Performing Classification Using Multiple Techniques

Janani Ravi

CO-FOUNDER, LOONYCORN

www.loonycorn.com

Overview

scikit-learn support for classification models

Discriminant Analysis

Stochastic Gradient Descent

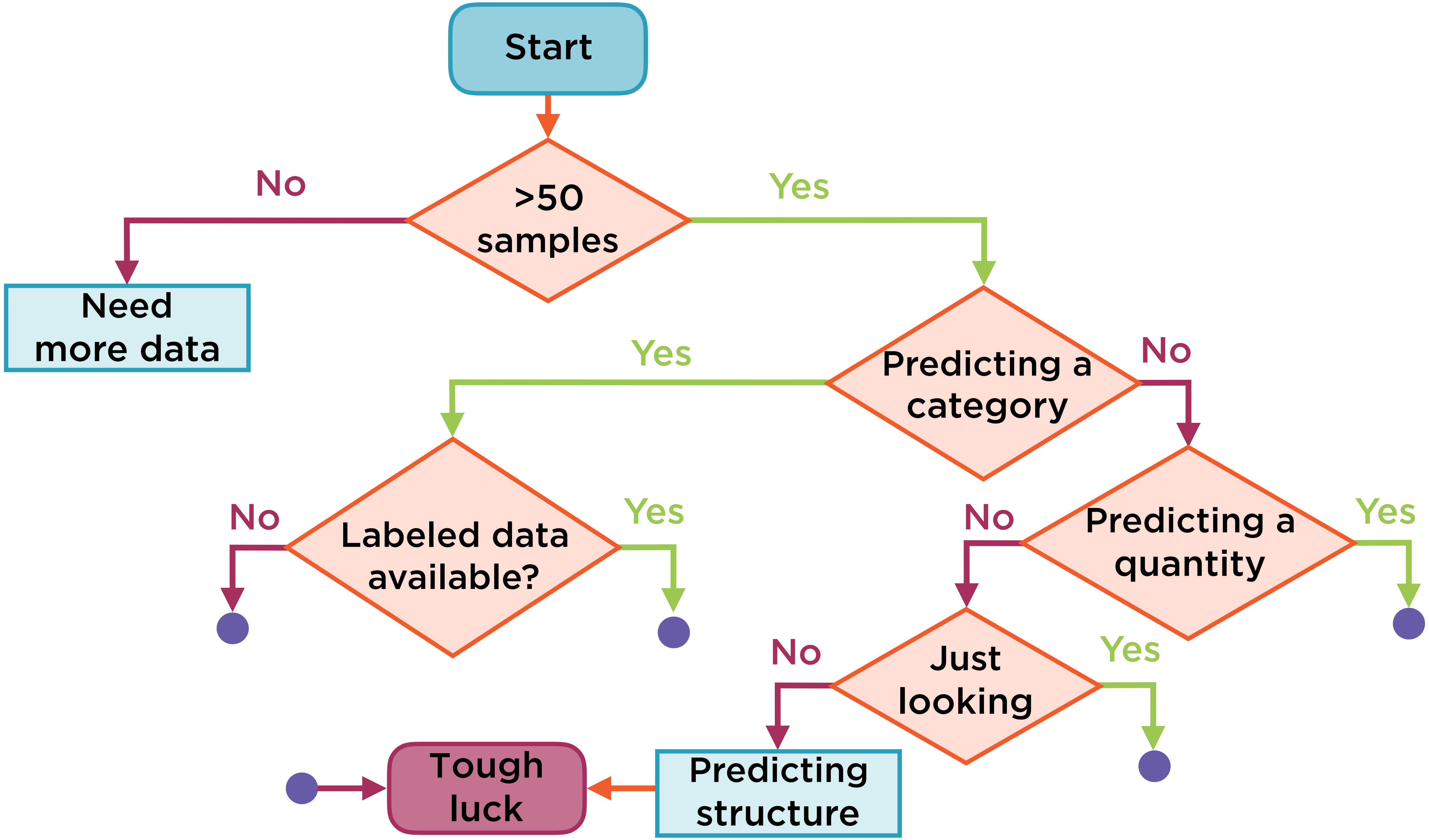
Support Vector Machines

Nearest Neighbors

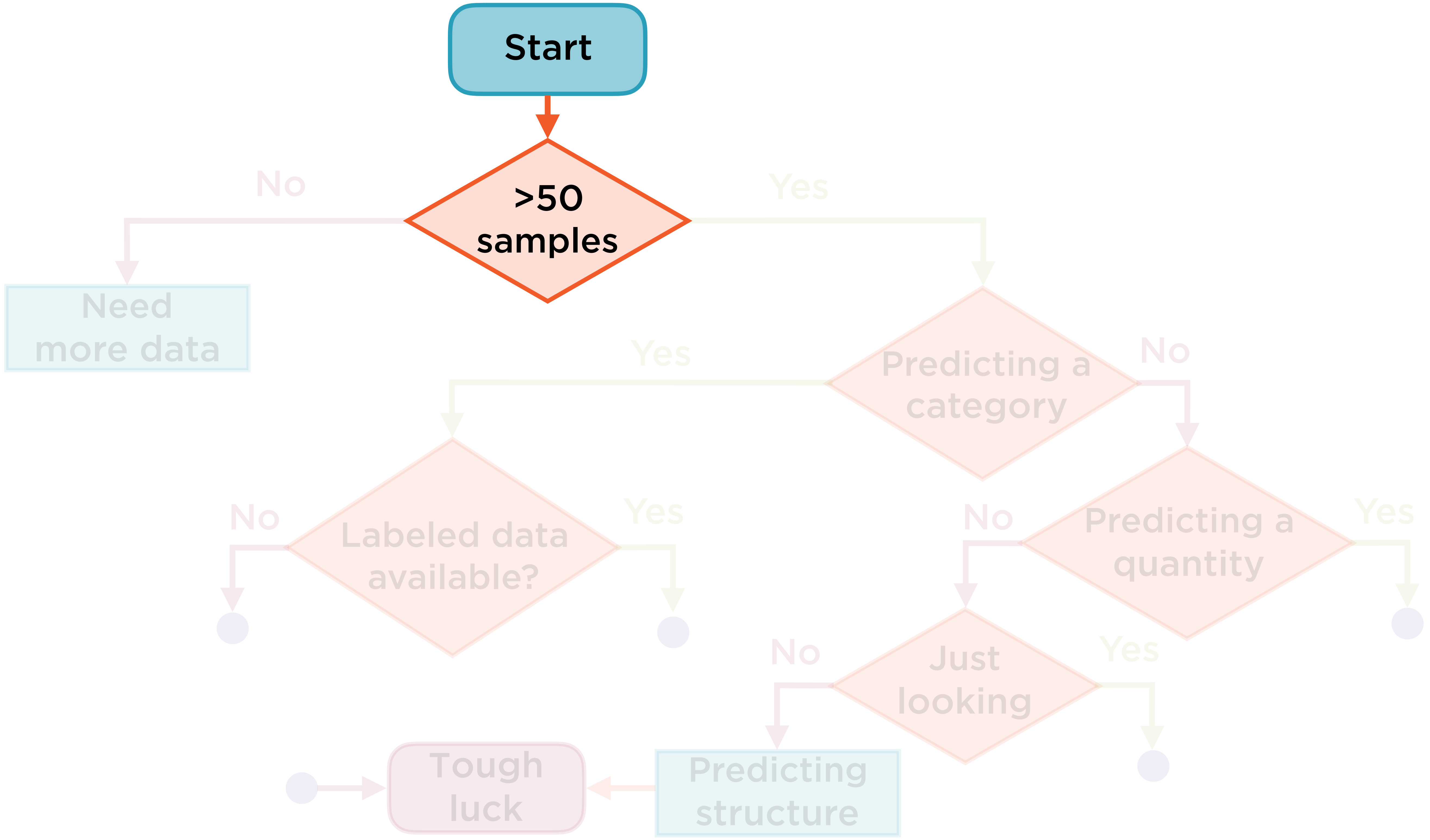
Decision Trees

Naive Bayes

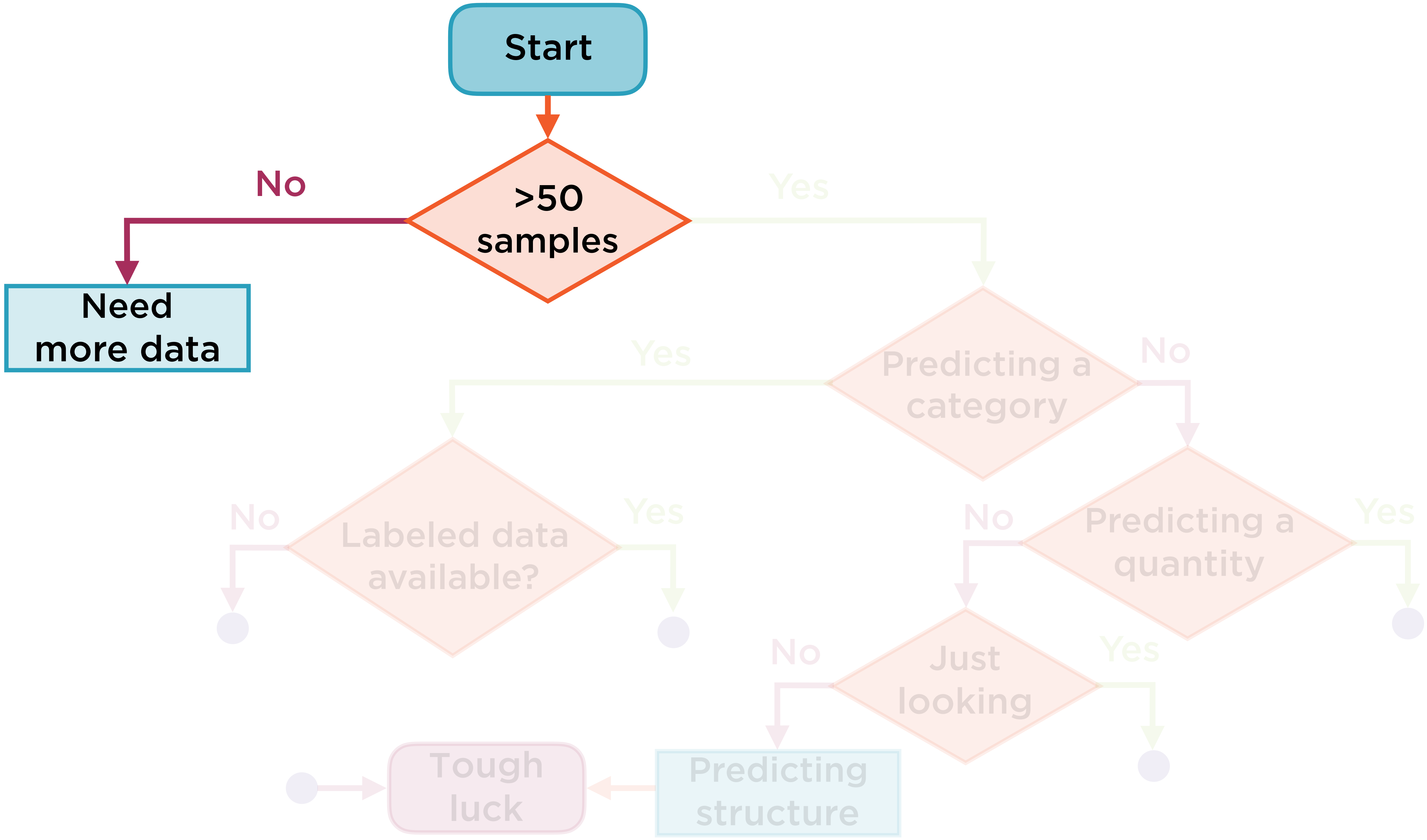
Choosing the Right Estimator



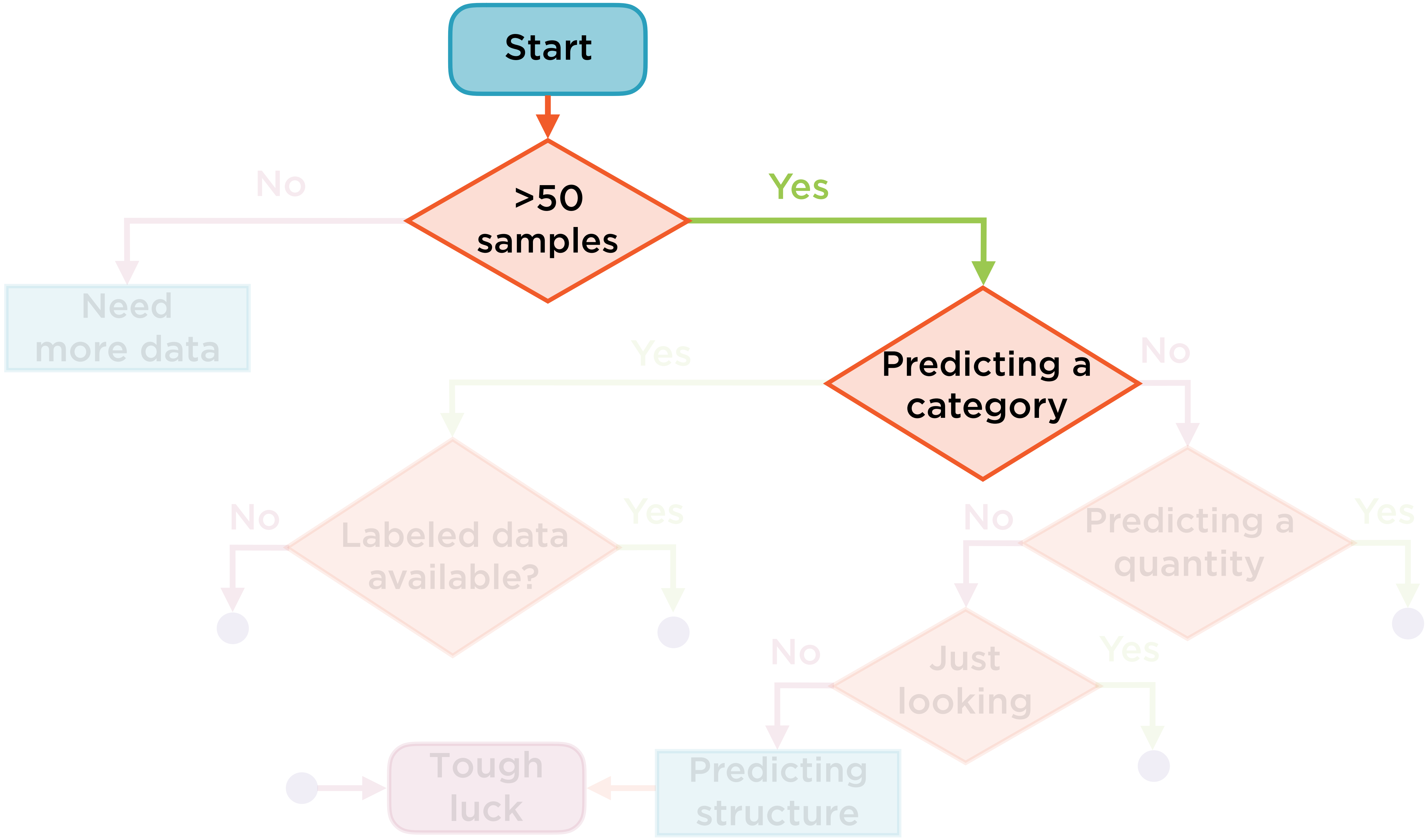
Choosing the Right Estimator



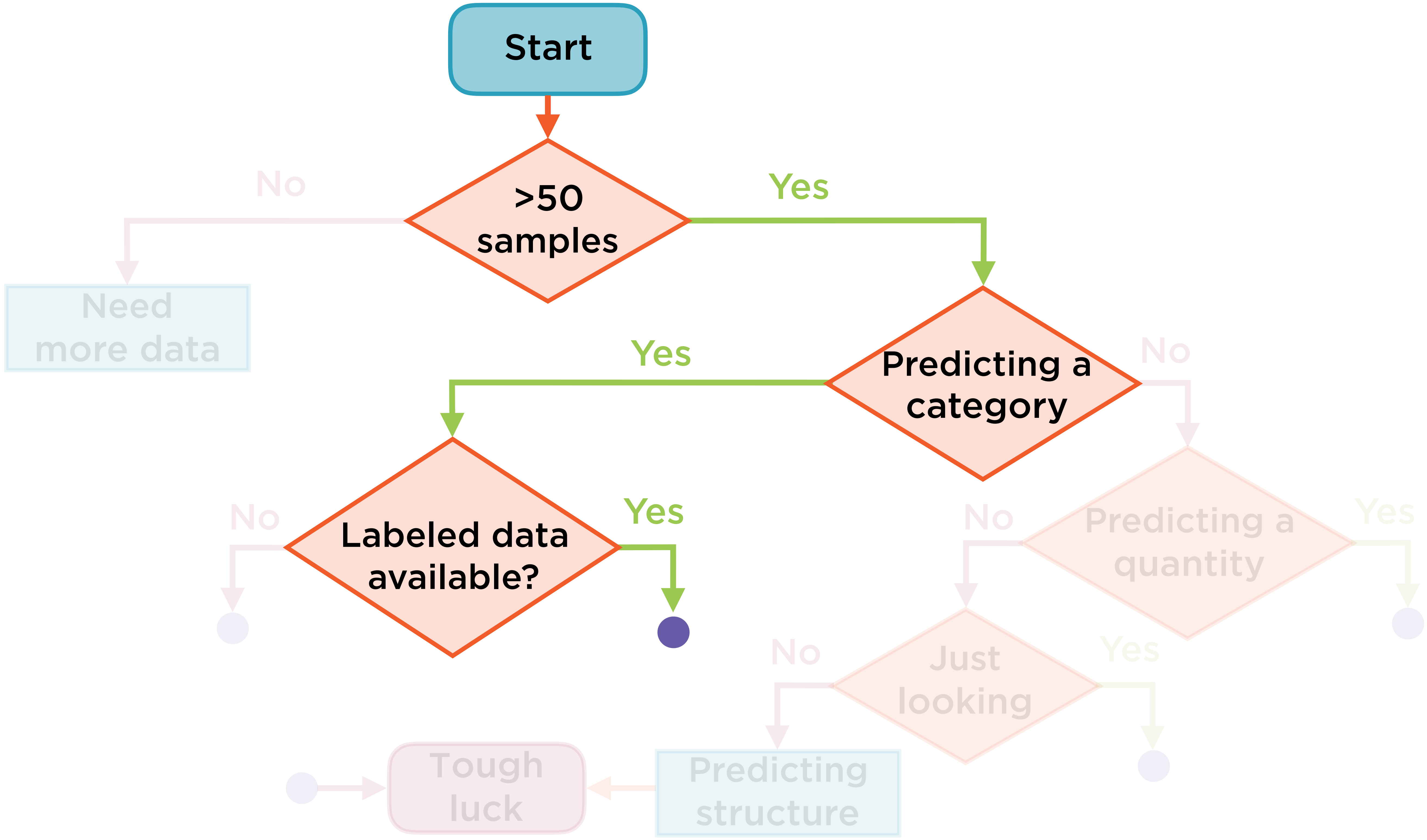
Choosing the Right Estimator



Choosing the Right Estimator



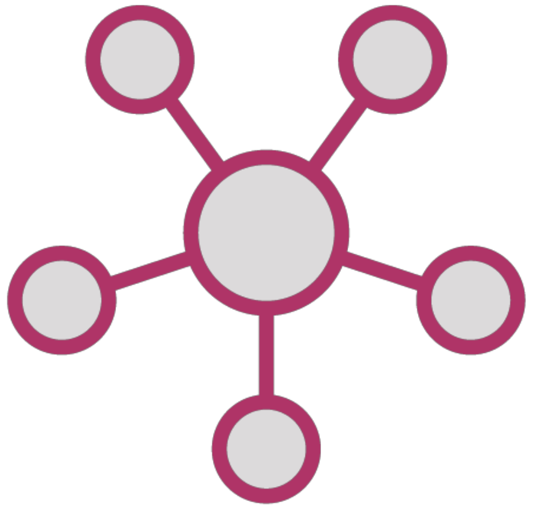
Choosing the Right Estimator



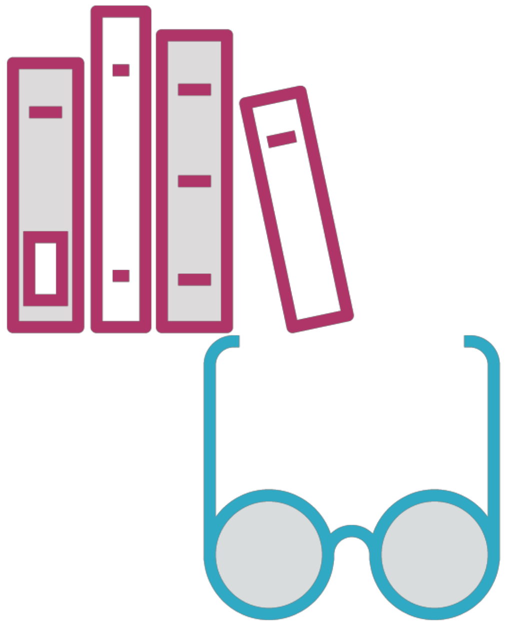
Types of Machine Learning Problems

Classification

Regression



Clustering



Dimensionality reduction

Types of Machine Learning Problems

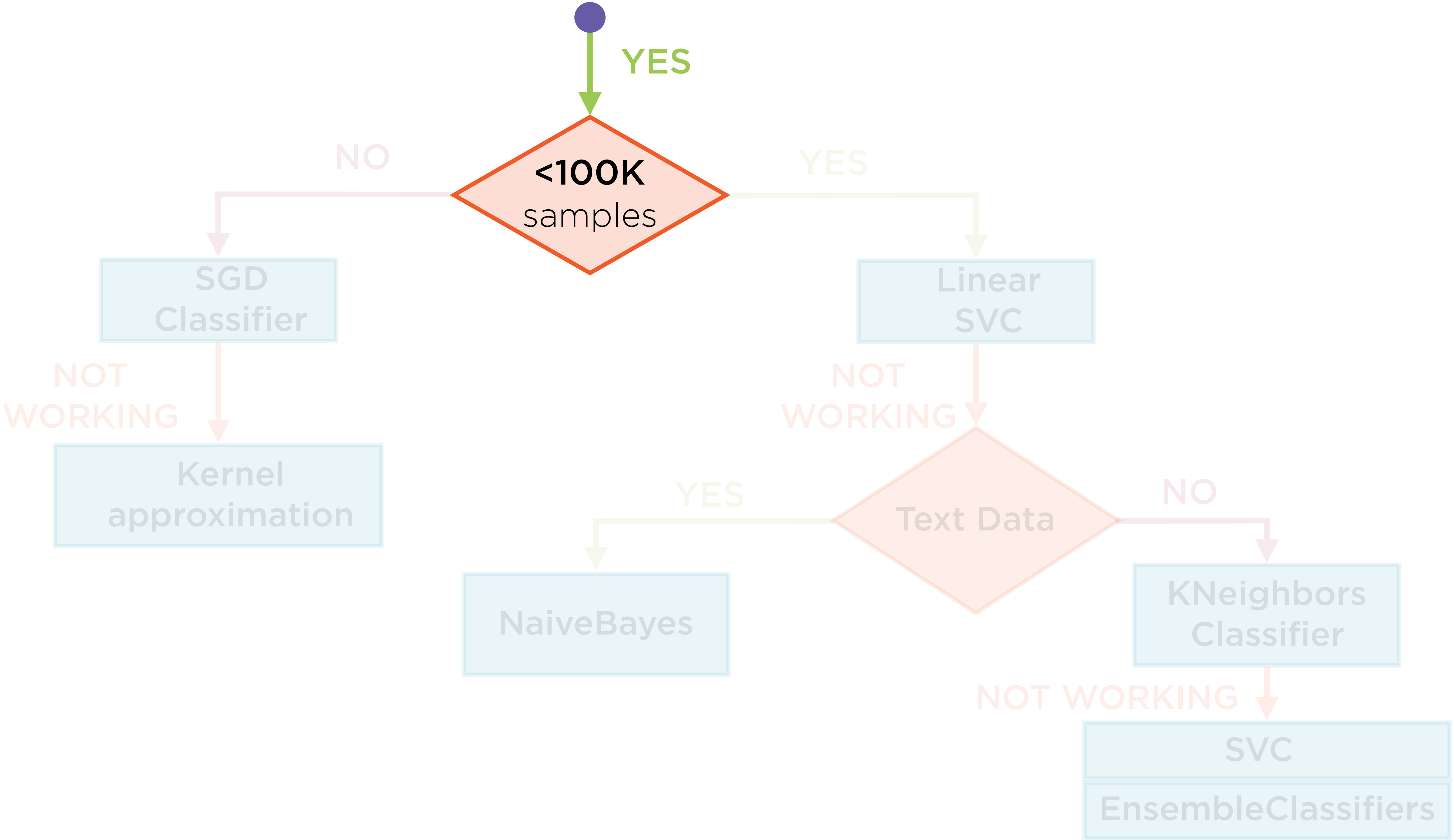
Classification

Regression

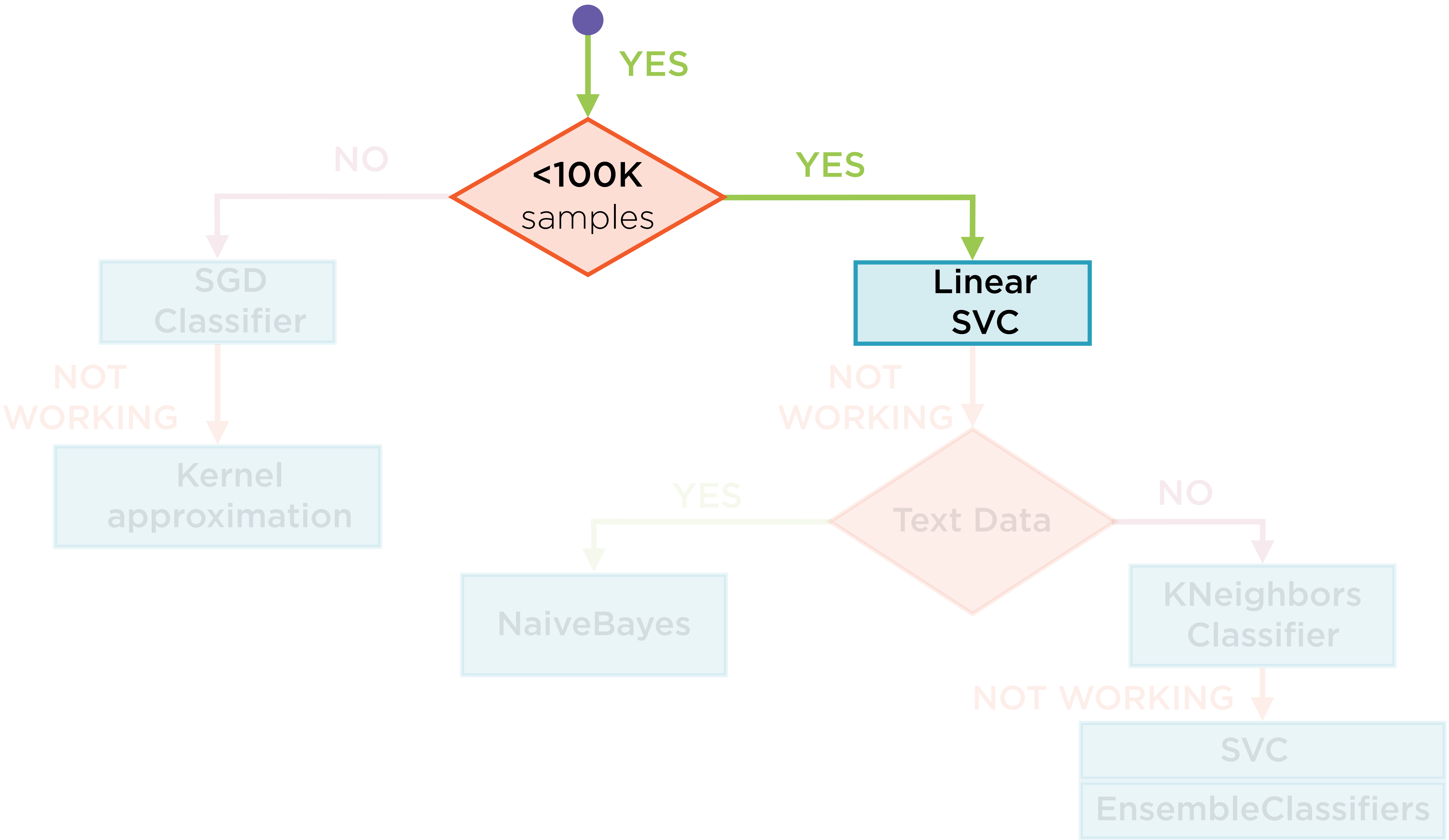
Clustering

Dimensionality reduction

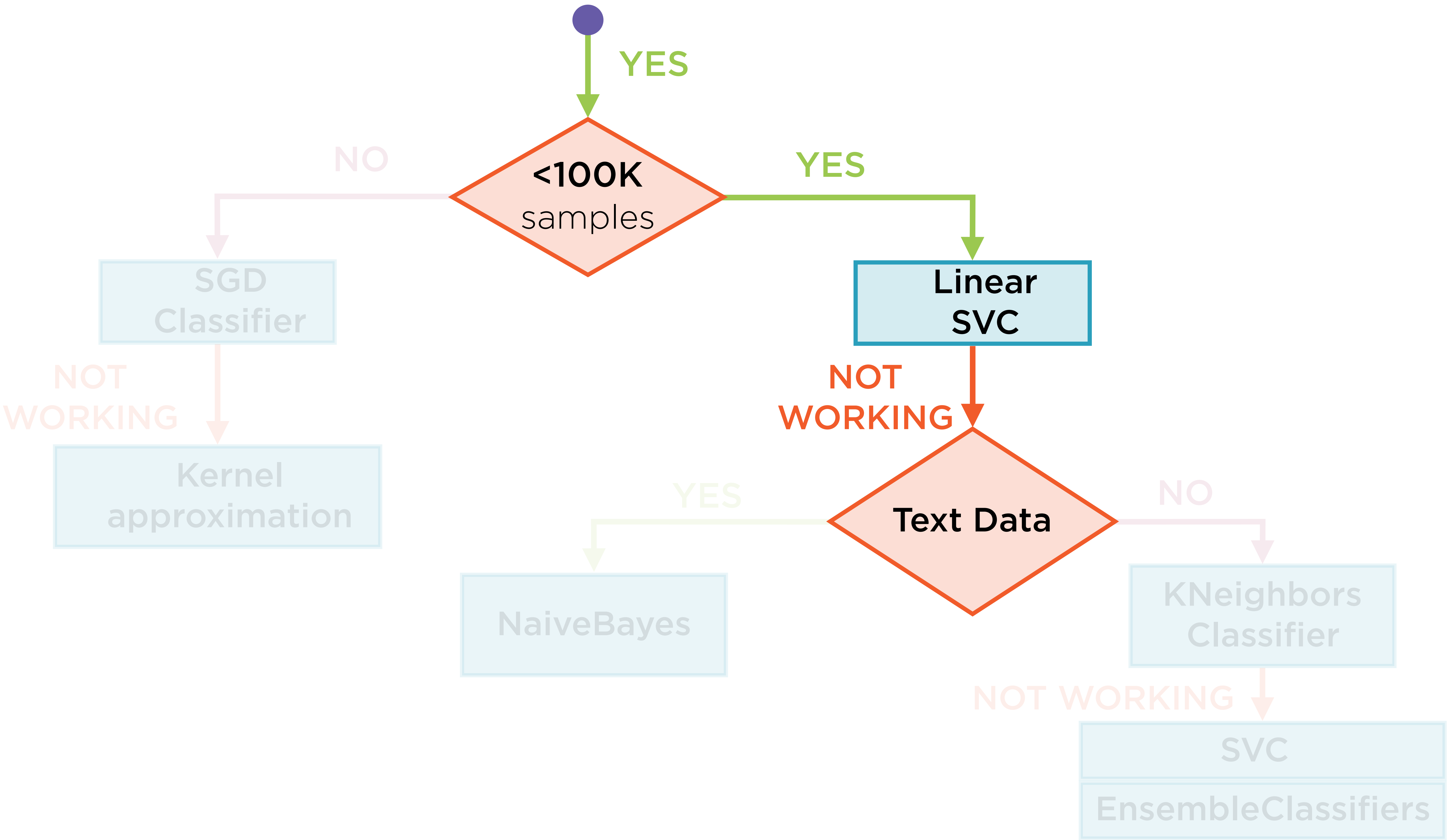
Classification



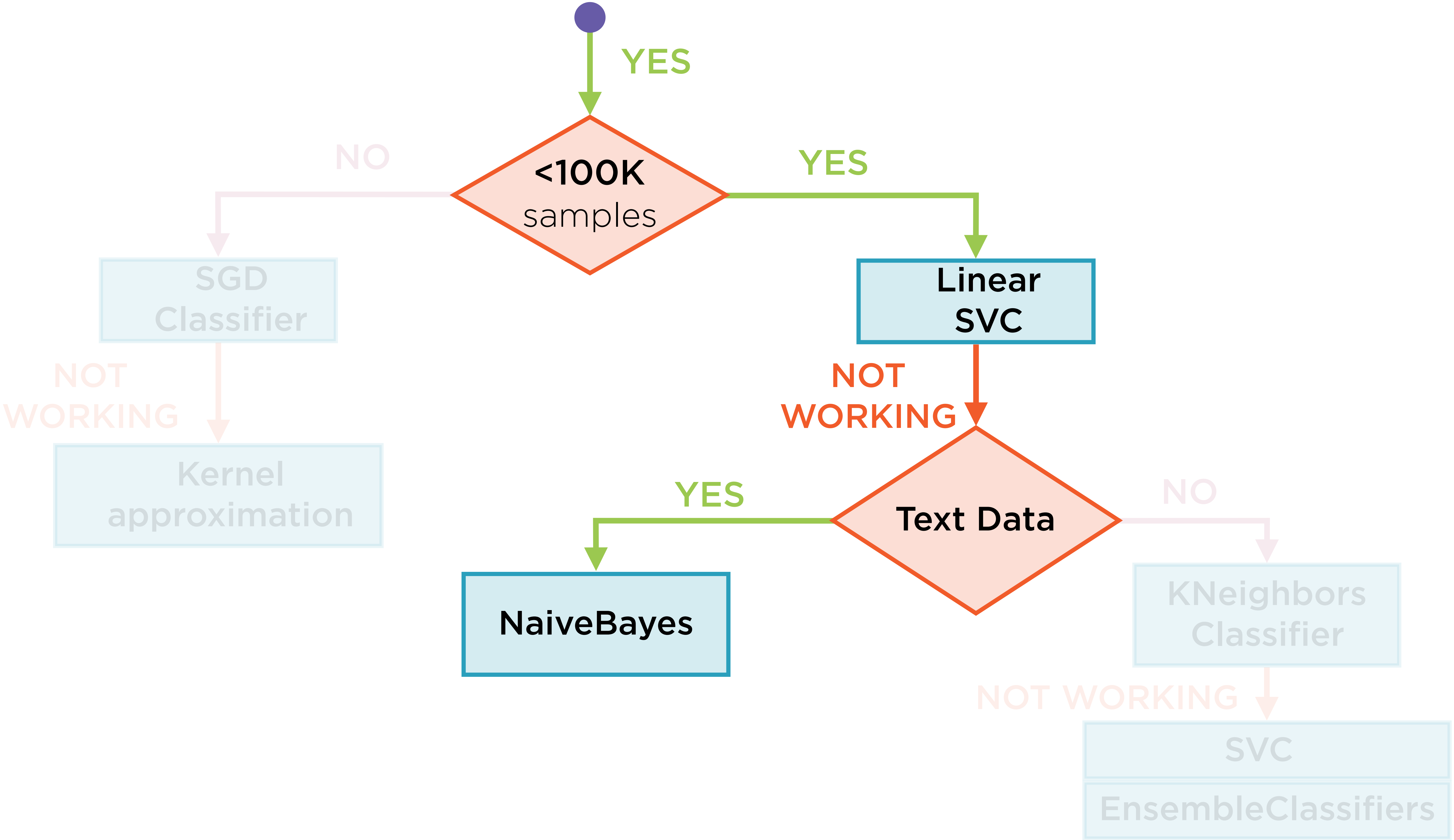
Classification



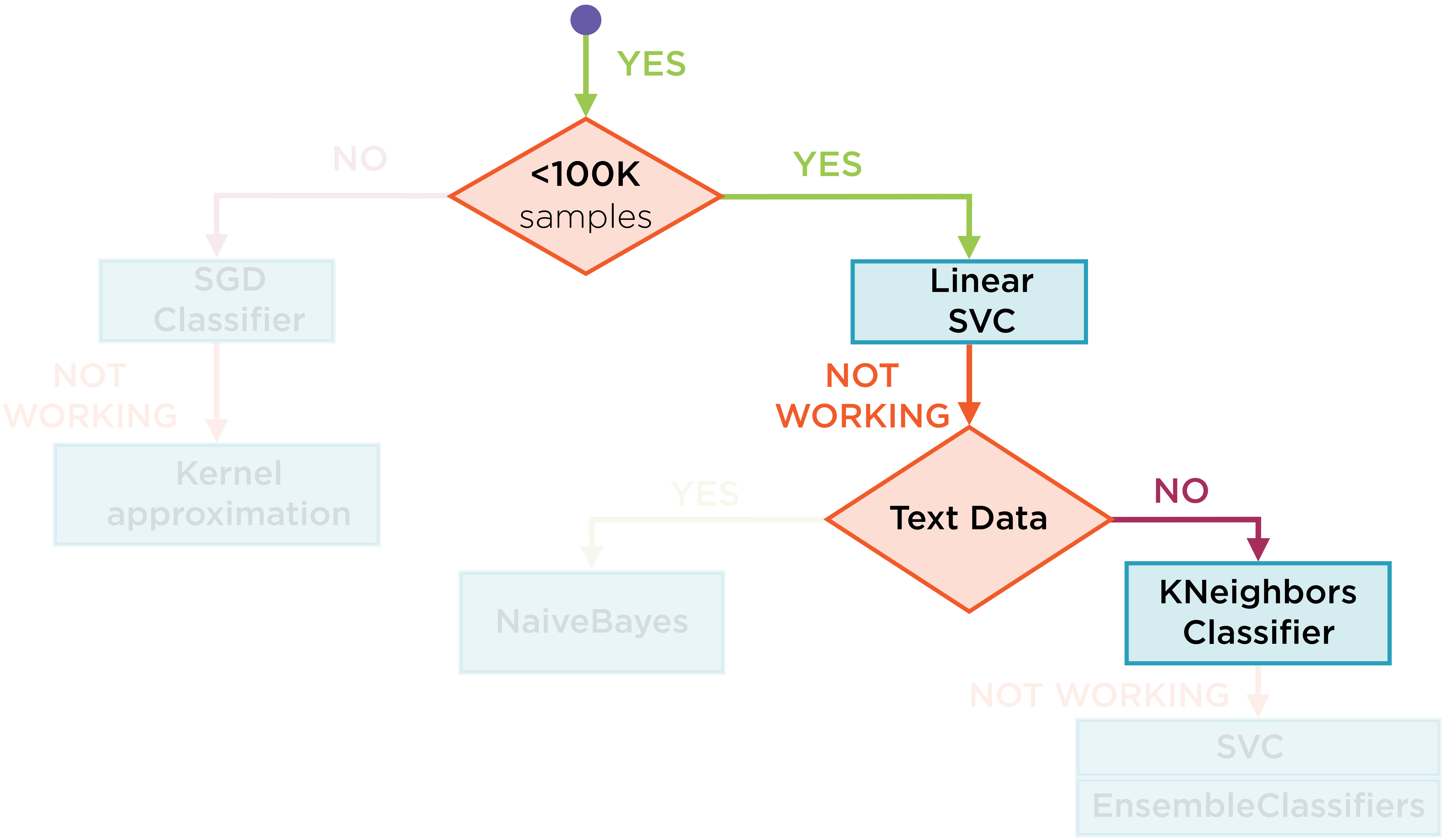
Classification



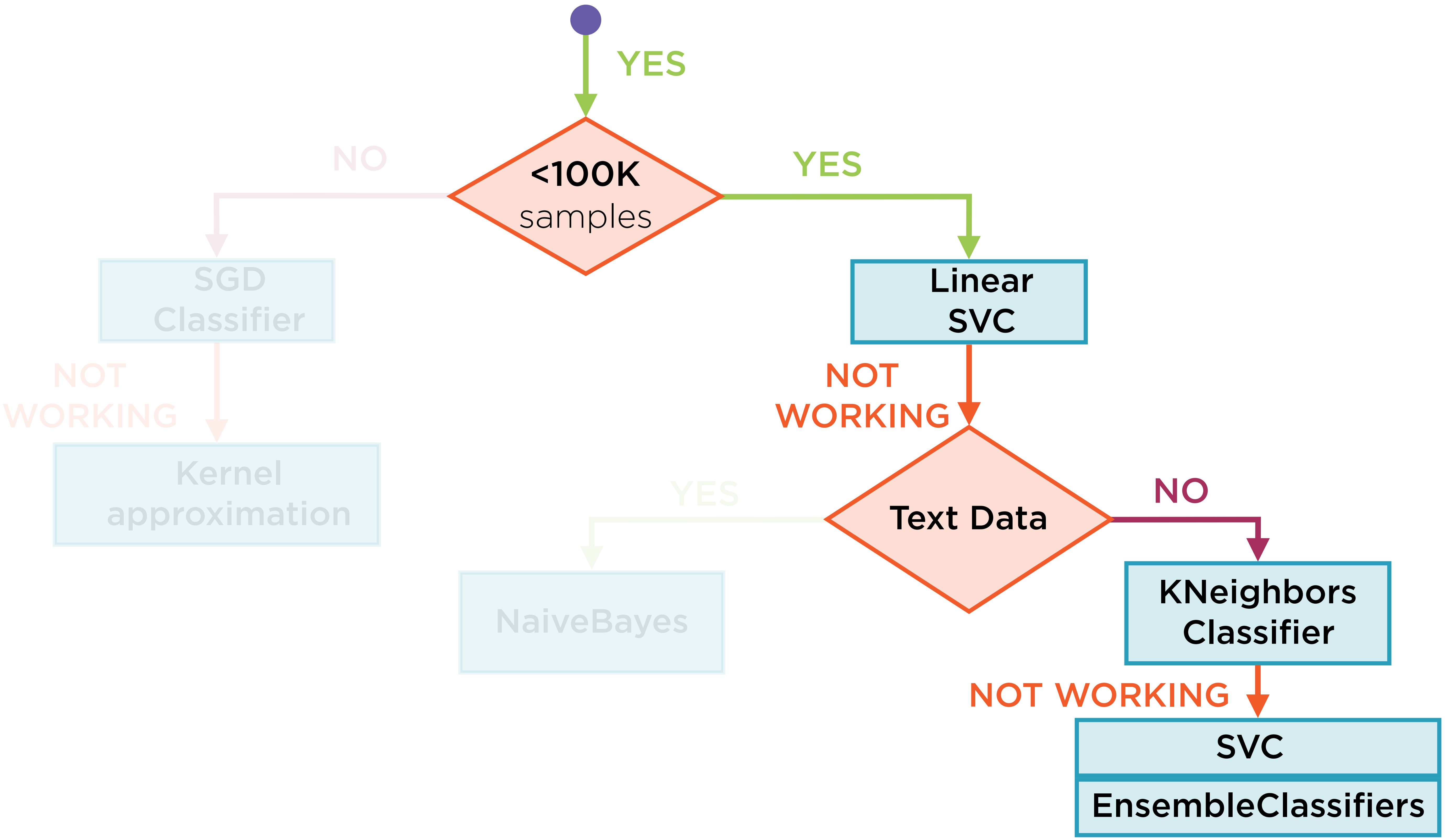
Classification



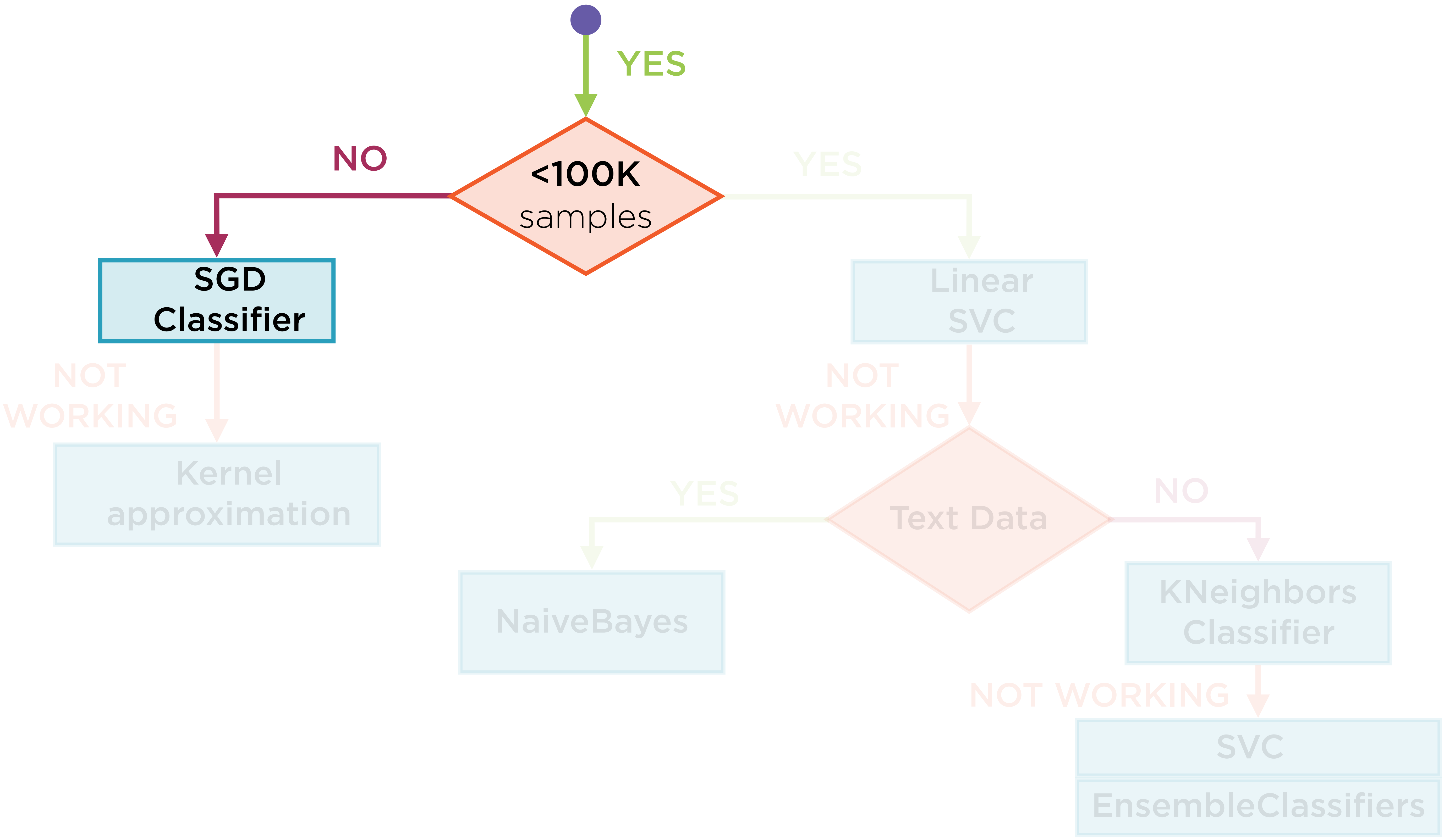
Classification



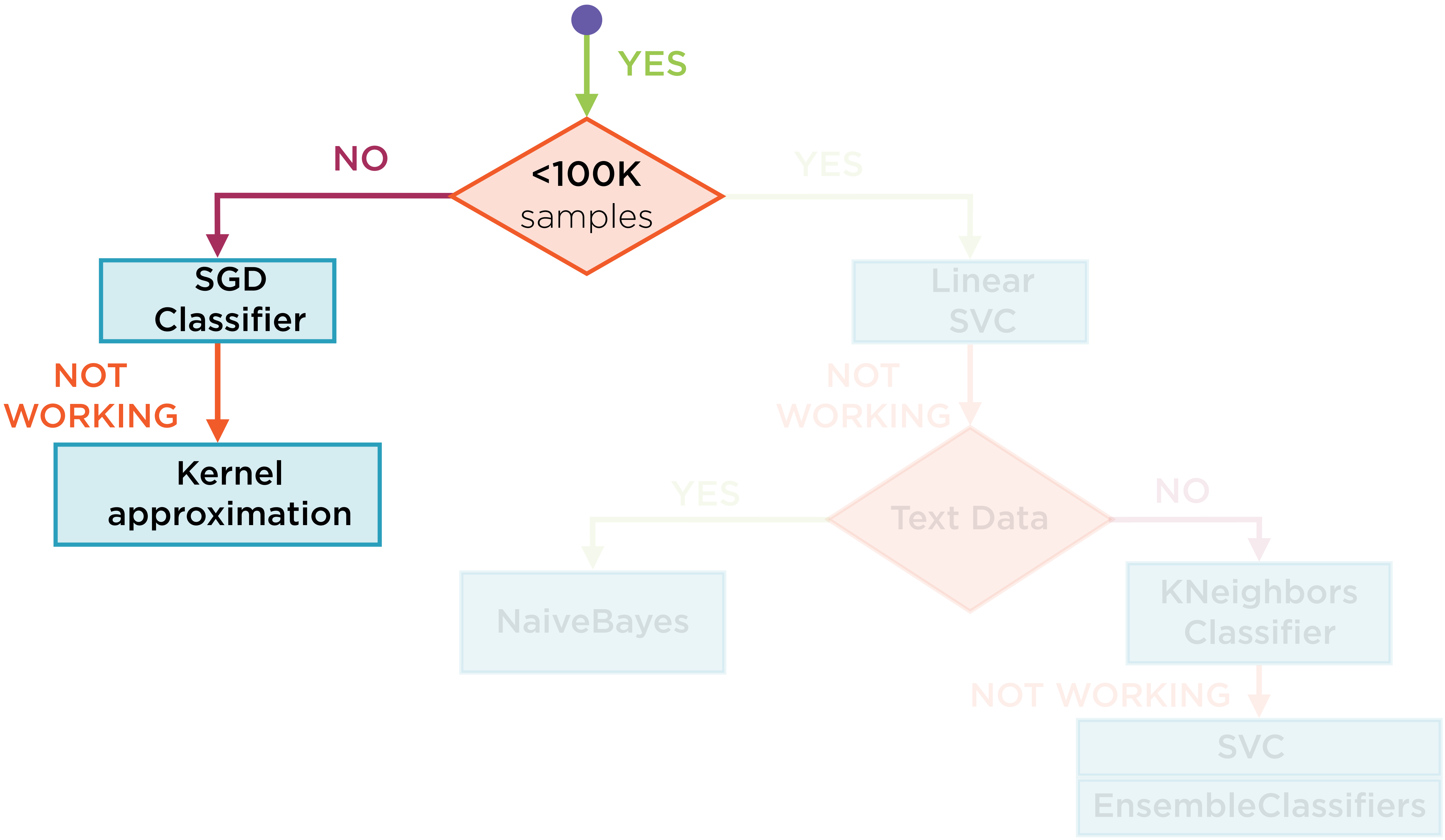
Classification



Classification

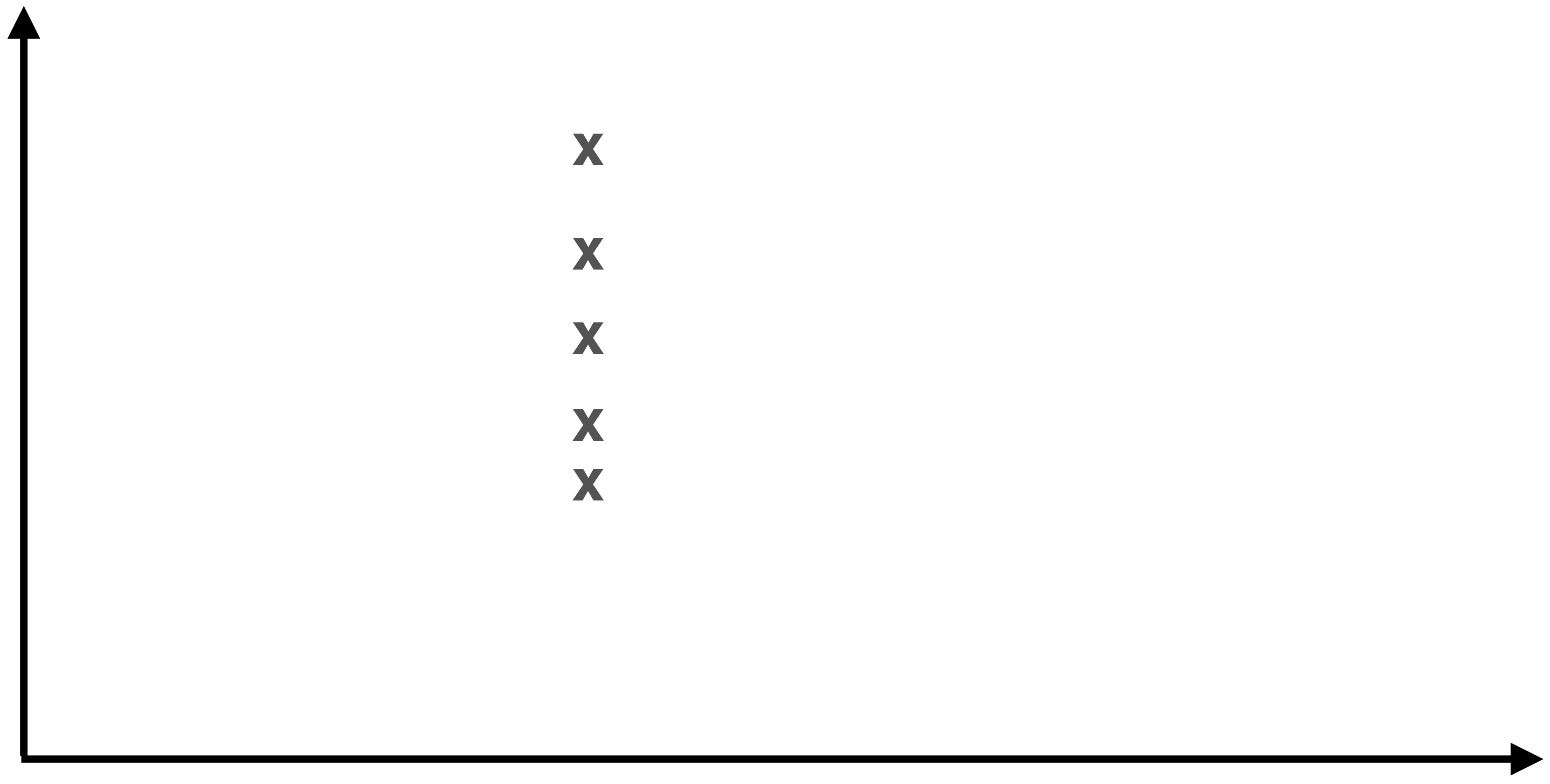


Classification



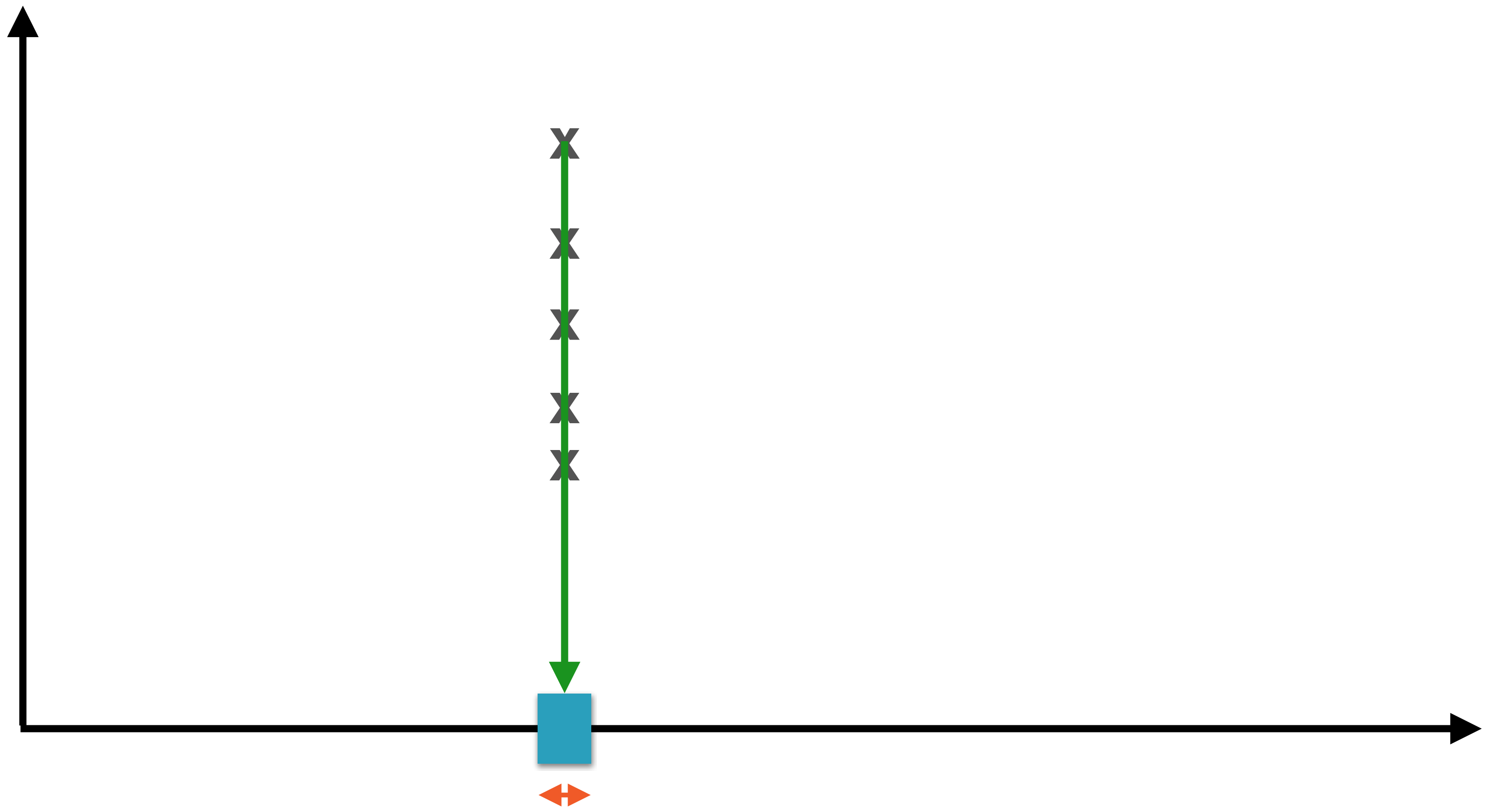
LDA and QDA Classifiers

A Question of Dimensionality



Pop quiz: Do we really need two dimensions to represent this data?

Bad Choice of Dimensions



