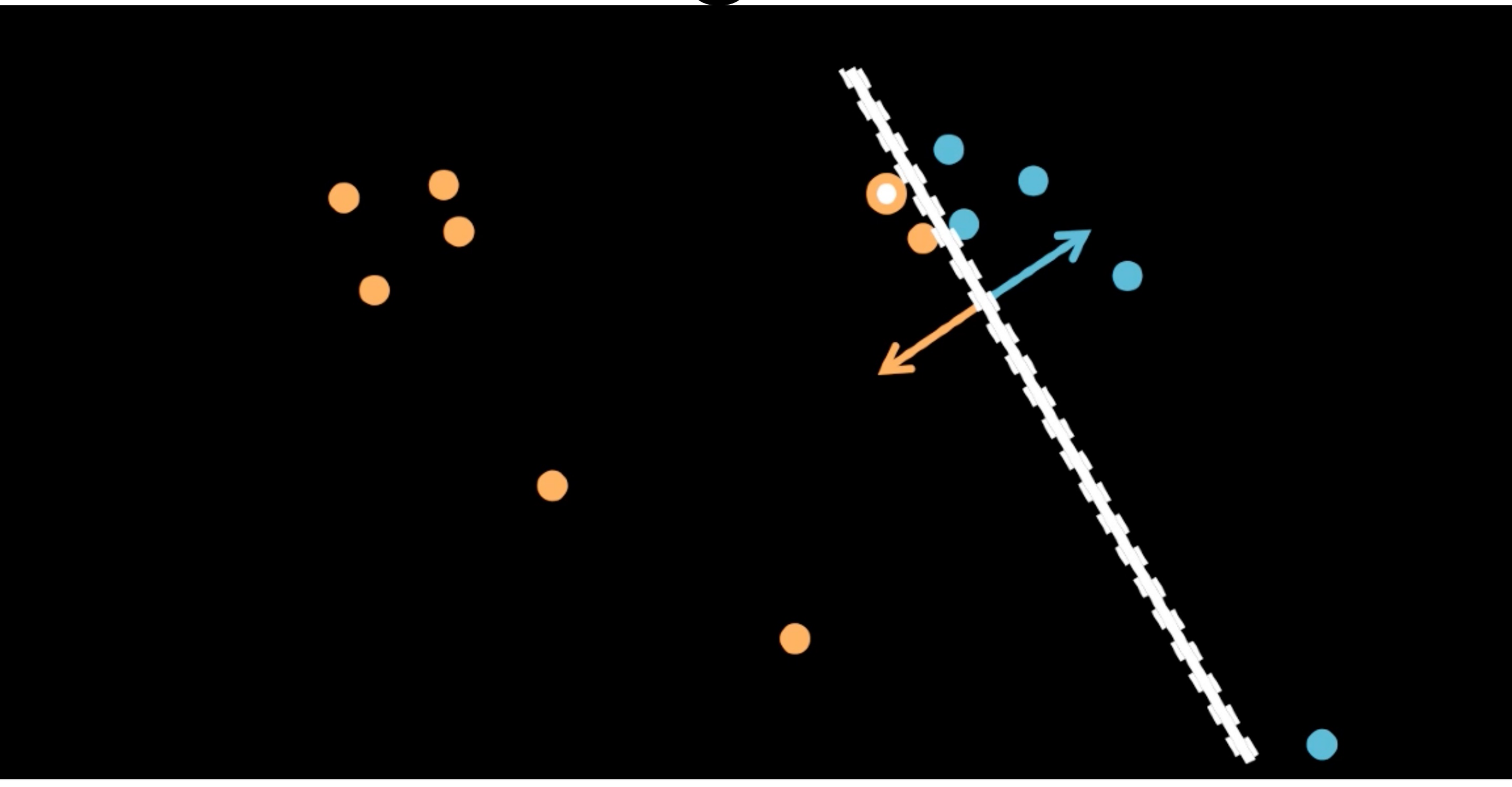
Credit: Greene, CU Boulder

Support Vector Machines

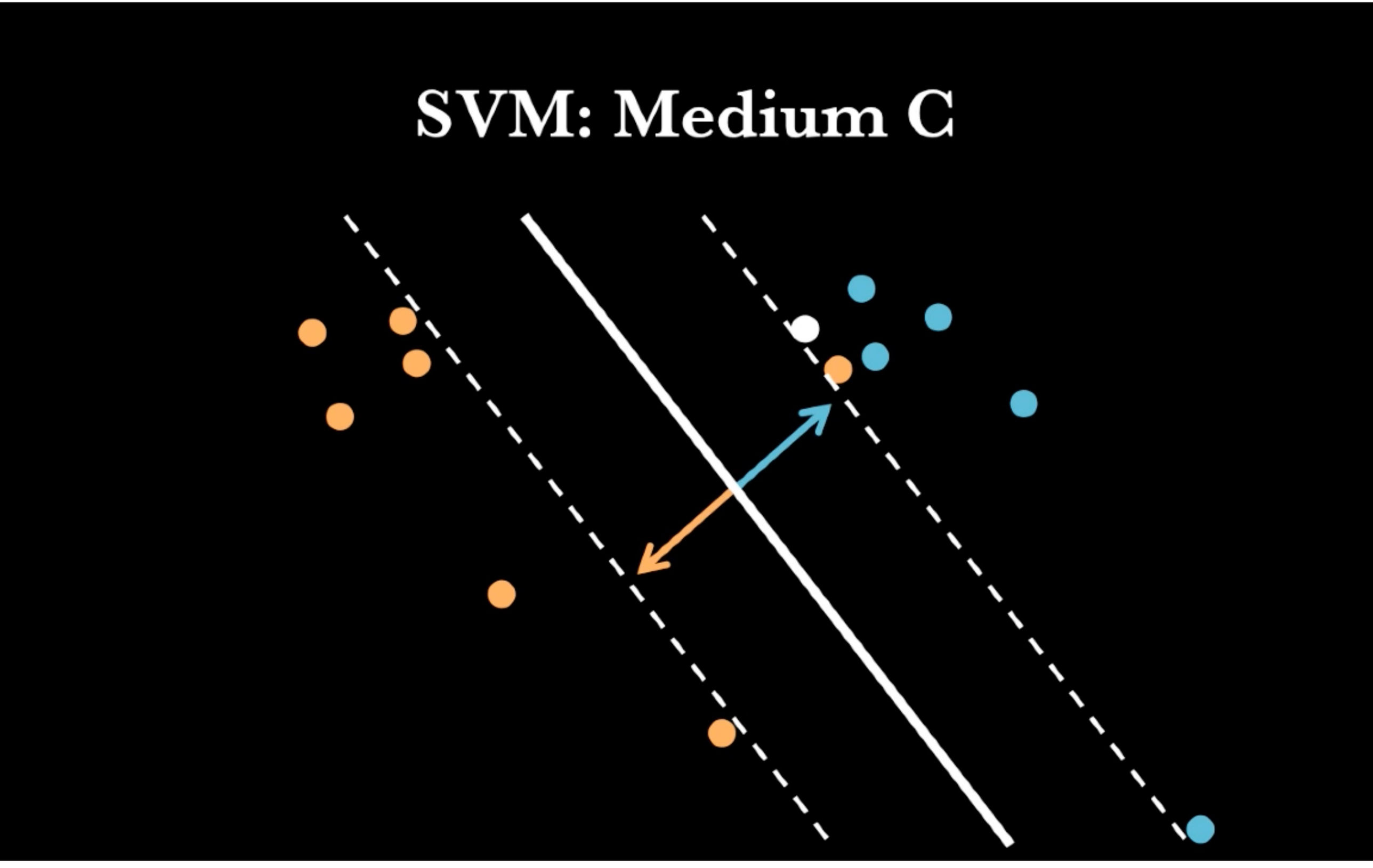
How much should an SVM care about getting everything right vs. getting the things that it gets right *very* right.