If we choose our axes (dimensions) poorly then we do need two dimensions

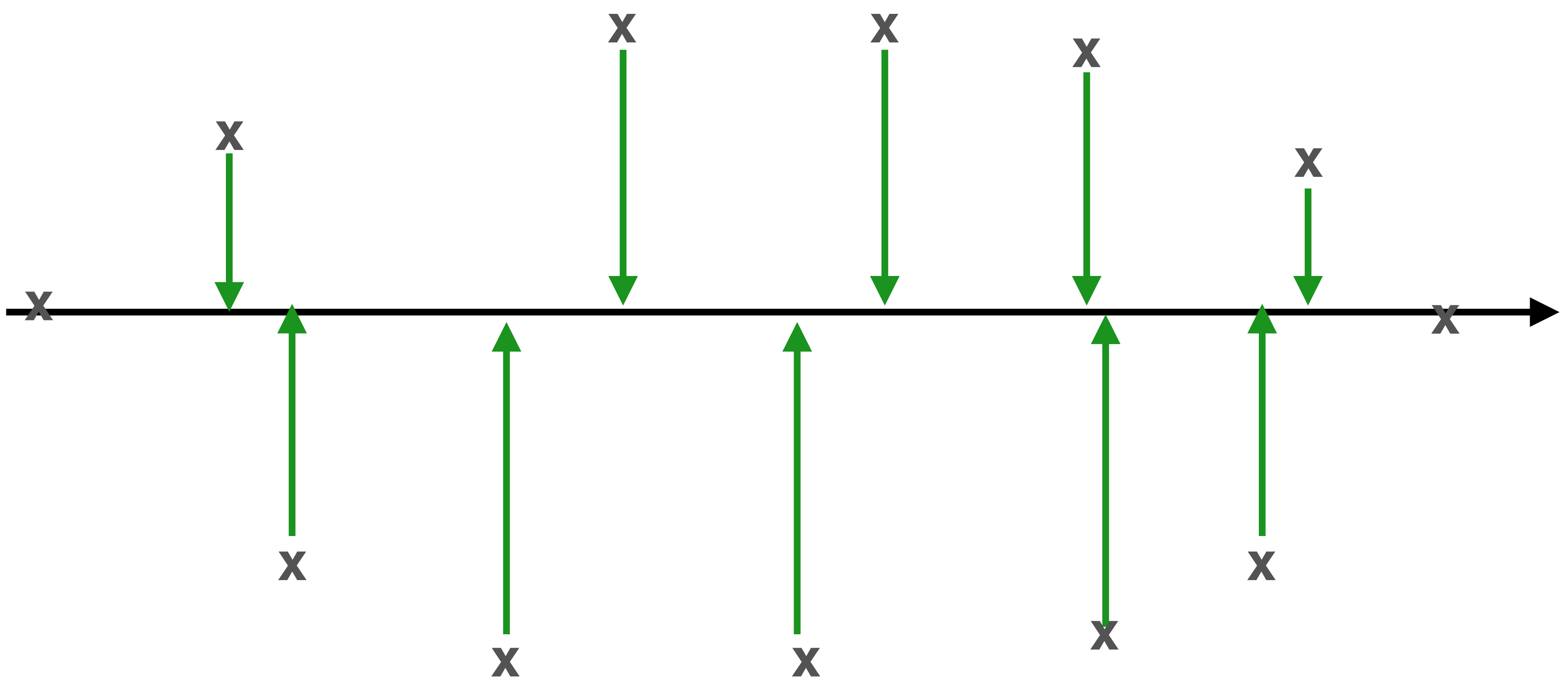
Good Choice of Dimensions

If we choose our axes (dimensions) well then one dimension is sufficient

Intuition: Principal Components Analysis

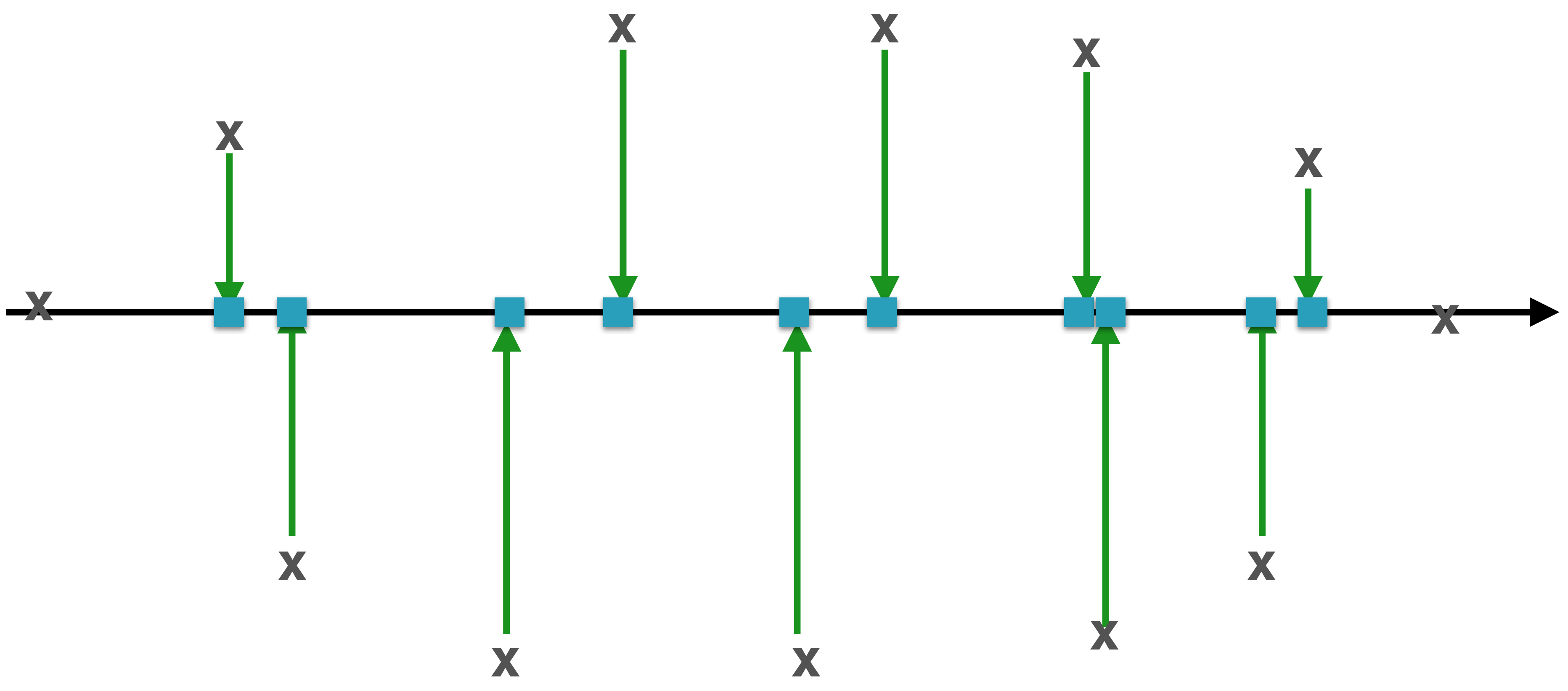
Objective: Find the “best” directions to represent this data

Intuition: Principal Components Analysis



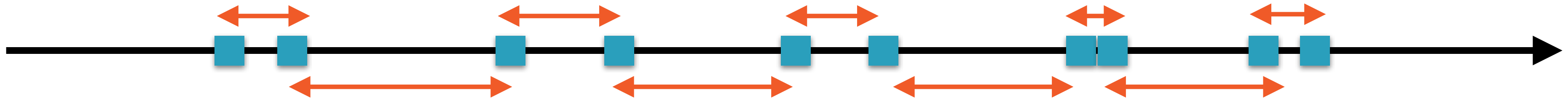
Start by "projecting" the data onto a line in some direction

Intuition: Principal Components Analysis



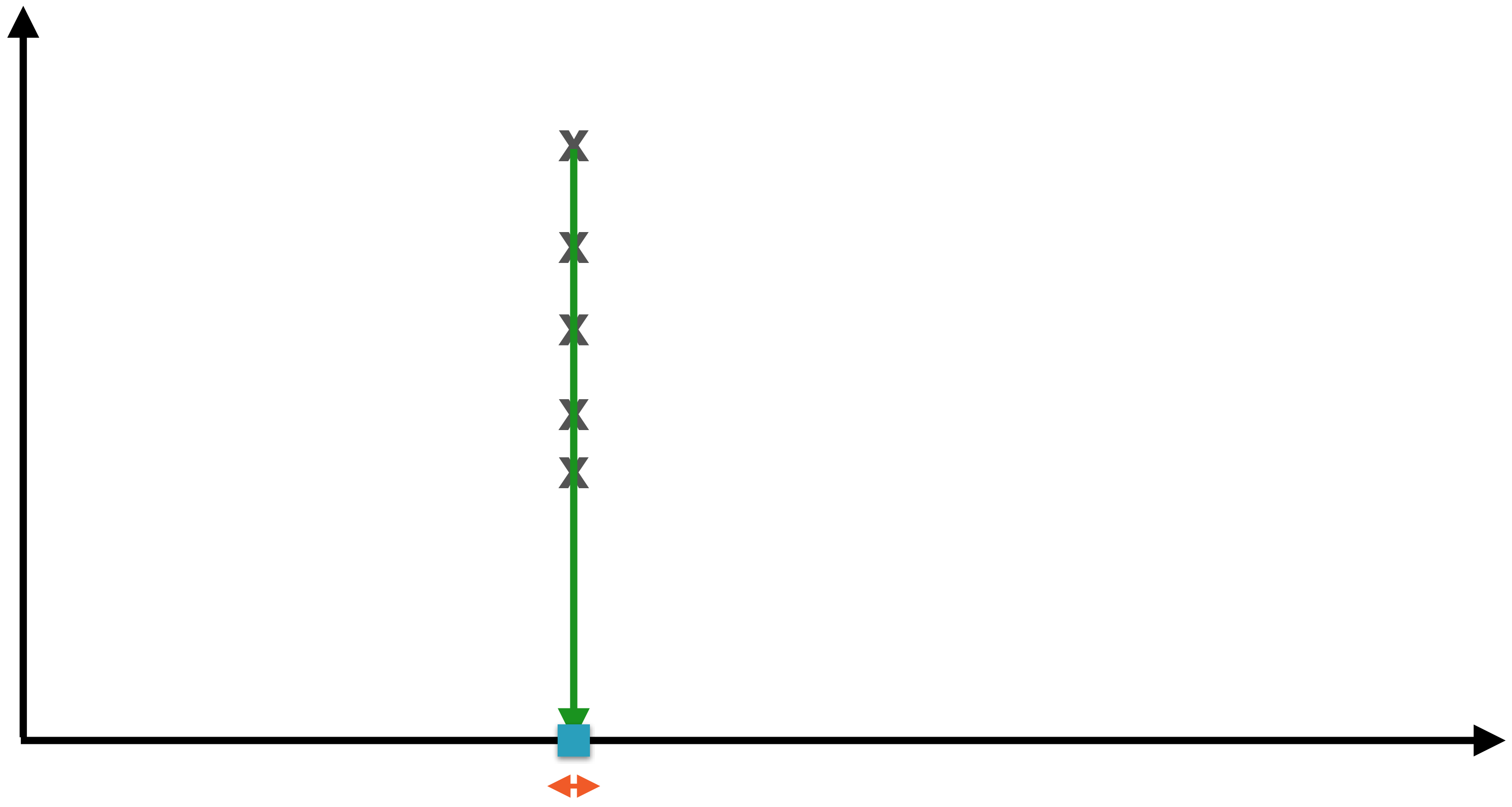
Start by "projecting" the data onto a line in some direction

Intuition: Principal Components Analysis



The greater the distances between these projections, the "better" the direction

Bad Projection

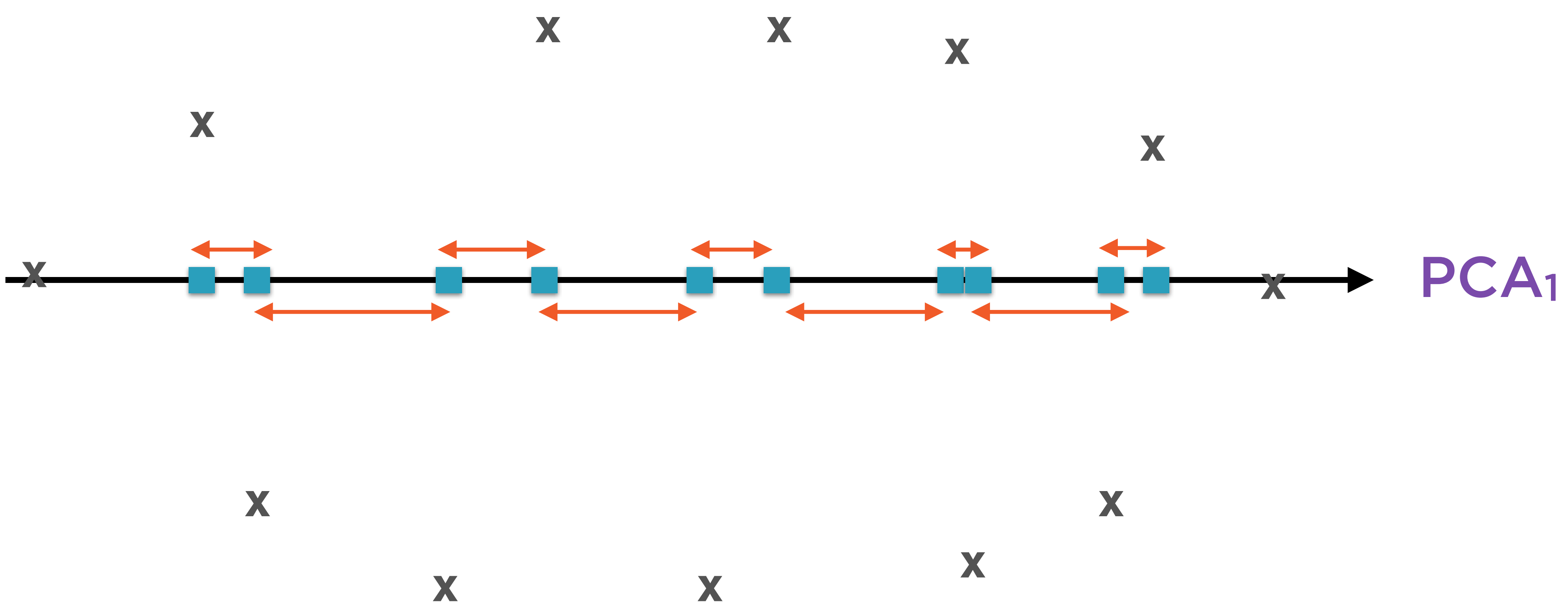


A projection where the distances are minimized is a bad one - information is lost

Good Projection

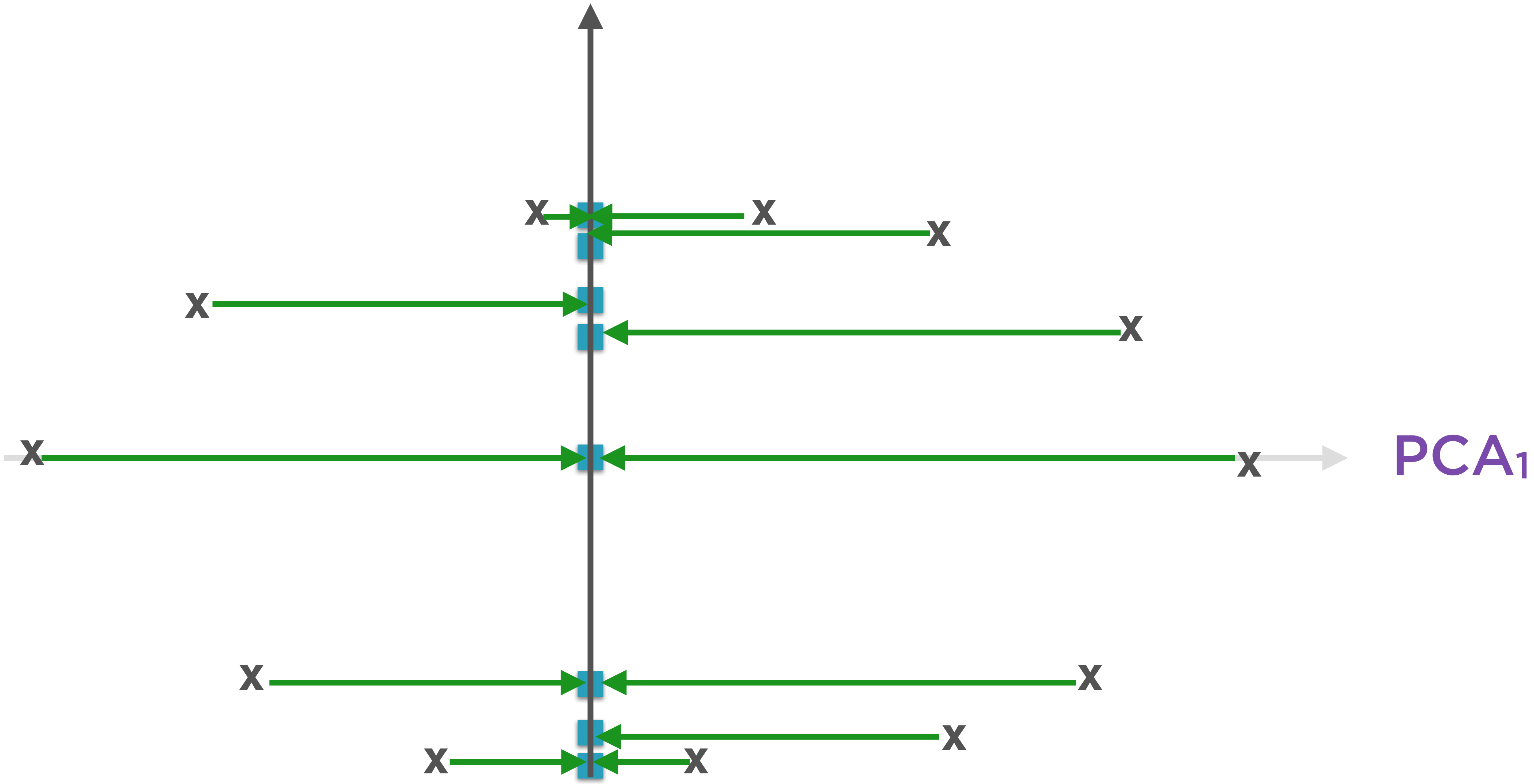
A projection where the distances are maximised is a good one - information is preserved

Intuition: Principal Components Analysis



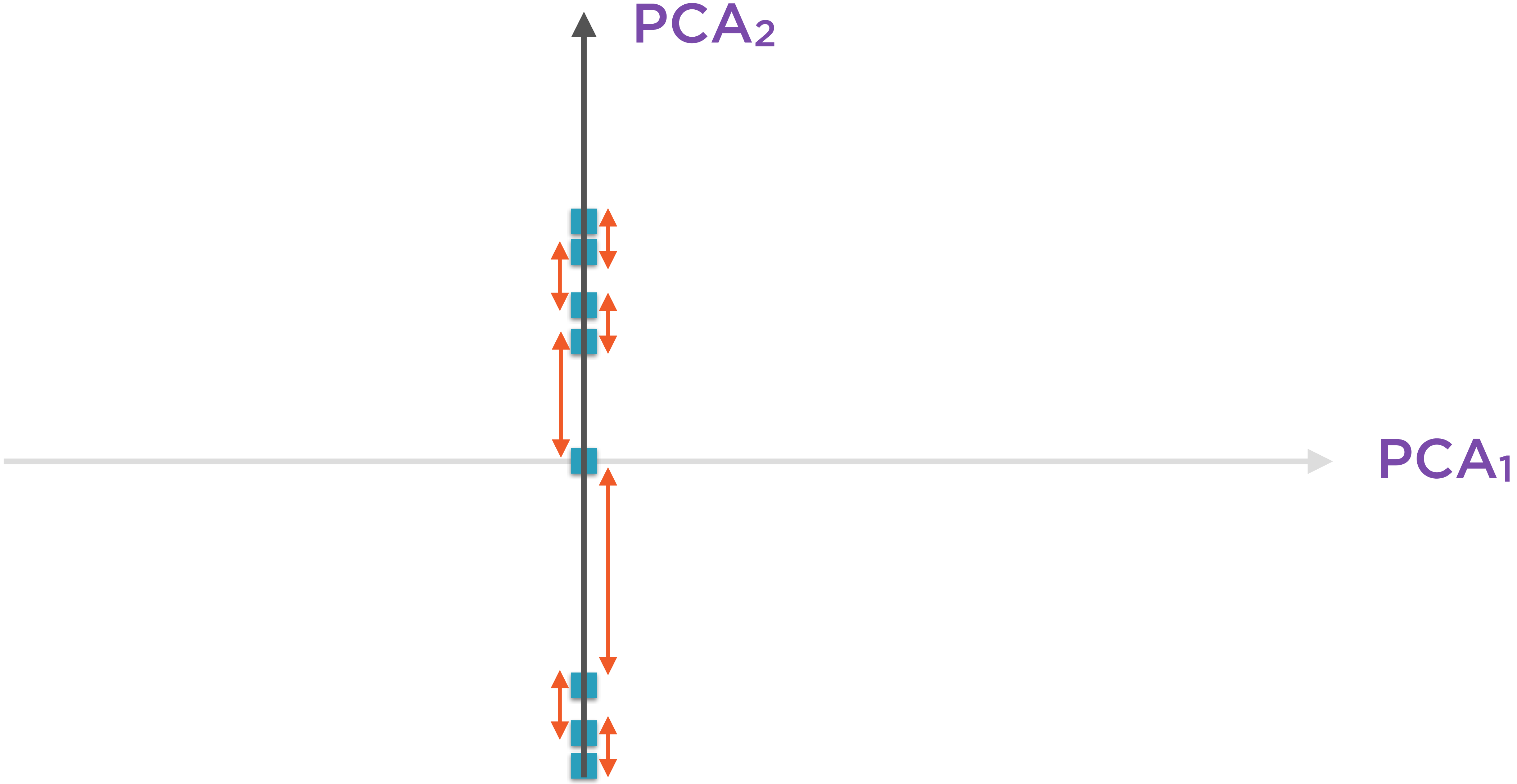
The direction along which this variance is maximised is the first principal component of the original data

Intuition: Principal Components Analysis



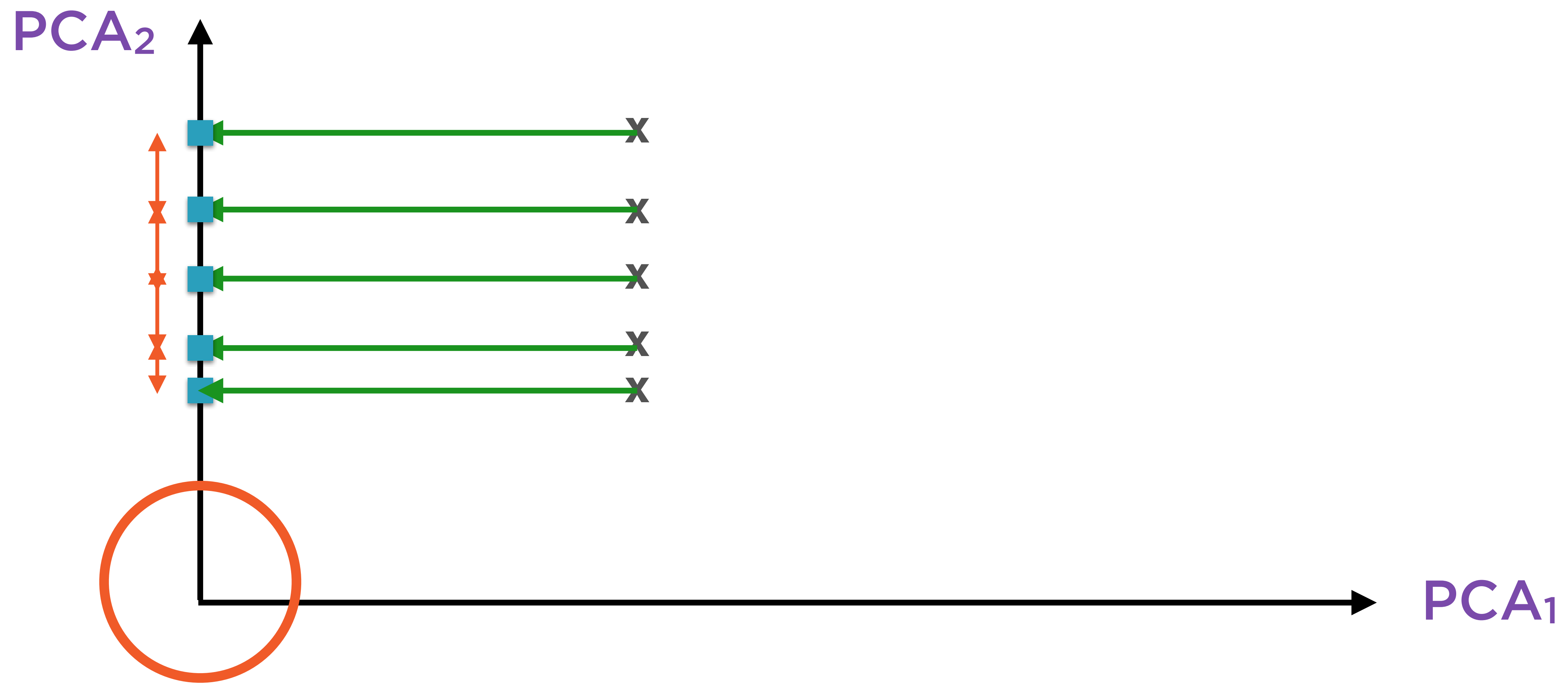
Find the next best direction, the **second principal component**, which must be at right angles to the first

Intuition: Principal Components Analysis



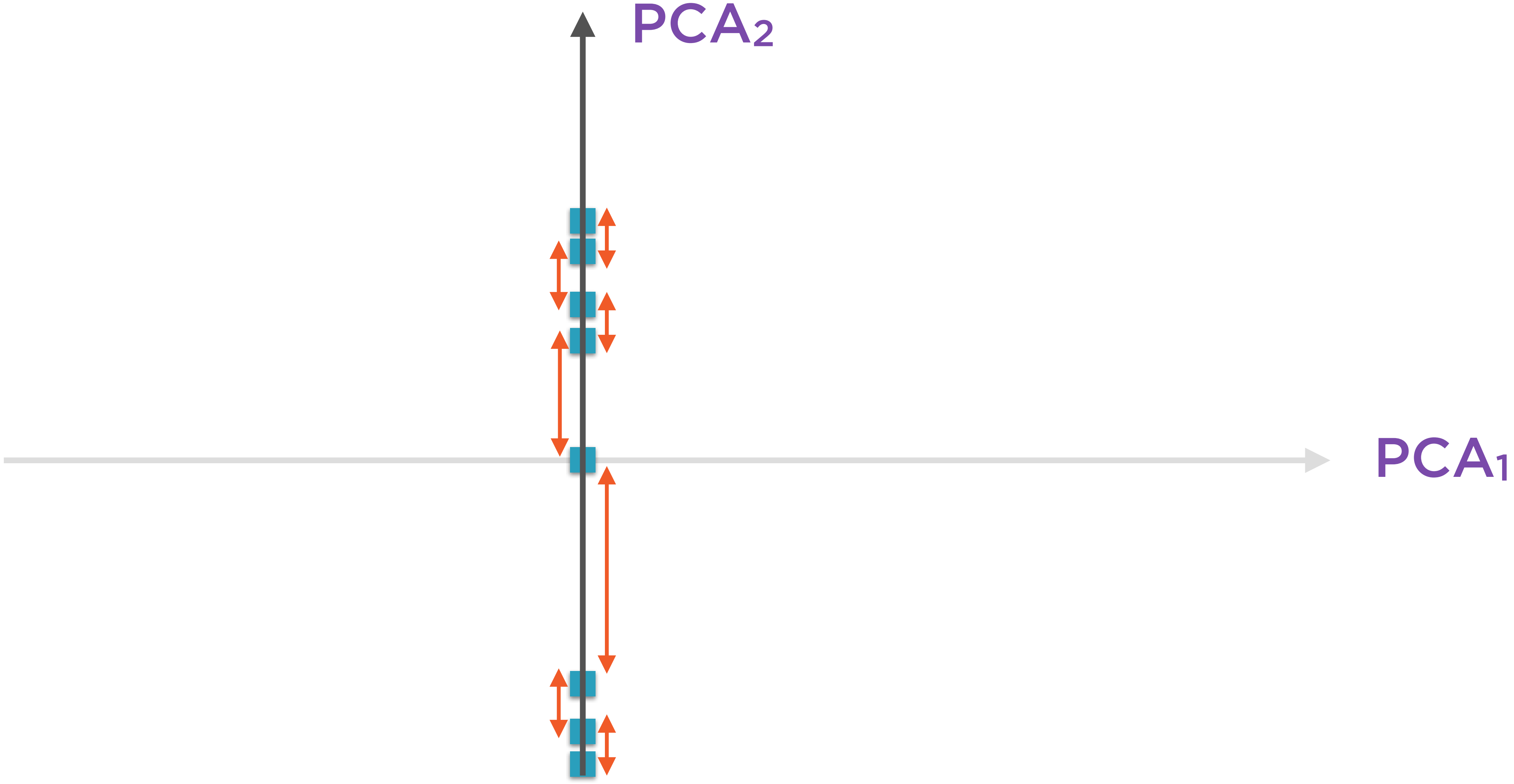
Find the next best direction, the second principal component, which must be at right angles to the first