Credit: Greene, CU Boulder

High C

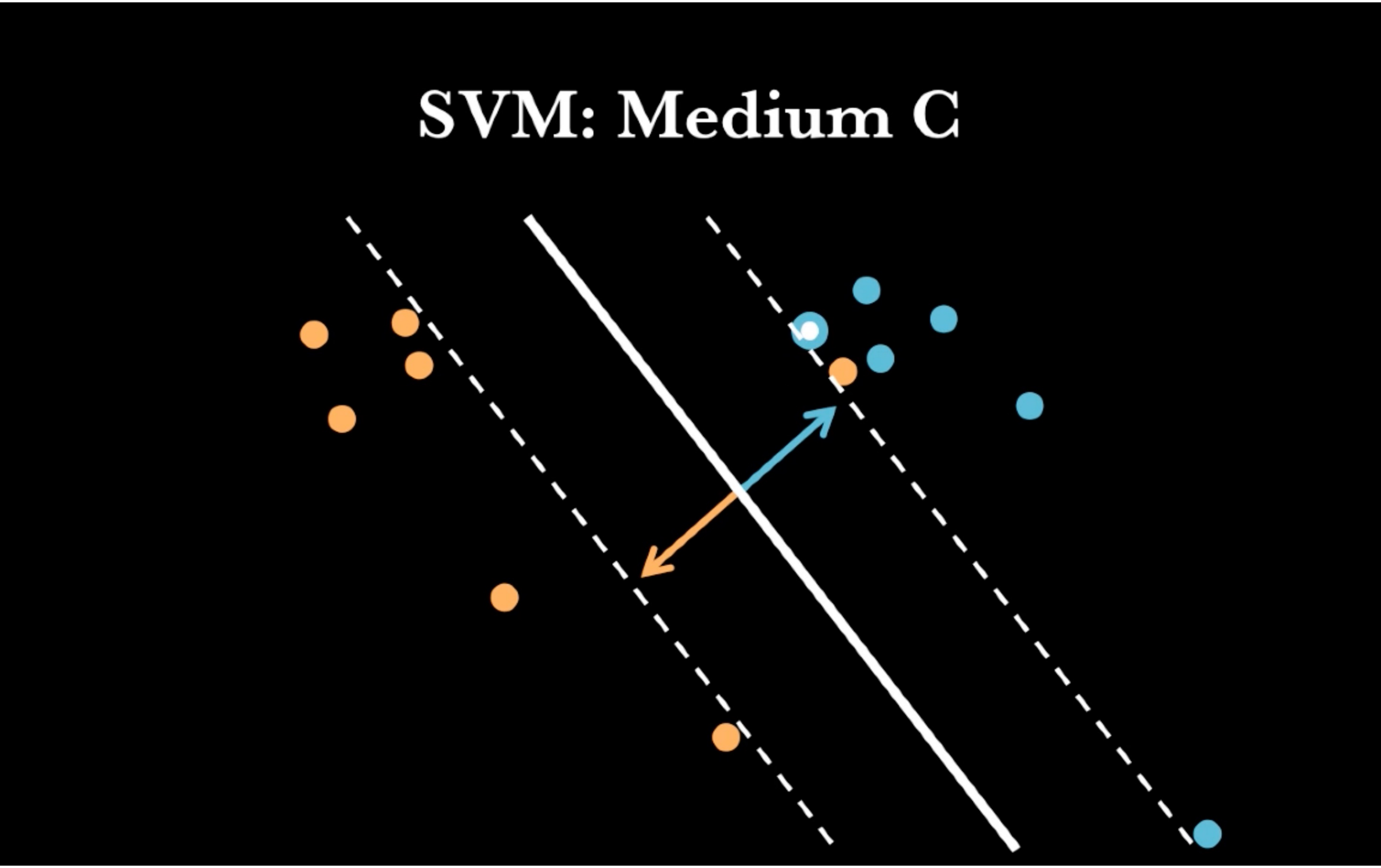


Credit: Greene, CU Boulder



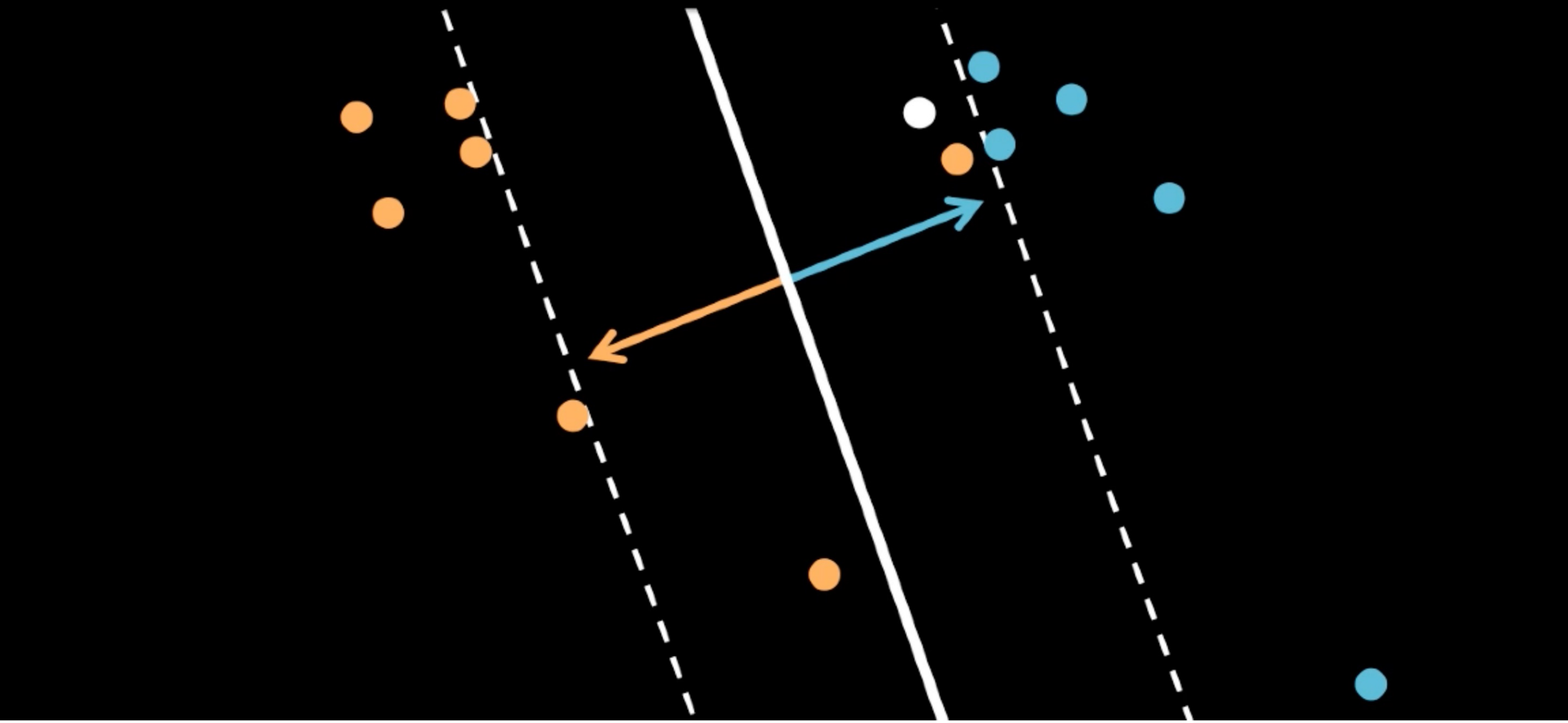
Credit: Greene, CU Boulder

SVM: Medium C



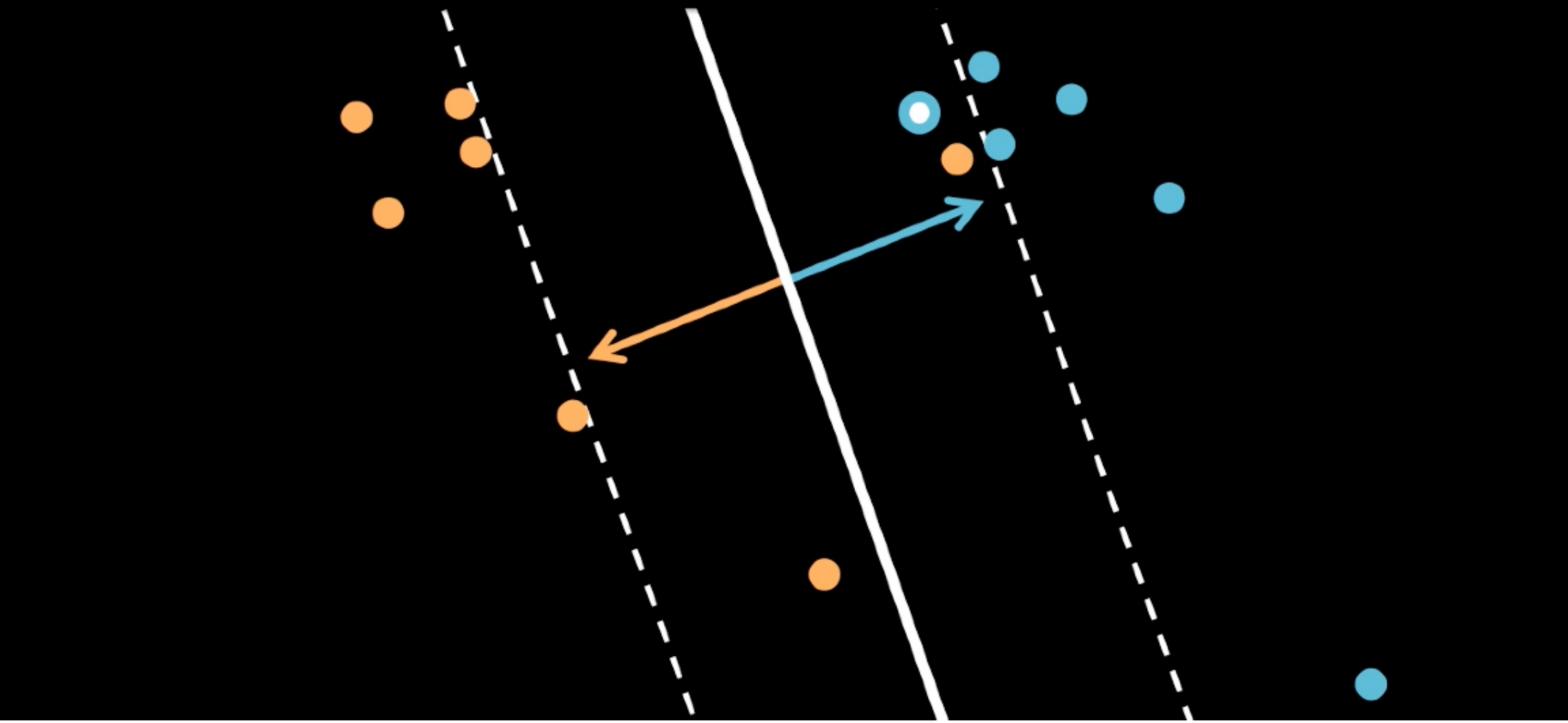
Credit: Greene, CU Boulder

SVM: Low C



Credit: Greene, CU Boulder

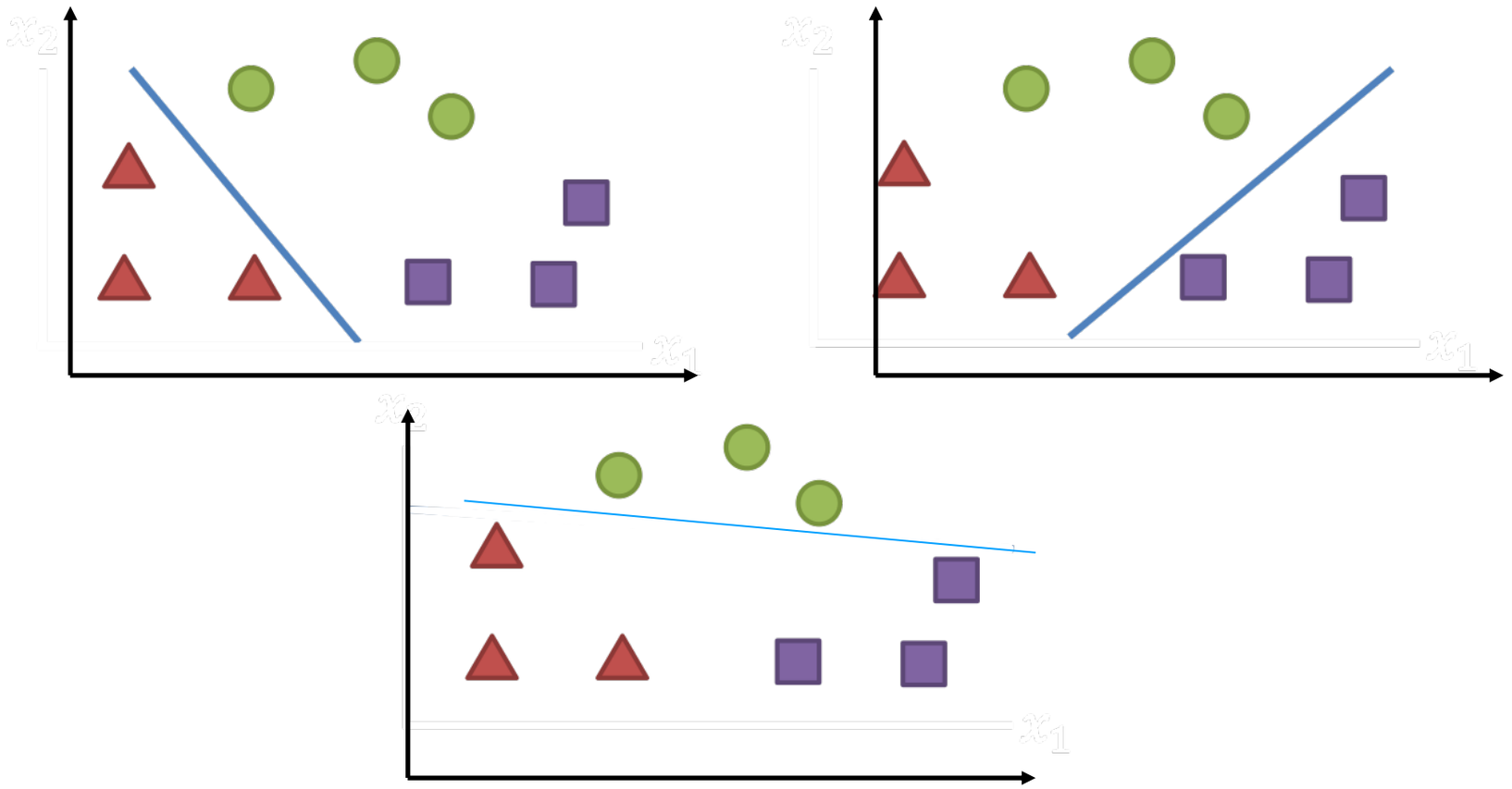
SVM: Low C



Credit: Greene, CU Boulder

Notebook time!

Multiclass Classification



One vs. Rest (OVR): N Class instances then N-Binary classifiers

Multiclass Classification