Principal Components at Right Angles



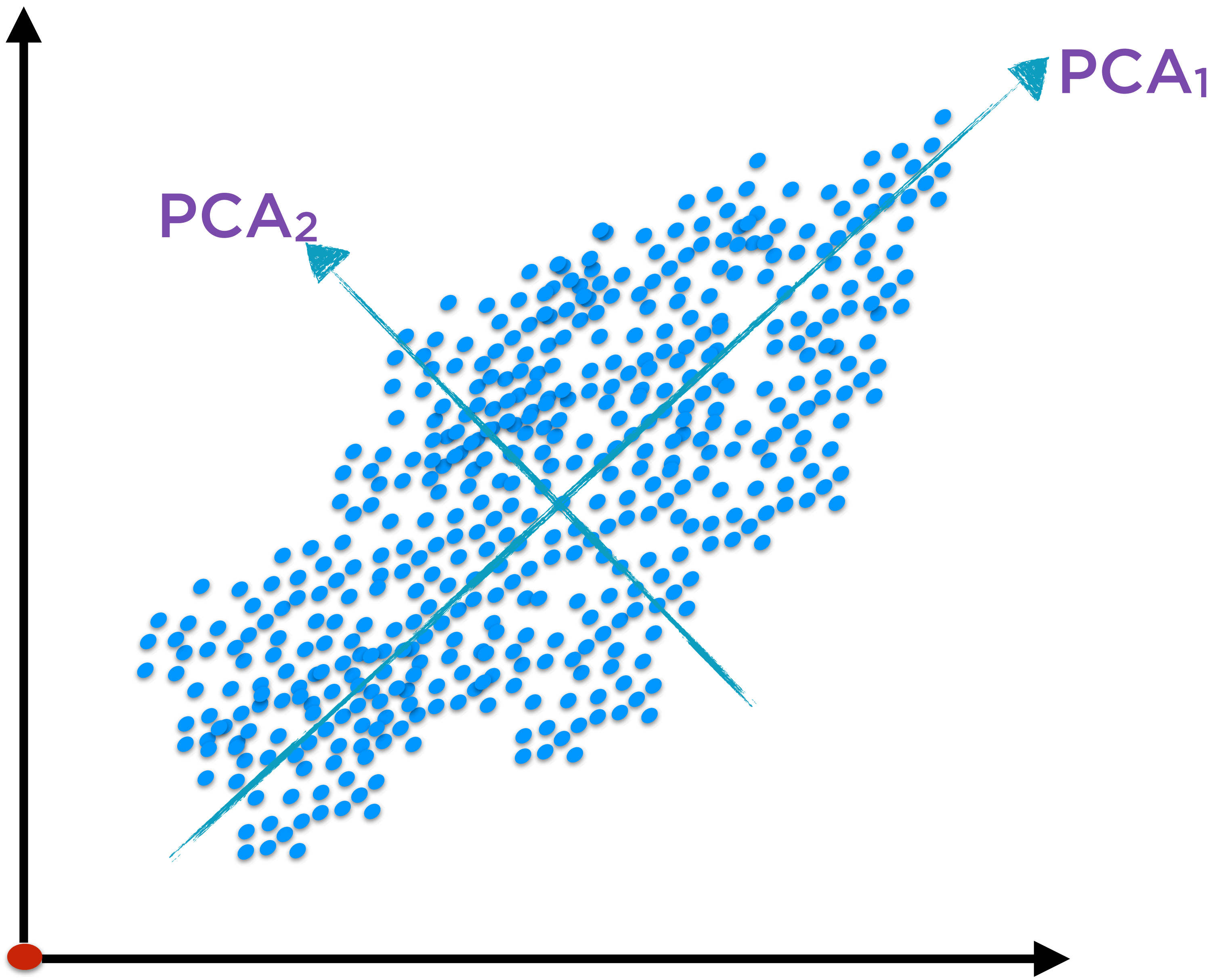
Directions at right angles help express the most variation with the smallest number of directions

Intuition Behind PCA



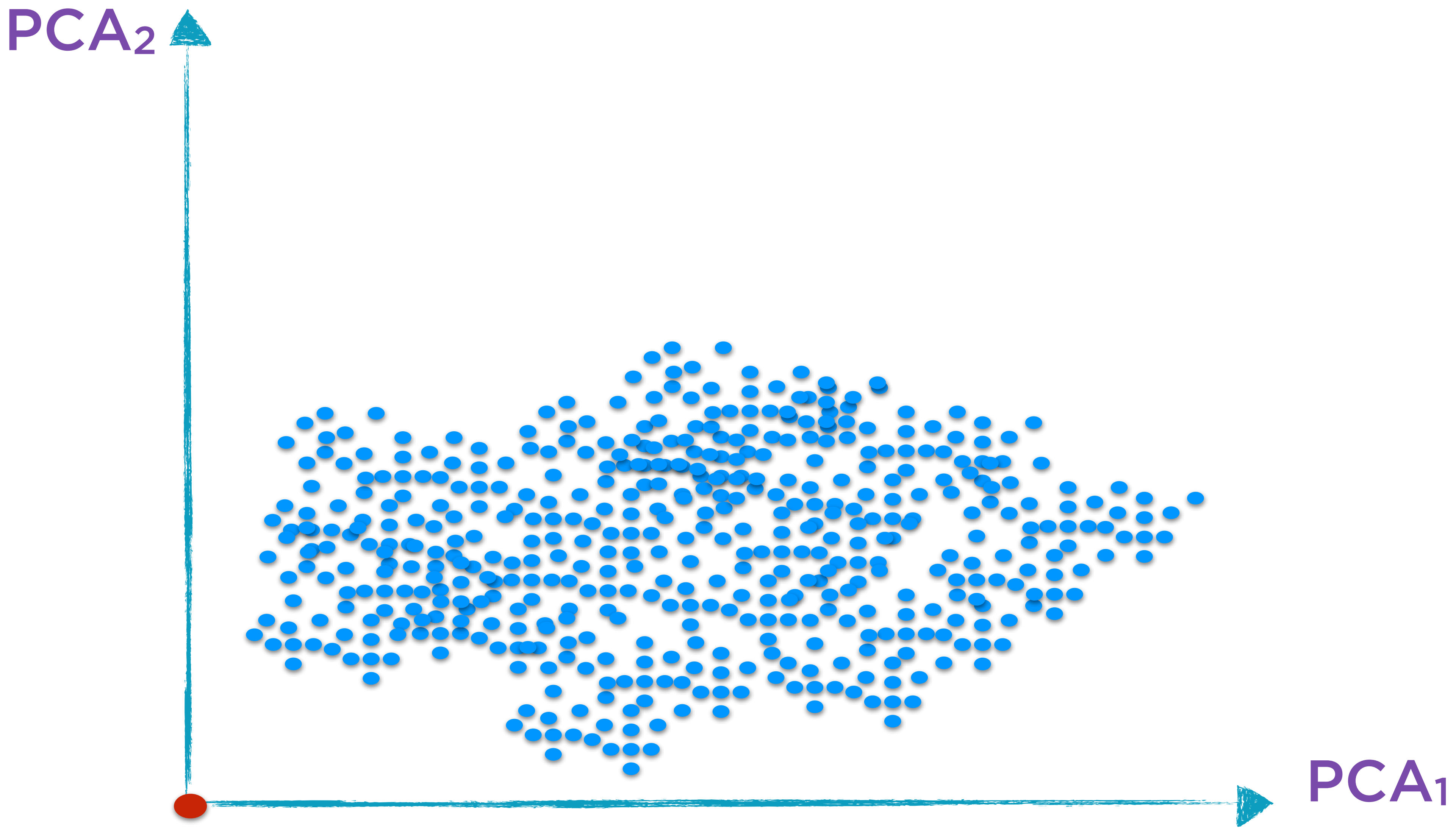
The variances are clearly smaller along this **second principal component** than along the first

Intuition Behind PCA



In general, there are as many principal components as there are dimensions in the original data

Intuition Behind PCA

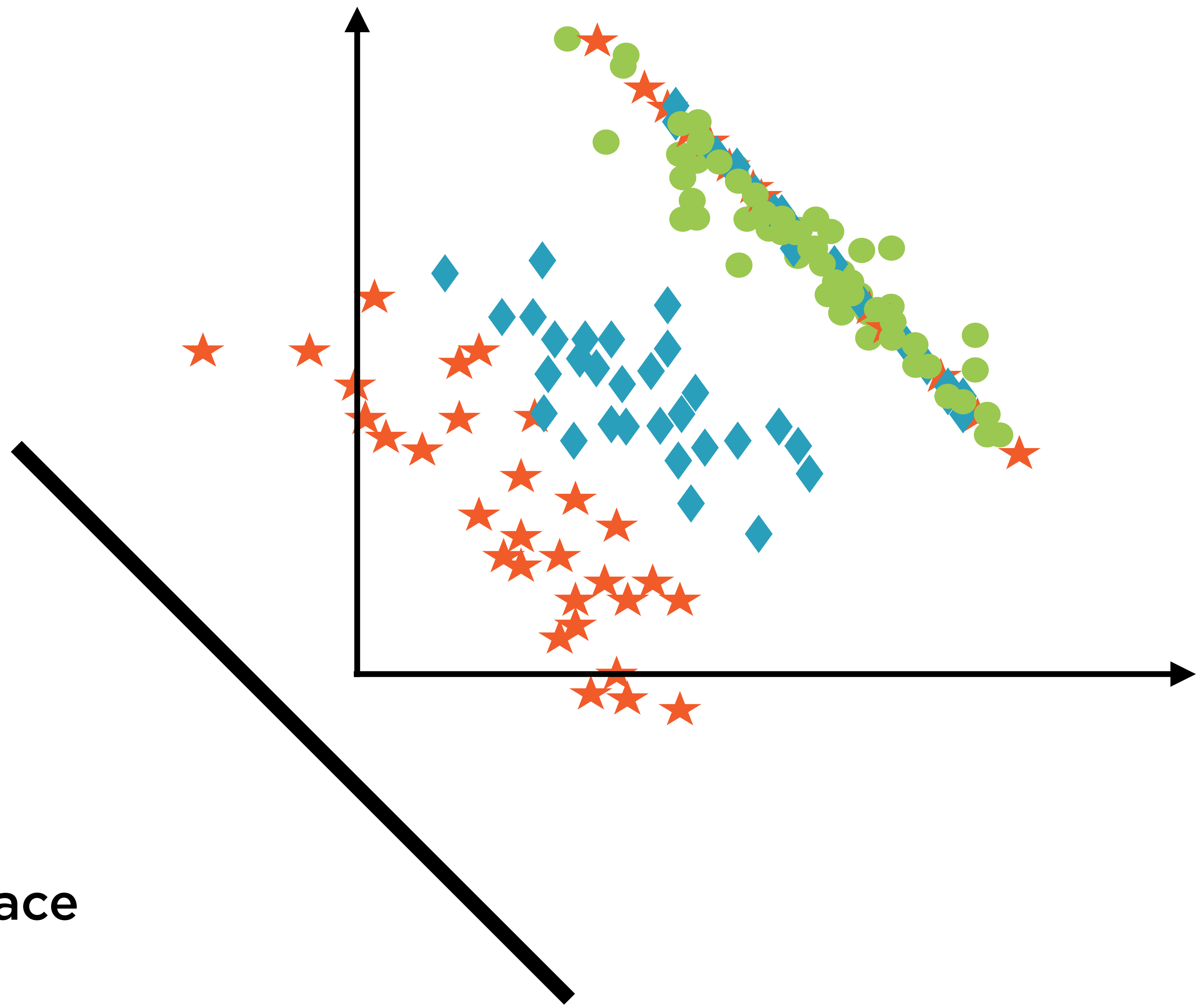


Re-orient the data along these new axes

Linear Discriminant Analysis (LDA) is similar to PCA - it uses the same underlying idea of projecting points onto different axes

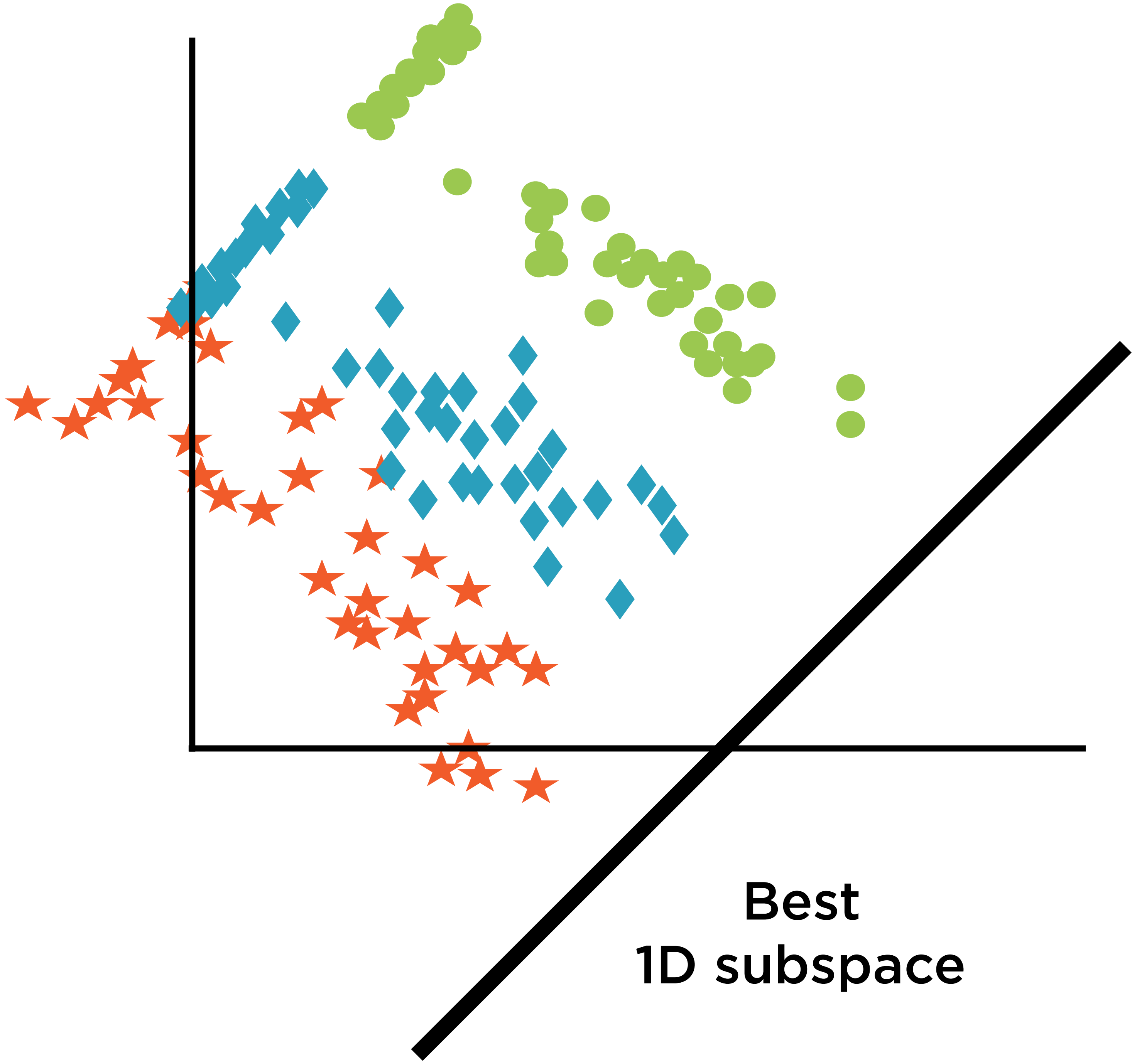
LDA chooses axis to maximize distance between points of different categories

Choosing Axes for Ternary Classification

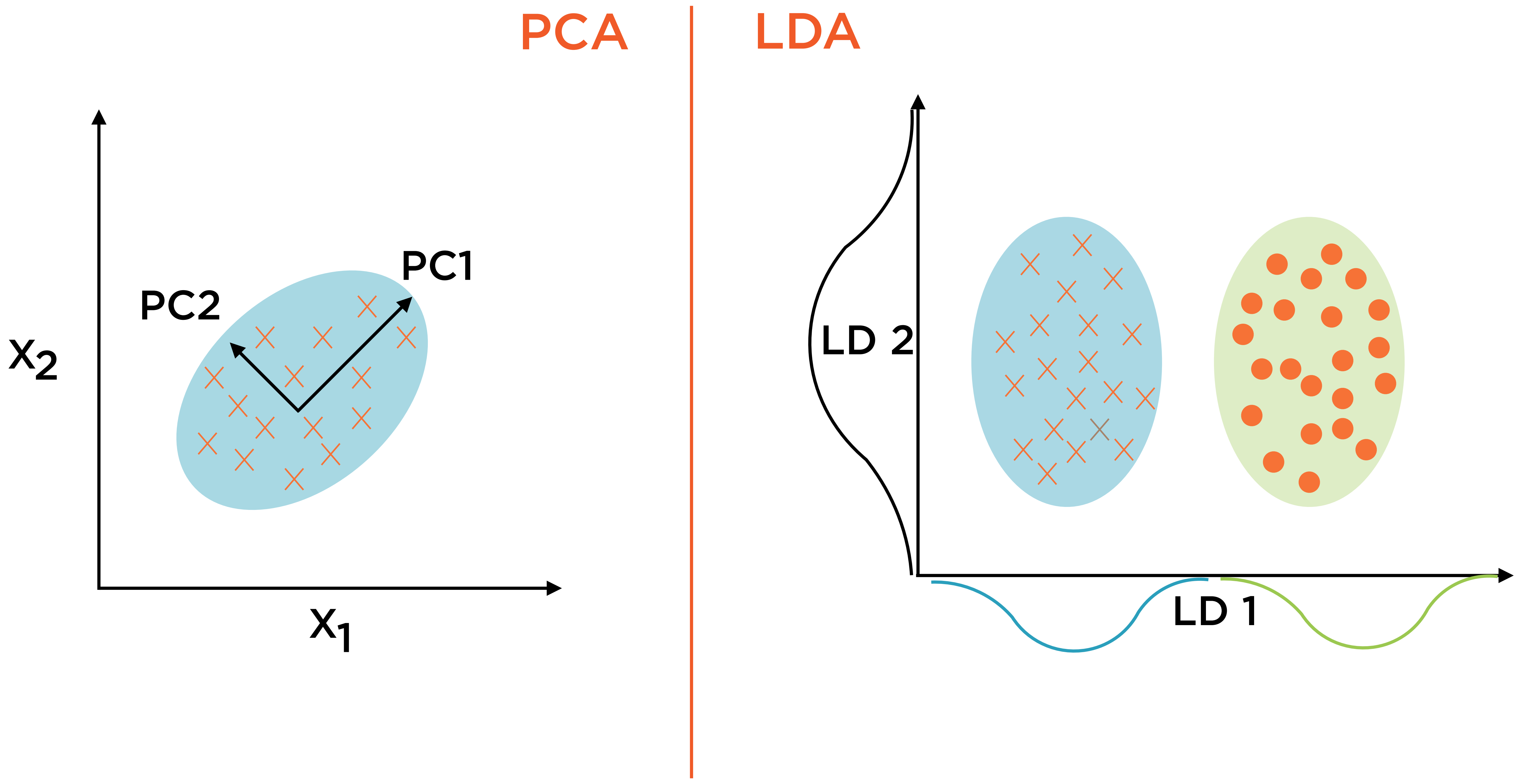


Worst
1D subspace

Choosing Axes for Ternary Classification



PCA vs. LDA



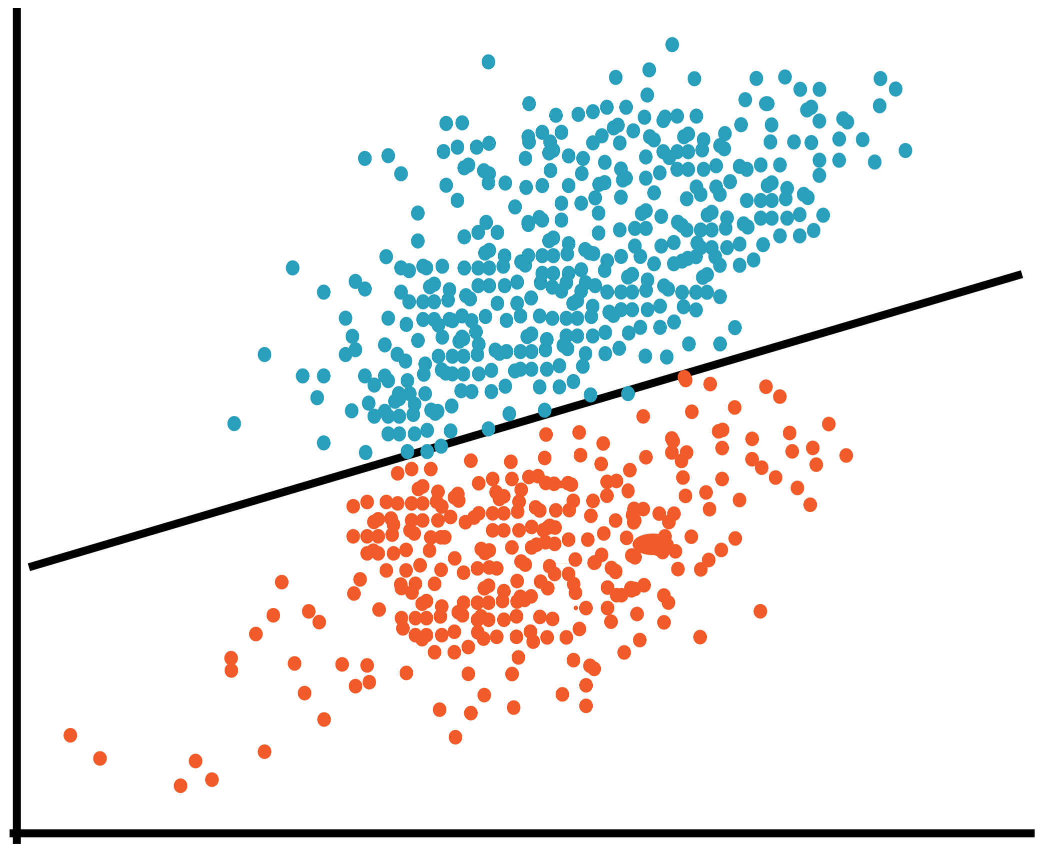
Quadratic Discriminant Analysis

Variant of LDA that is better suited to cases where X-variables corresponding to different y-labels have different covariances

Covariance

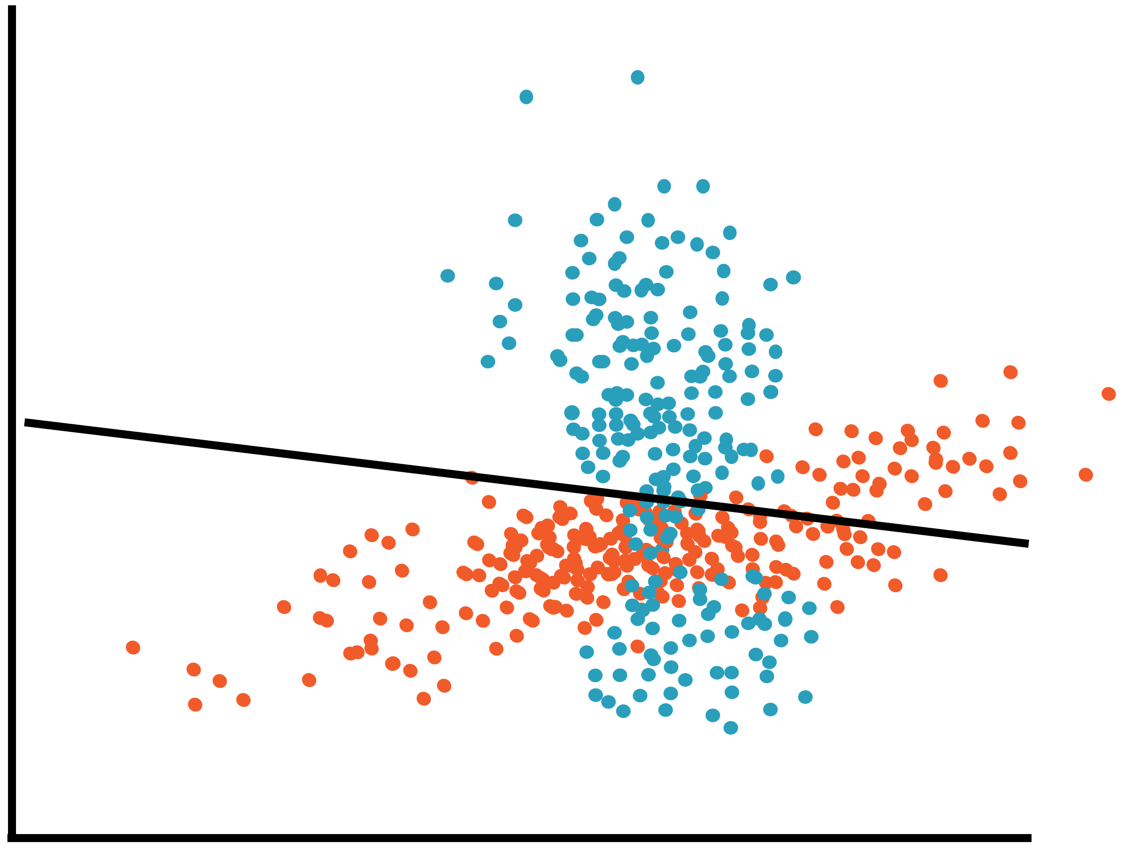
Measures relationship between two variables, specifically whether greater values of one variable correspond to greater values in the other.

QDA vs. LDA

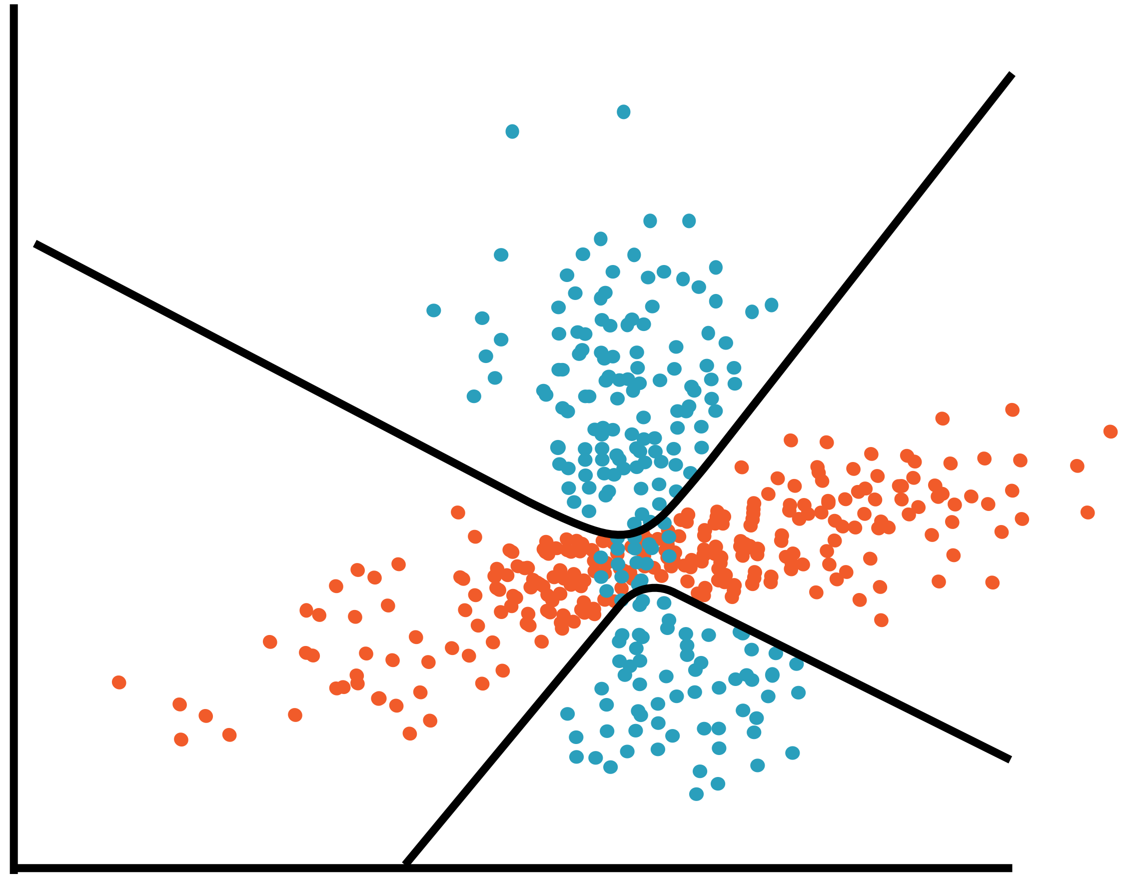


LDA works fine when X-variables share uniform covariances (independent of Y)

QDA vs. LDA



LDA fails when covariances of X are a function of value of Y



QDA correctly separates such points

Demo

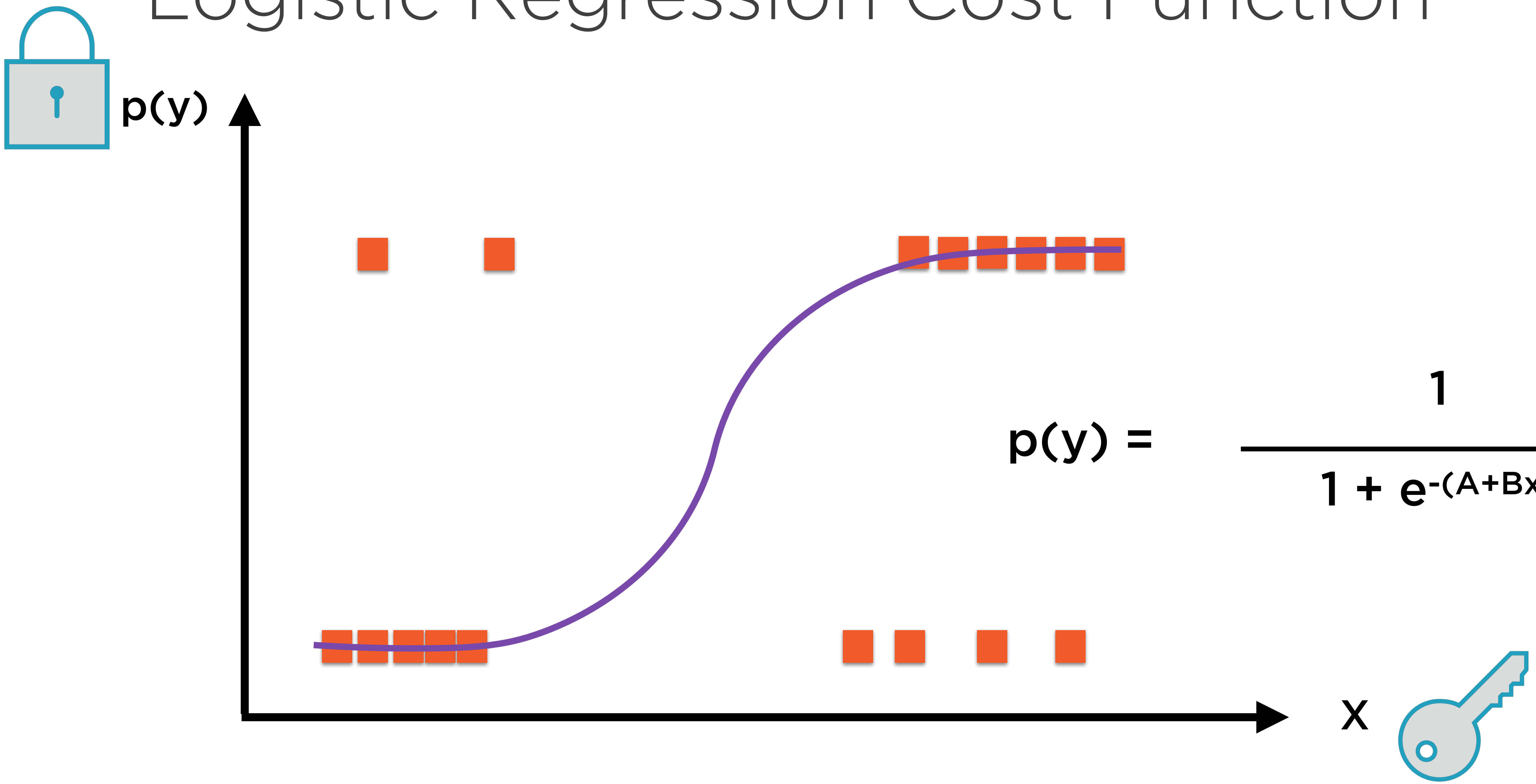
Linear Discriminant Analysis for classification