Say new data with feature (x1,y1) comes in. Then we pass it to all three classifier to get a probability score,

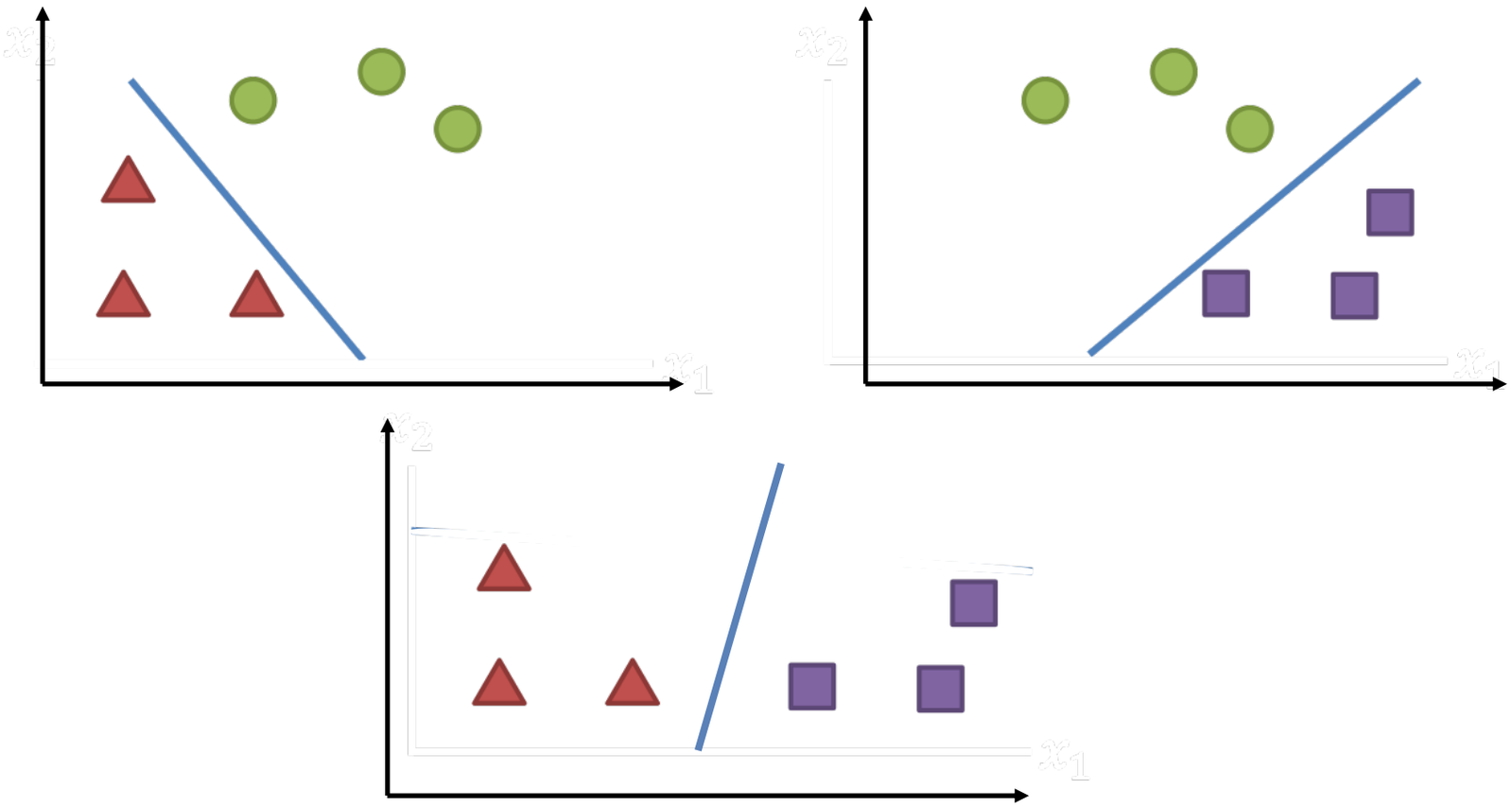
Green Classifier: Positive with a probability score of (0.9)

Purple Classifier: Positive with a probability score of (0.4)

Red Classifier: Negative with a probability score of (0.5)

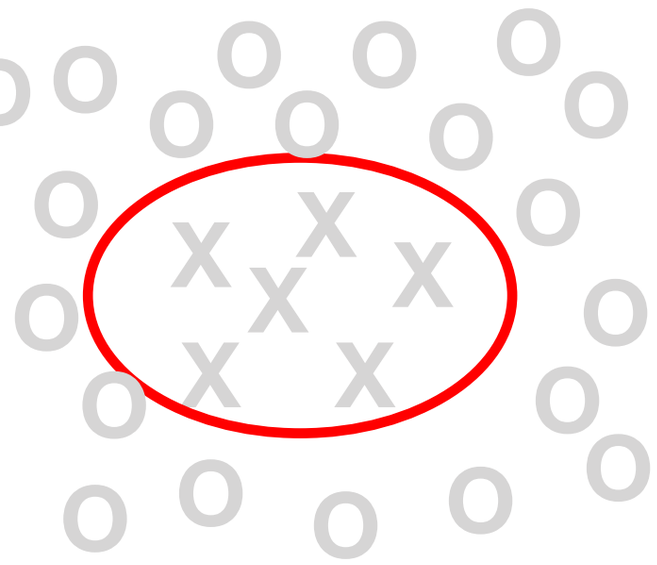
One vs. Rest (OVR): N Class instances then N-Binary classifiers

Multiclass Classification



One vs. One (OVO): We must generate n* (n-1)/2 binary classifier models

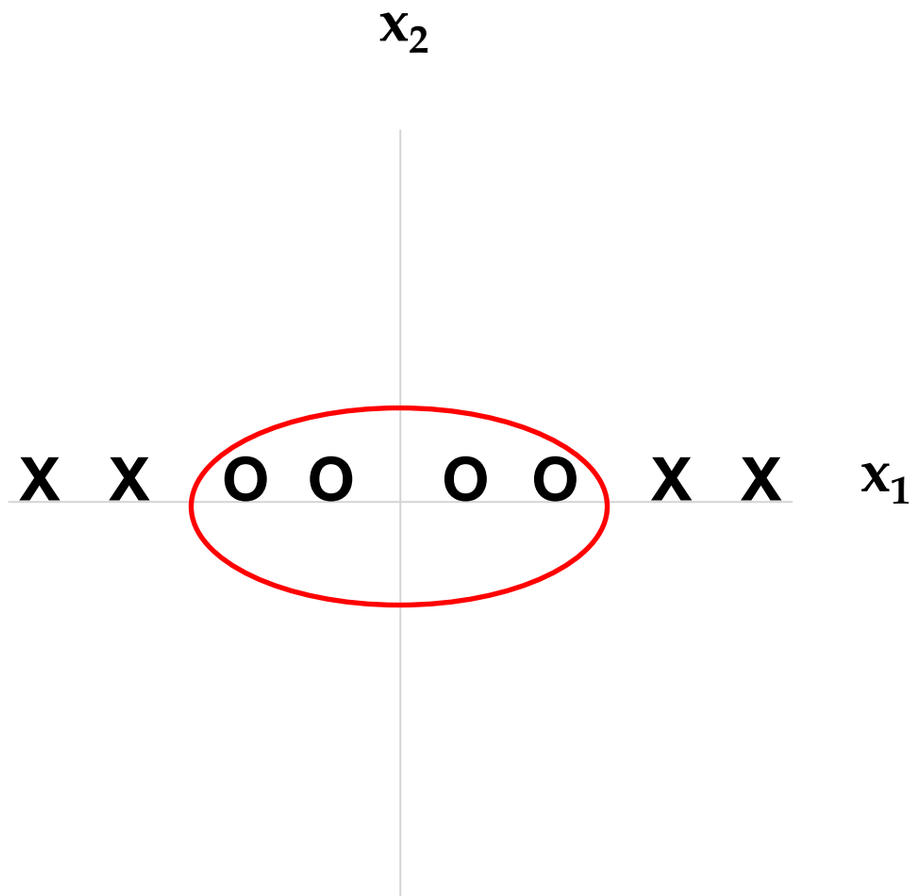
What if classes are not linearly seperable?

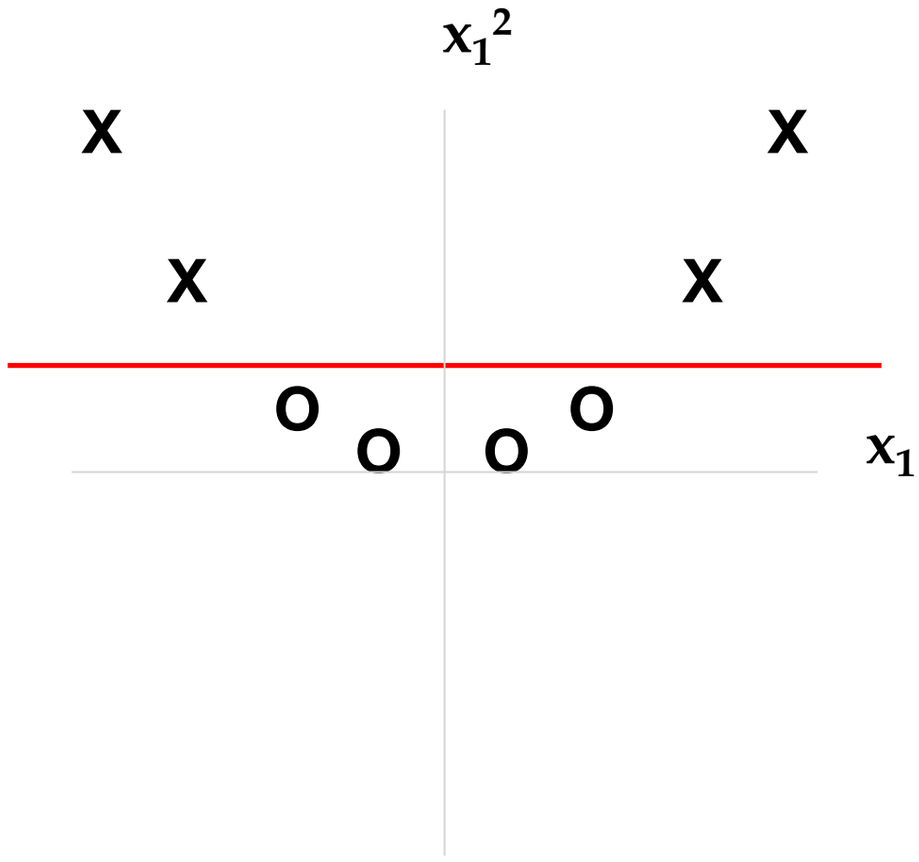


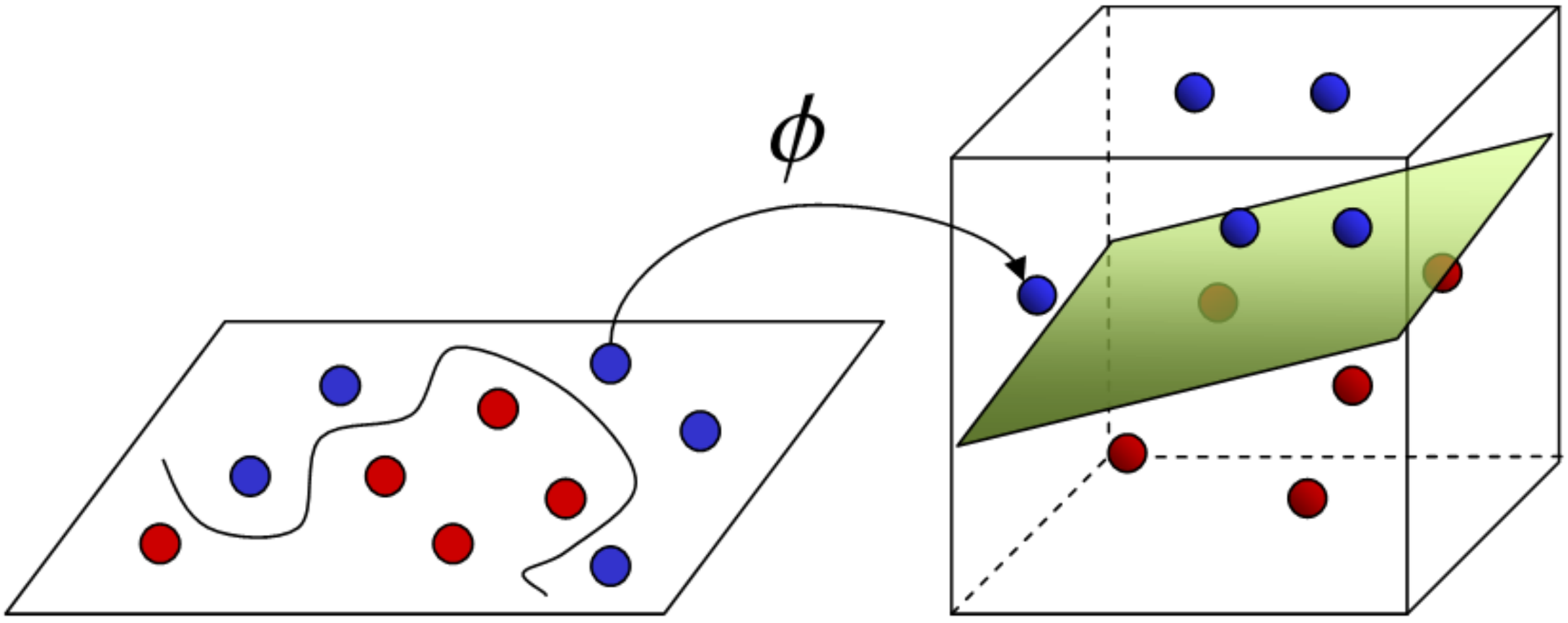
Kernel Methods

Making the Non-Linear Linear

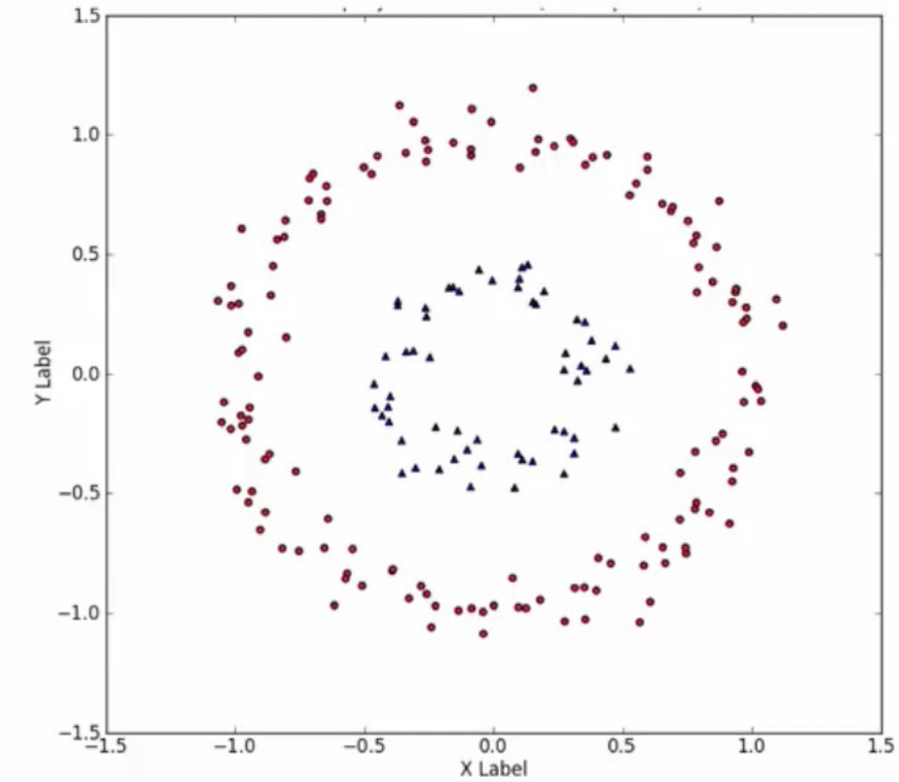
When Linear Separator Fails