Demo

Quadratic Discriminant Analysis for classification

SGD Classifiers

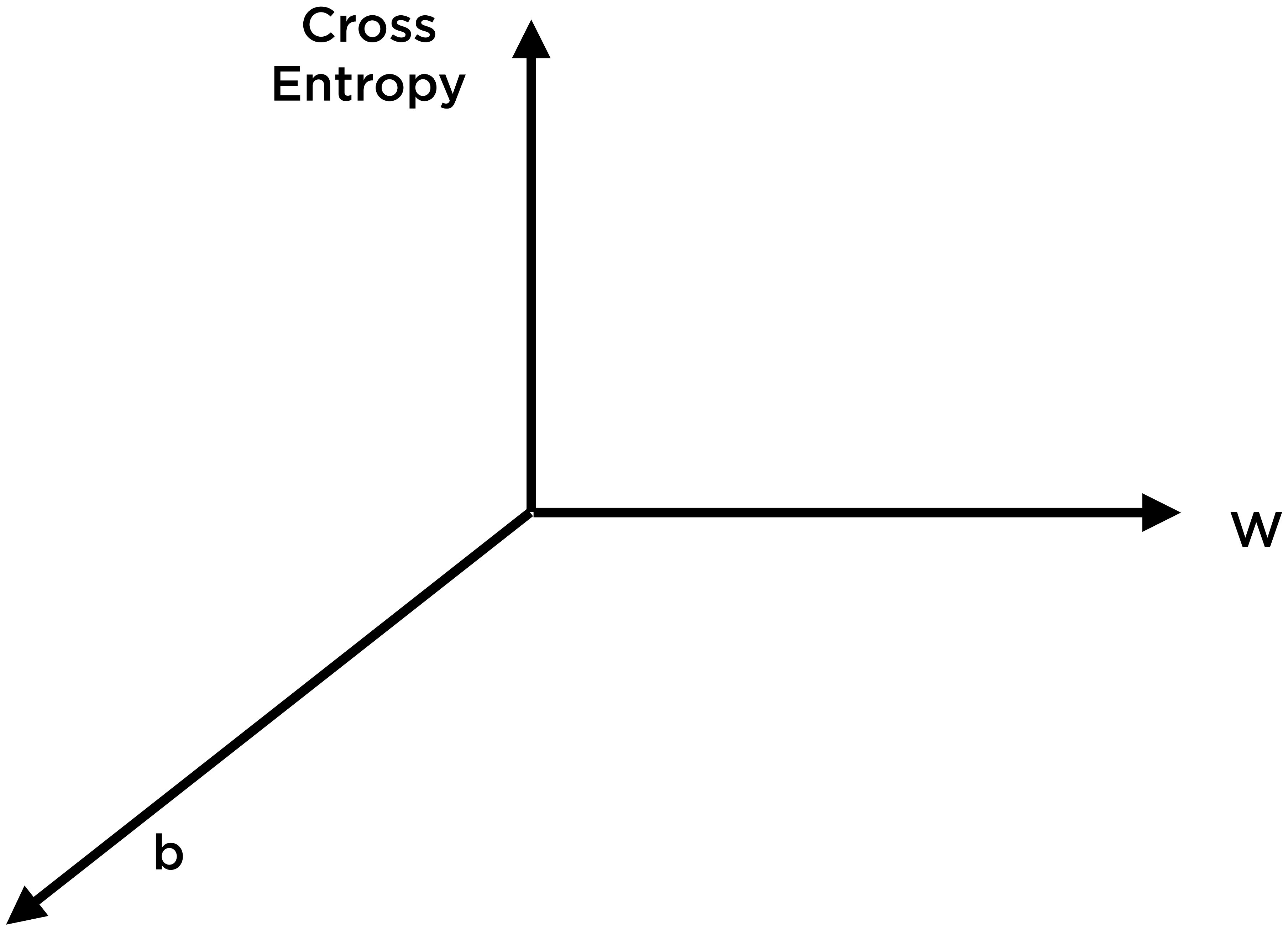
Logistic Regression Cost Function



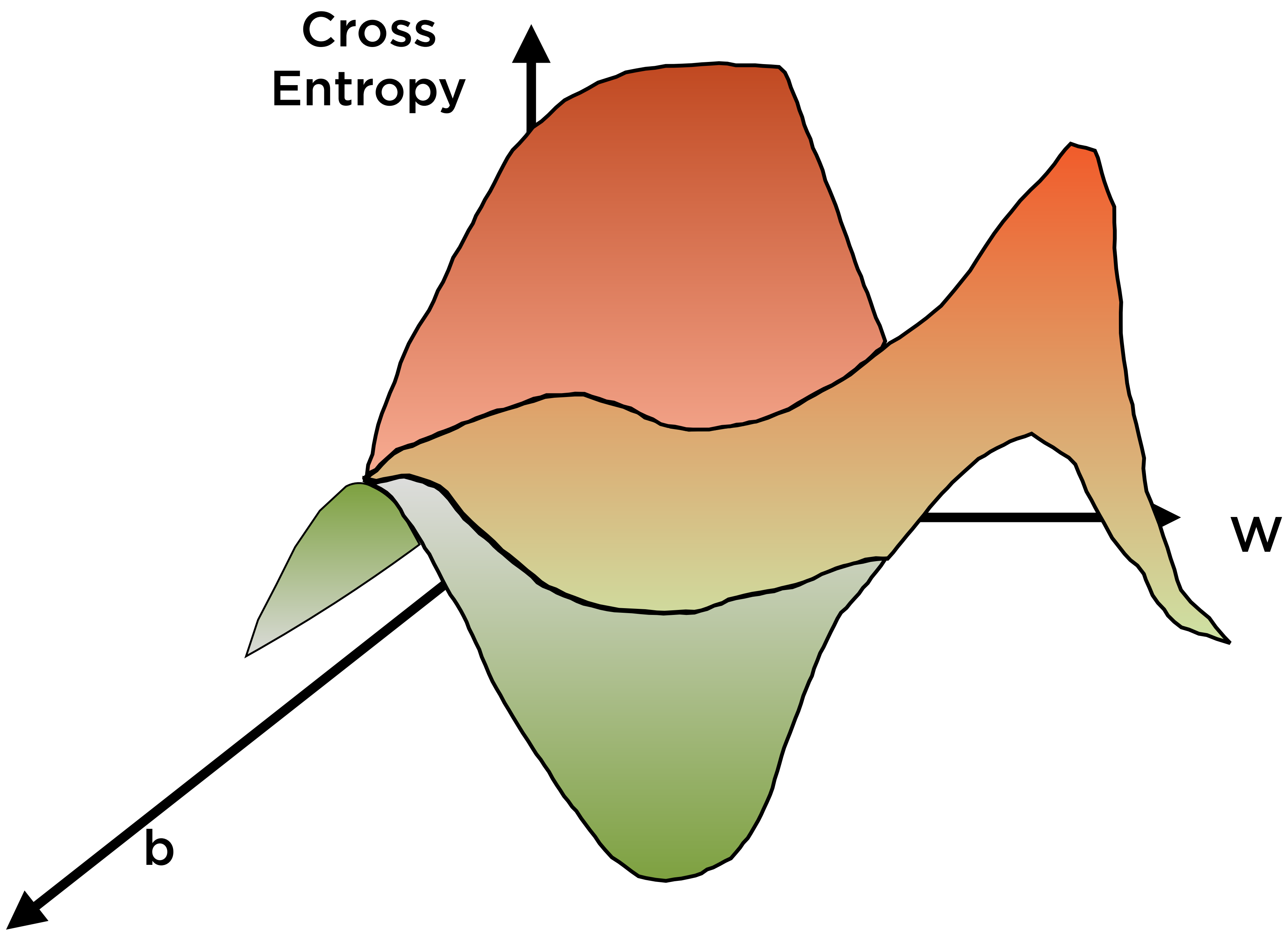
$$p(y) = \frac{1}{1 + e^{-(A+Bx)}}$$

Cross entropy measures how well the estimated probabilities match actual labels

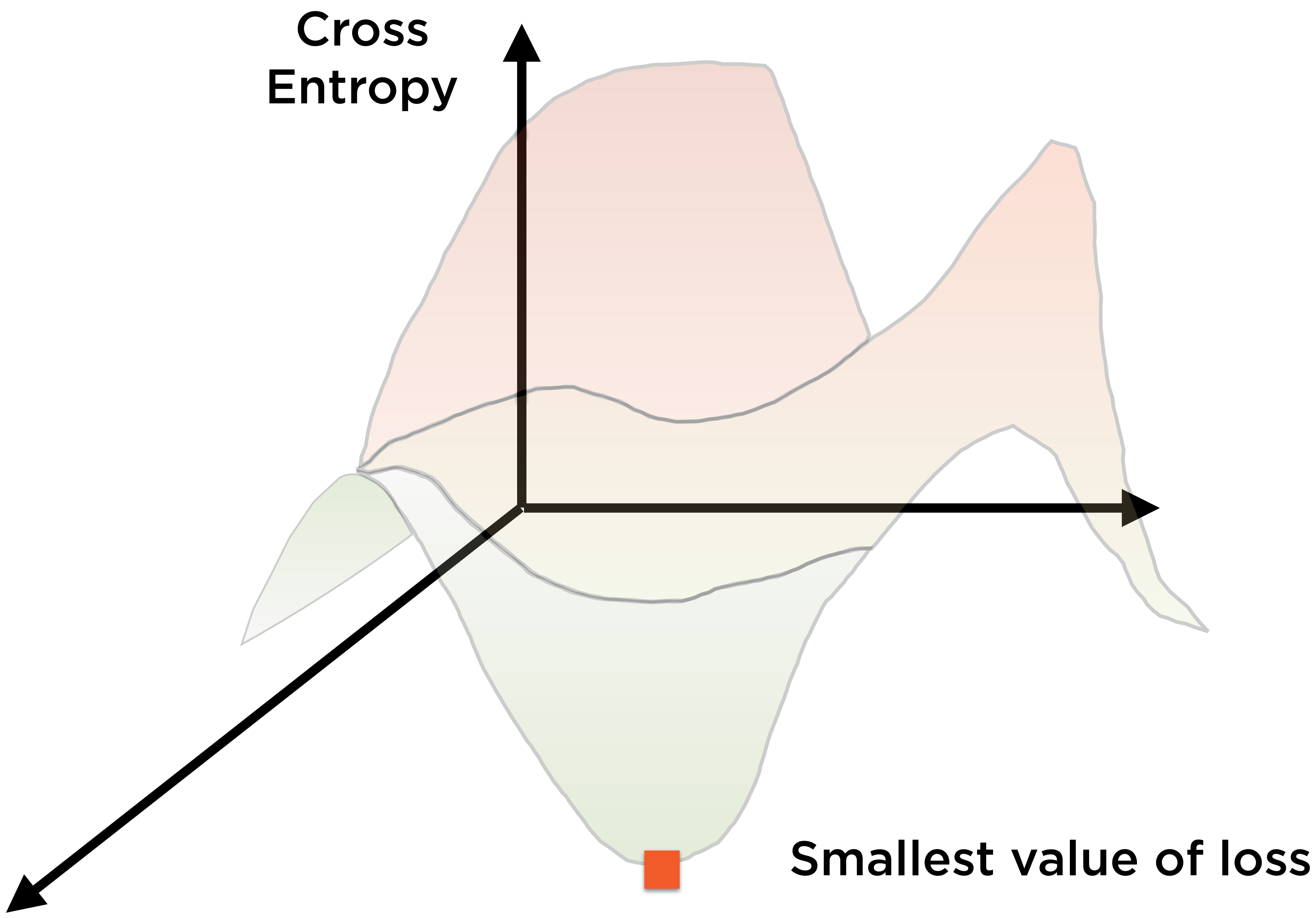
Minimizing Cross Entropy



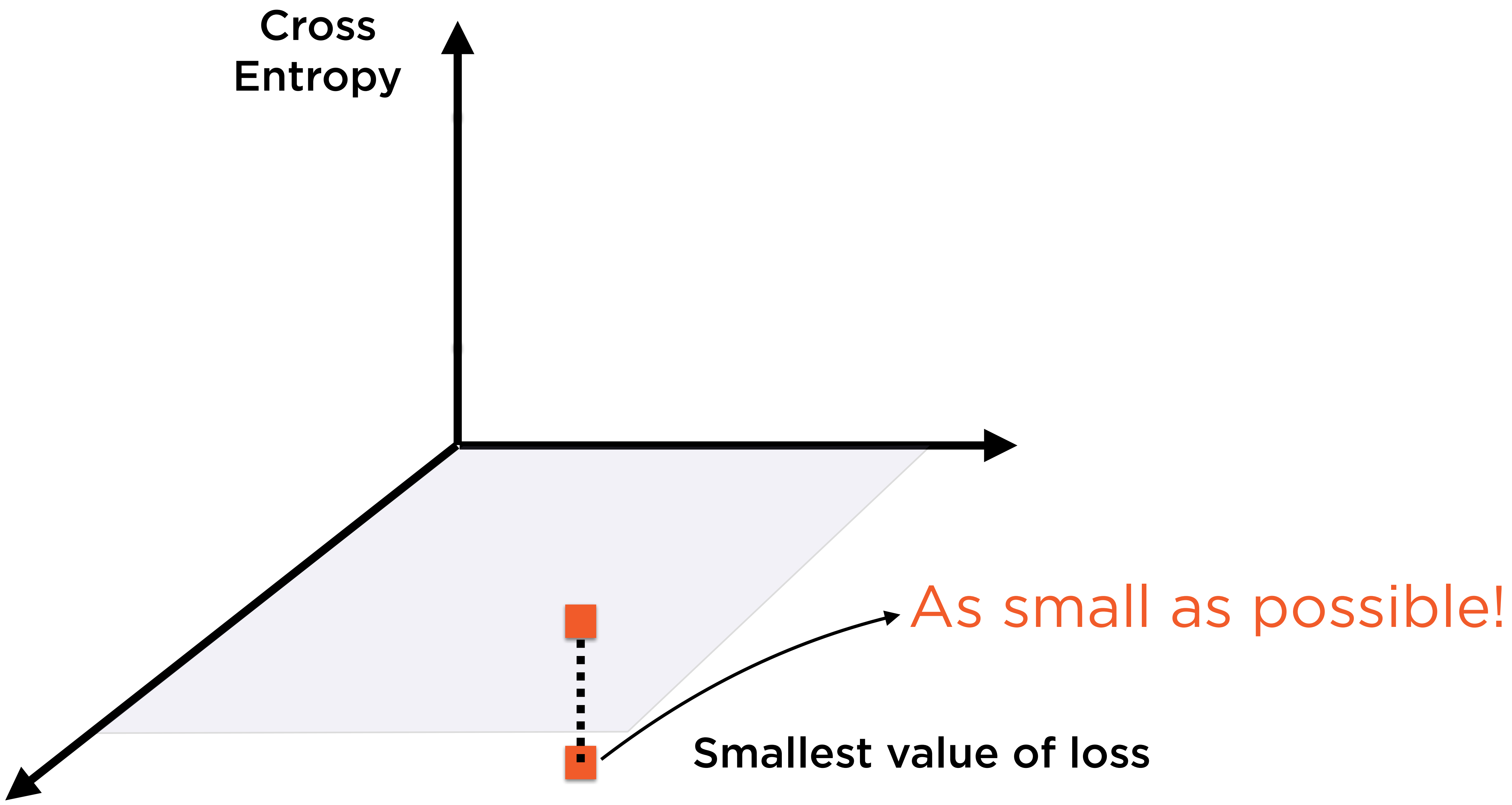
Minimizing Cross Entropy



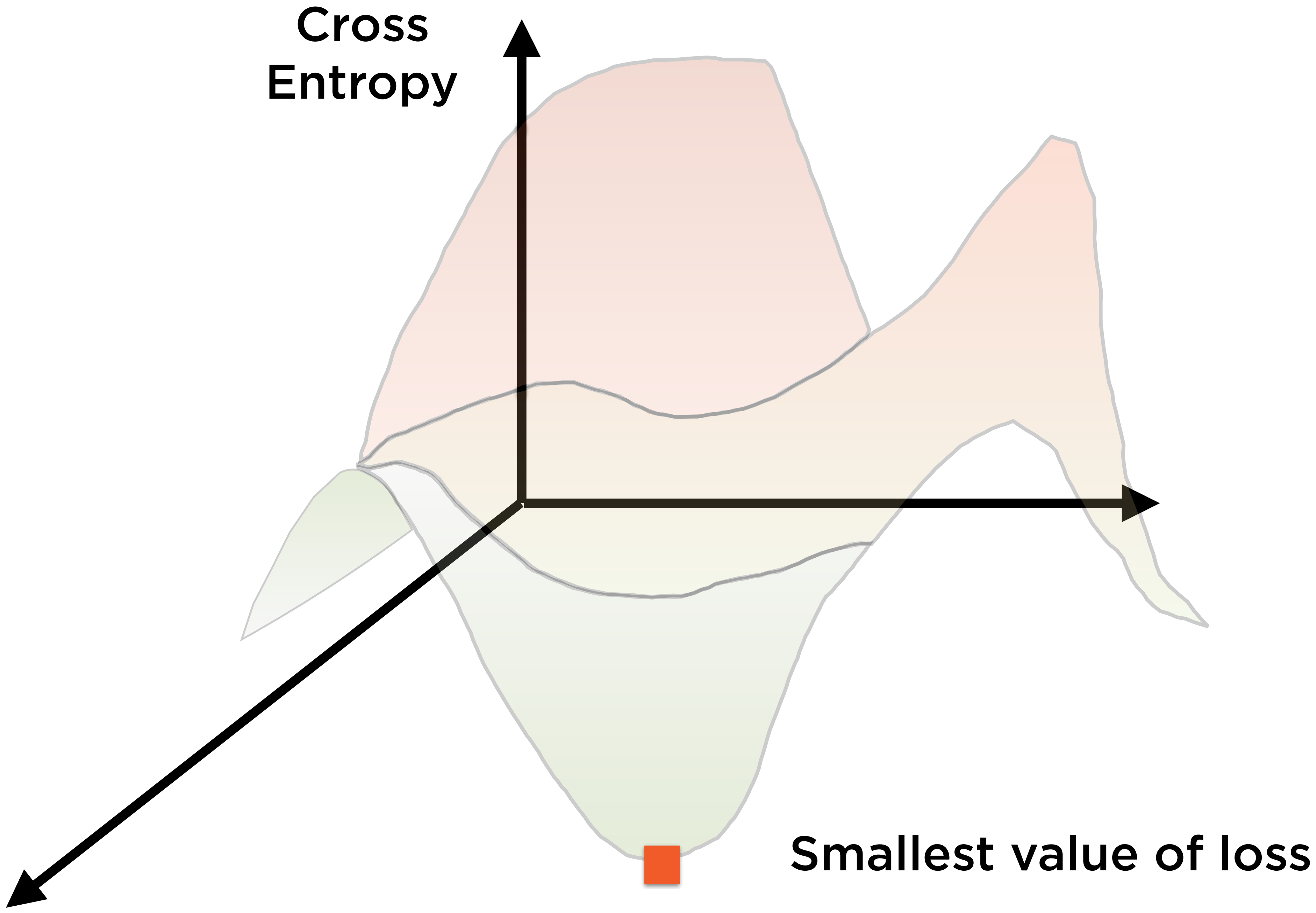
Minimizing Cross Entropy



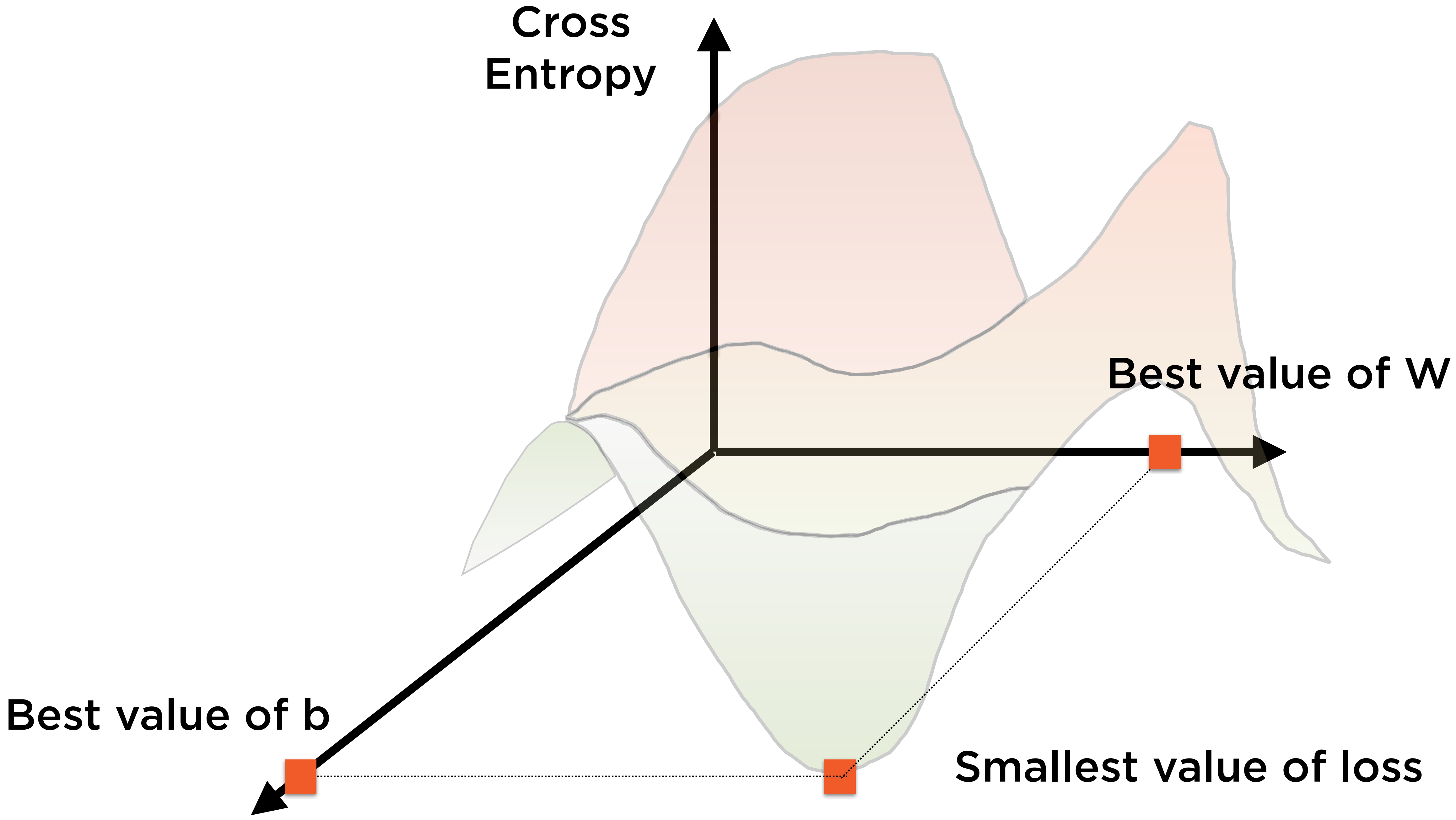
Minimizing Cross Entropy



Minimizing Cross Entropy

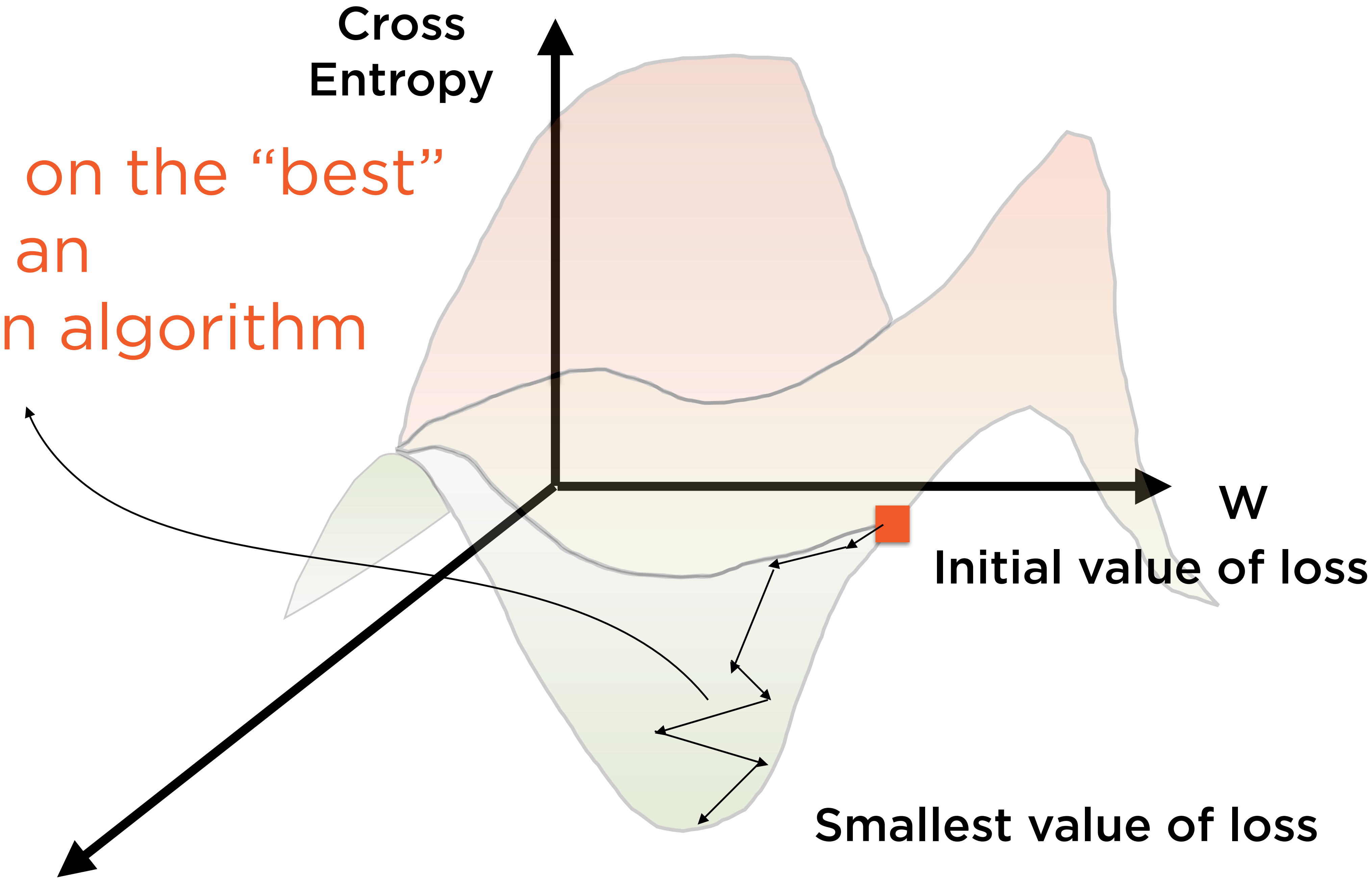


Minimizing Cross Entropy

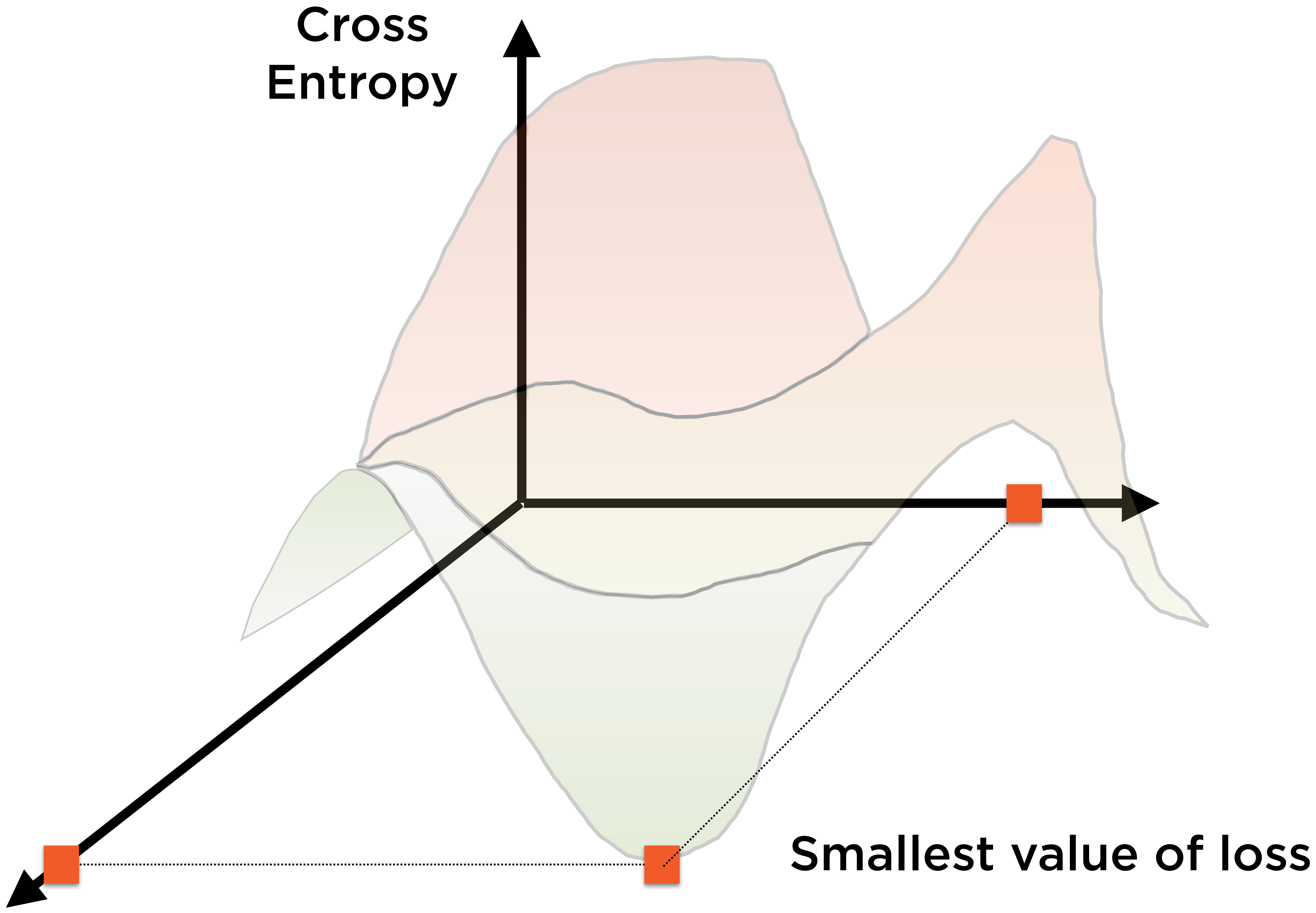


"Gradient Descent"

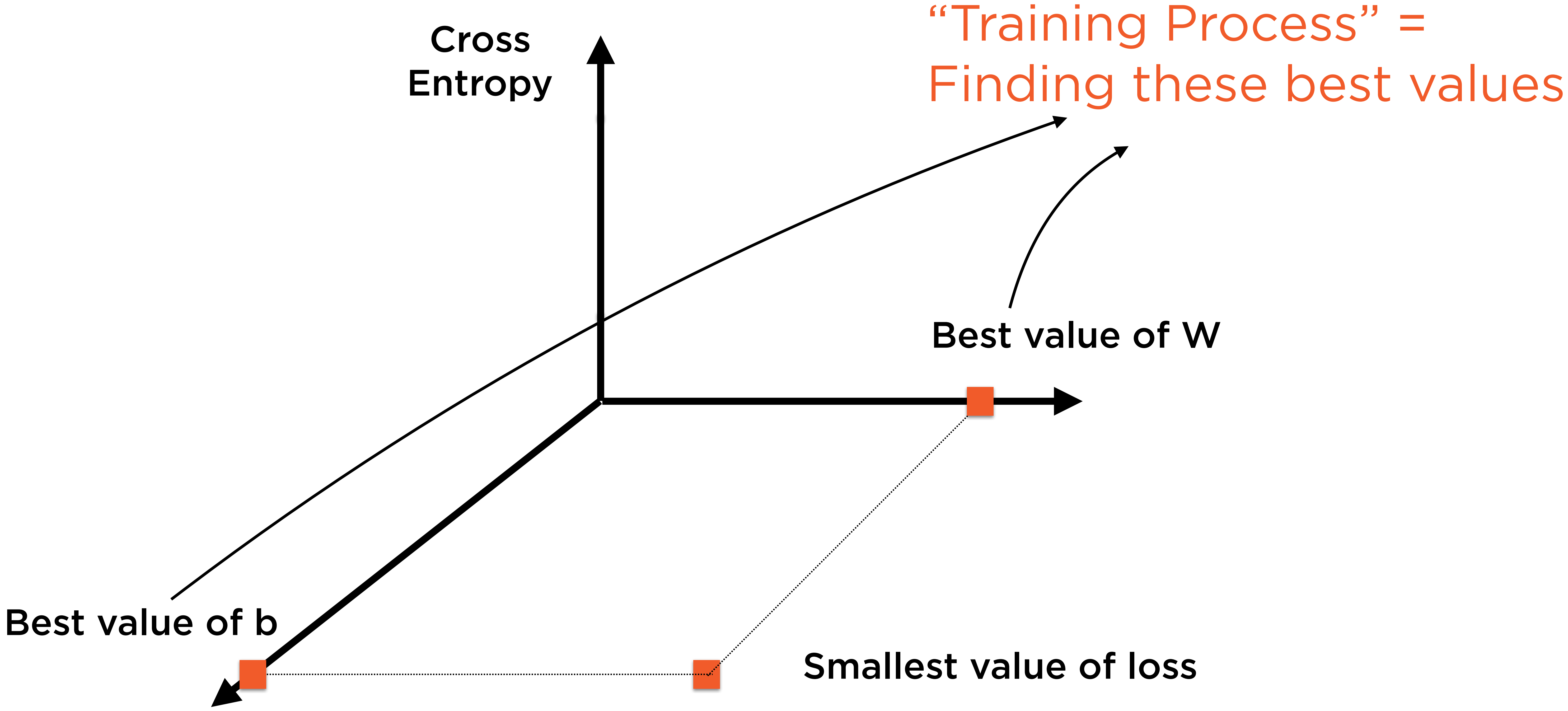
Converging on the "best" value using an optimization algorithm



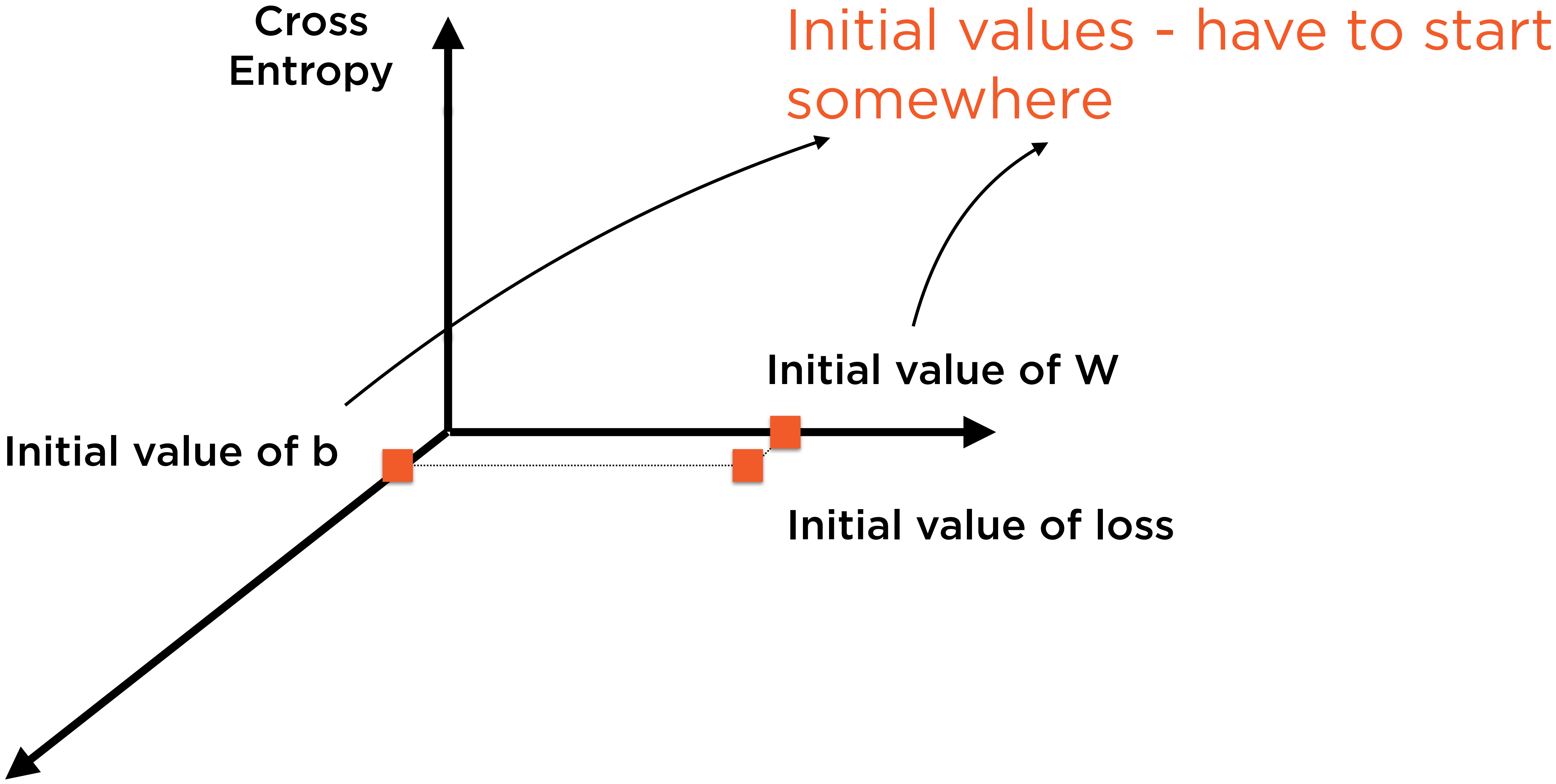
Minimizing Cross Entropy



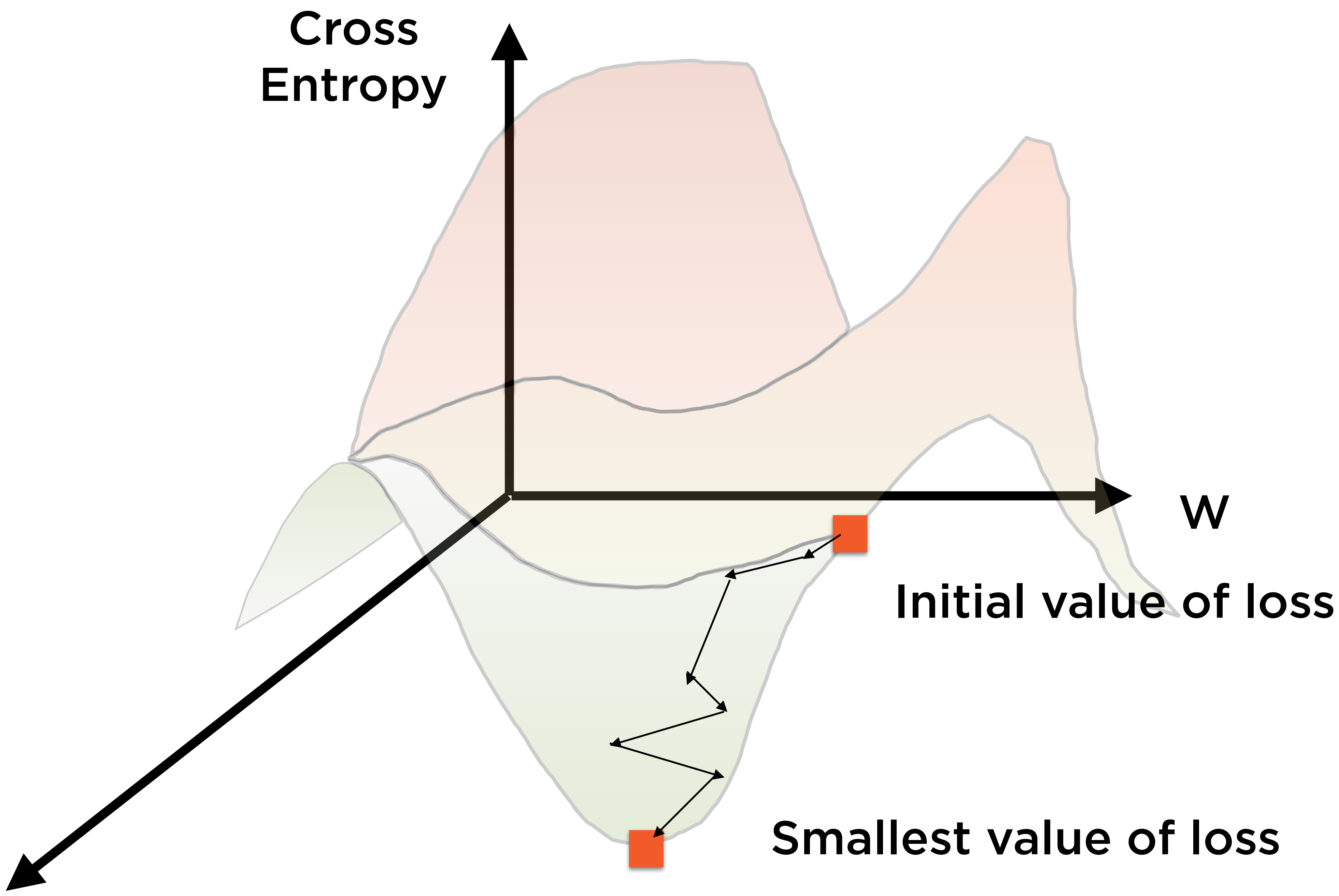
"Training" the Algorithm



Start Somewhere



"Gradient Descent"



Stochastic Gradient Descent **iteratively** converges to the best model

Demo

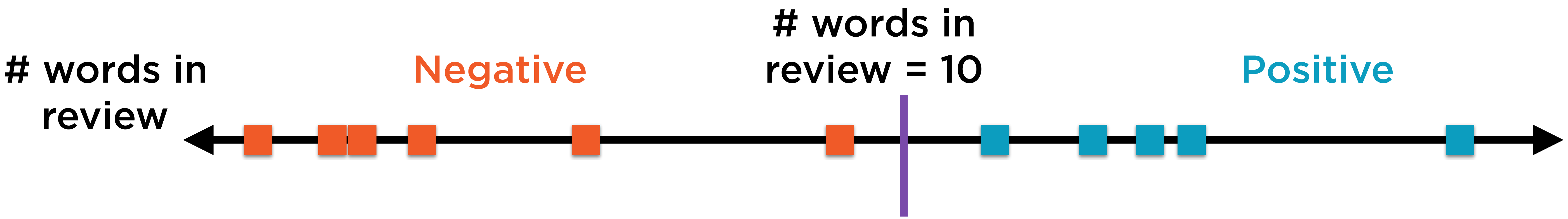
Stochastic Gradient Descent classifier

Support Vector Classifiers (SVC)

Data in One Dimension

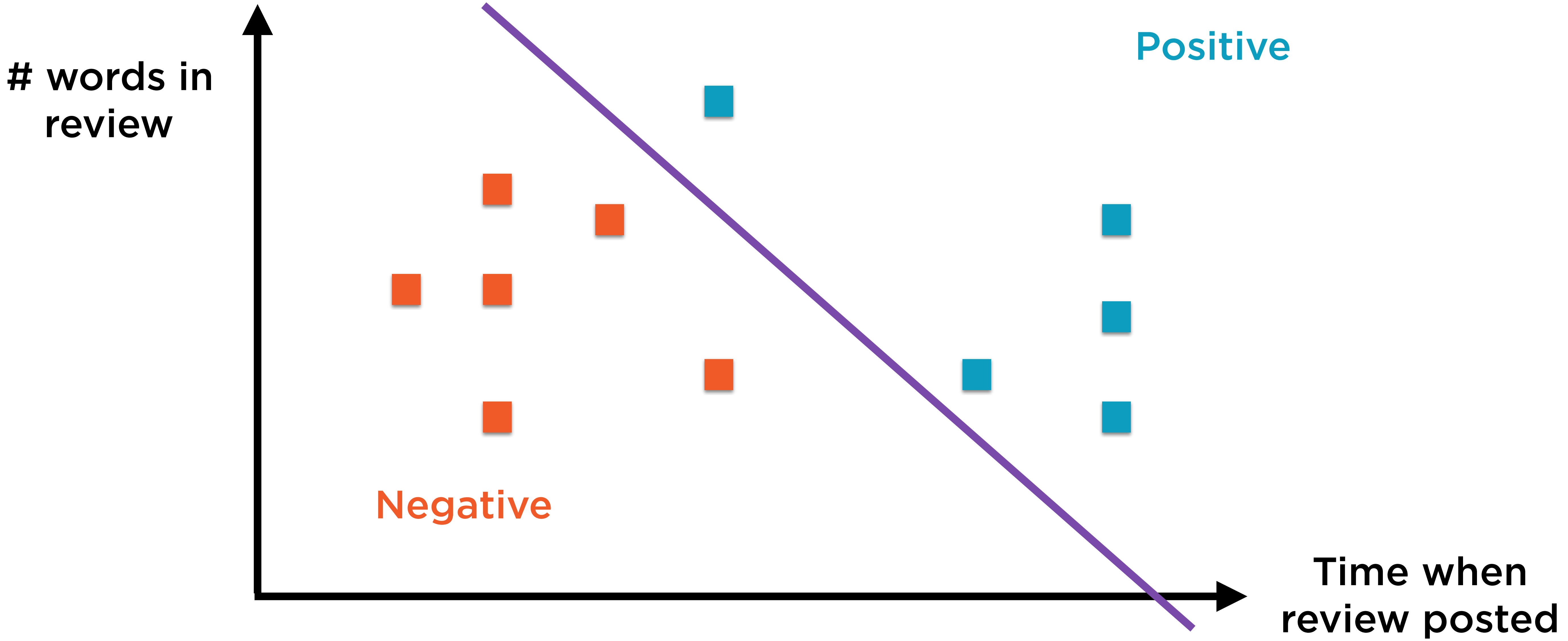
Unidimensional data points can be represented using a line, such as a number line

Data in One Dimension



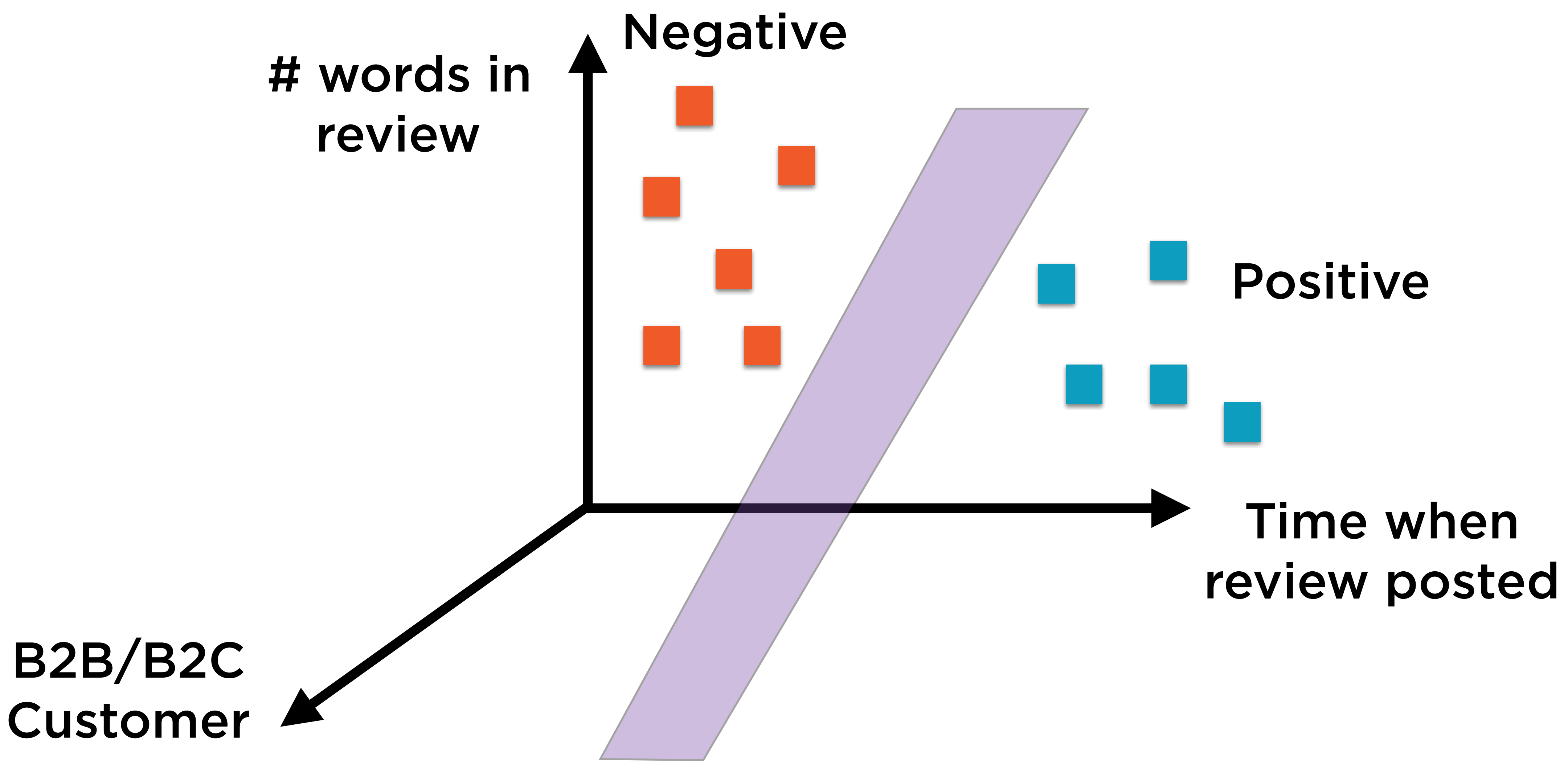
Unidimensional can also be separated, or classified, using a point

Data in Two Dimensions



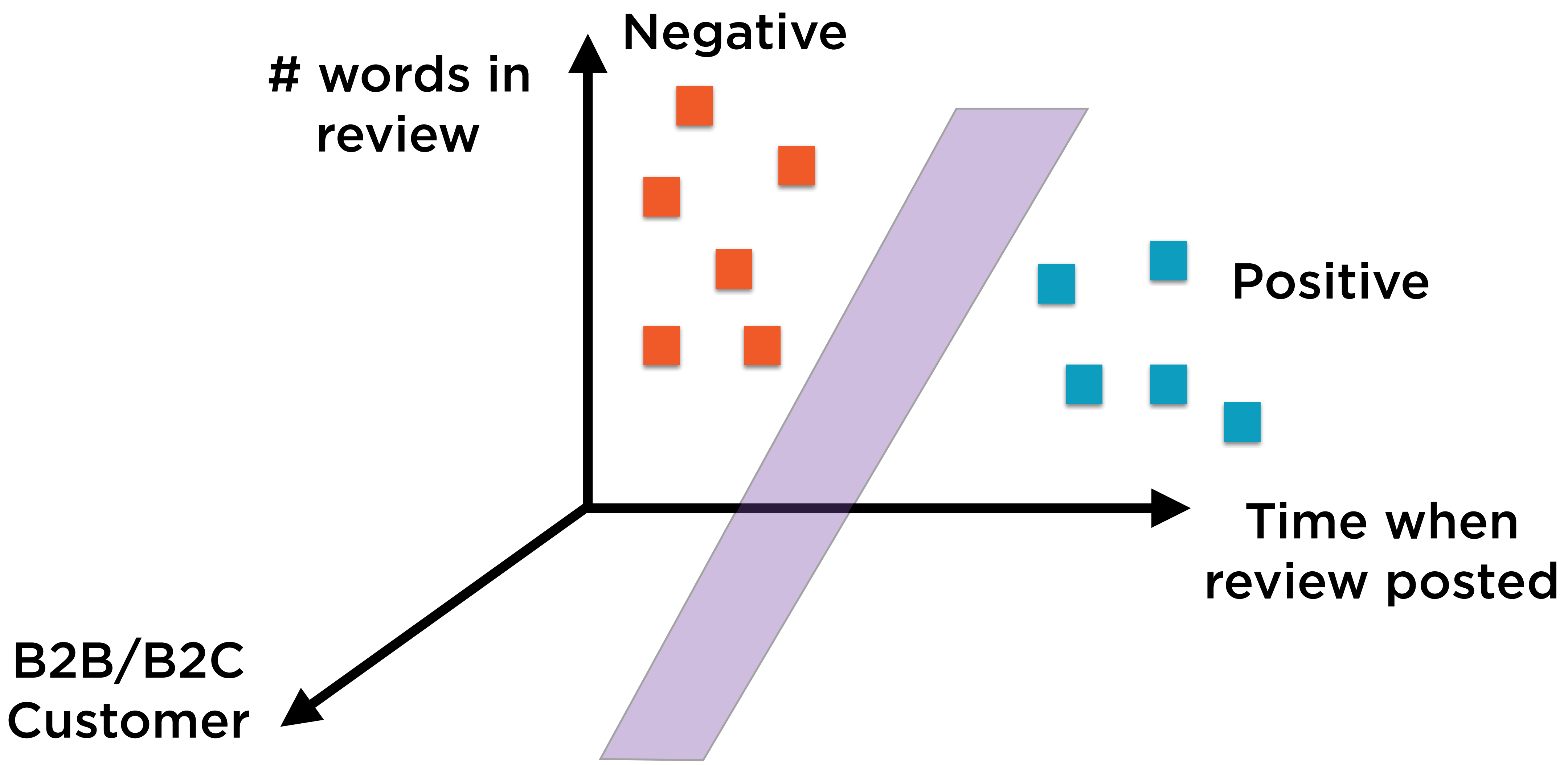
Bidimensional data points can be represented using a plane, and classified using a line

Support Vector Machines



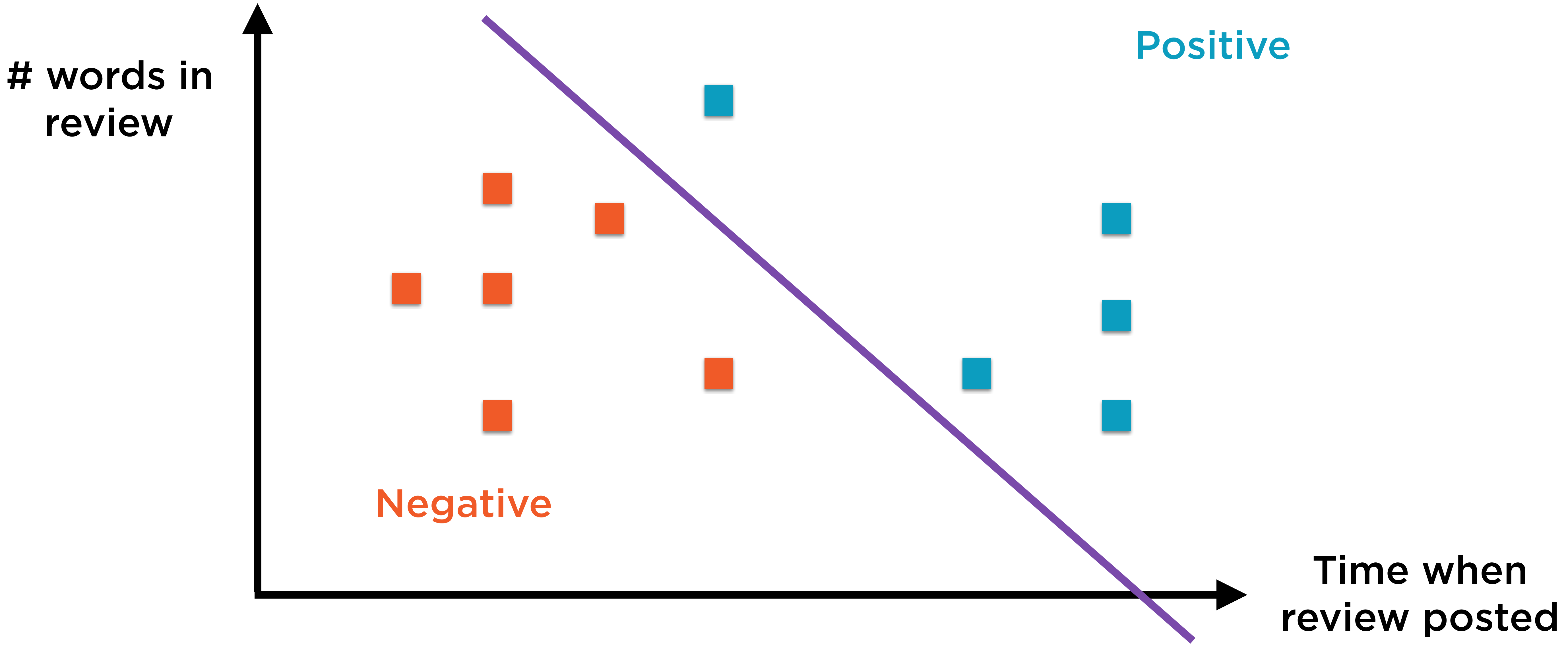
SVM classifiers find the hyperplane that best separates points in a hypercube