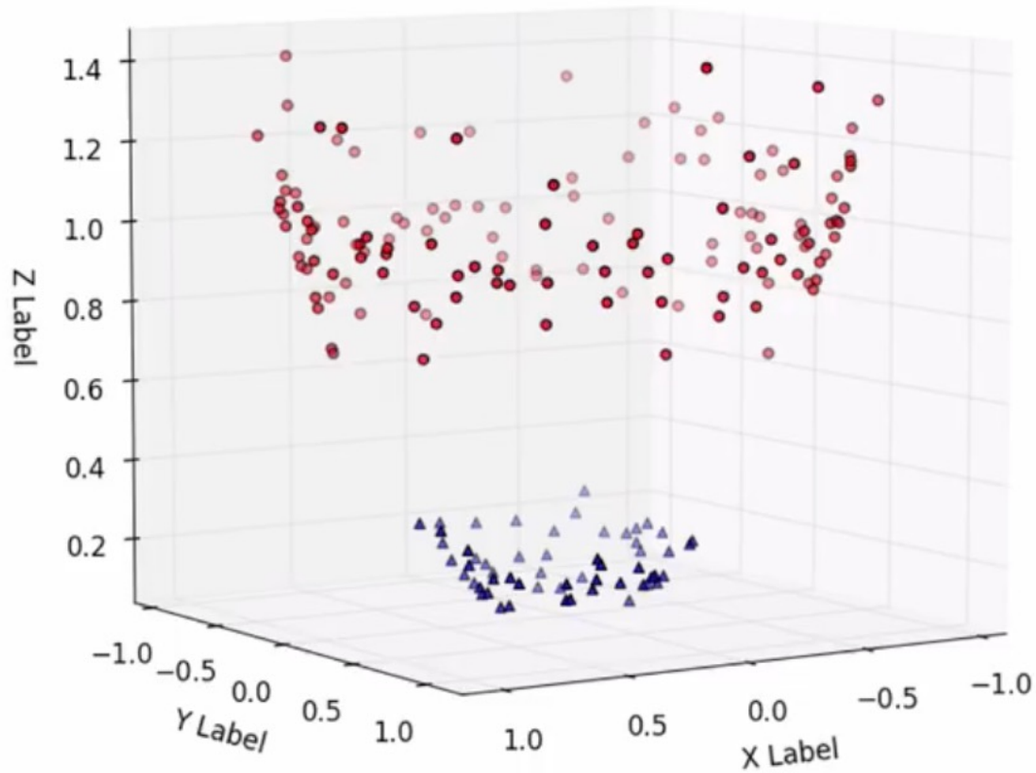




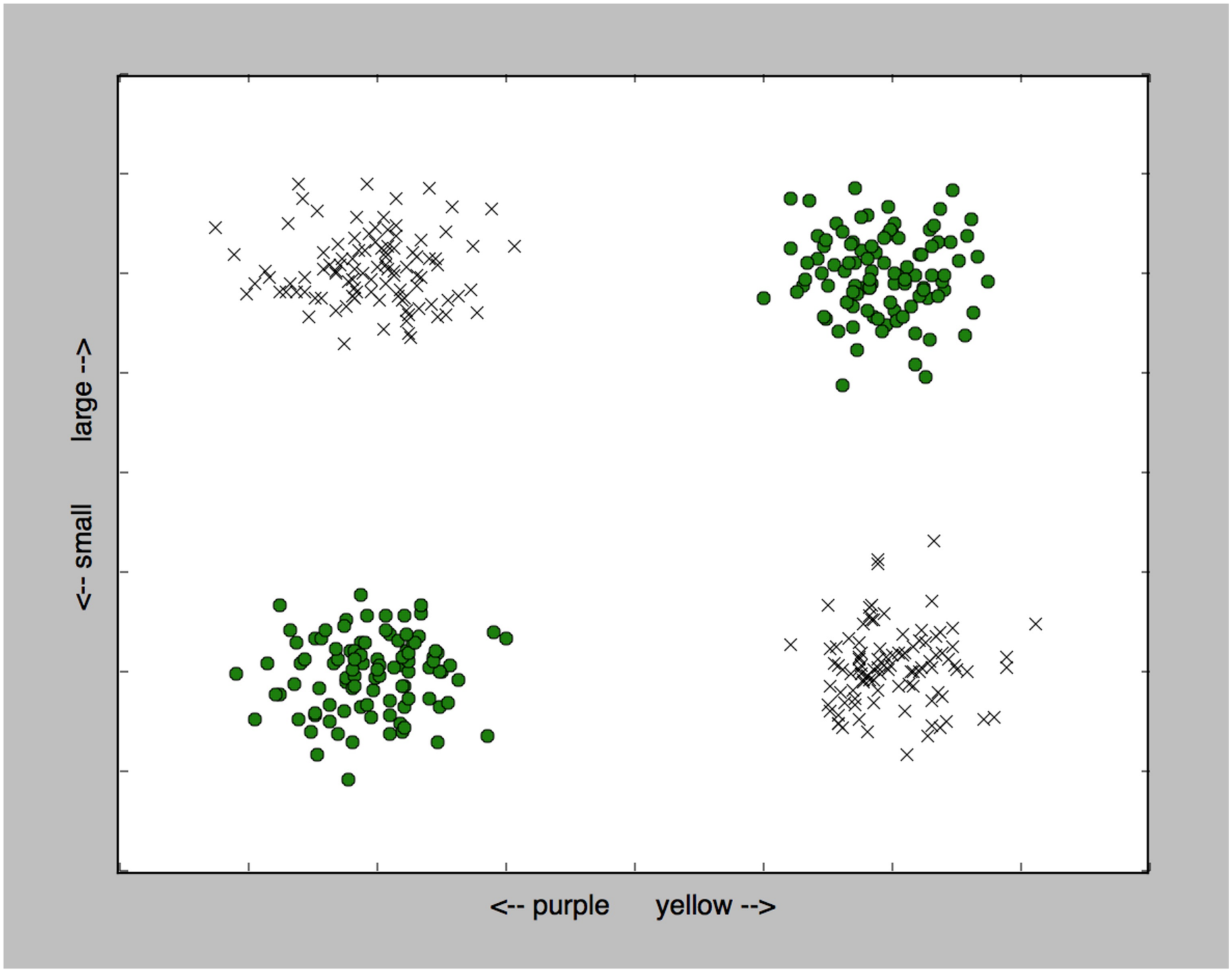

ϕ
Input Space
Feature Space

SVM: Kernel Trick

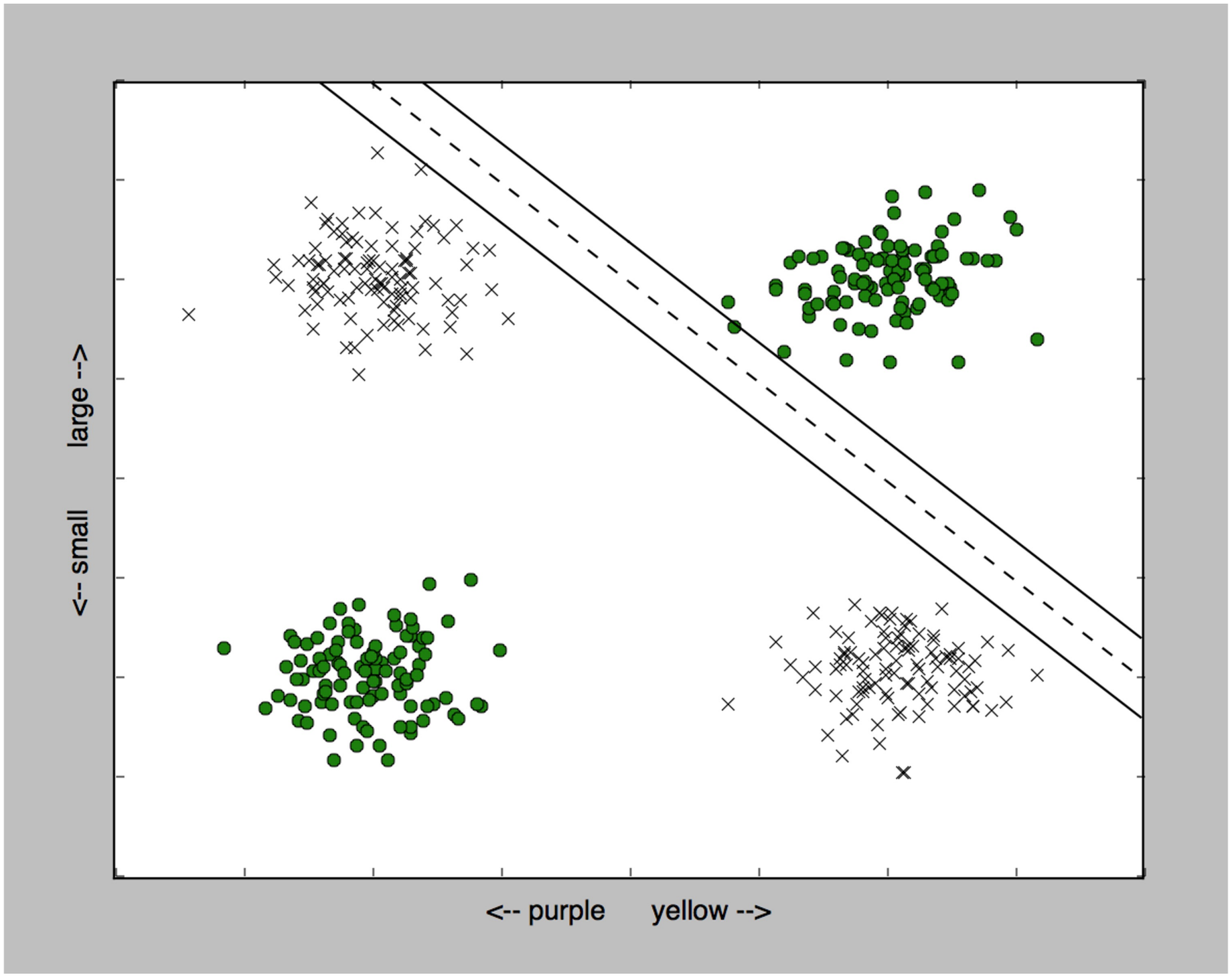




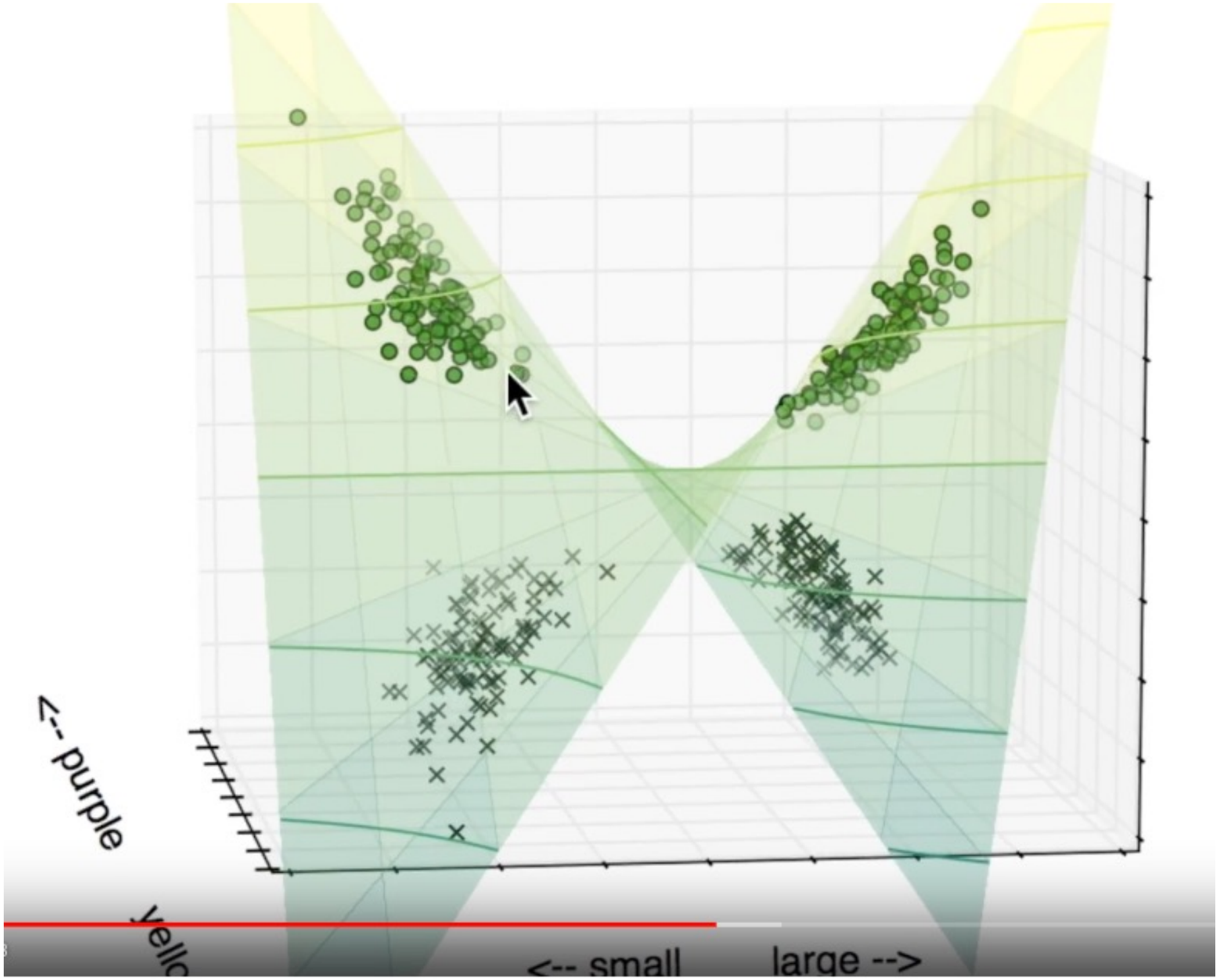
Graph courtesy: Alice Zhao, Metis