Data in N Dimensions



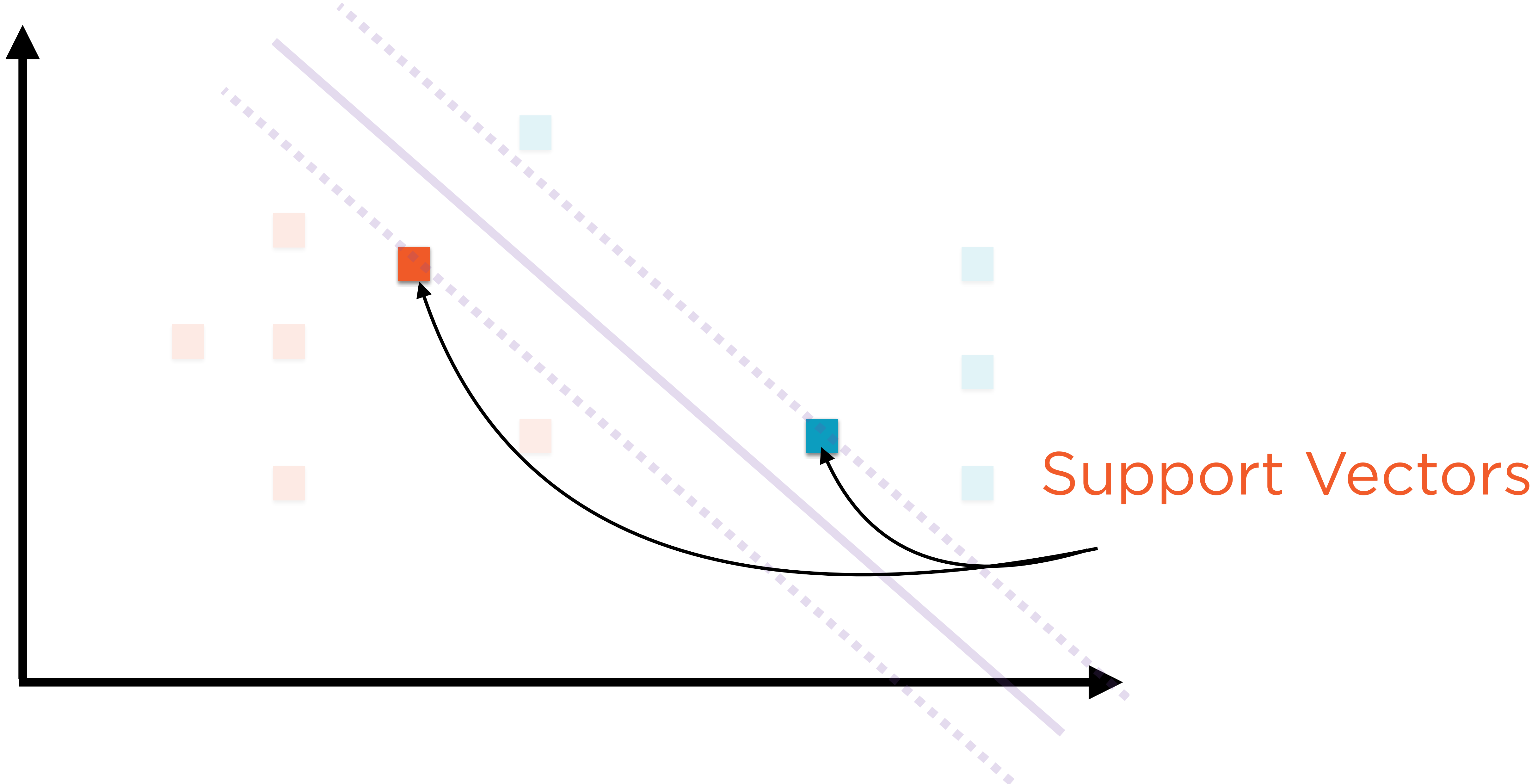
N-dimensional data can be represented in a hypercube, and classified using a hyperplane

Hard Margin Classification



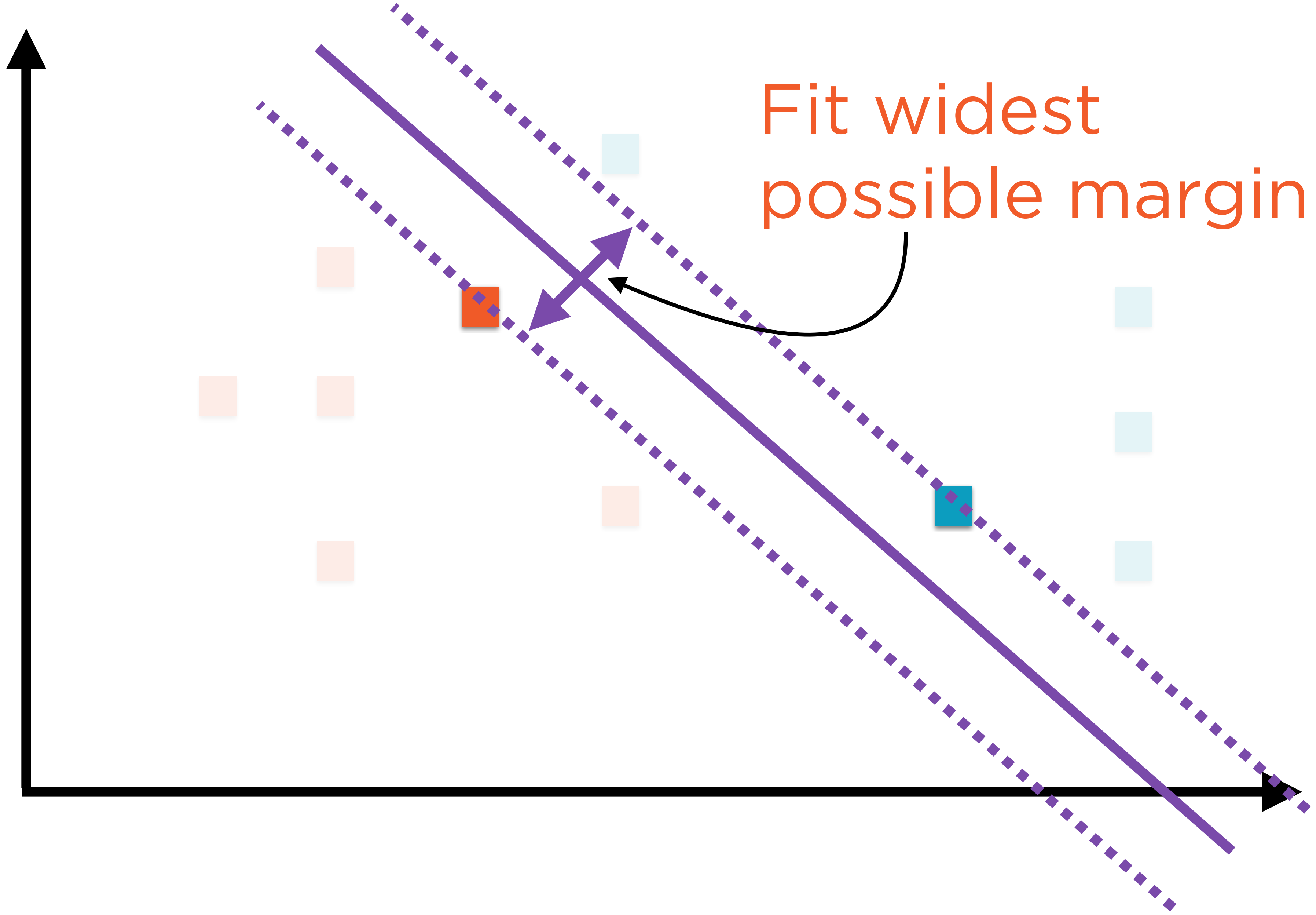
Ideally, data is linearly separable - hard decision boundary

Hard Margin Classification



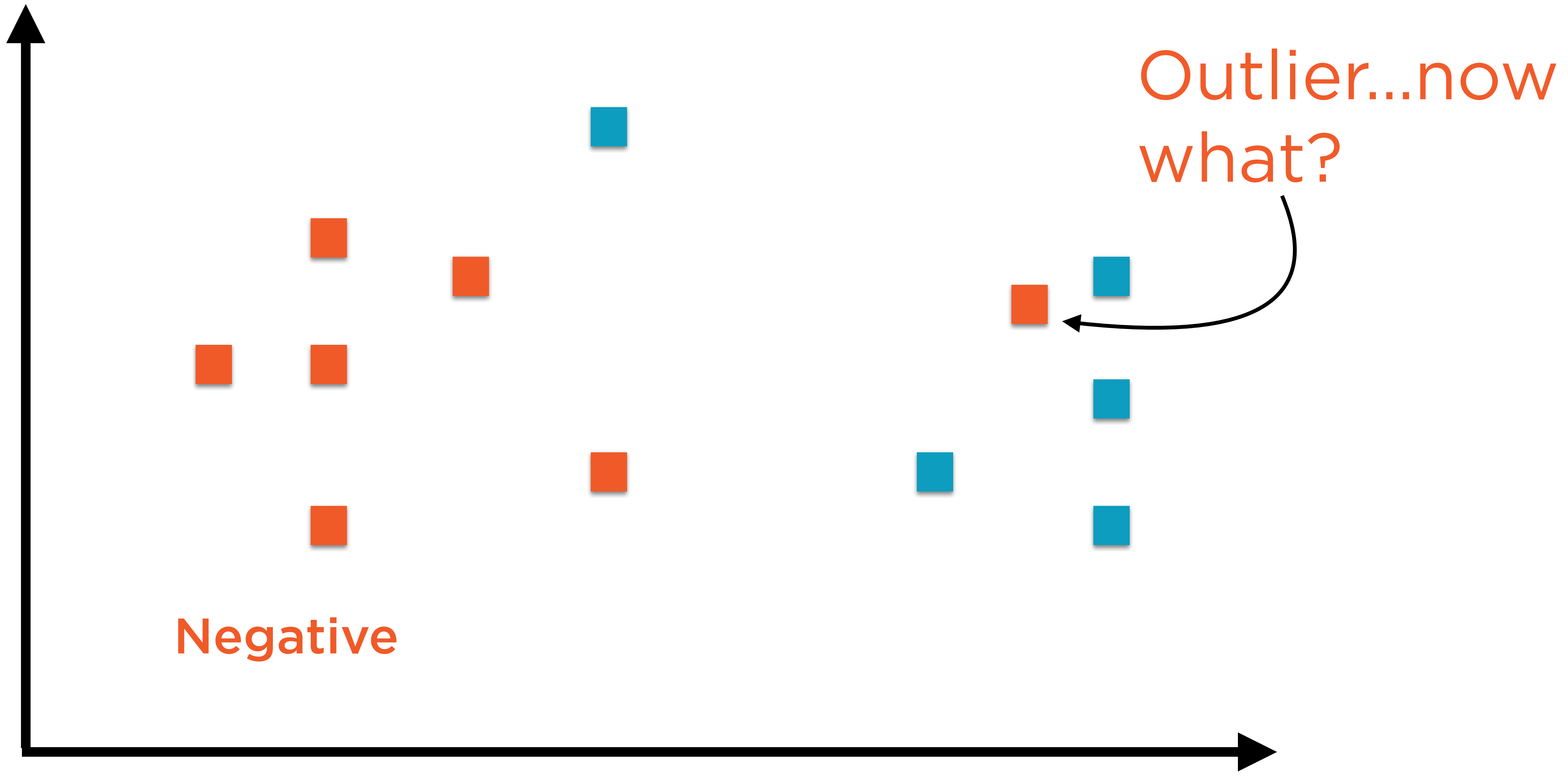
The nearest instances on either side of the boundary are called the support vectors

Hard Margin Classification



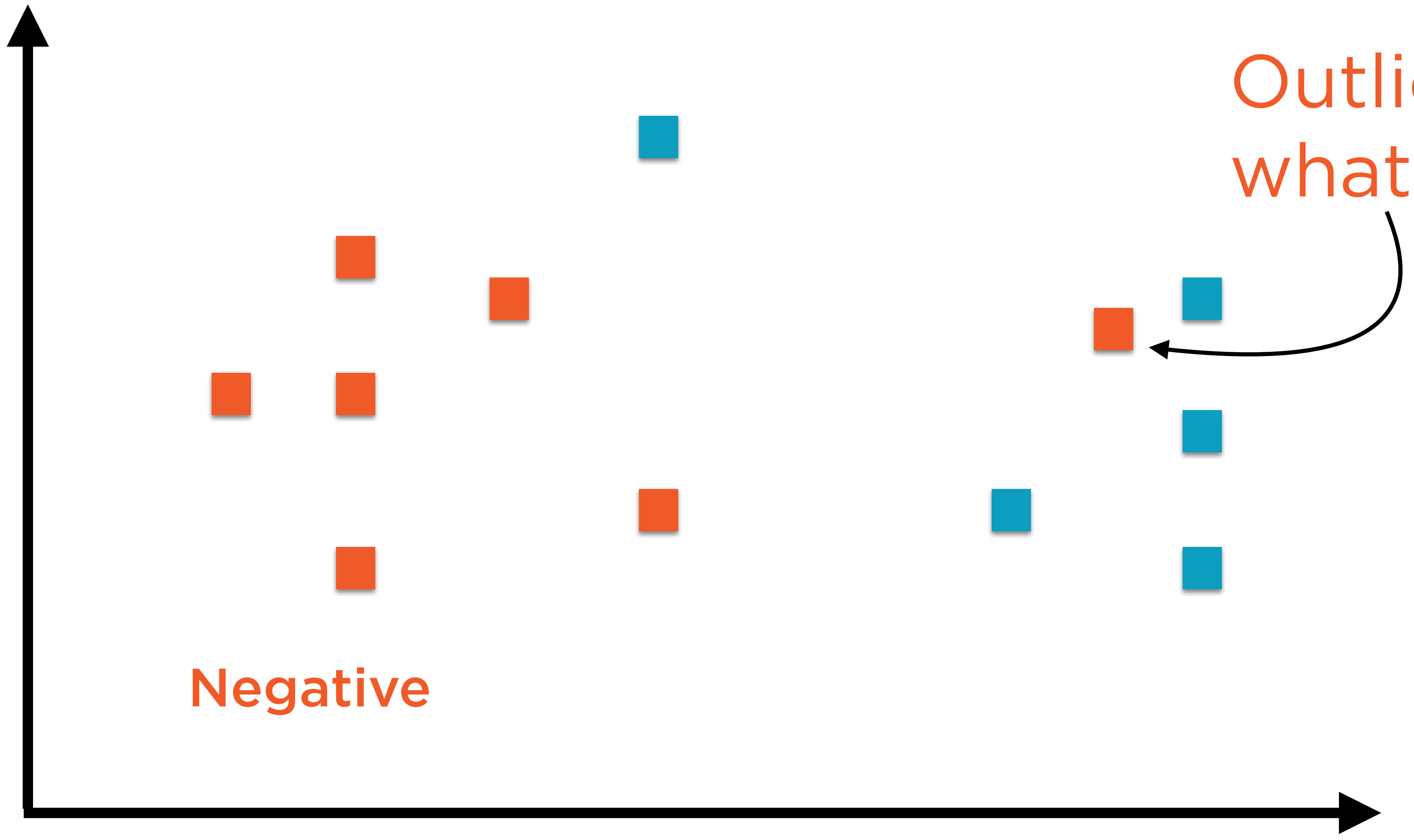
SVM finds the widest street between the nearest points on either side

Soft Margin Classification



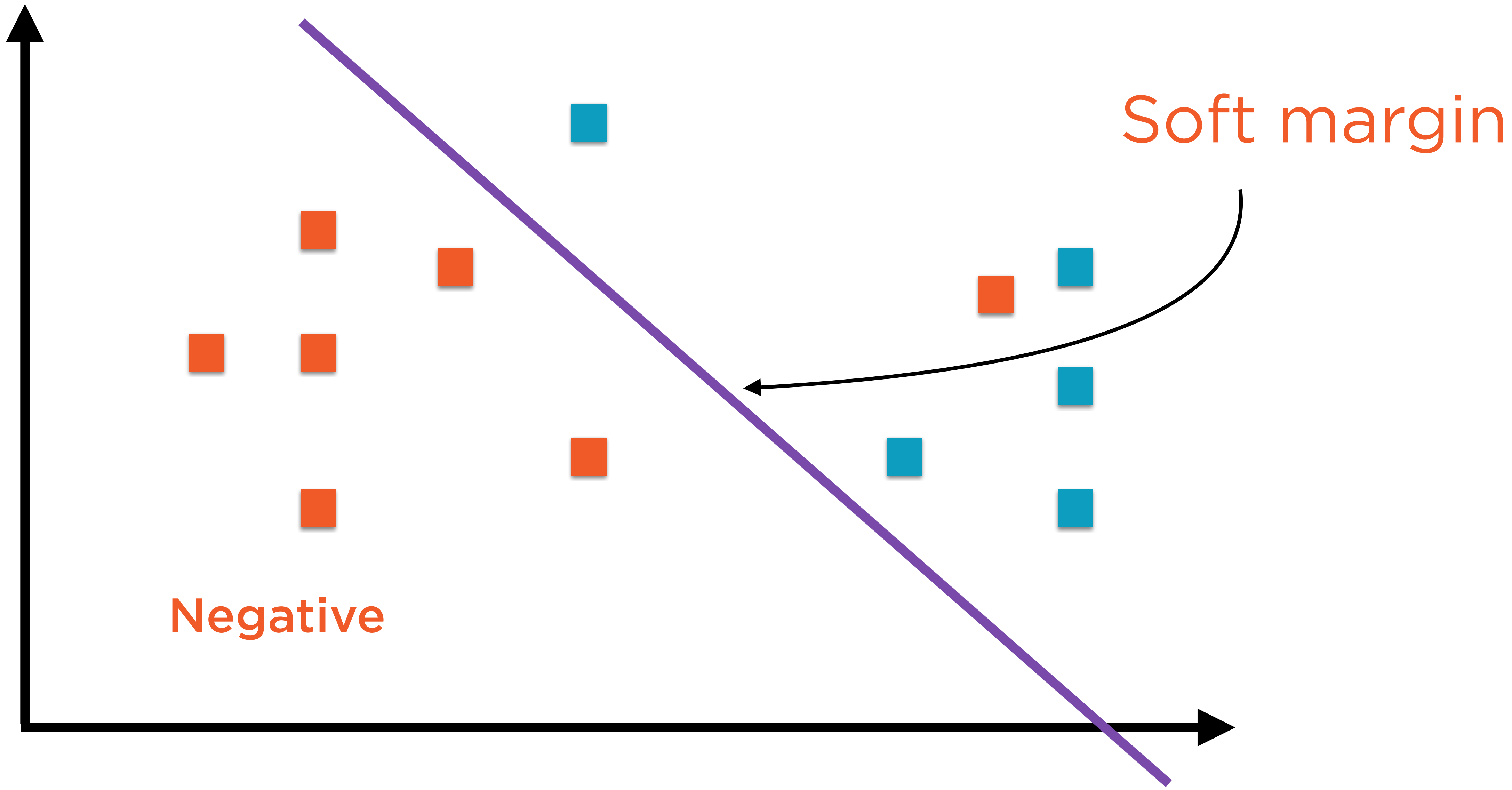
Hard margin classifiers are sensitive to outliers...

Soft Margin Classification



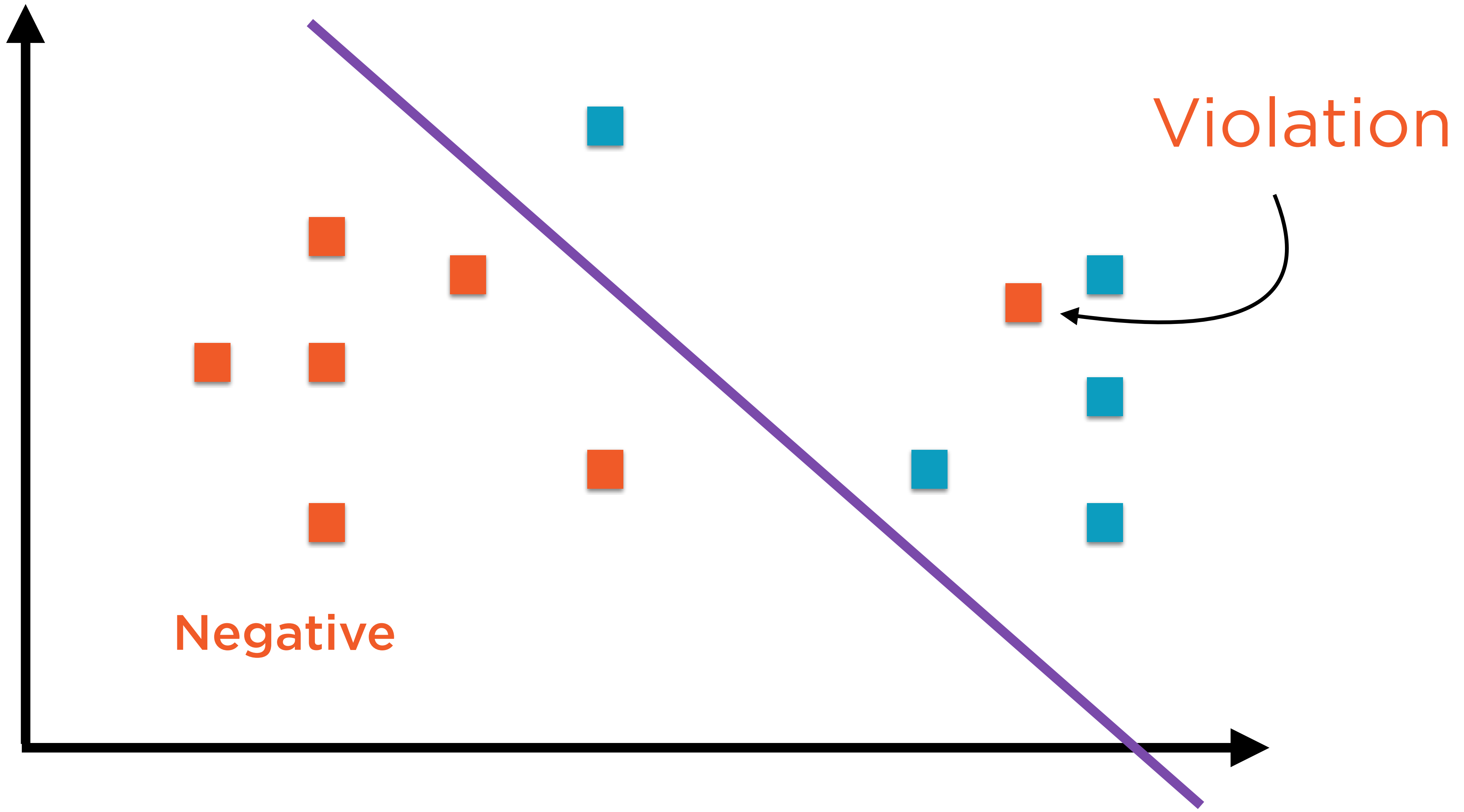
...and require perfectly linear separability in data

Soft Margin Classification



Soft margin classifiers allow some violations of the decision boundary

Soft Margin Classification



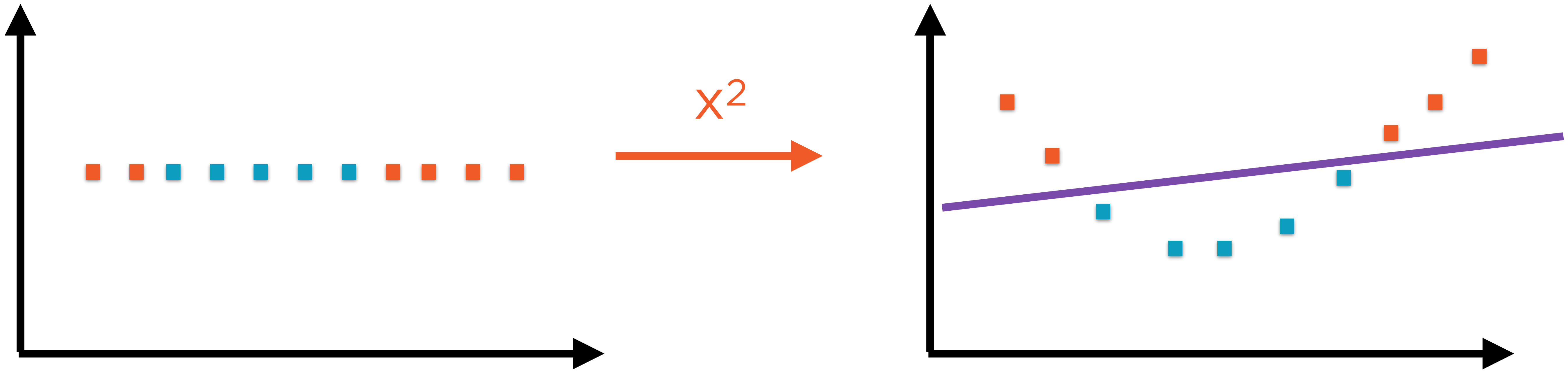
Soft margin classifiers allow some violations of the decision boundary

Non-separable Data

Smart transformations resolve surprisingly many such cases

SVM classification can be extended to almost any data using something called the kernel trick

Nonlinear SVM



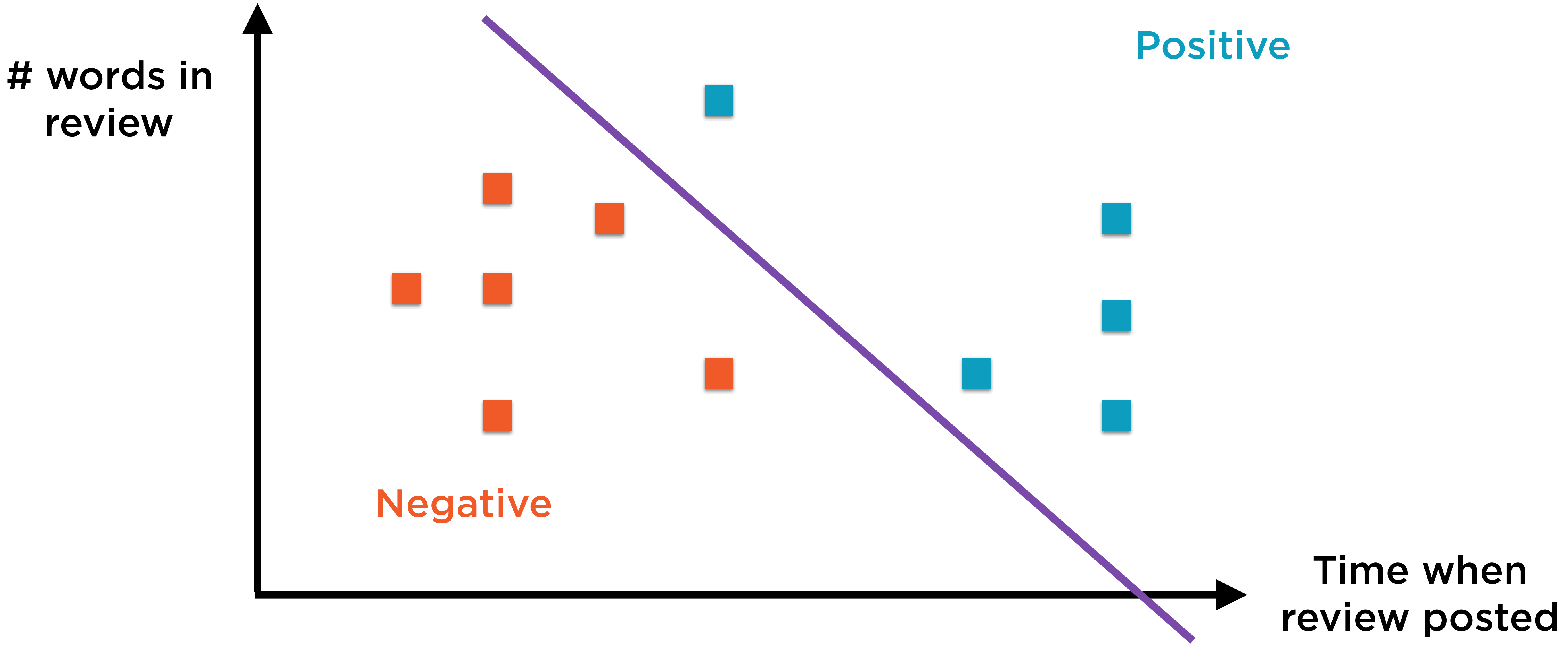
Original Data

Not linearly separable

Square of original data

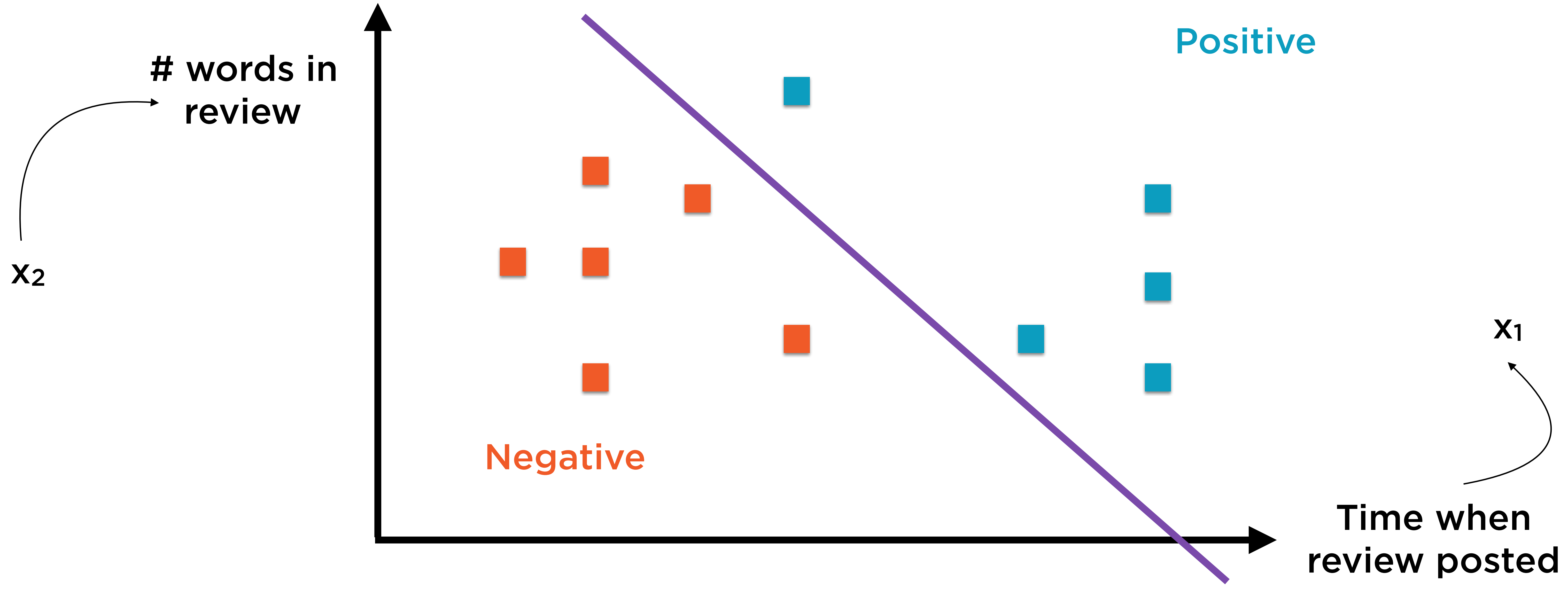
Now linearly separable!

Margin Classification



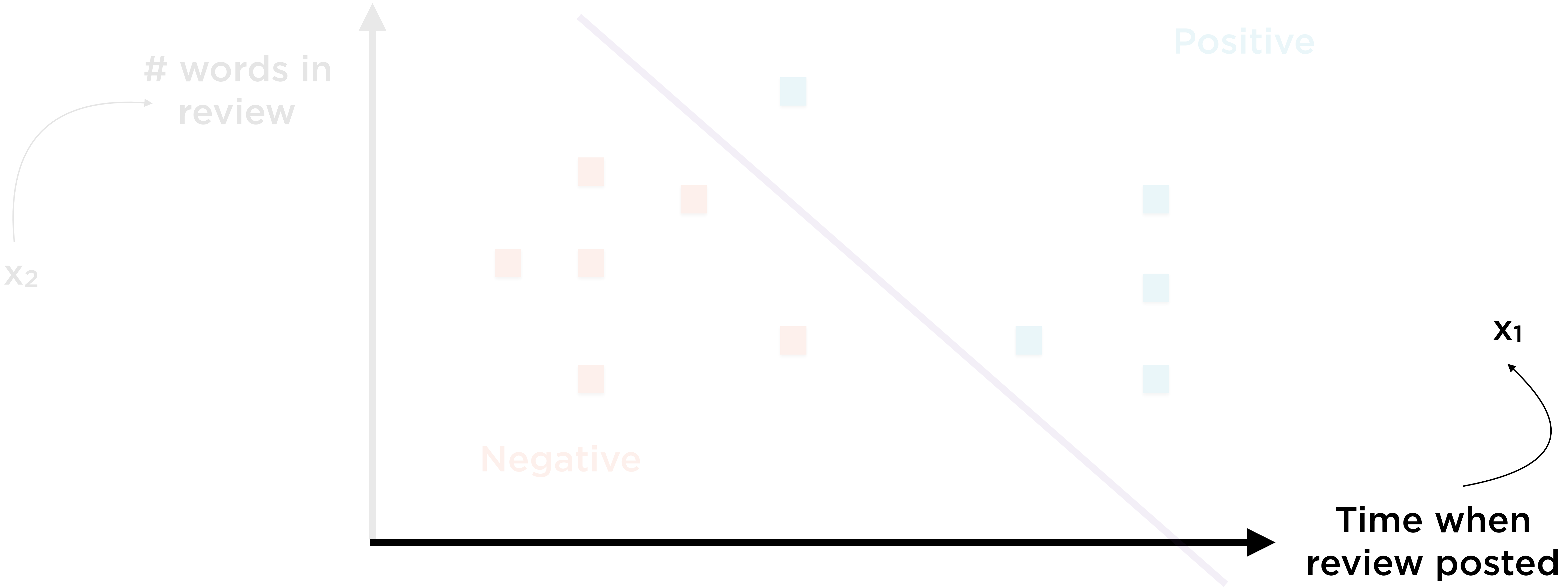
Classify review as positive or negative based on length of review, and time when posted

Margin Classification



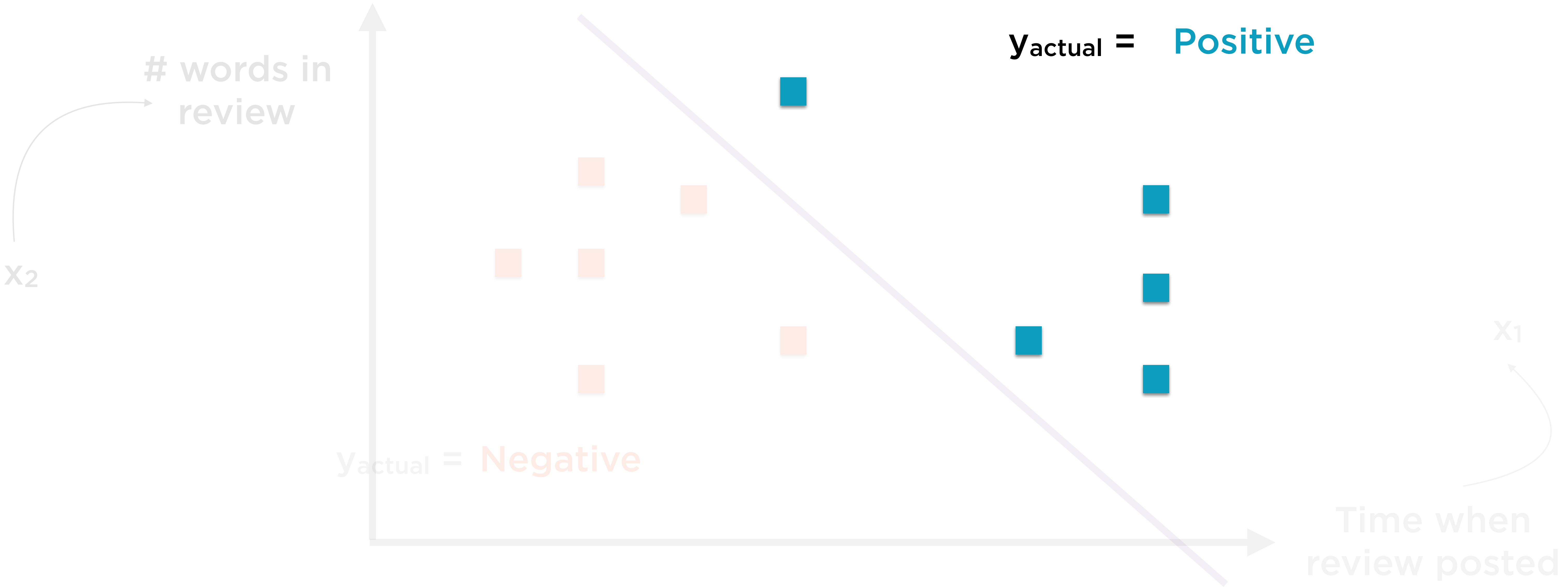
Margin Classification

Margin Classification

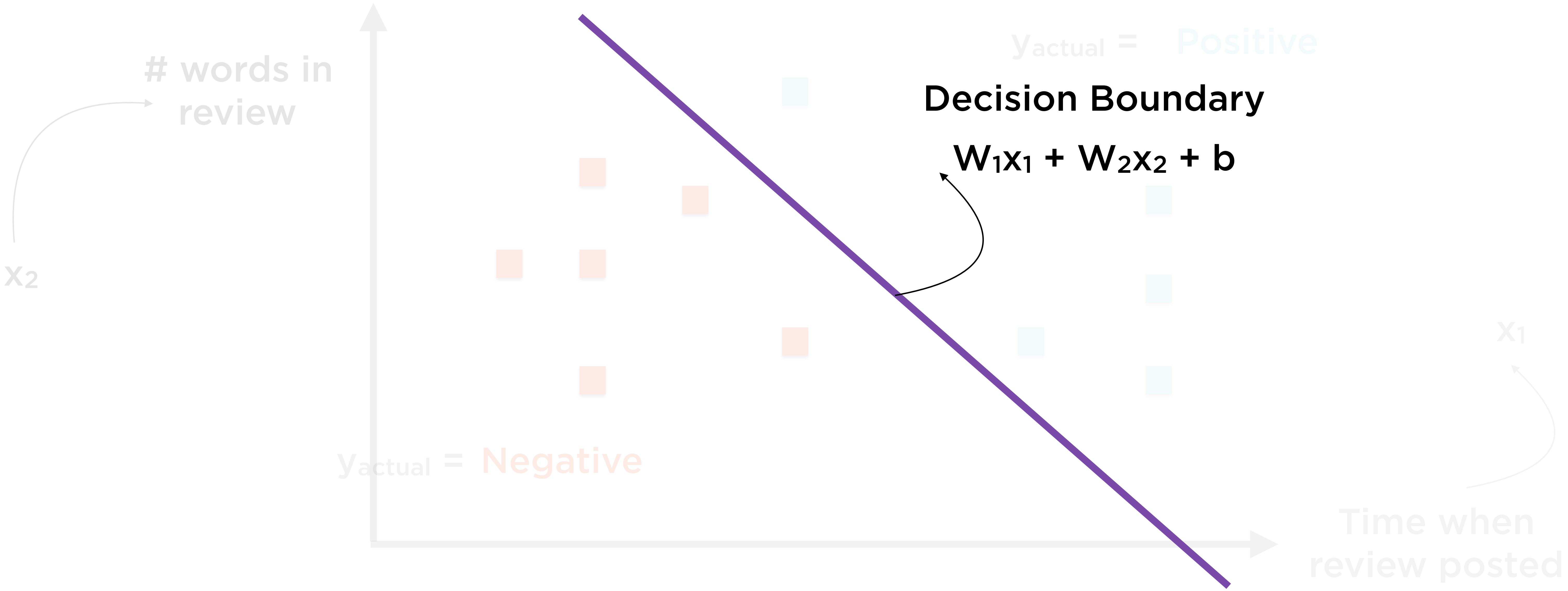


Margin Classification

Margin Classification

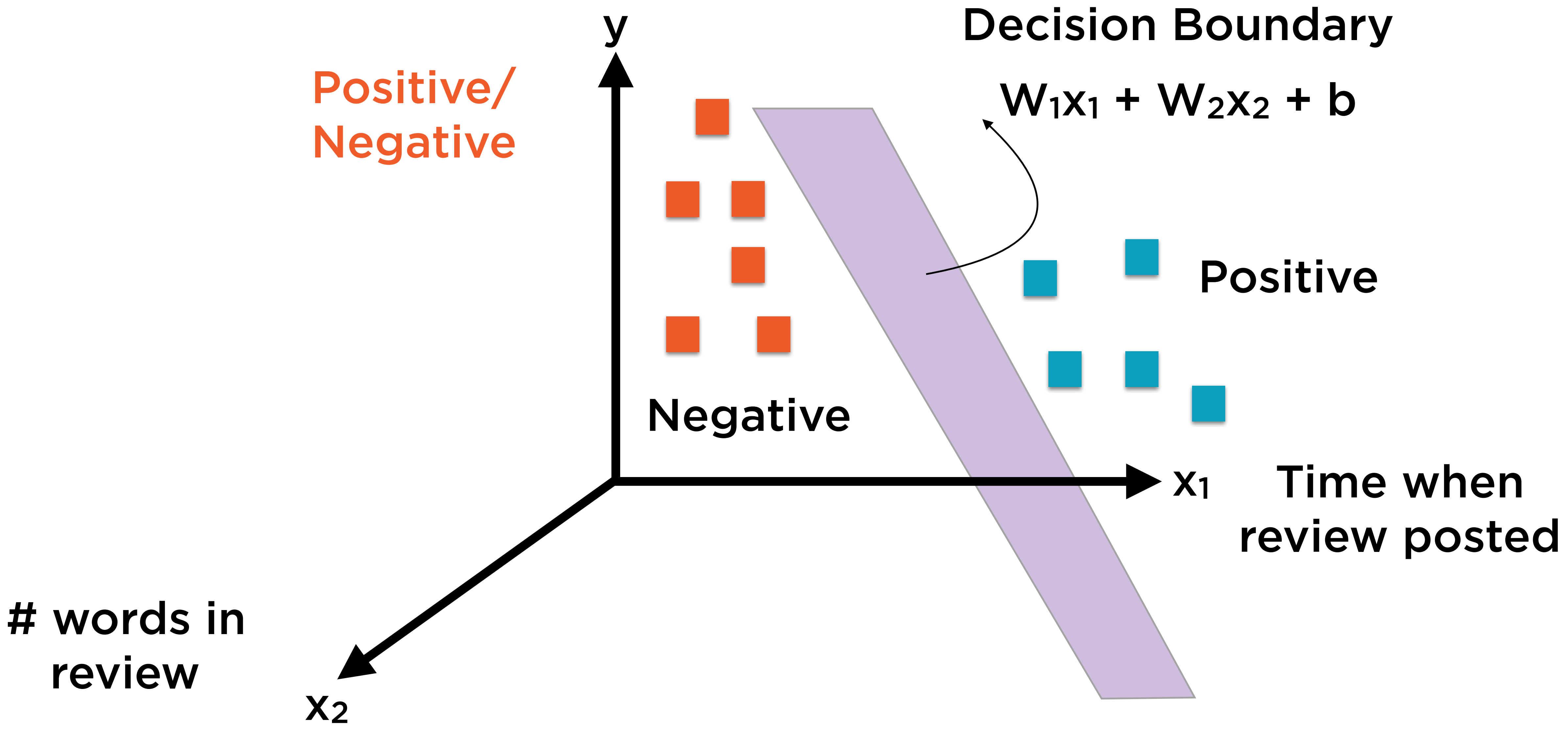


Margin Classification



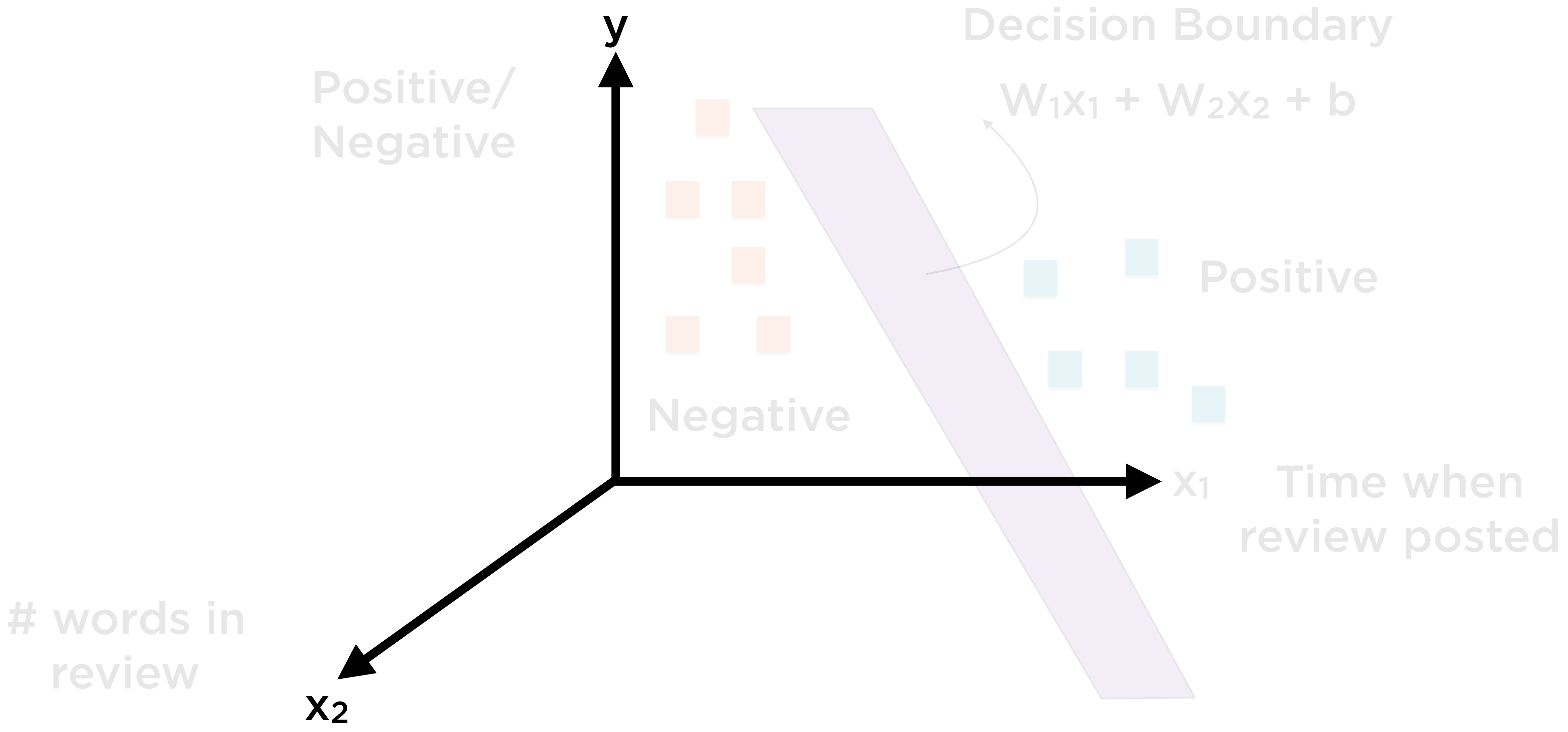
Actually, we need three dimensions to visualize decision boundary correctly