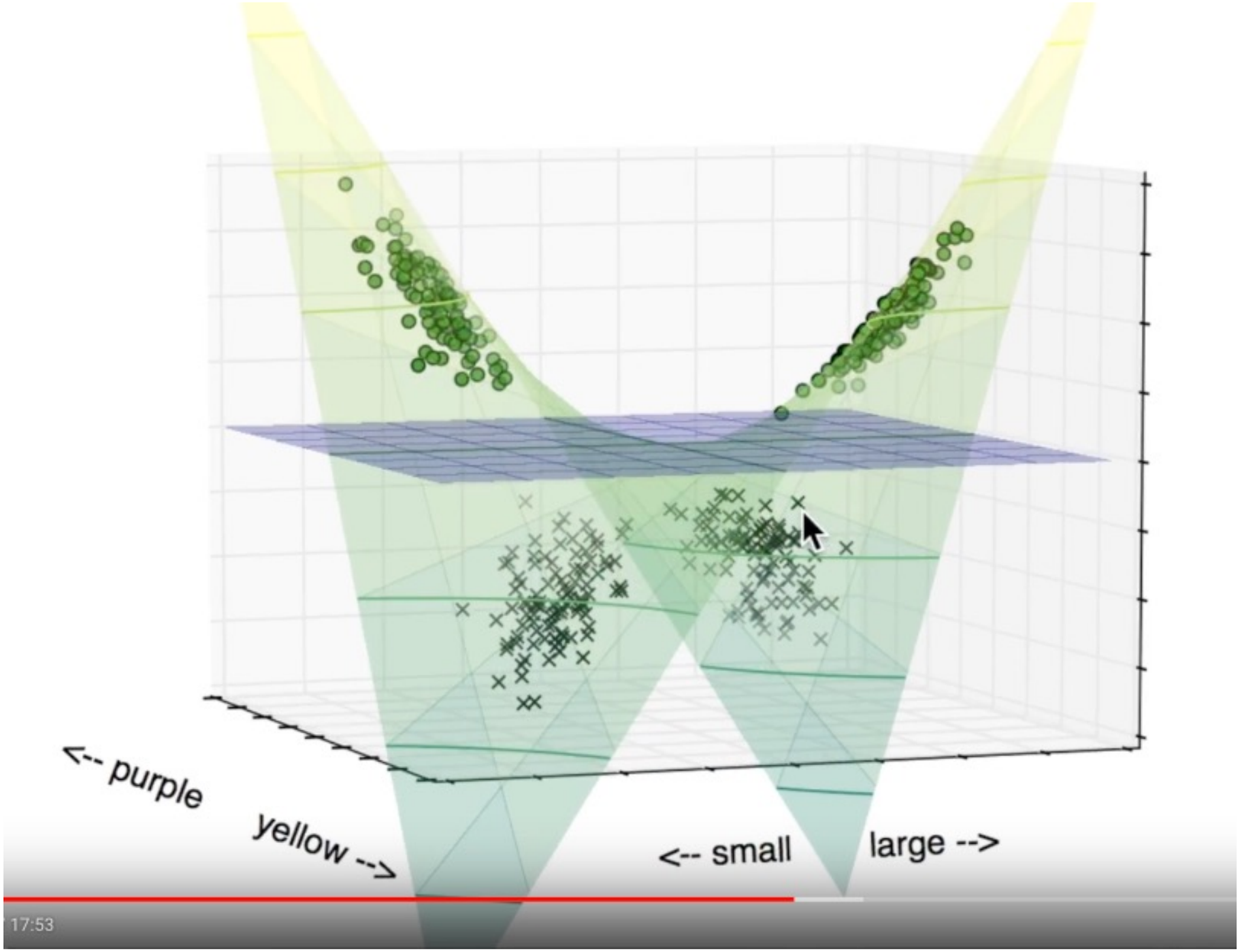
Graph courtesy: Brandon



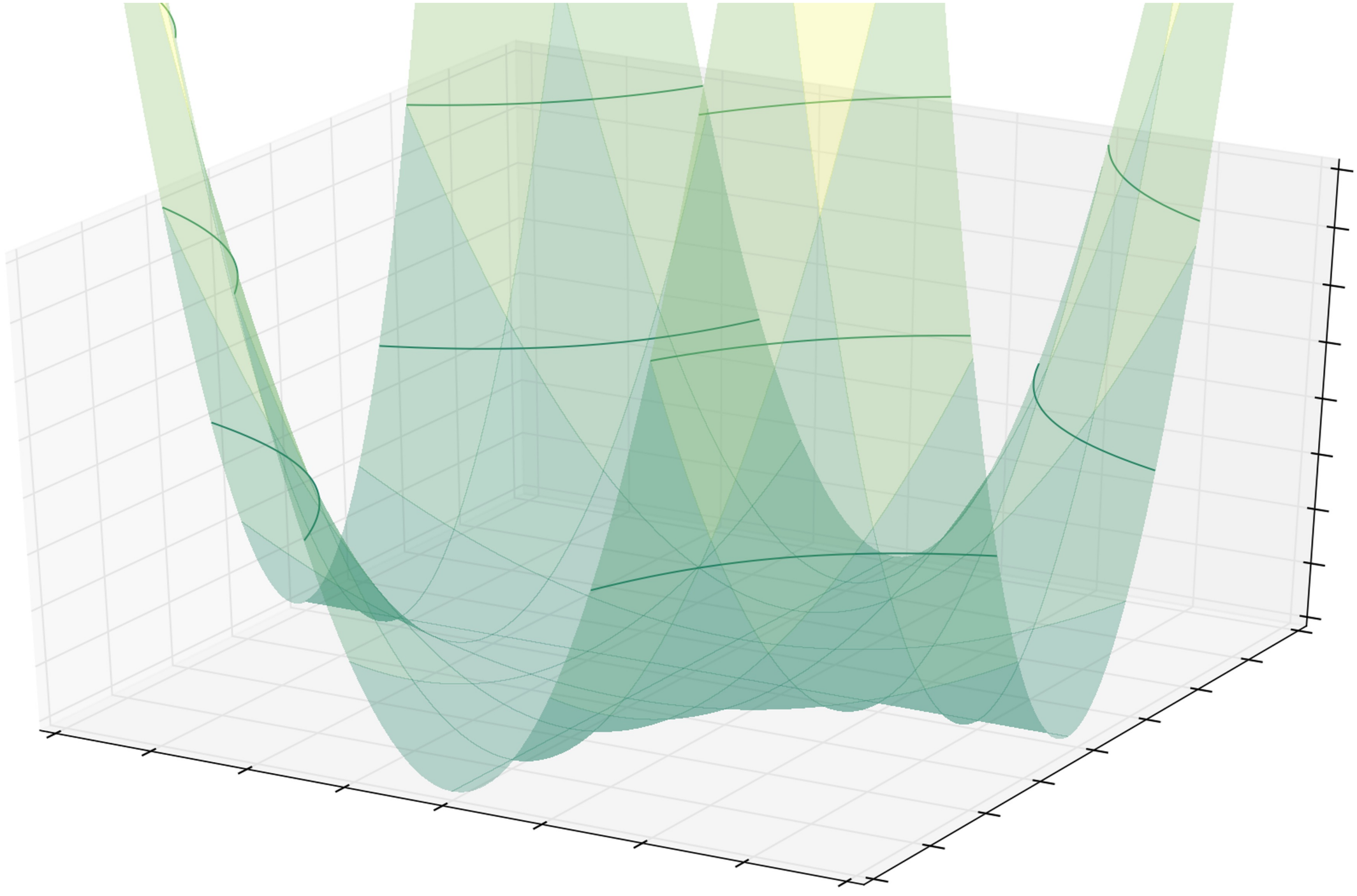
Graph courtesy: Brandon



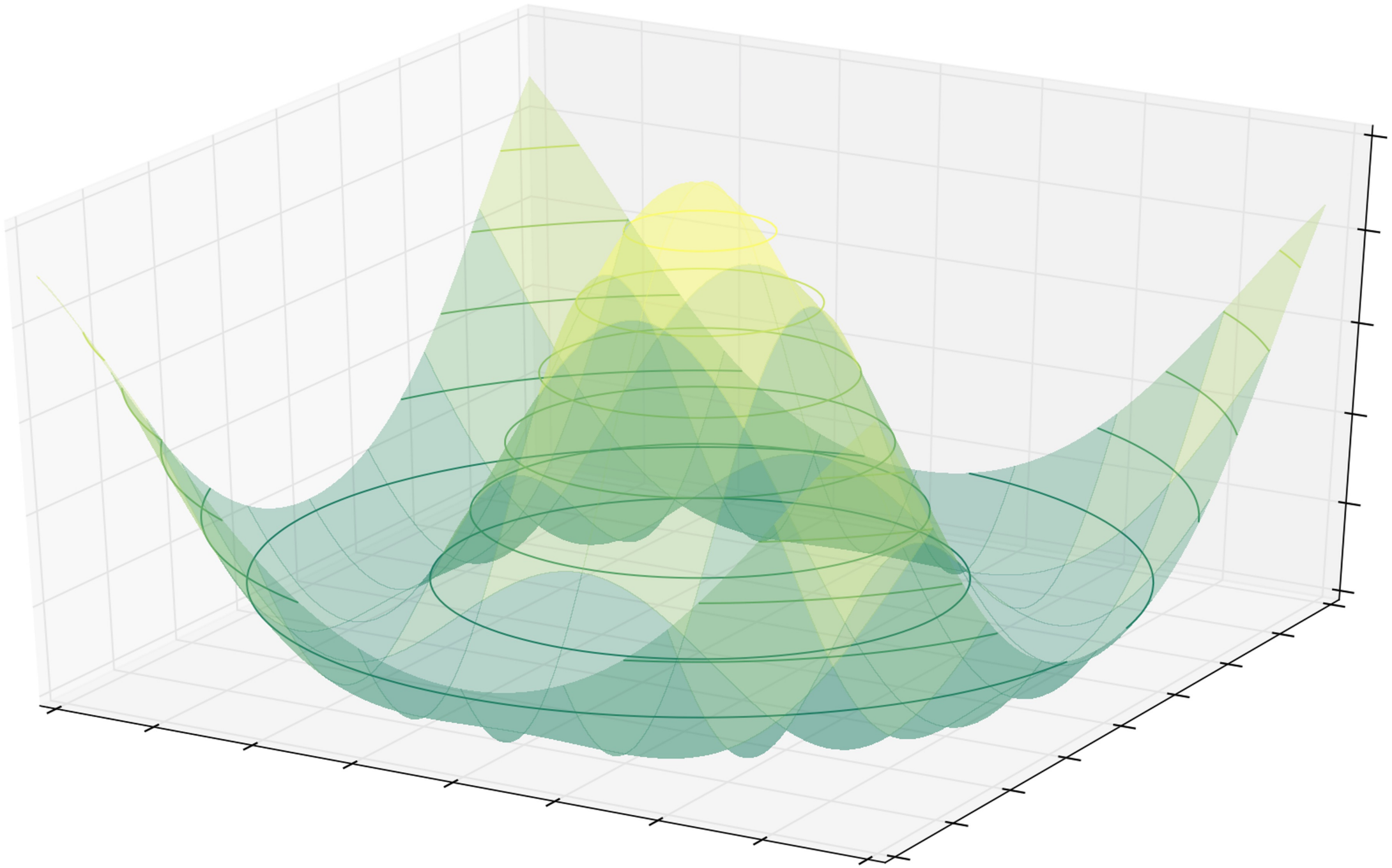
Graph courtesy: Brandon