Margin Classification



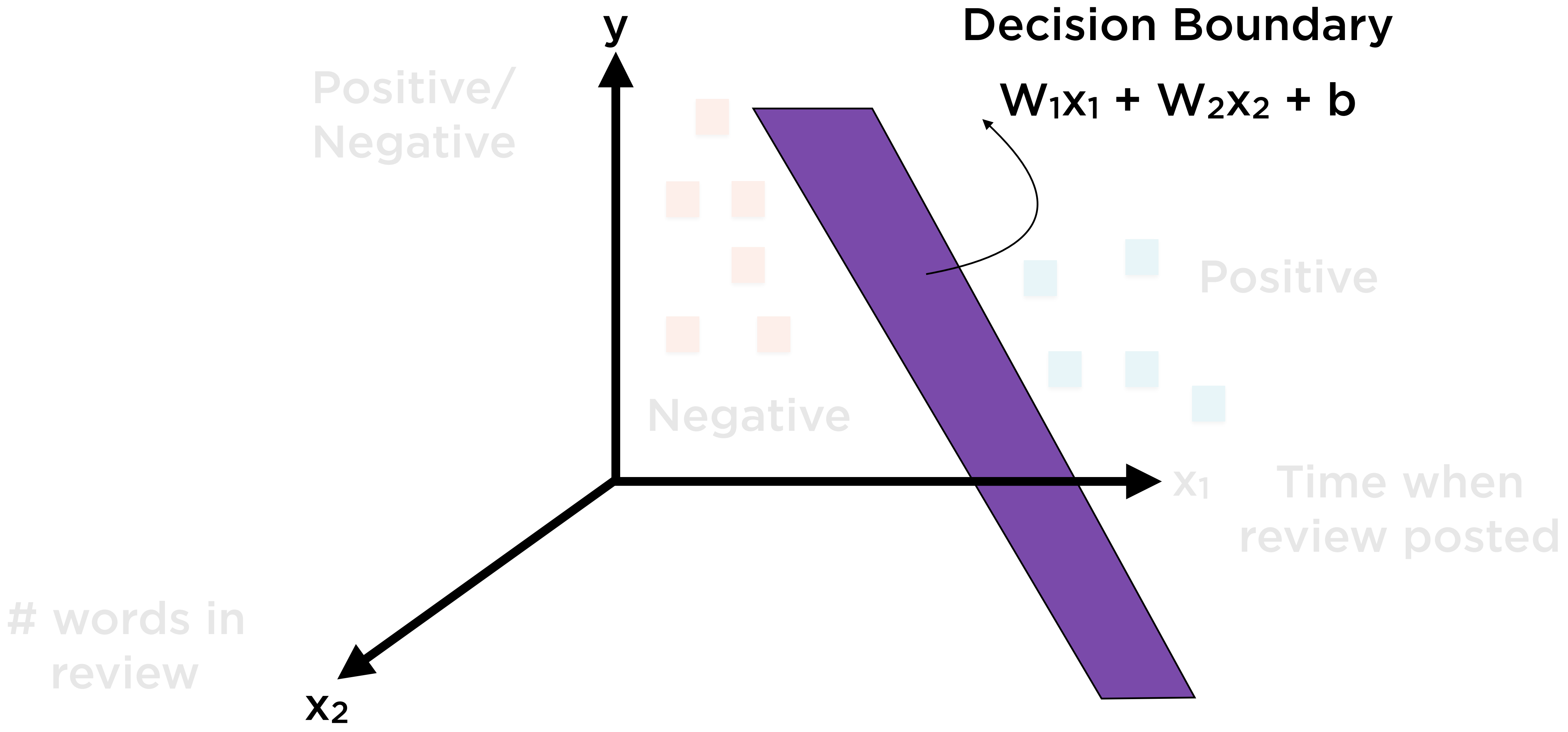
Actually, we need three dimensions to visualize decision boundary correctly

Margin Classification



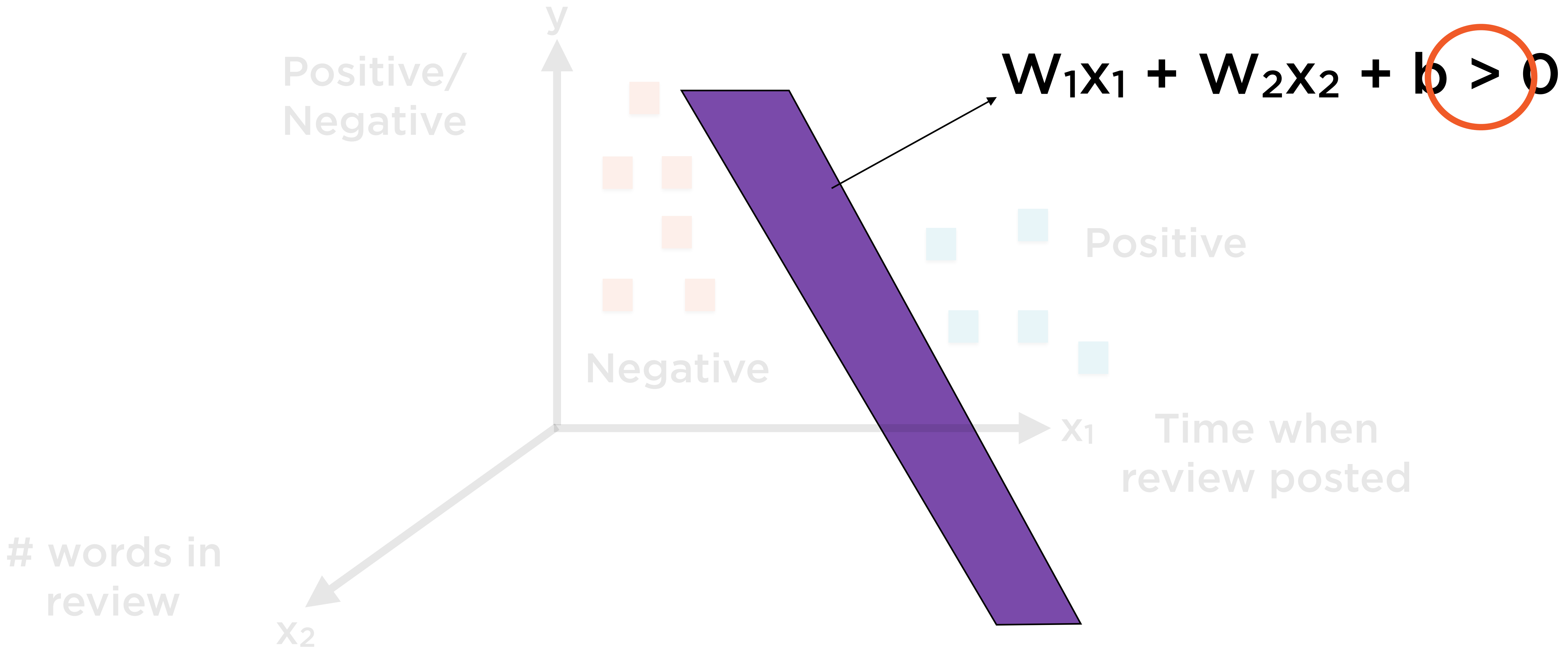
Actually, we need three dimensions to visualize decision boundary correctly

Margin Classification



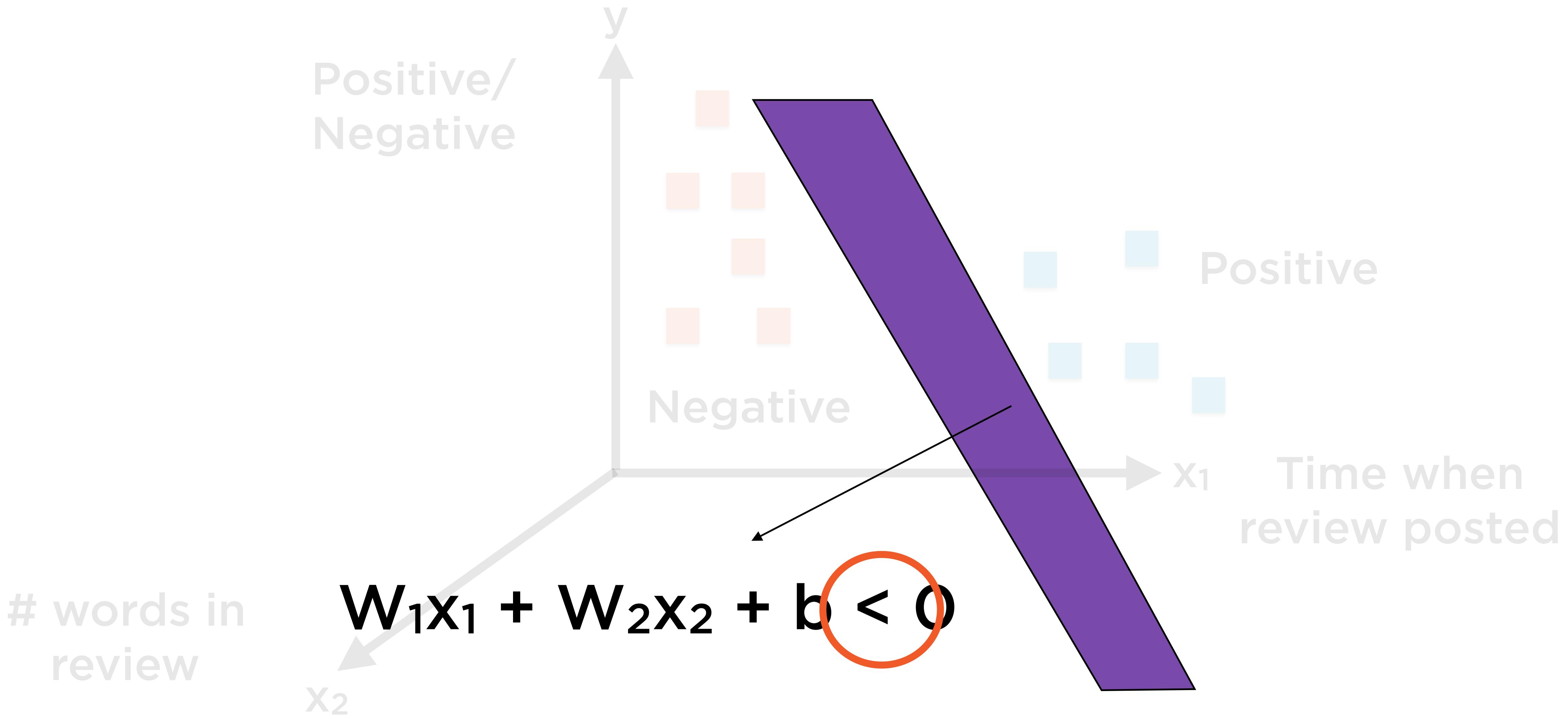
Actually, we need three dimensions to visualize decision boundary correctly

Margin Classification



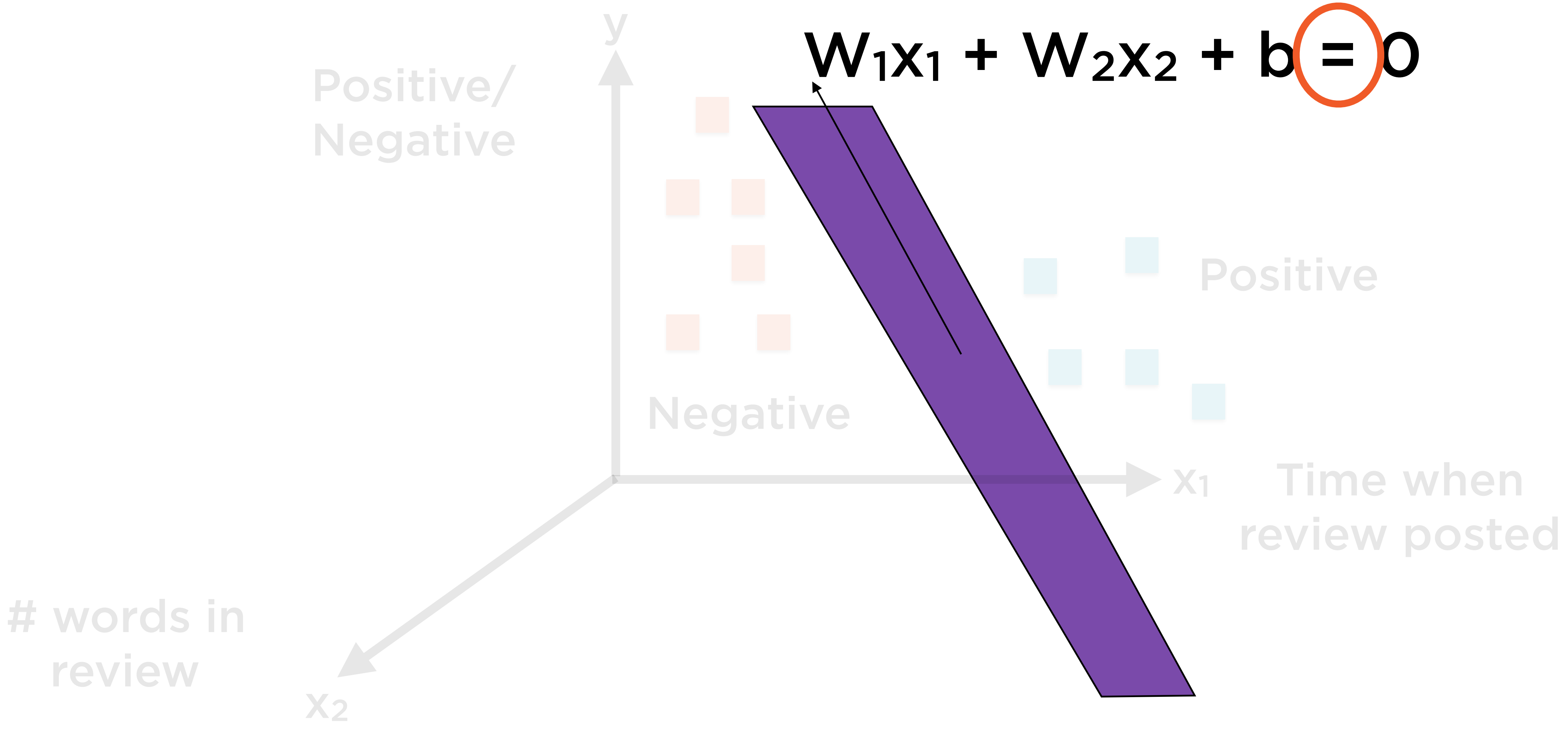
Decision plane separates points based on whether
$W_1x_1 + W_2x_2 + b$ =,<,> 0

Margin Classification



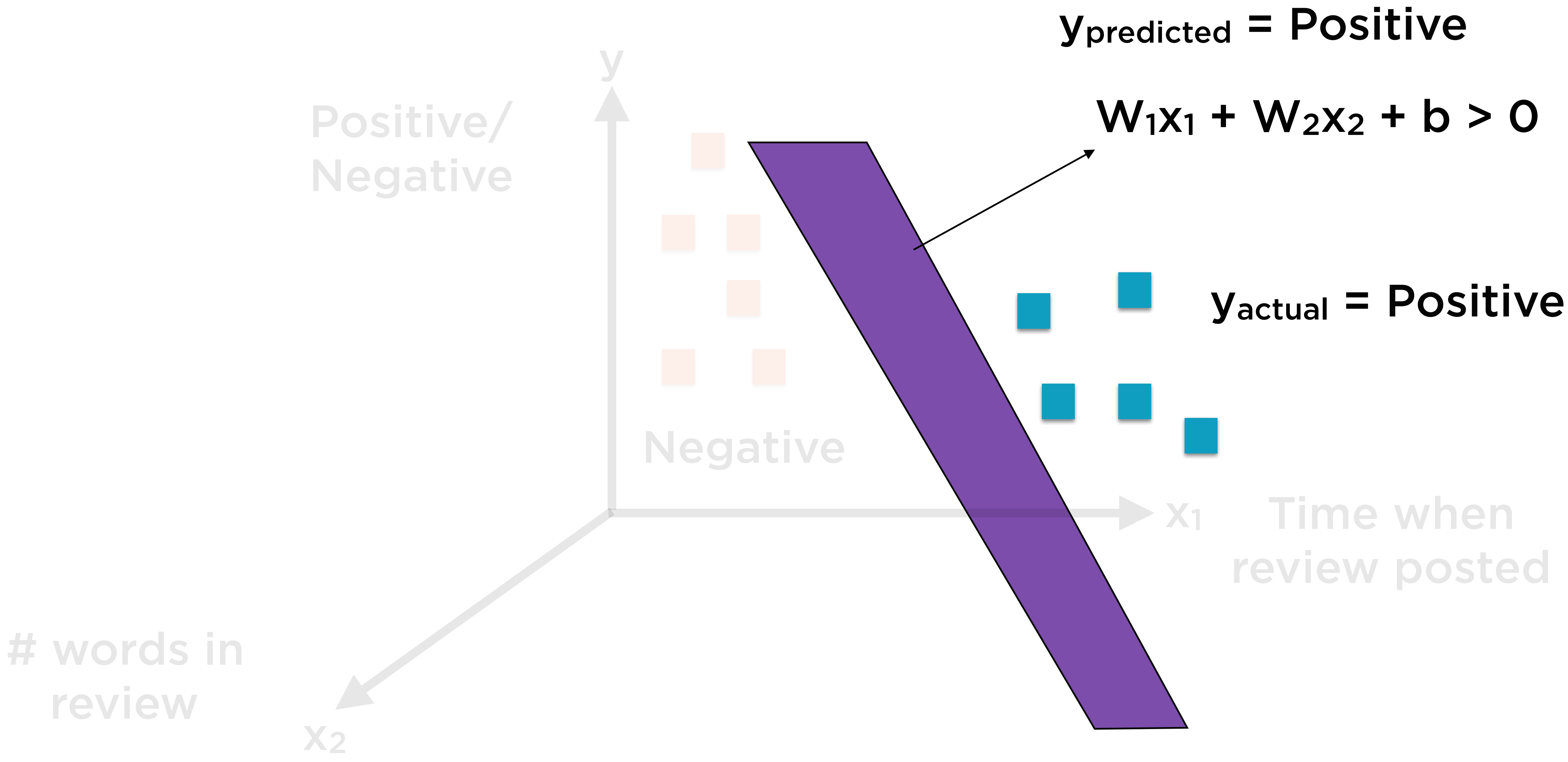
Decision plane separates points based on whether
$W_1x_1 + W_2x_2 + b$ =,<,> 0

Margin Classification



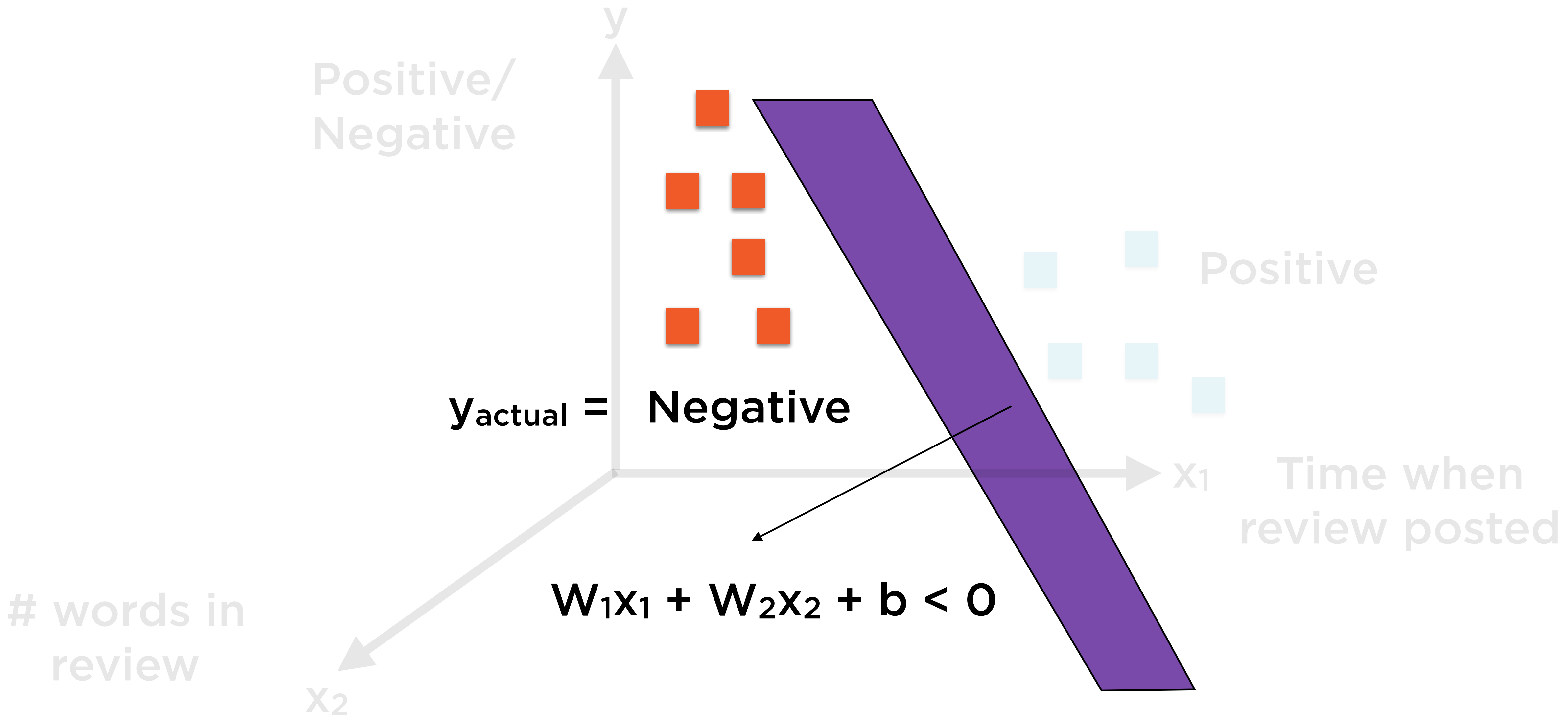
Decision plane separates points based on whether
$W_1x_1 + W_2x_2 + b$ =,<,> 0

Margin Classification



If $W_1x_1 + W_2x_2 + b > 0$ $y_{predicted}$ = Positive

Margin Classification



If $W_1x_1 + W_2x_2 + b <= 0$ $y_{predicted}$ = Negative

Classification Using the Decision Boundary

Find the "best" values of

$$W_1 , W_2 , b$$

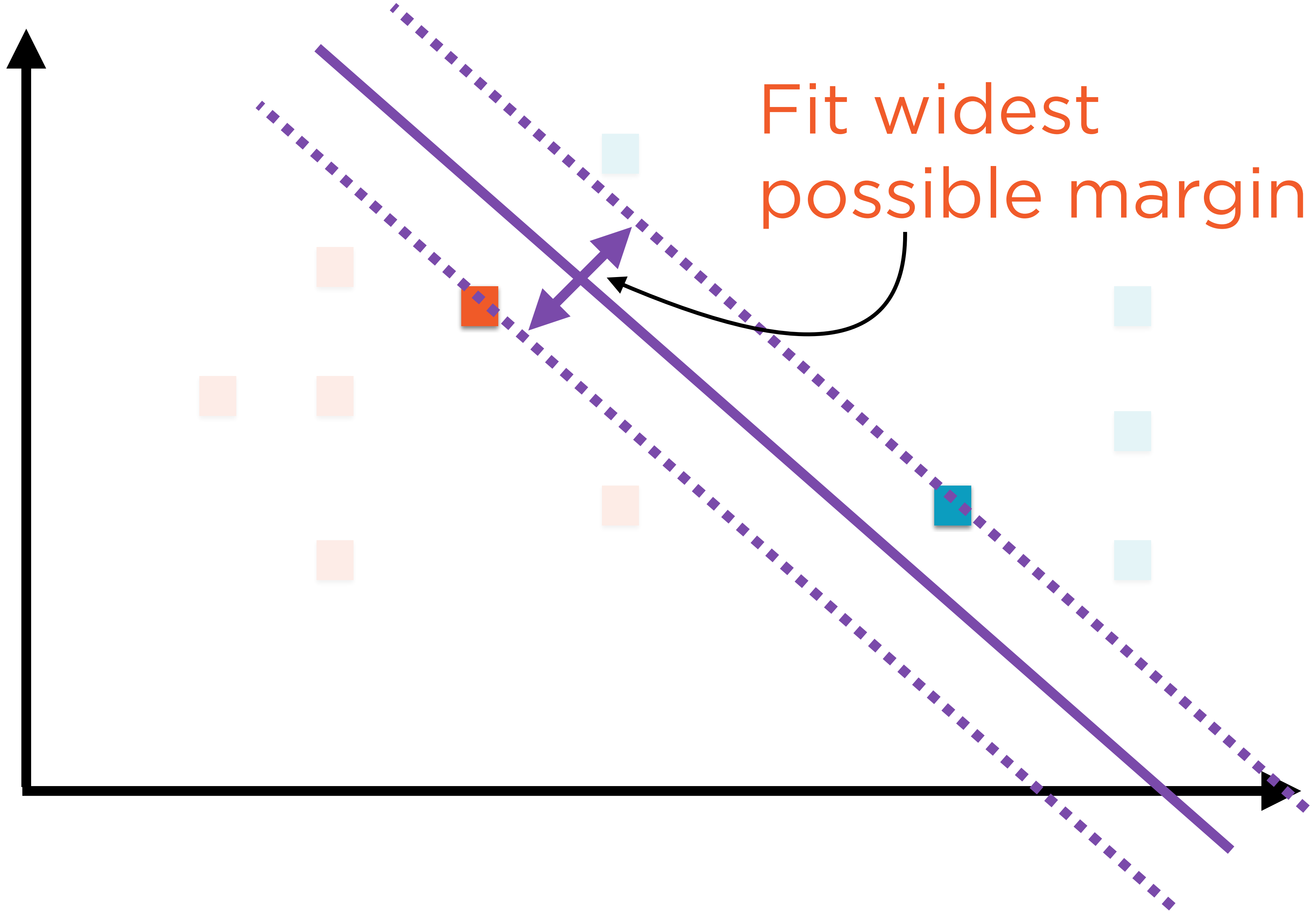
Such that

$$\text{If } W_1x_1 + W_2x_2 + b <= 0$$
$$y_{predicted} = 0$$

$$\text{If } W_1x_1 + W_2x_2 + b > 0$$
$$y_{predicted} = 1$$

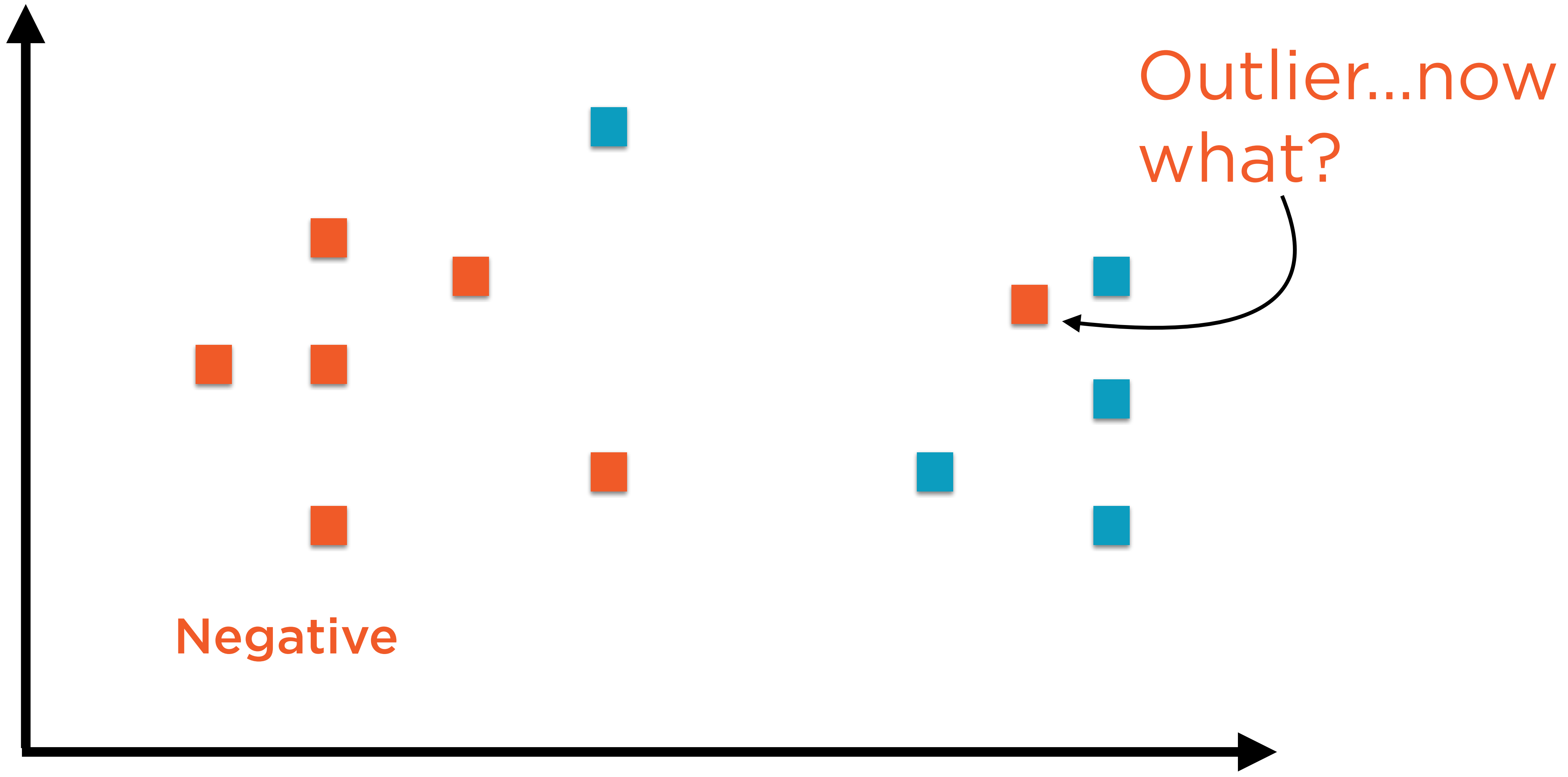
Represent
Negative = 0
Positive = 1

Classification Using the Decision Boundary



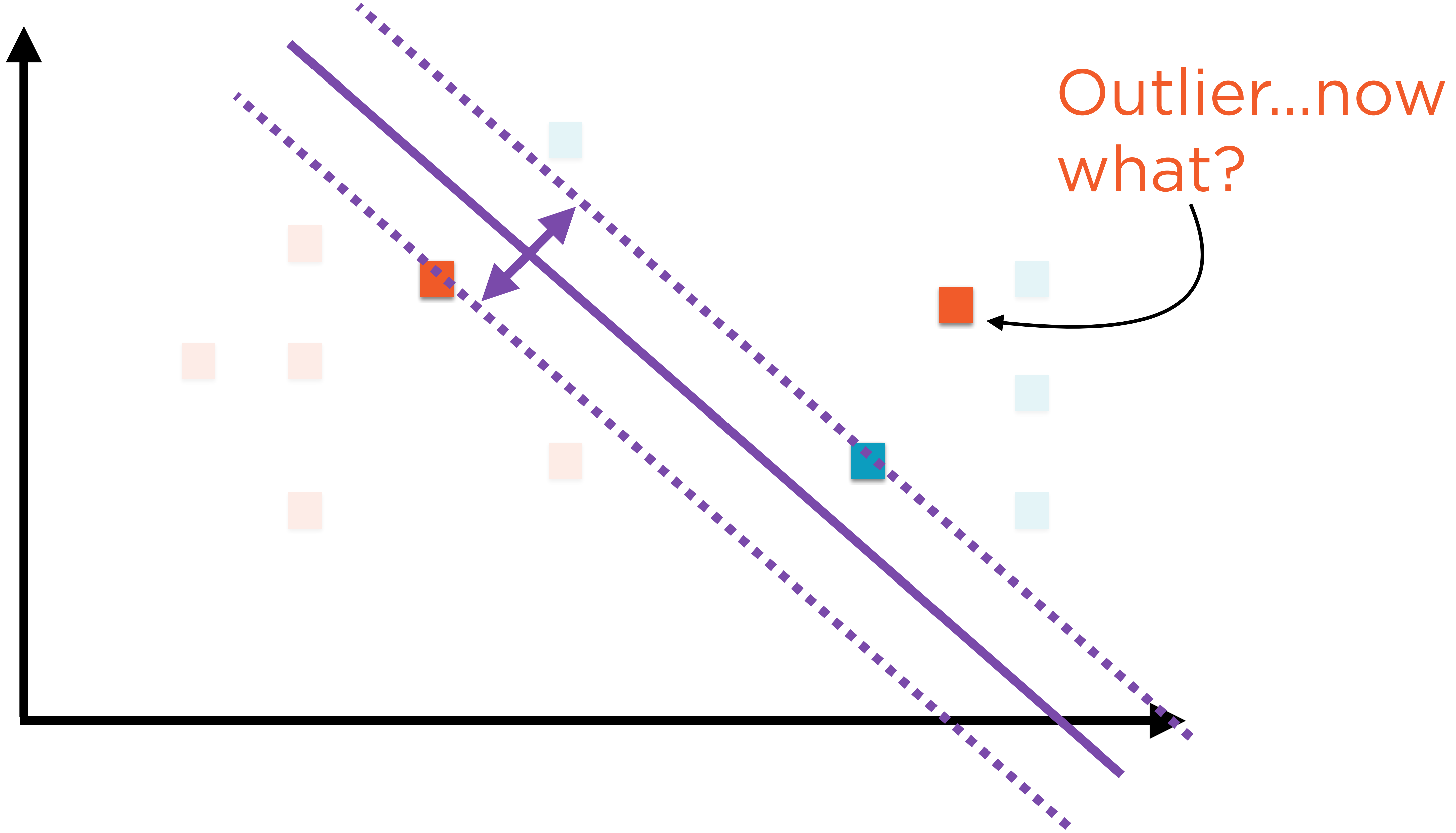
SVM finds the widest street between the nearest points on either side

Classification Using the Decision Boundary



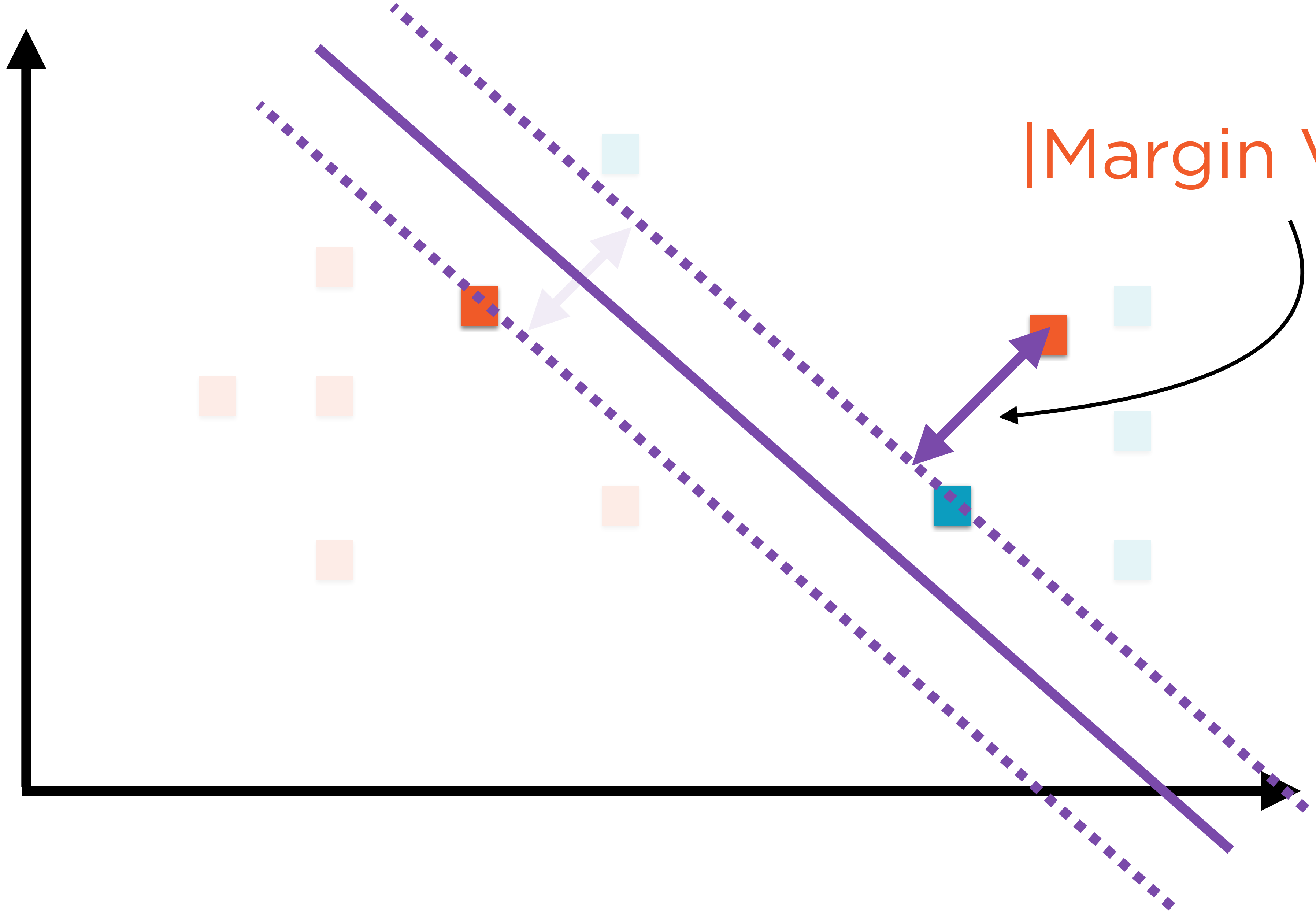
But "best" must also avoid or minimize outliers (by penalizing them during the optimization)

Classification Using the Decision Boundary



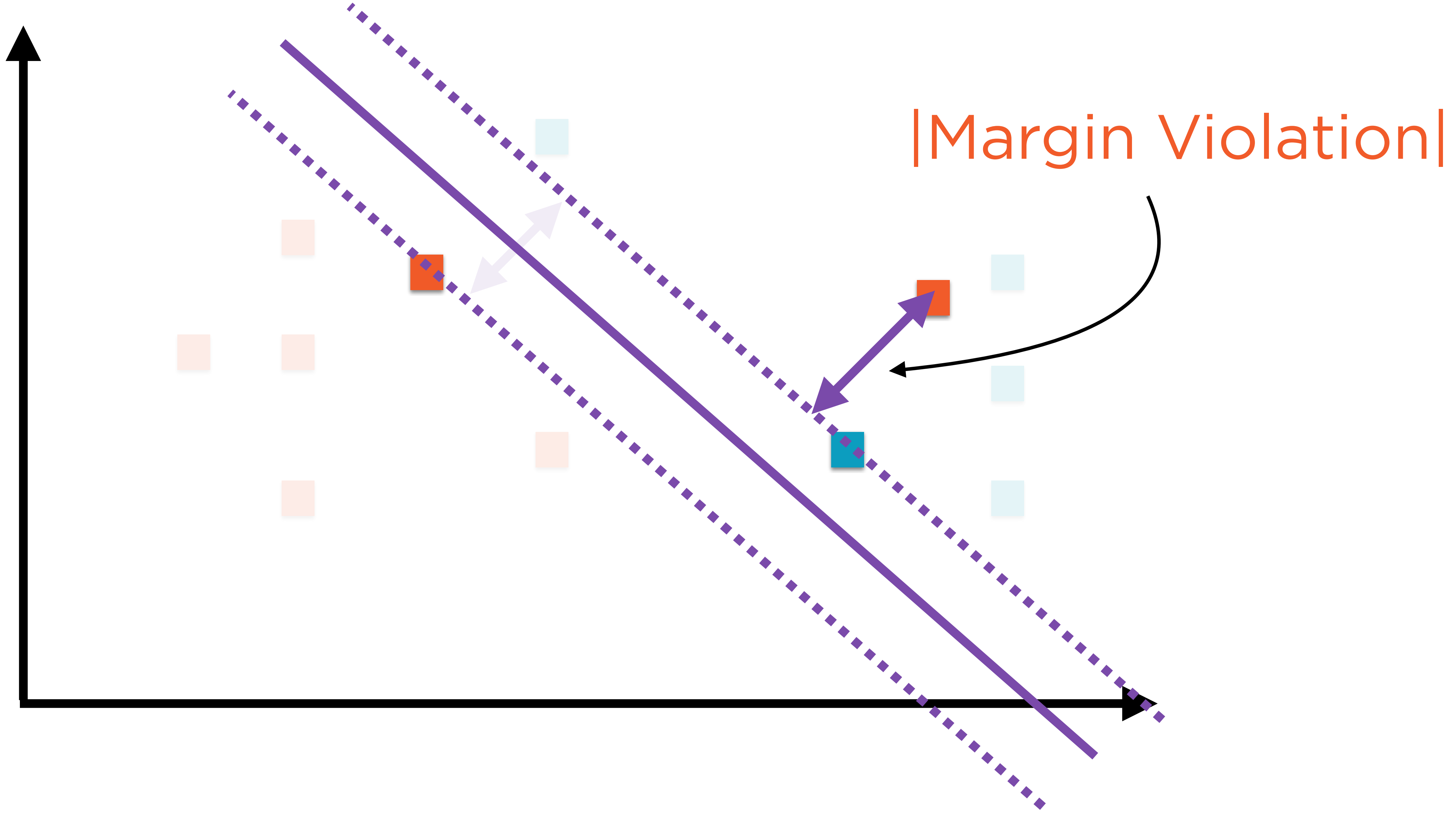
But "best" must also avoid or minimize outliers (by penalizing them during the optimization)

Classification Using the Decision Boundary



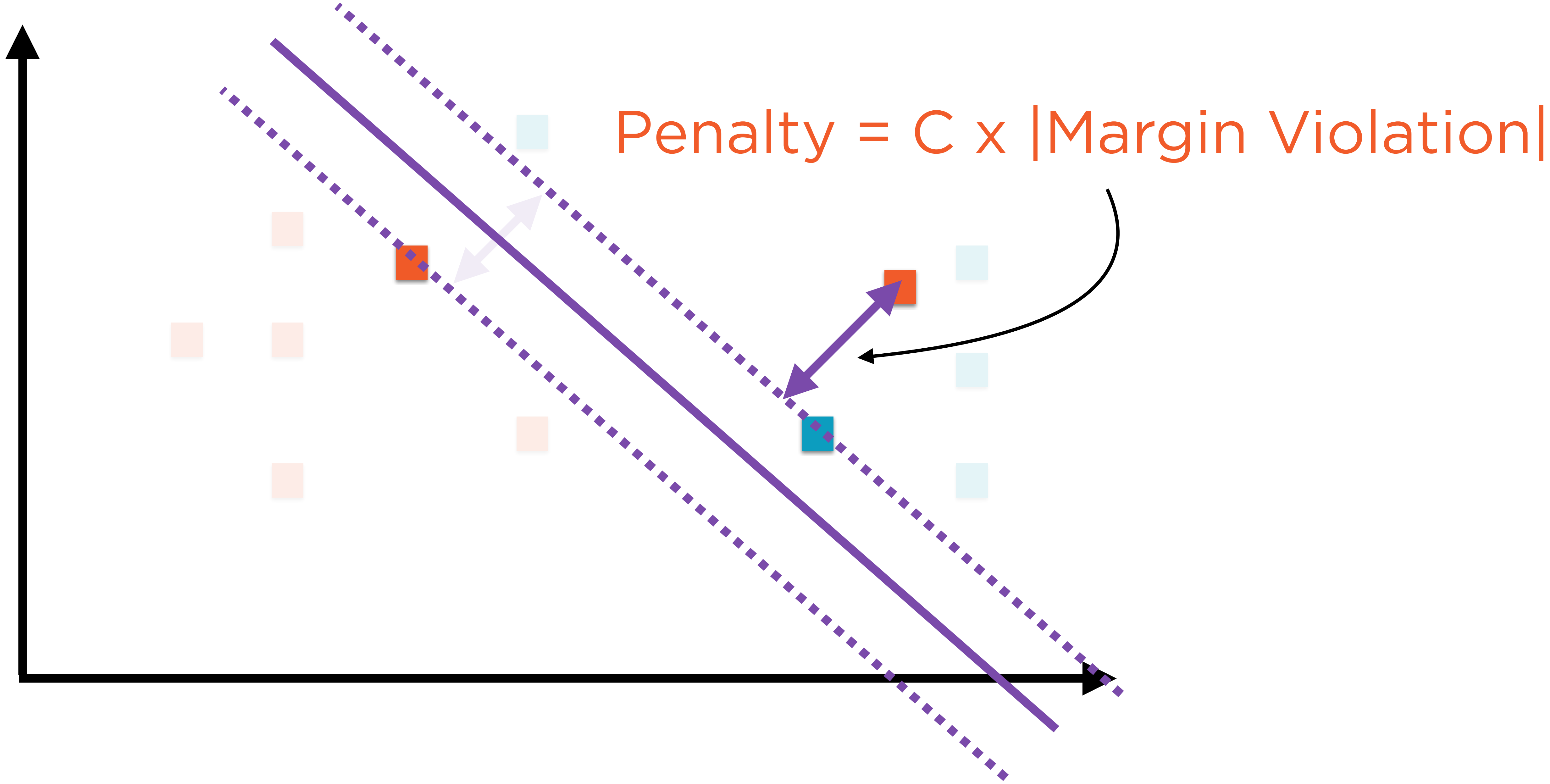
Calculate the magnitude of the margin violation for each point on the wrong side of the boundary

Classification Using the Decision Boundary



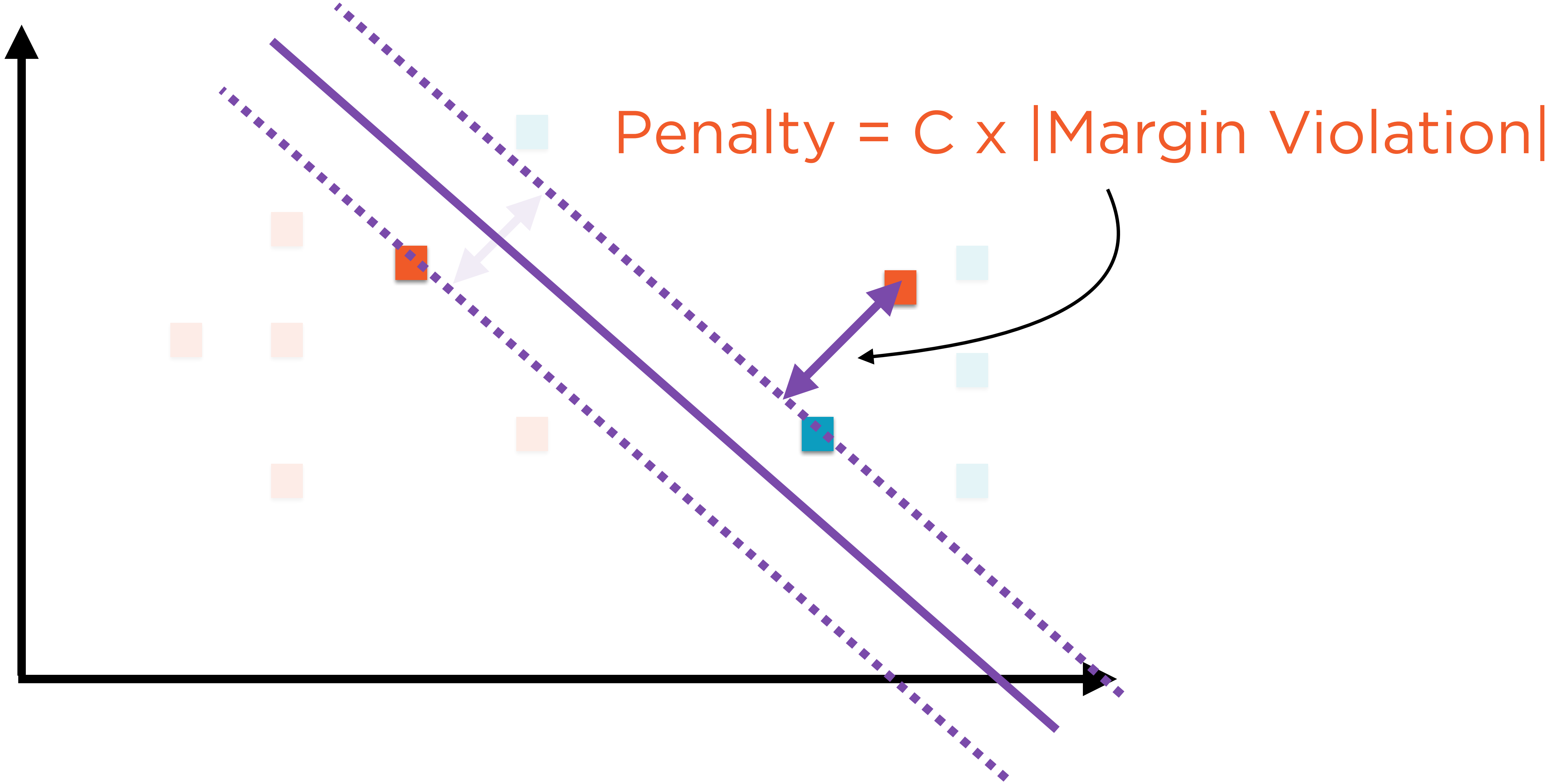
Multiply this magnitude of margin violation by a penalty factor C

Classification Using the Decision Boundary



Penalize each outlier using hyperparameter C

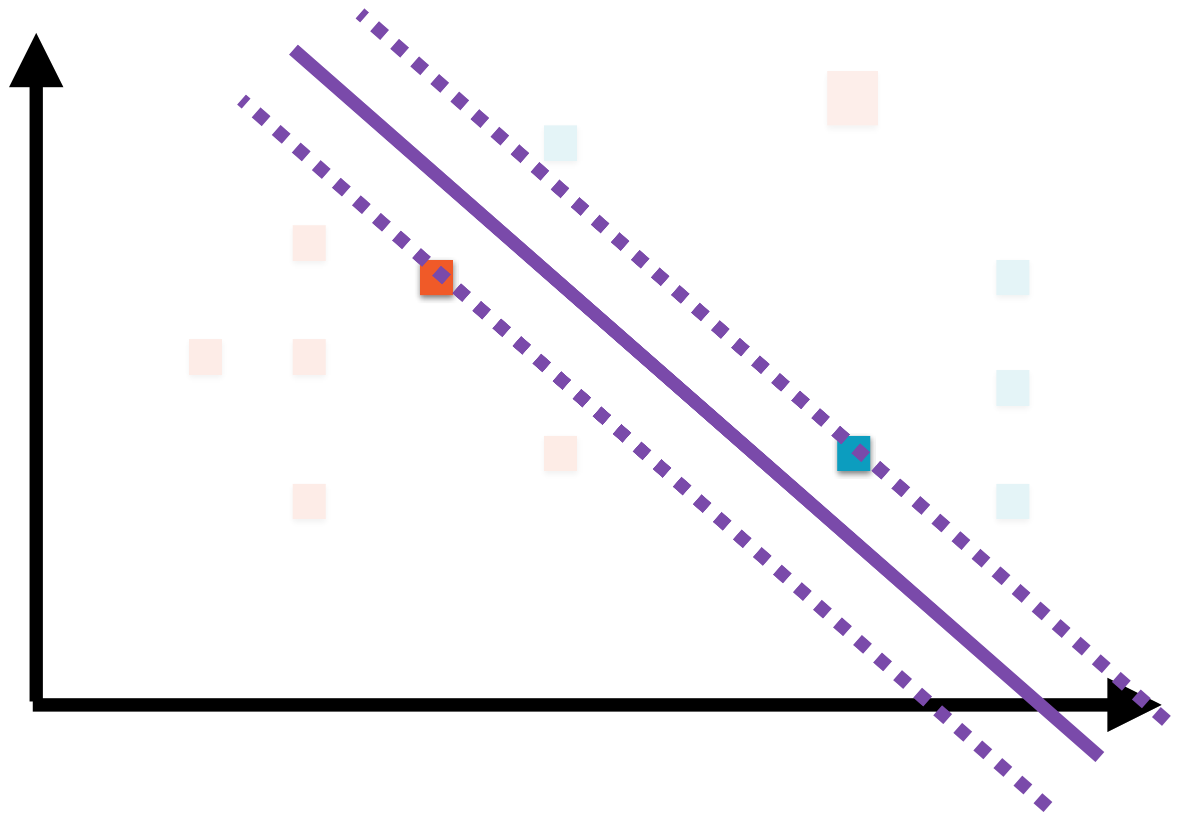
Classification Using the Decision Boundary



Very large values of C ~ hard margin classification

Very small values of C ~ soft margin classification

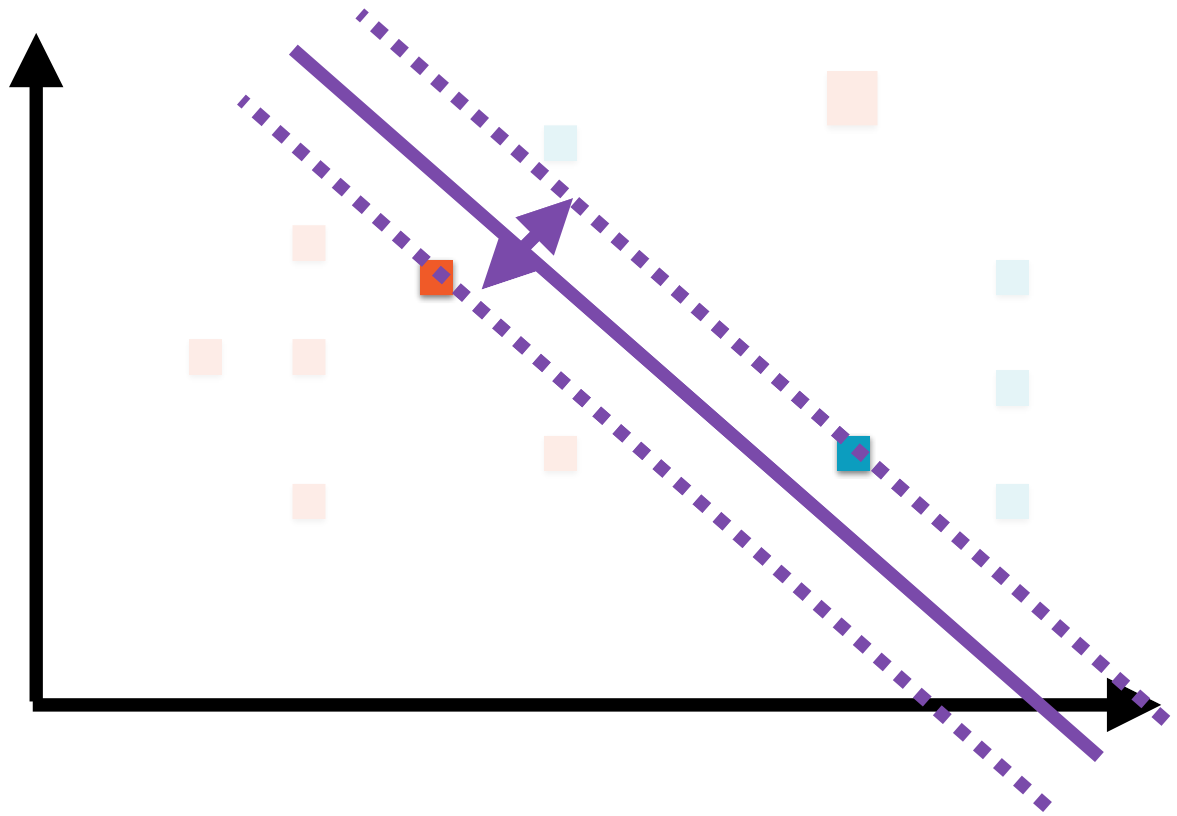
Solving SVM Optimization



Don't need to know precise math

Understand that "best" decision boundary

Solving SVM Optimization

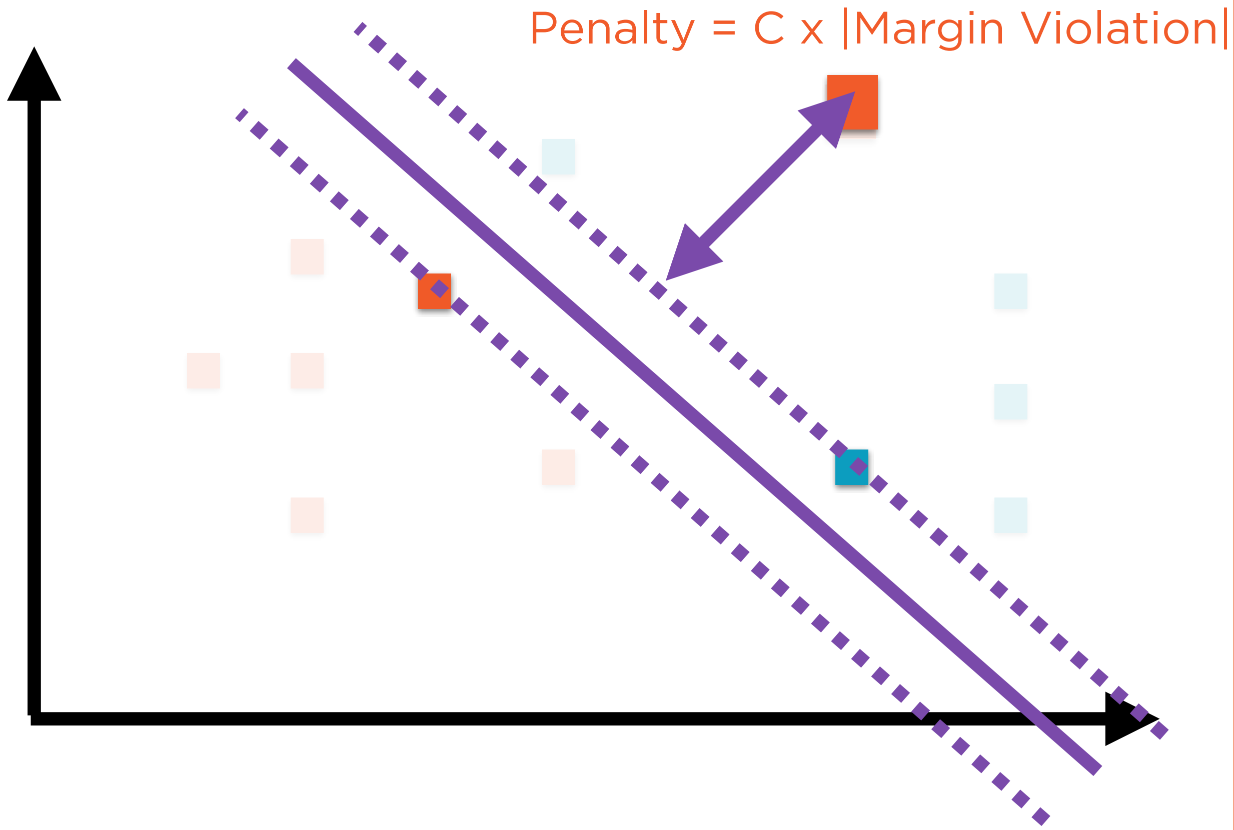


Don't need to know precise math

Understand that "best" decision boundary

- seeks to maximize width of street

Solving SVM Optimization

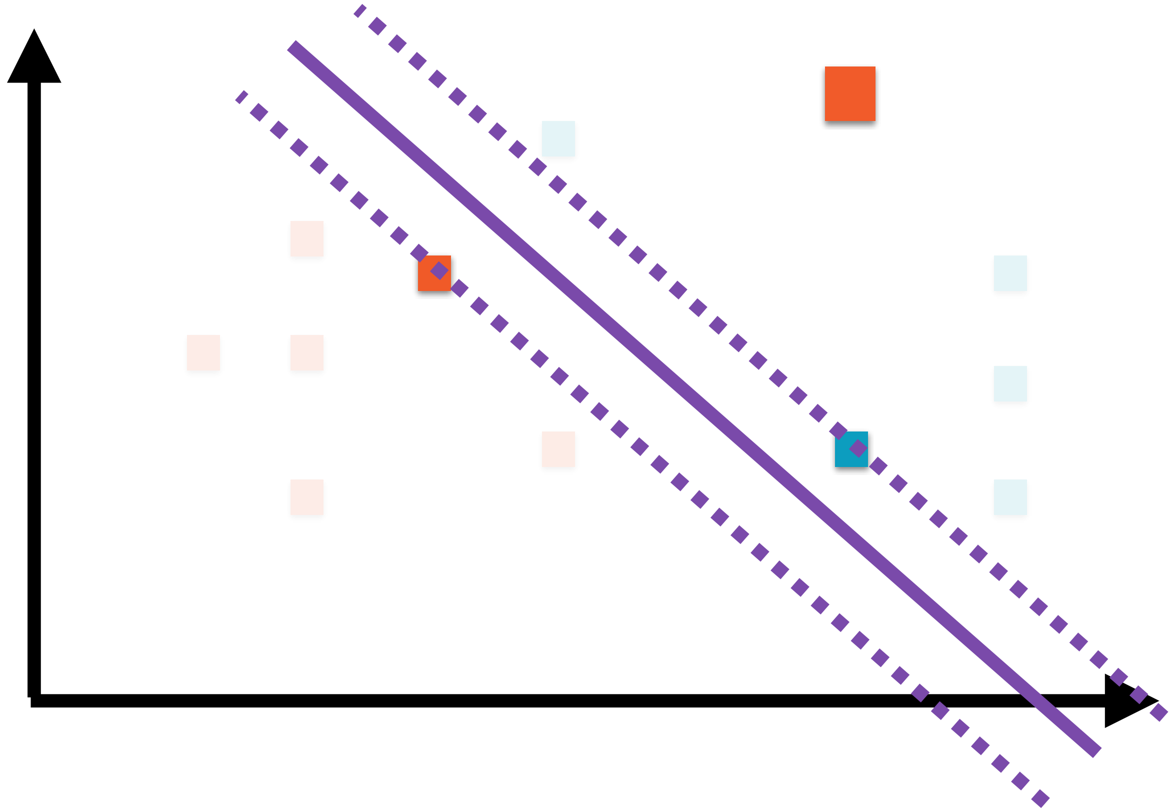


Don't need to know precise math

Understand that "best" decision boundary

- seeks to maximize width of street
- seeks to minimize margin violations

Solving SVM Optimization



Don’t need to know precise math

Understand that “best” decision boundary

- seeks to maximize width of street
- seeks to minimize margin violations

These two objectives are in conflict with each other

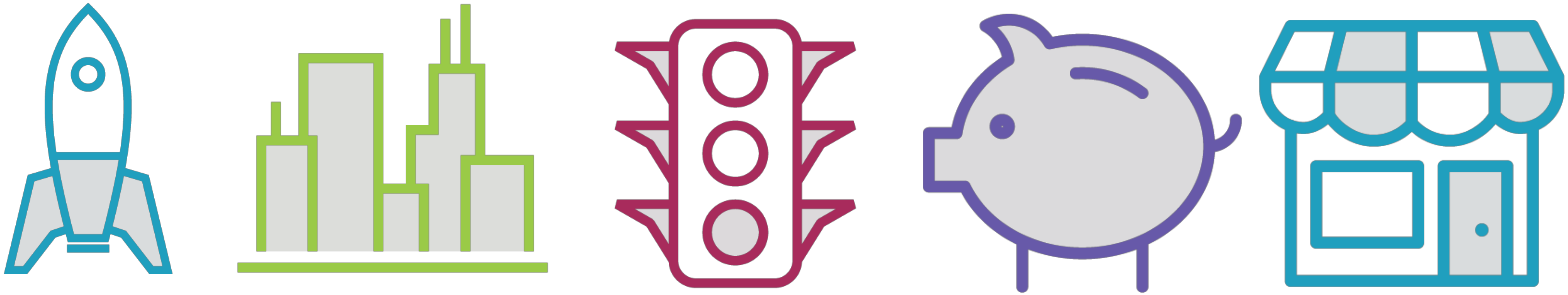
Demo

Classification using Support Vector Machines

Nearest Neighbors Classifiers

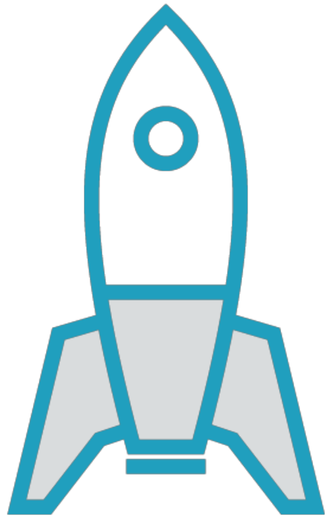
Nearest Neighbors Classification uses training data to find what is most similar to the current sample

Nearest Neighbors Classification



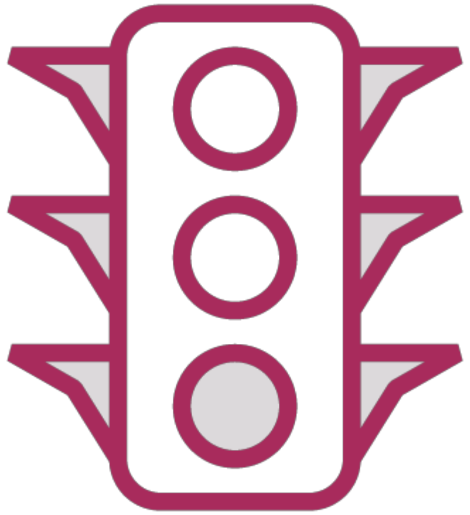
Uses the entire training dataset as a model

Nearest Neighbors Classification

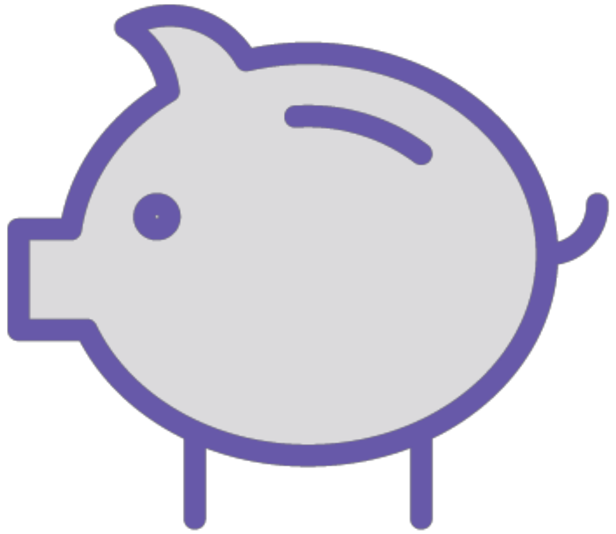


Rocket

Buildings



Signal

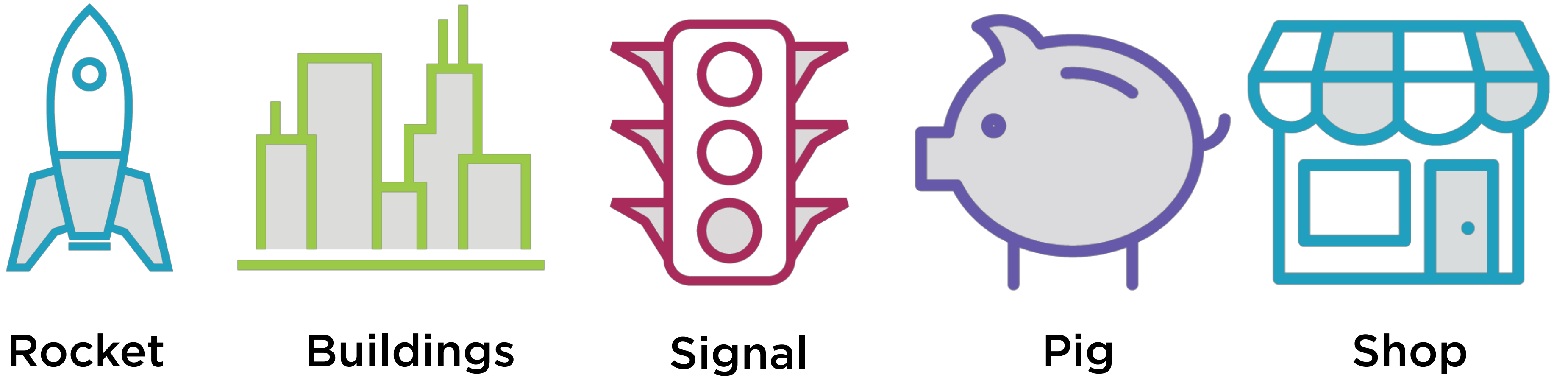


Pig

Shop

Each element in training data has an associated label

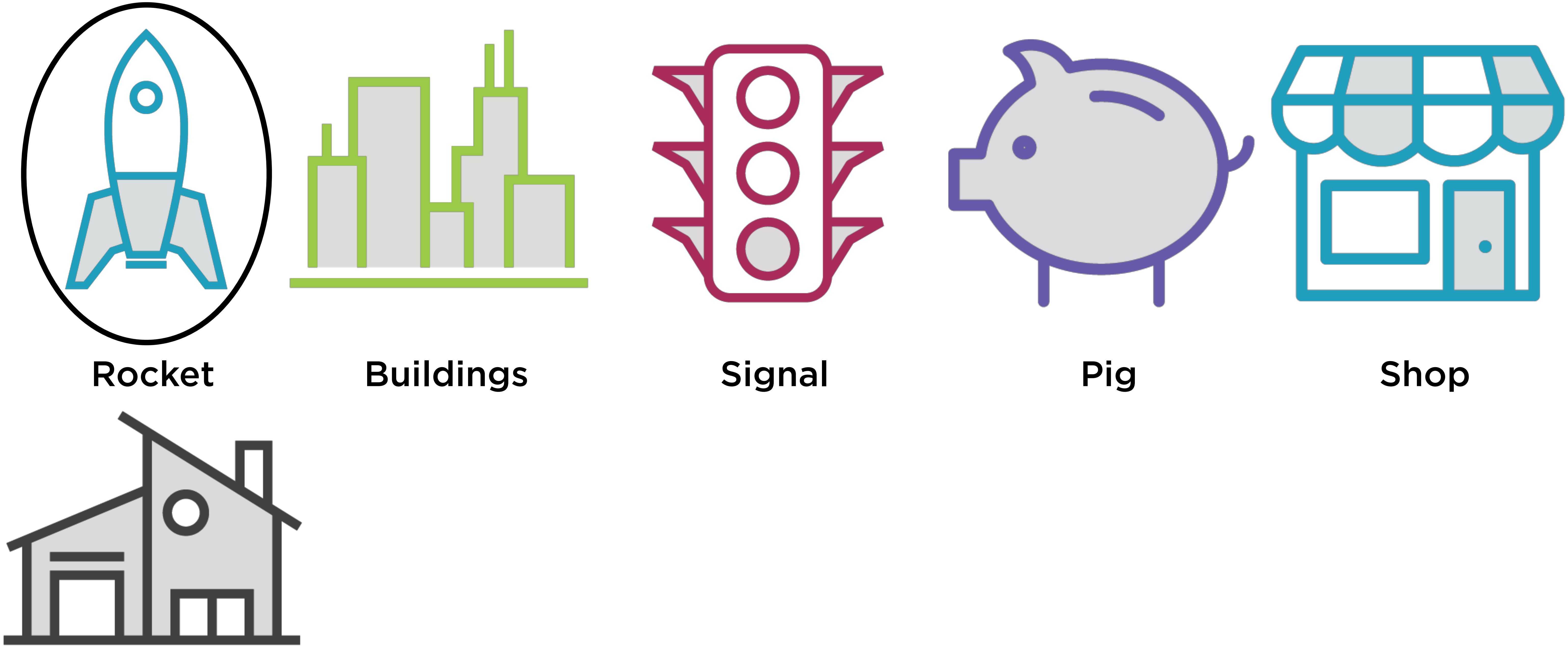
Nearest Neighbors Classification



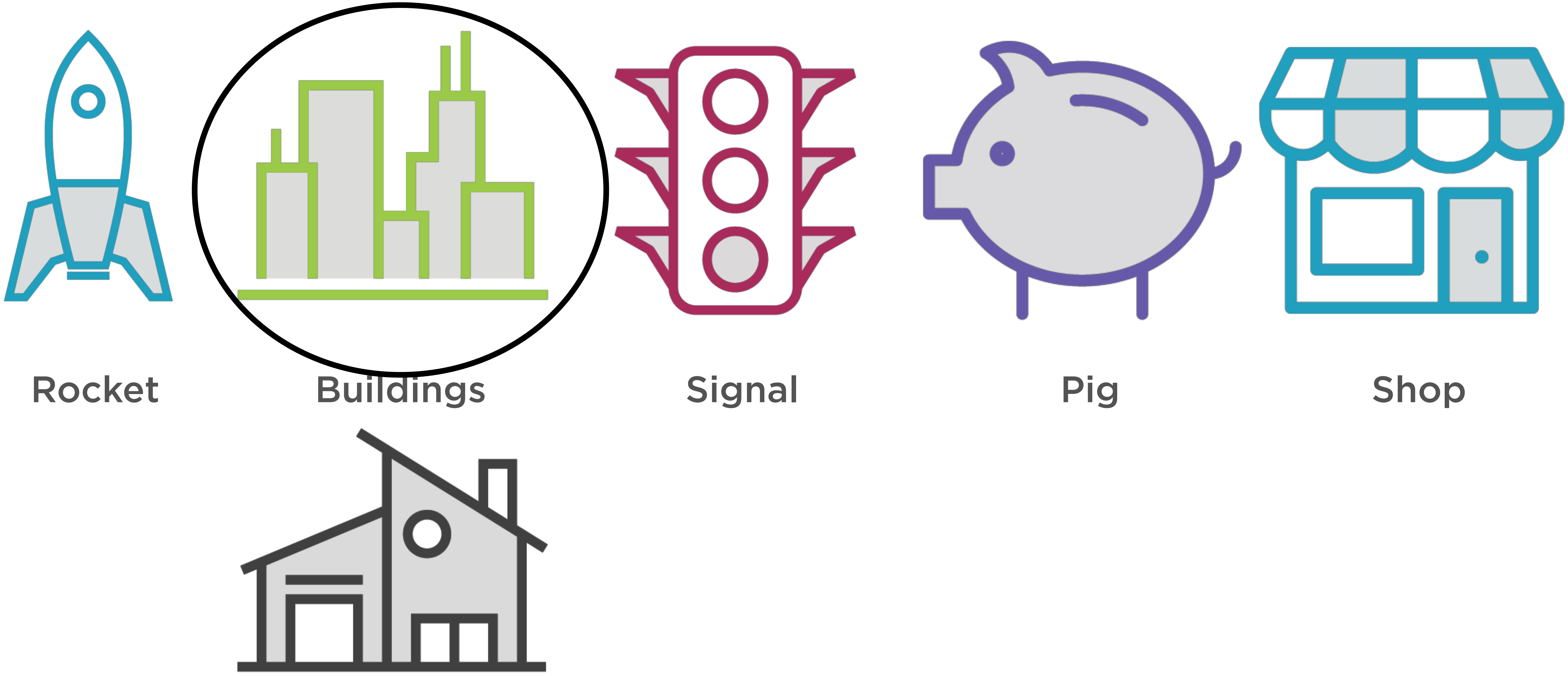
Predictions for a new sample involves figuring out which element in the training data it is similar to

The nearest neighbor

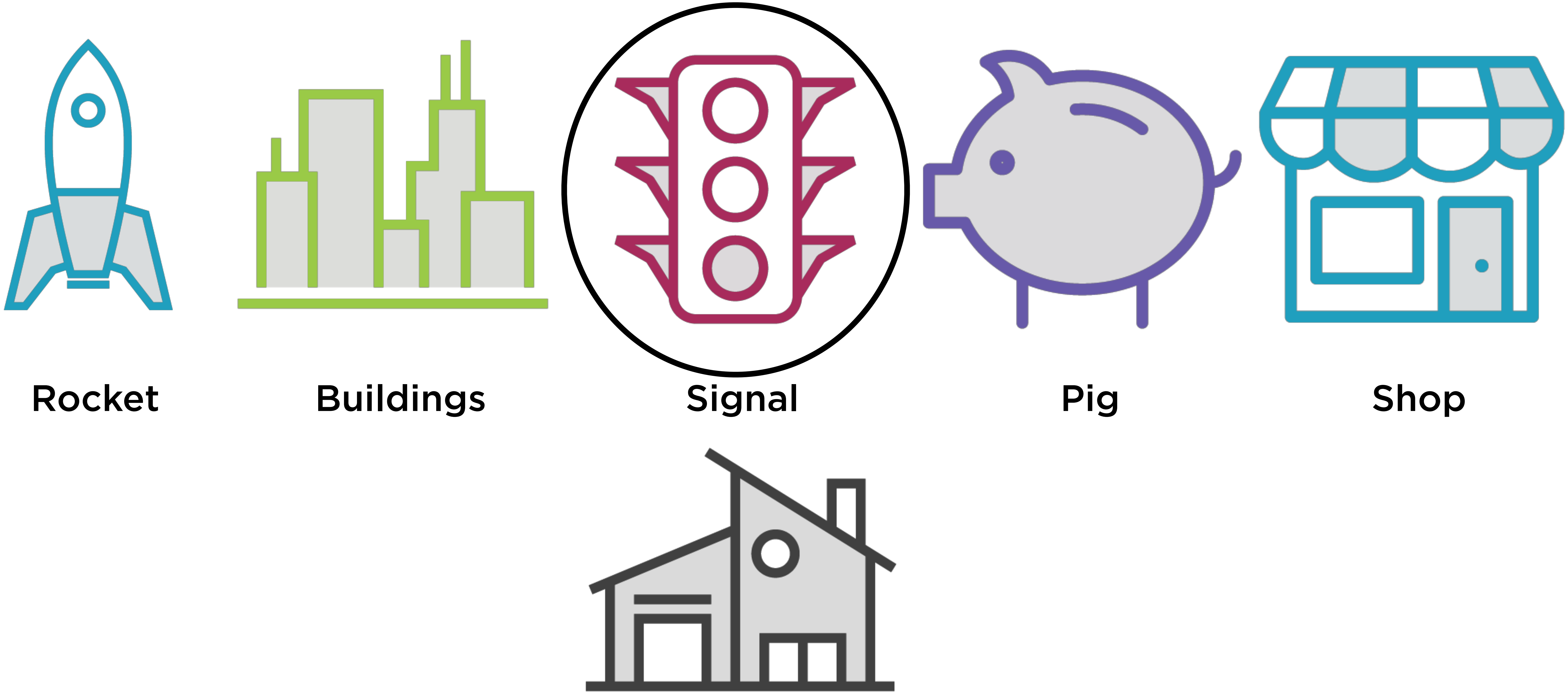
Nearest Neighbors Classification



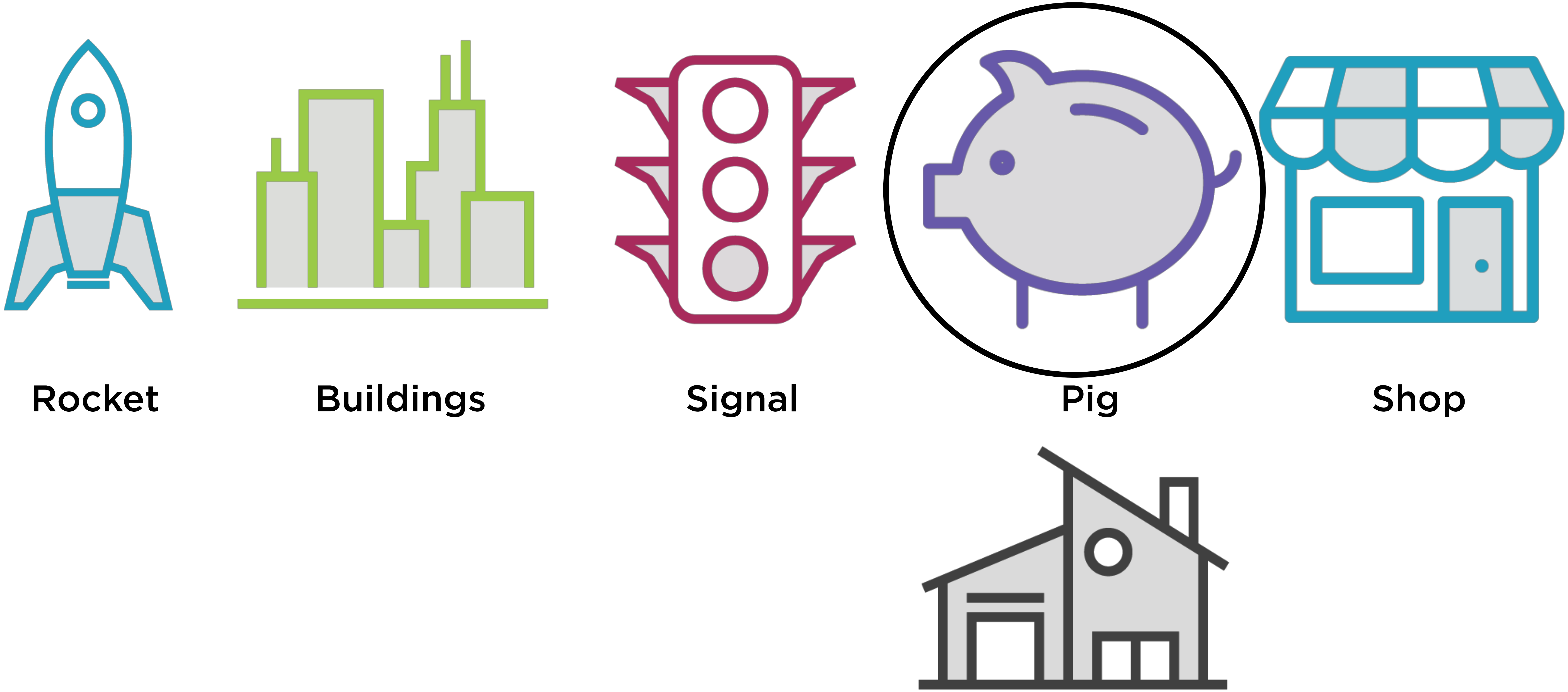
Nearest Neighbors Classification



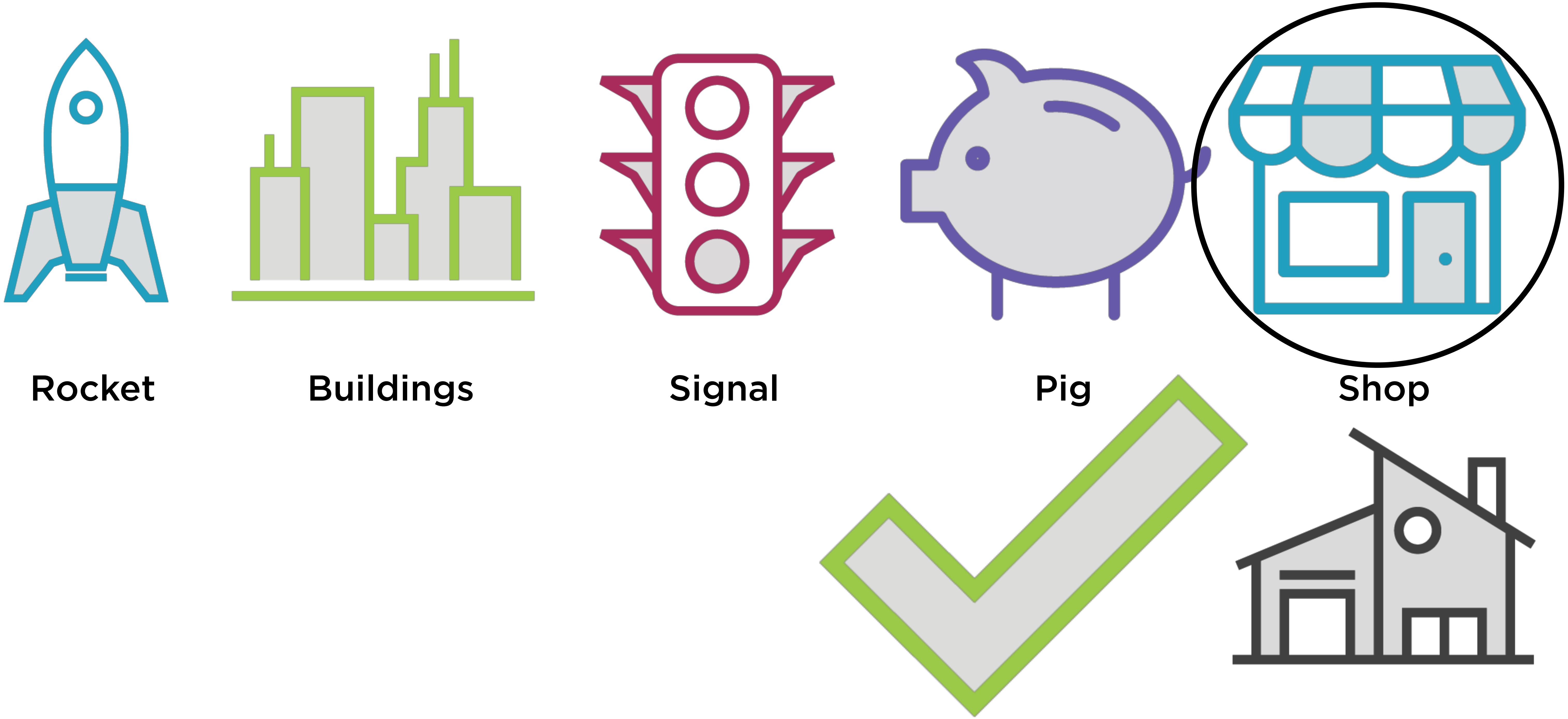
Nearest Neighbors Classification



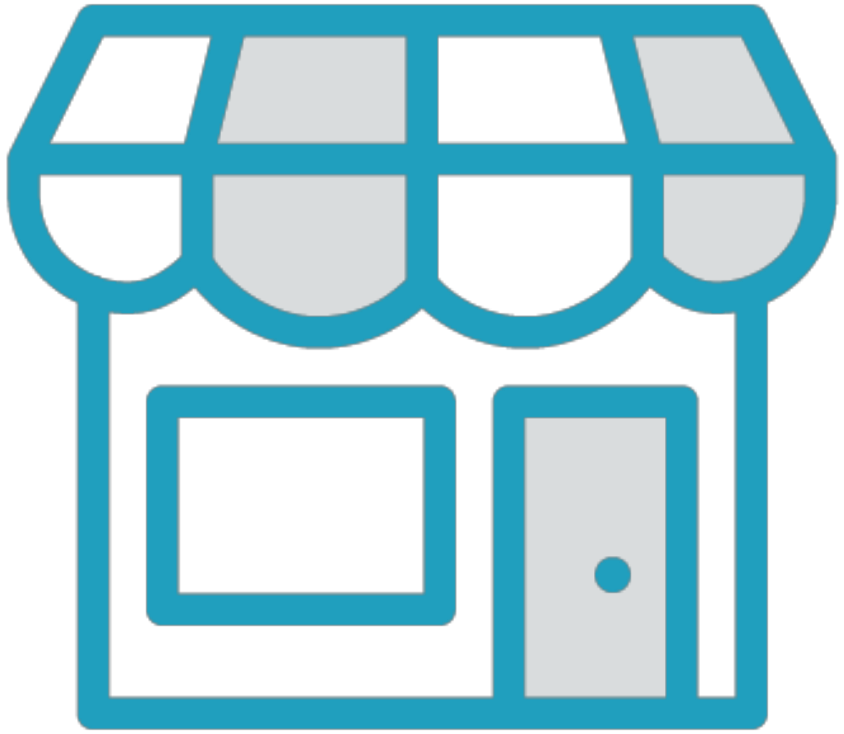
Nearest Neighbors Classification



Nearest Neighbors Classification

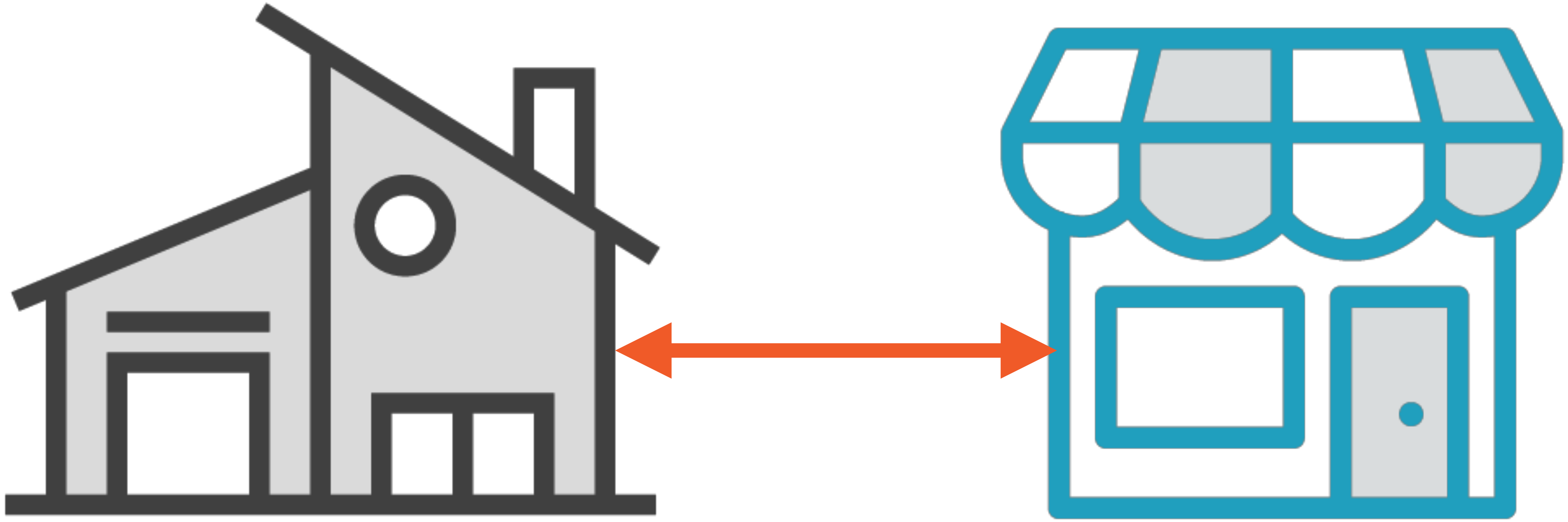


Nearest Neighbors Classification



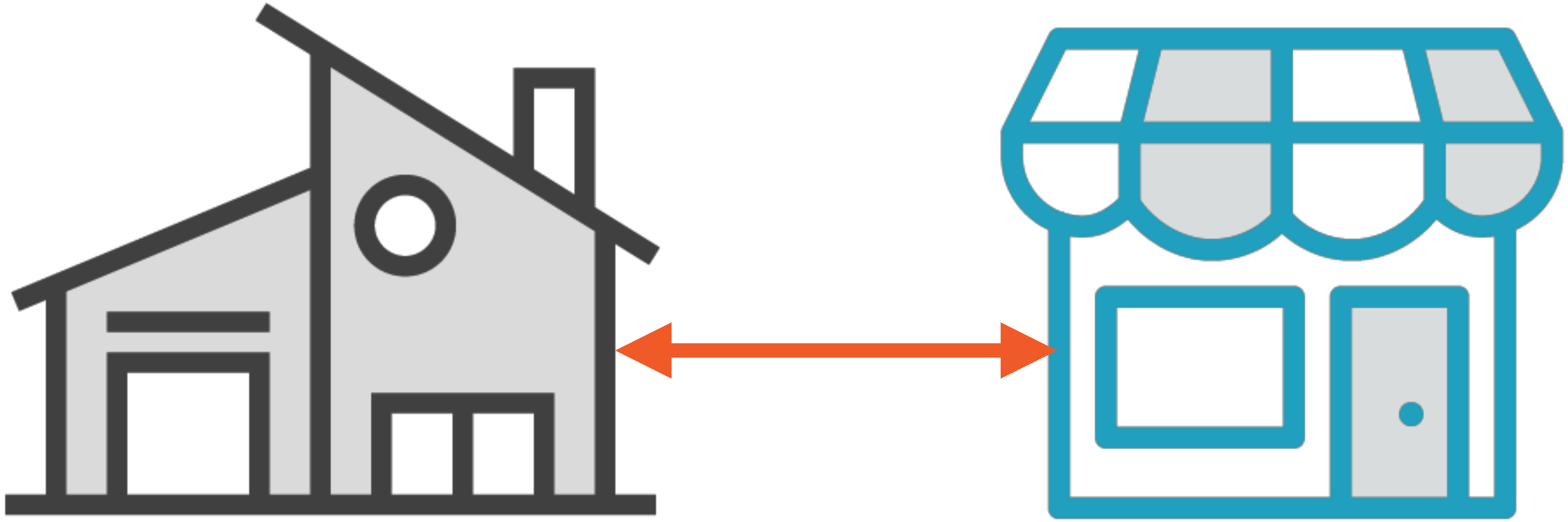
How do we calculate neighbors of a sample?