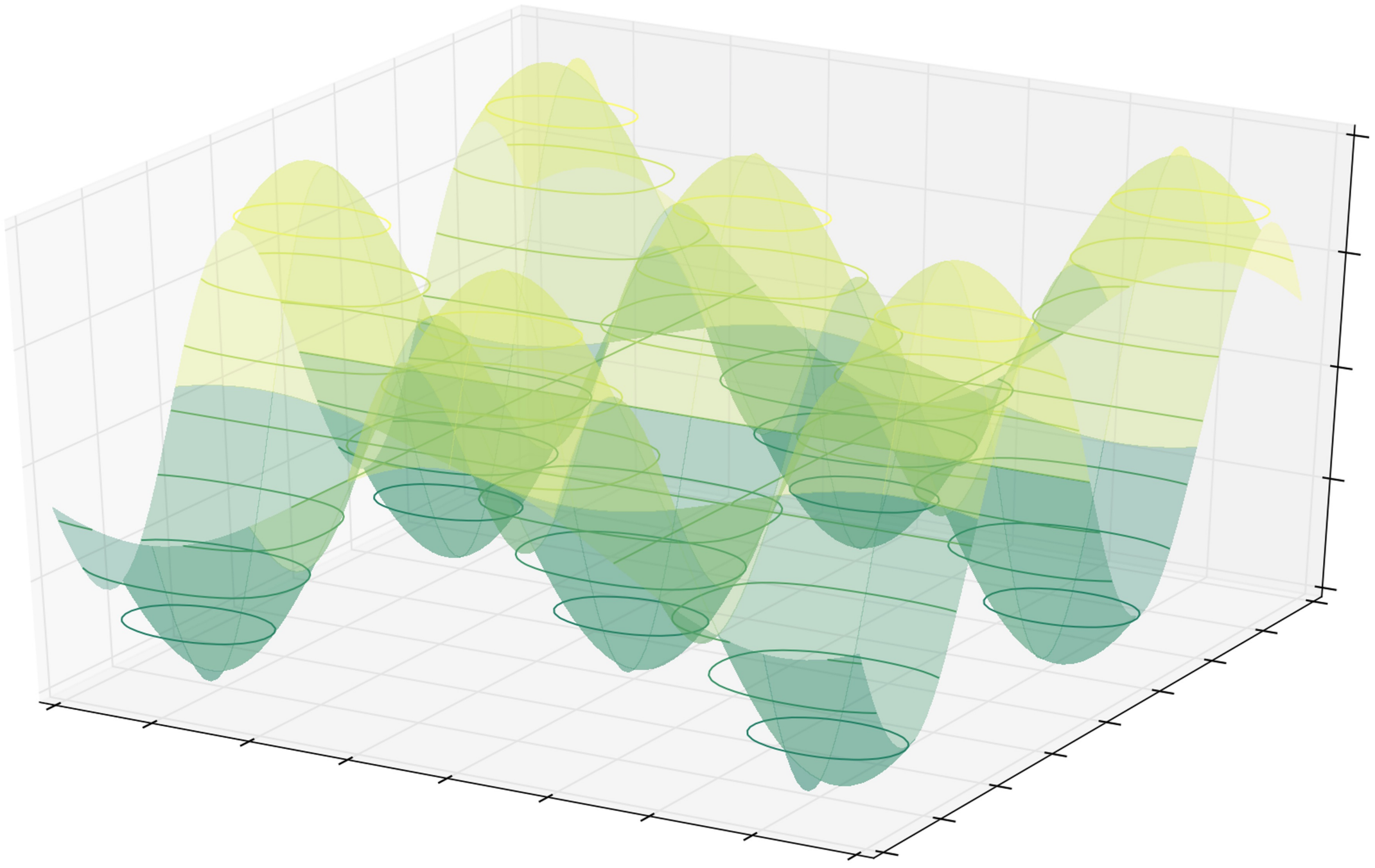
Graph courtesy: Brandon



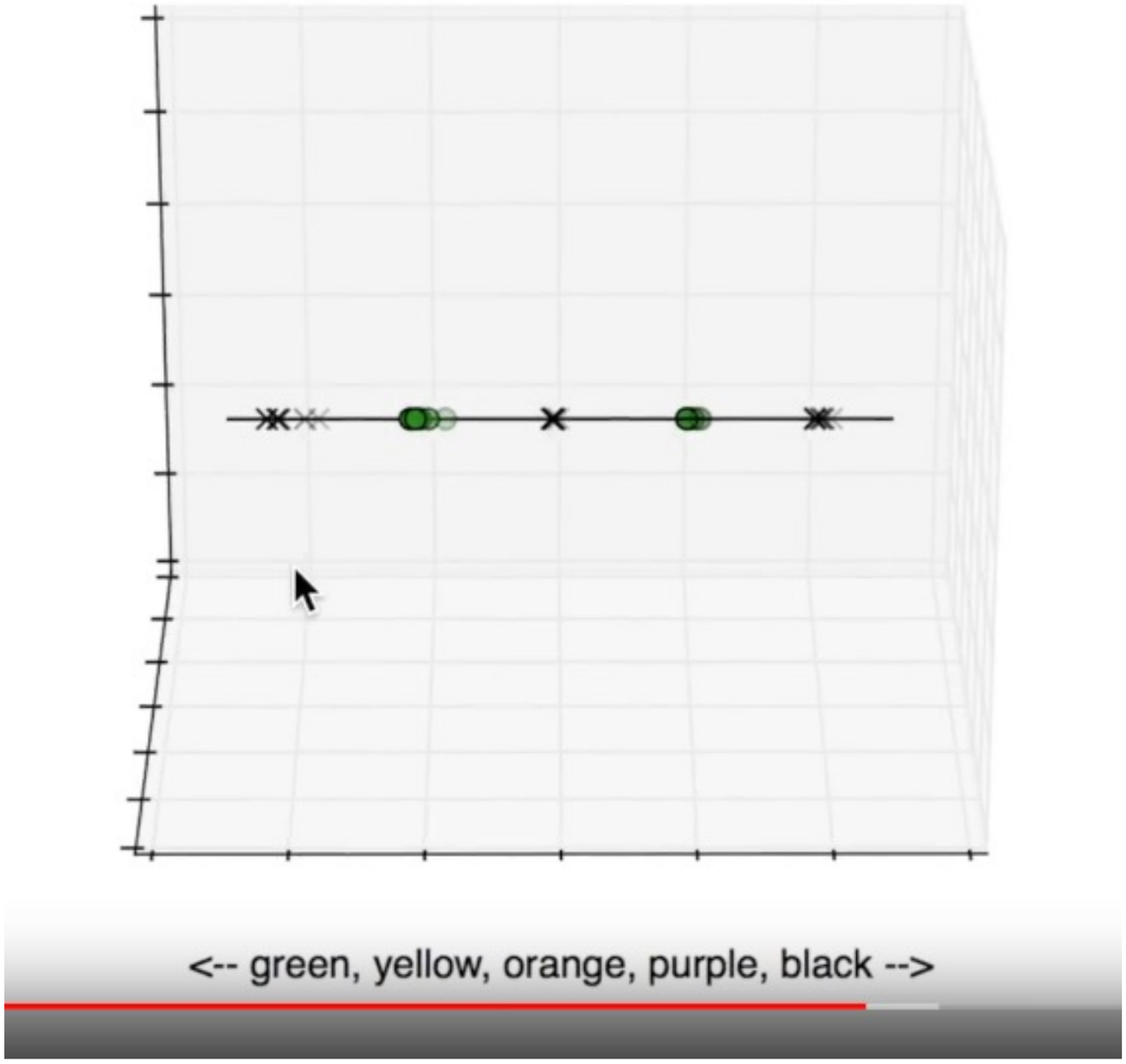
Graph courtesy: Brandon



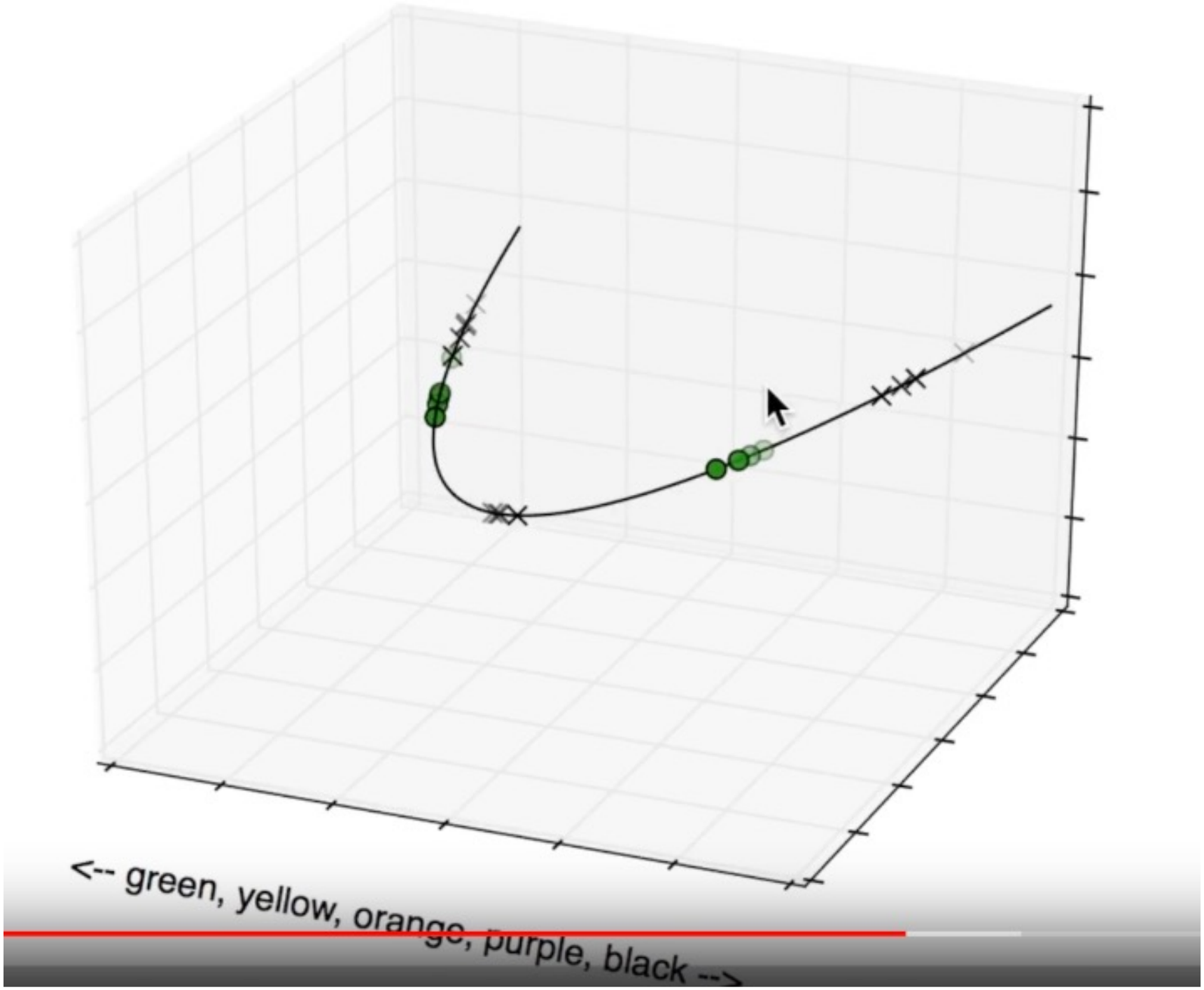
Graph courtesy: Brandon



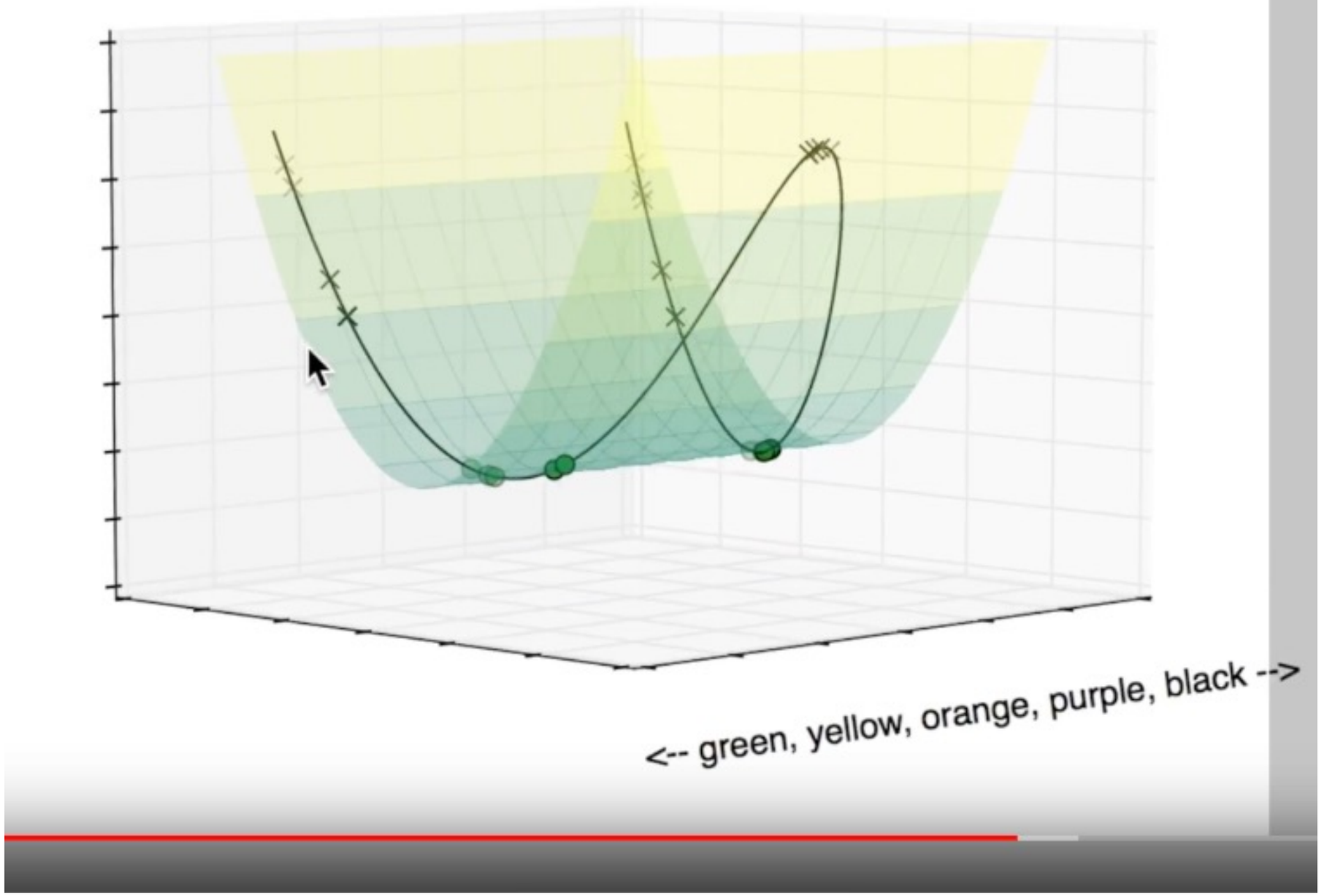
Graph courtesy: Brandon



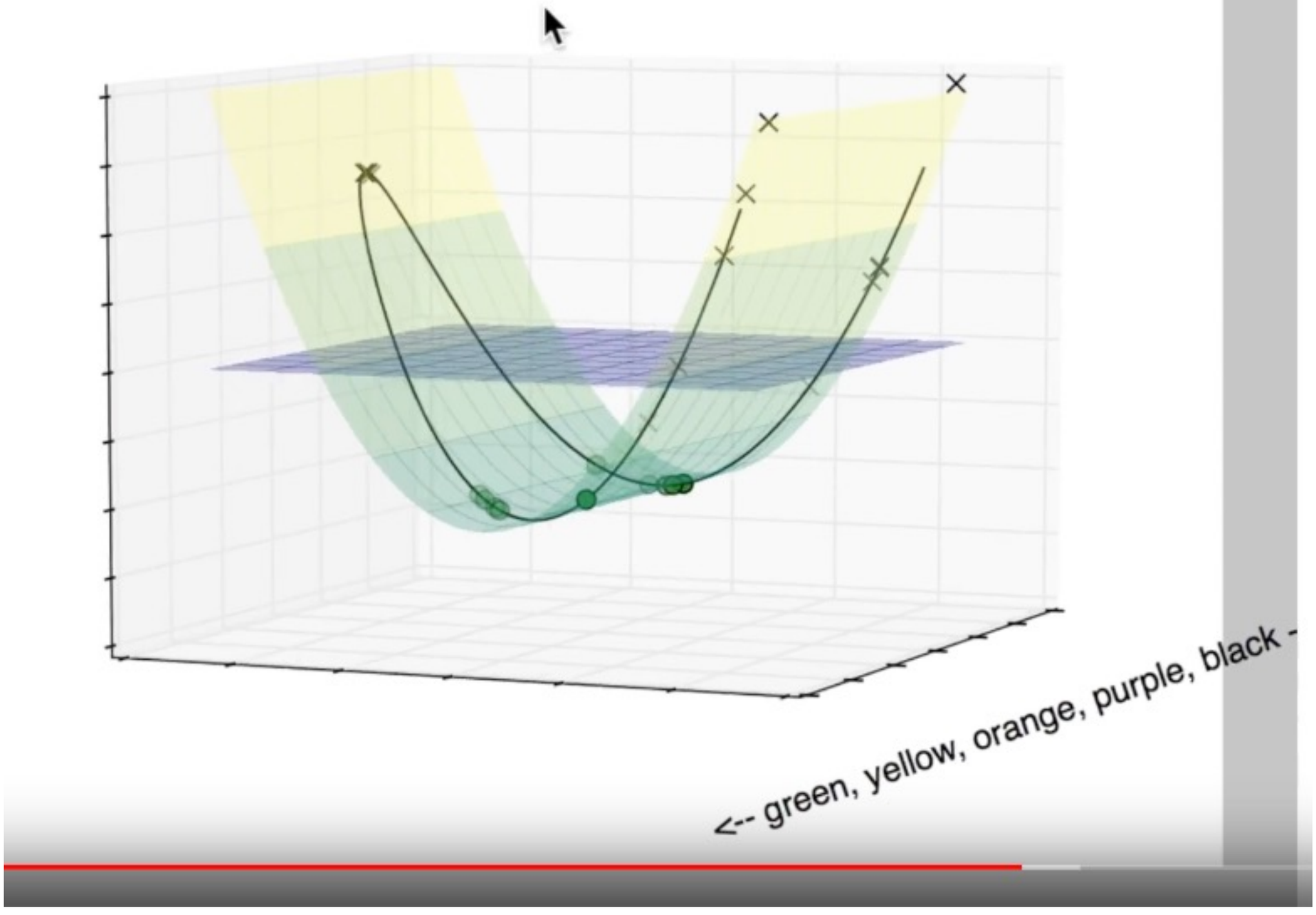
Graph courtesy: Brandon



Graph courtesy: Brandon



Graph courtesy: Brandon



Graph courtesy: Brandon

Output of Linear *vs.* RBF