Nearest Neighbors Classification



Distance measures

Nearest Neighbors Classification



Euclidean distance, Hamming distance, Manhattan distance

Nearest Neighbors Classification

K-nearest-neighbors Classification

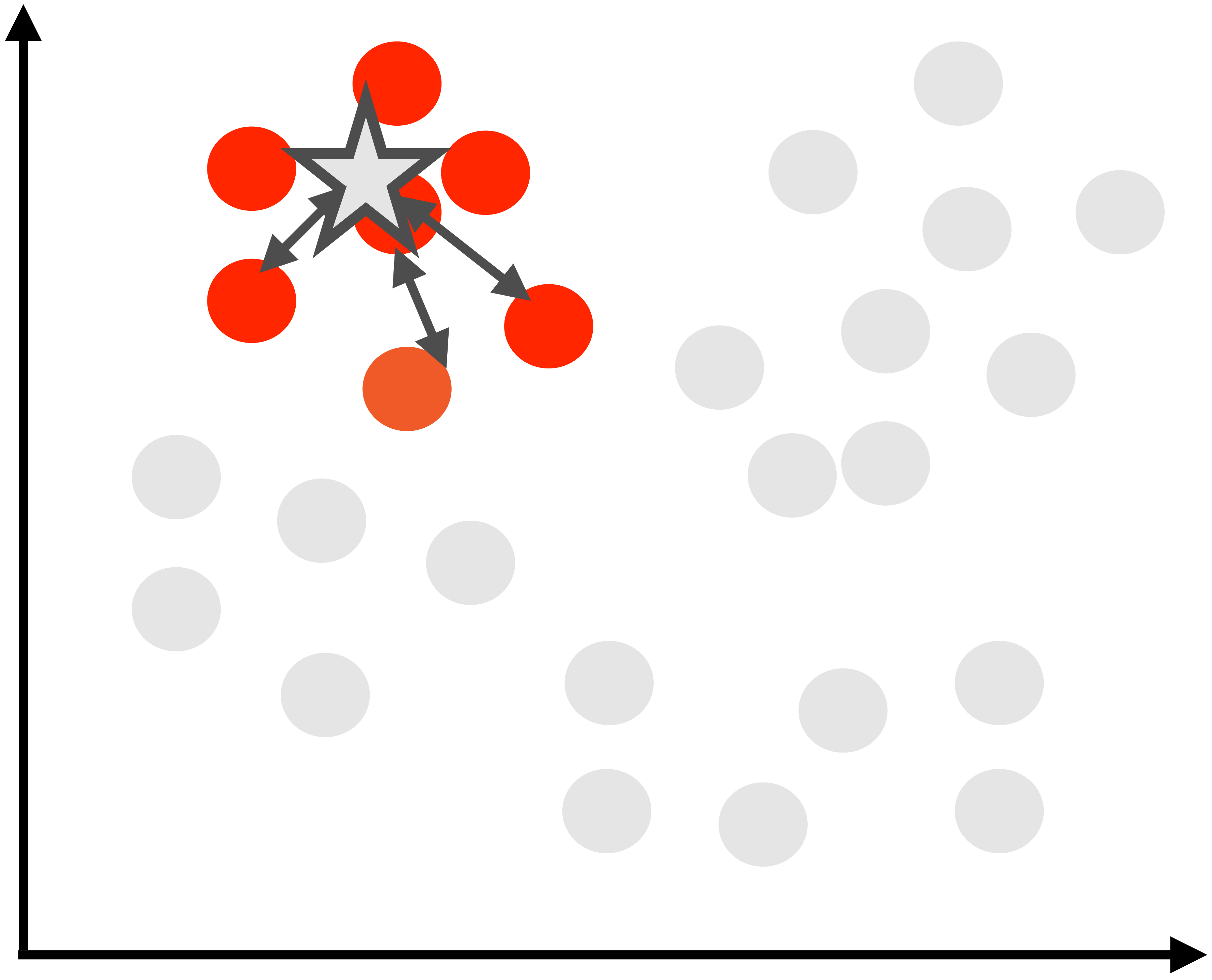
Radius Neighbors Classification

Nearest Neighbors Classification

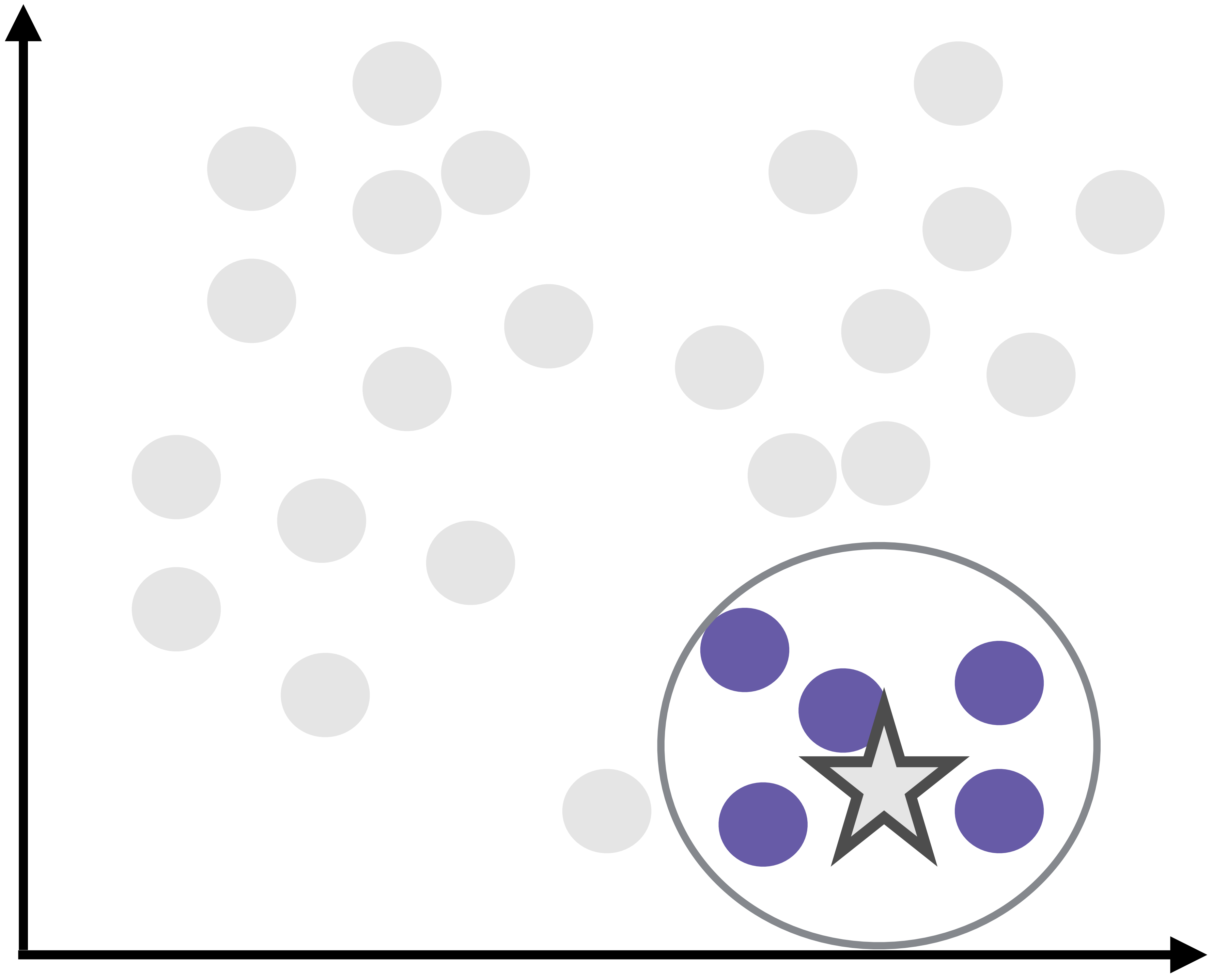
Voting among K nearest neighbors

Voting among all neighbors within radius

K-nearest-neighbors



Radius Neighbors



Demo

Classification using Nearest Neighbors

Decision Trees for Classification

Jockey or Basketball Player?

Jockeys

Tend to be light to meet horse carrying limits

Basketball Players

Tend to be tall, strong and heavy

Jockey or Basketball Player?

Intuitively know

Jockeys tend to be light

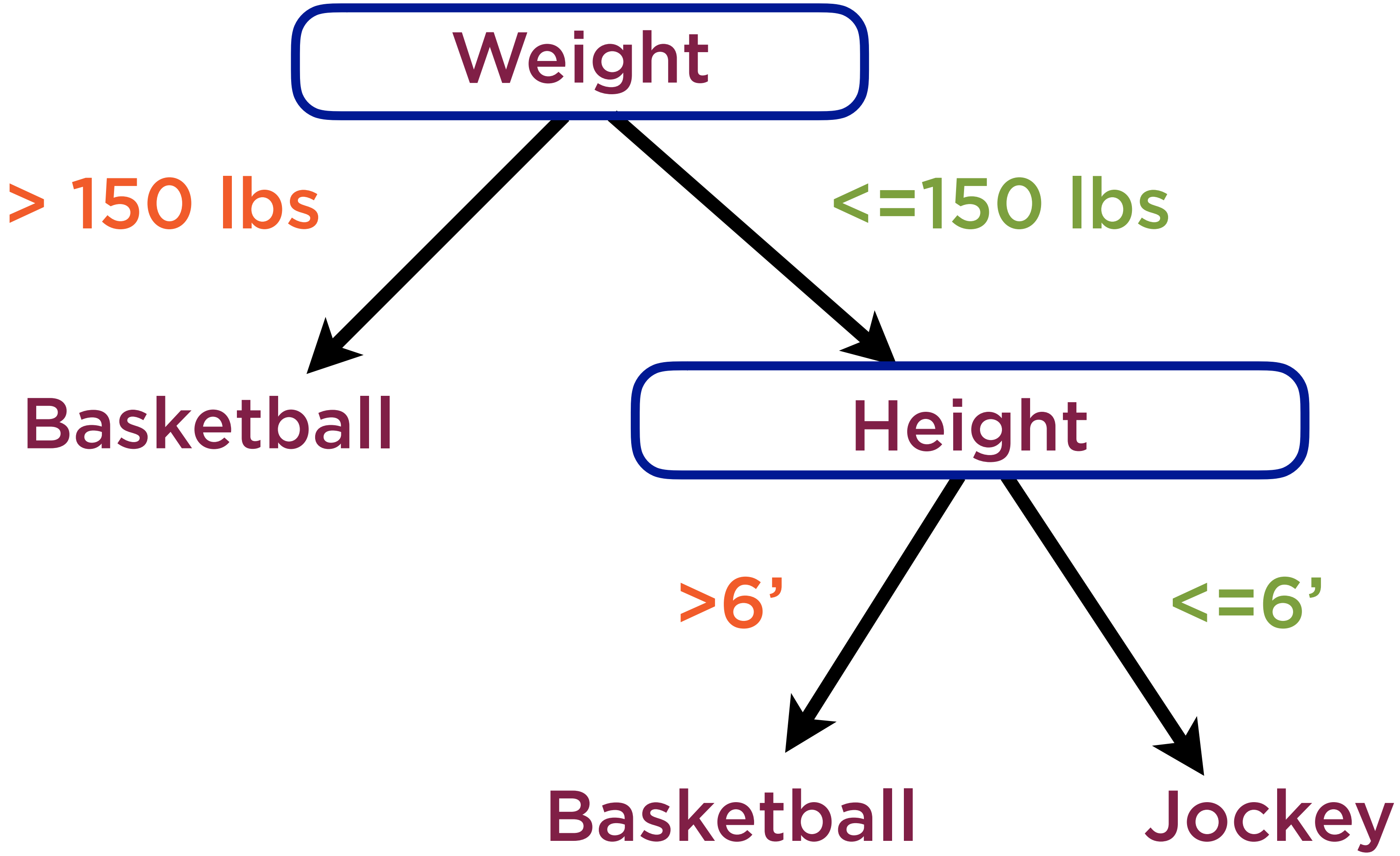
And not very tall

Basketball players tend to be tall

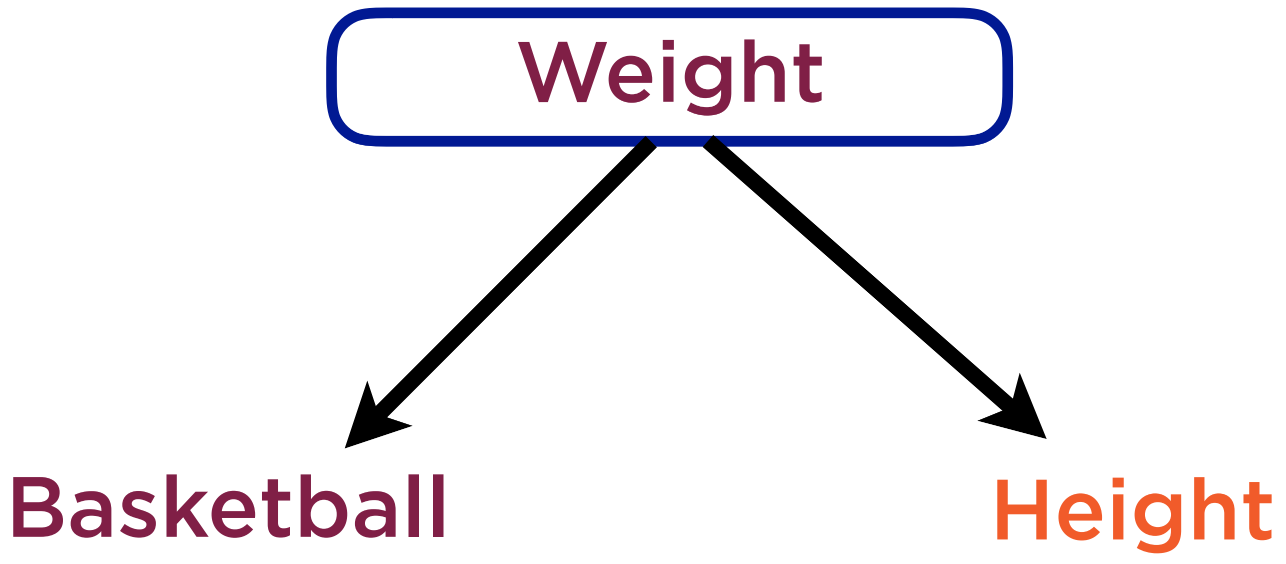
And also quite heavy

Decision trees set up a tree structure on training data which helps make **decisions** based on **rules**

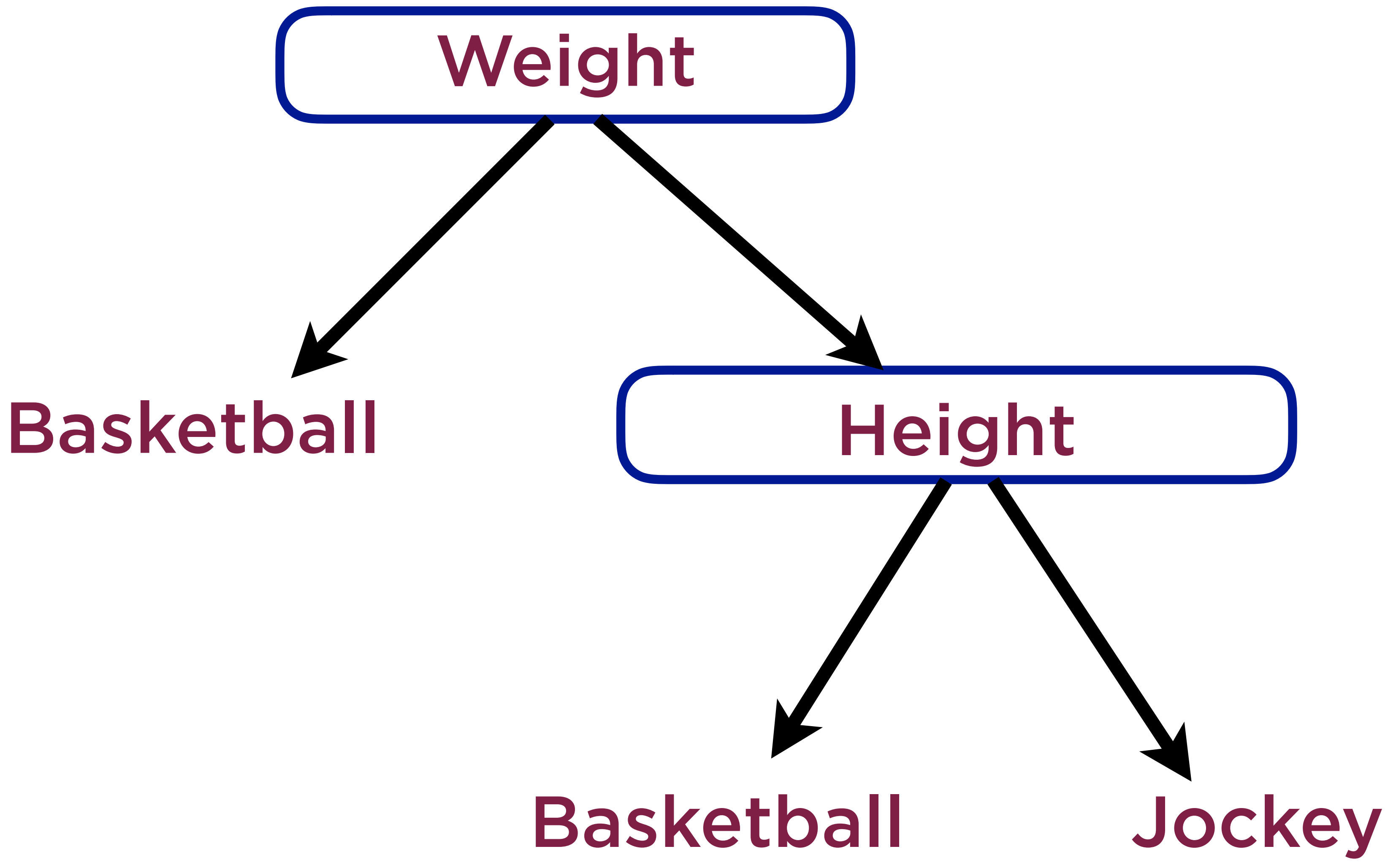
Fit Knowledge into Rules



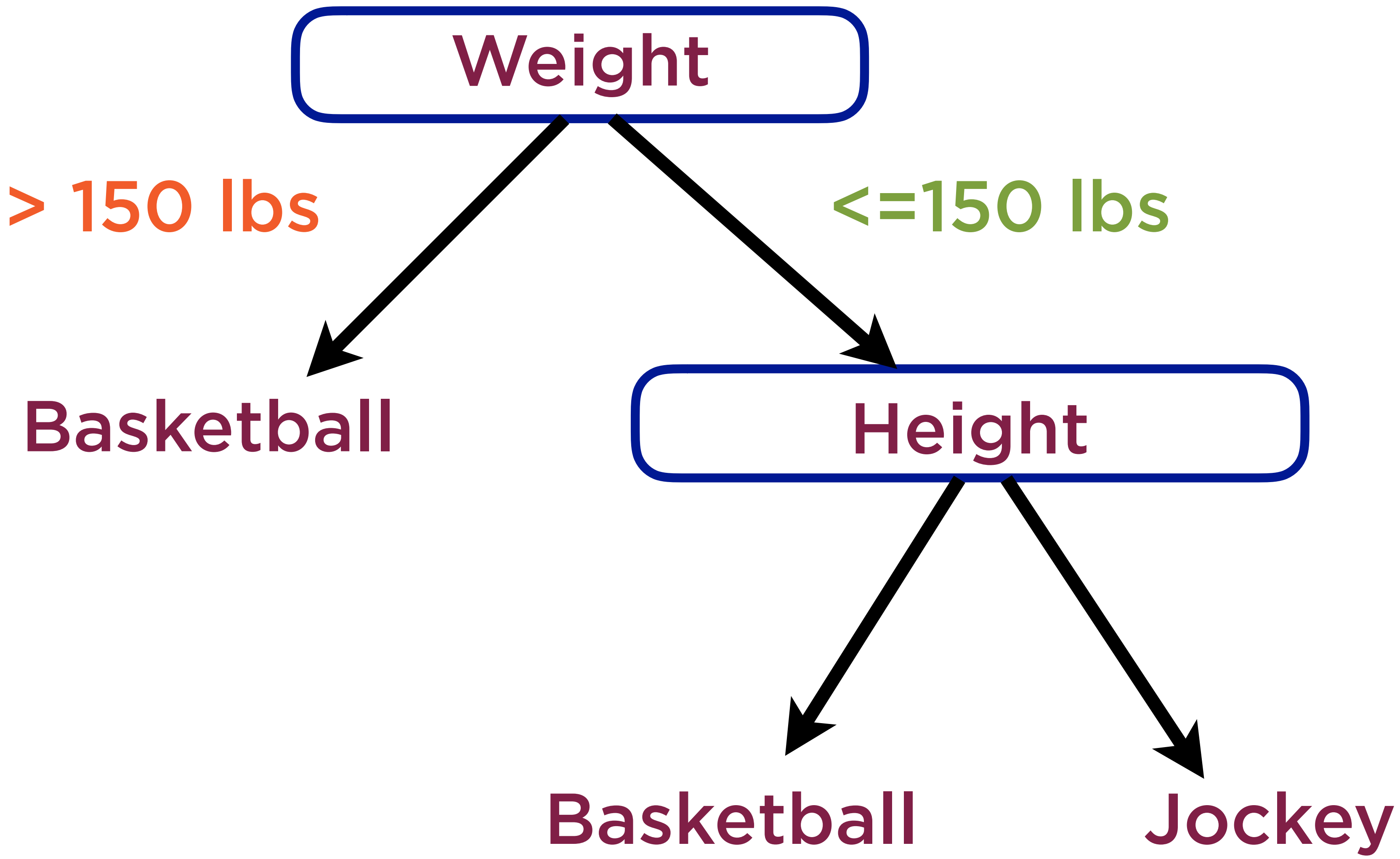
Decision Based on Weight



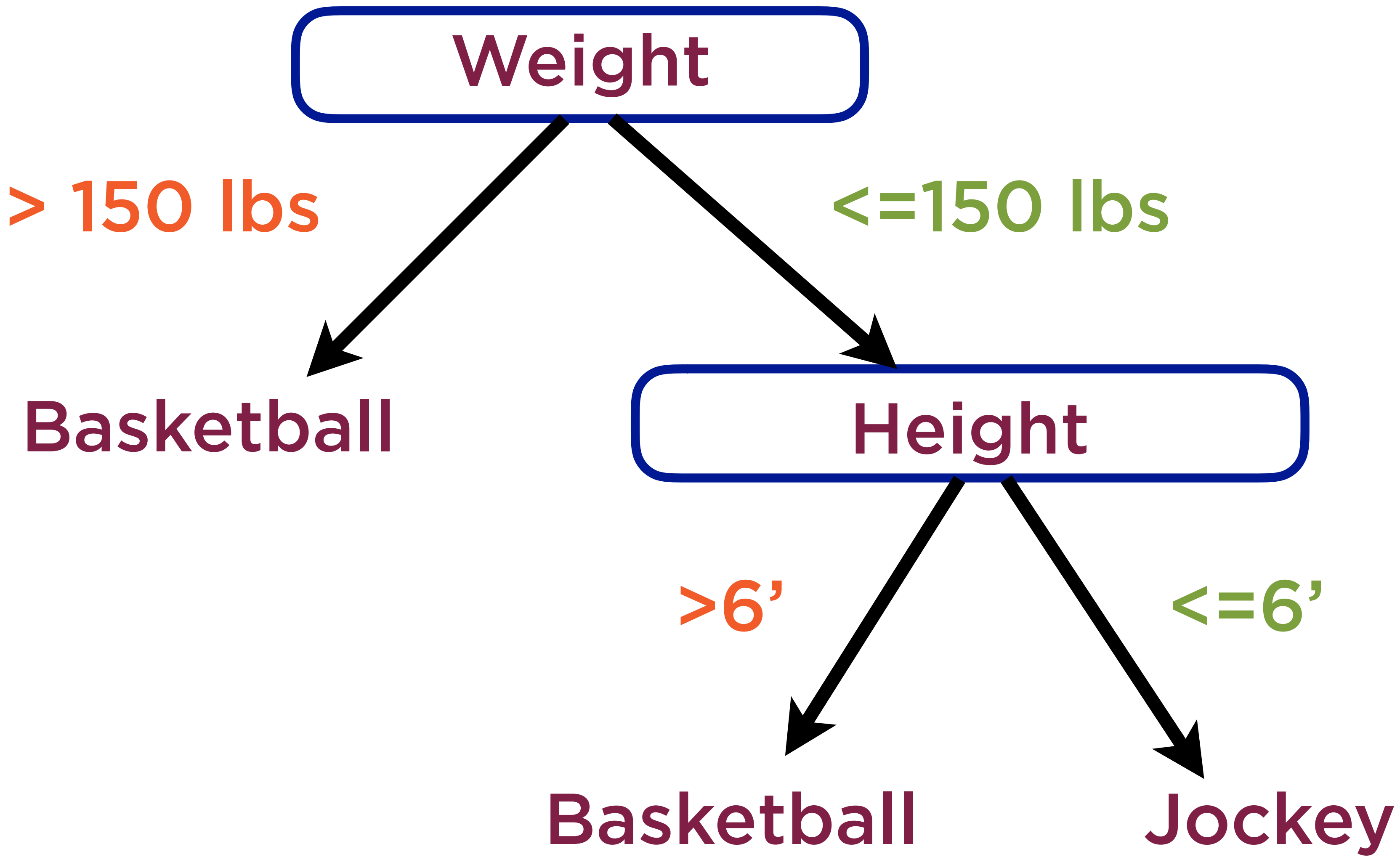
Decision Based on Height



Fit Knowledge into Rules



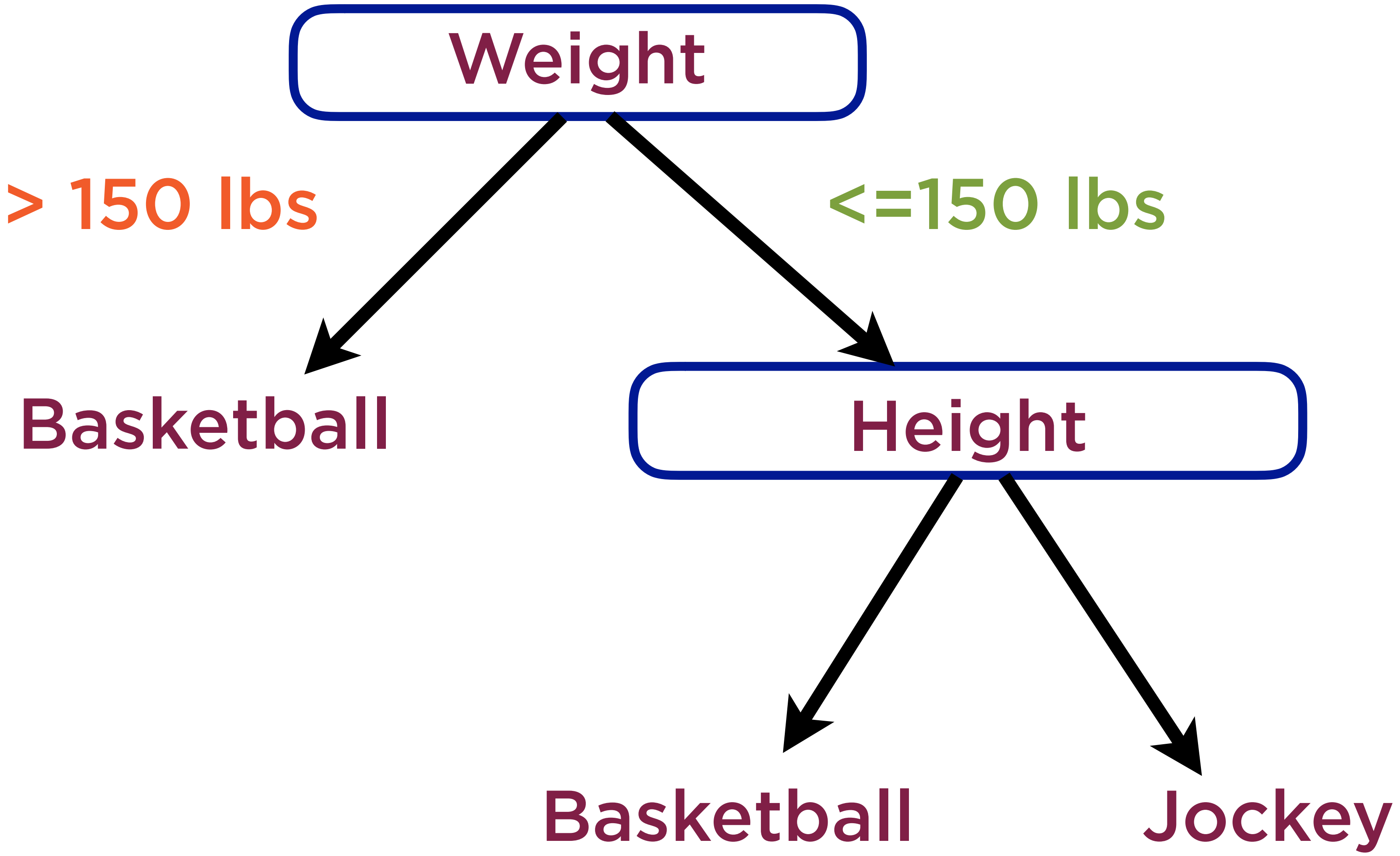
Fit Knowledge into Rules



Decision Tree

Fit knowledge into rules

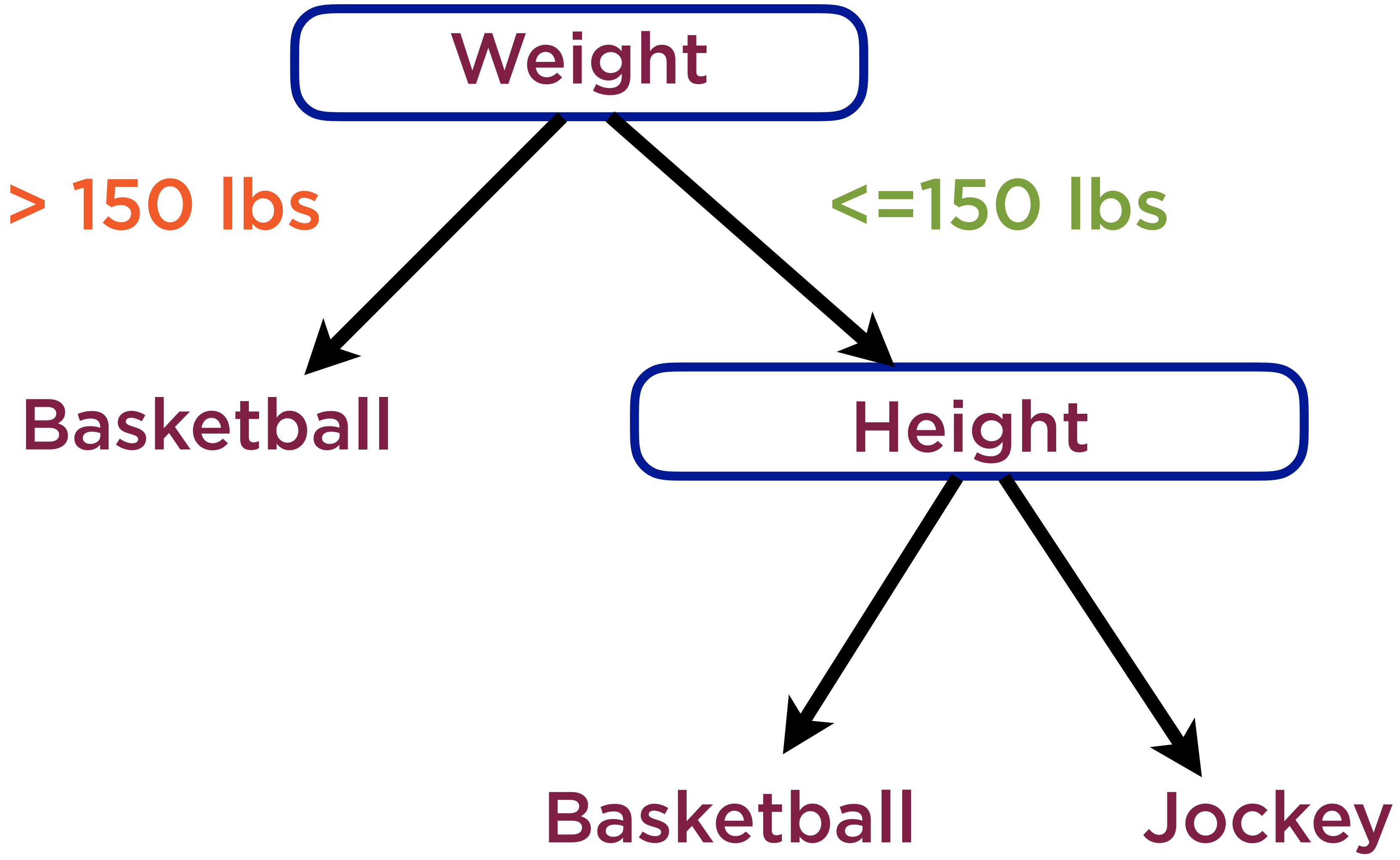
Each rule involves a threshold



Decision Tree

Order of decision variables matters

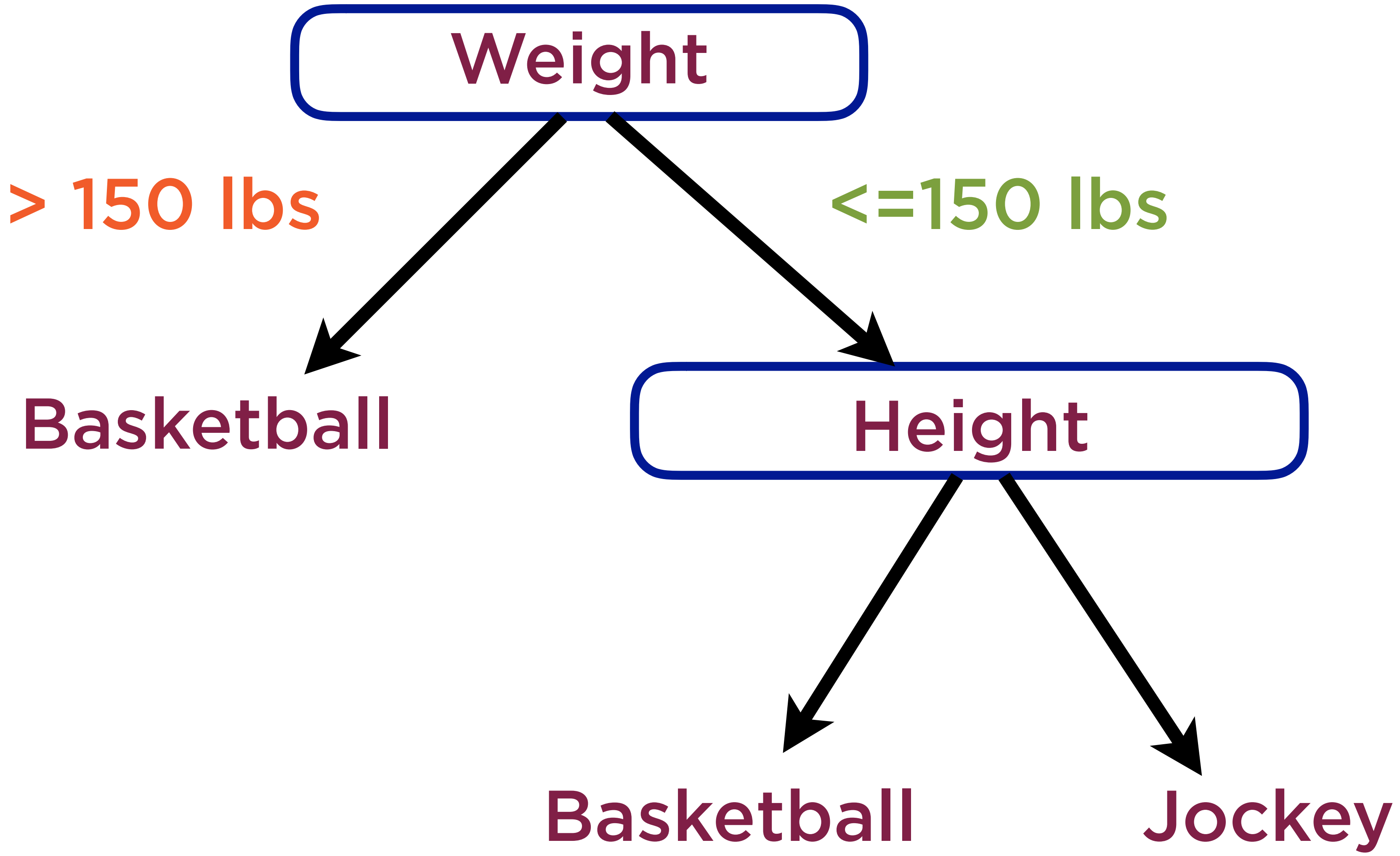
Rules and order found using ML



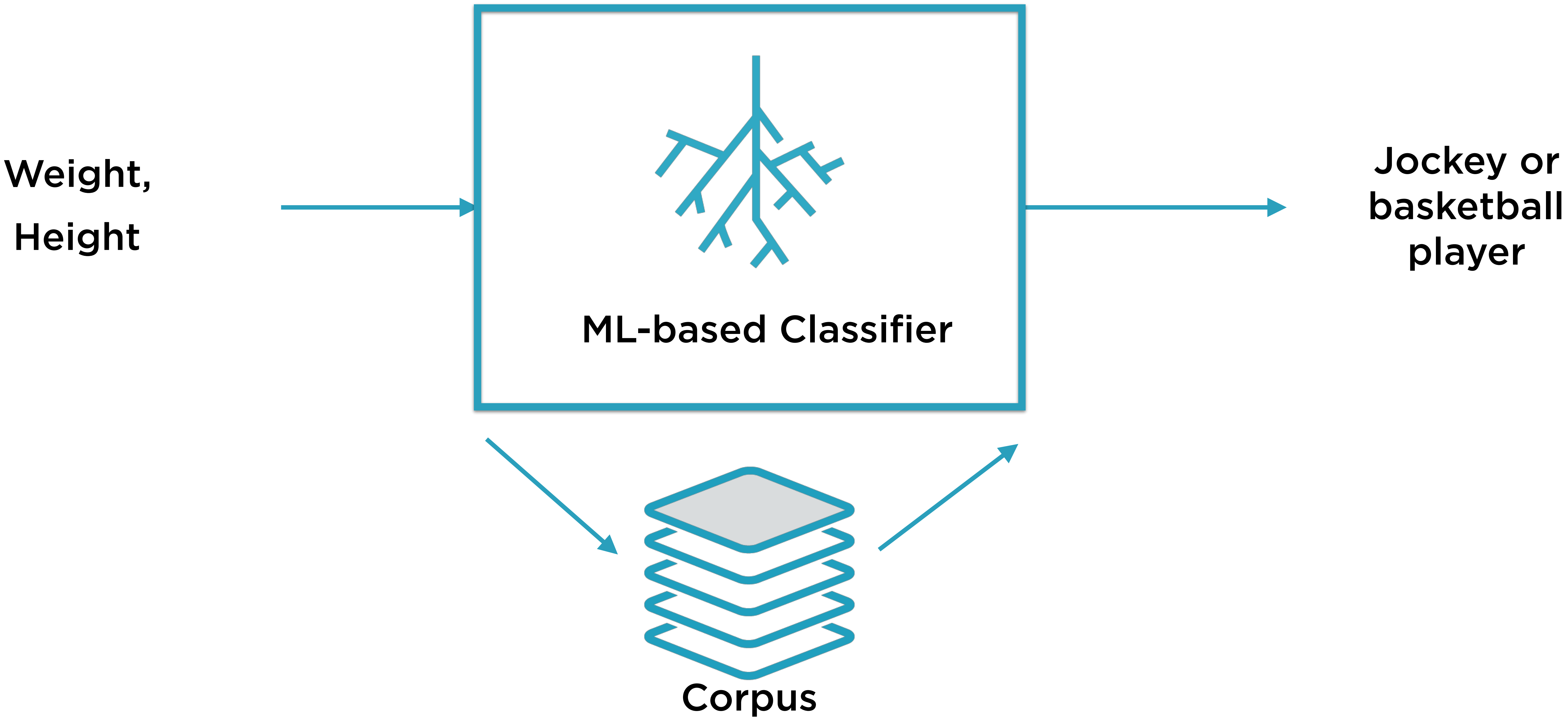
Decision Tree

"CART"

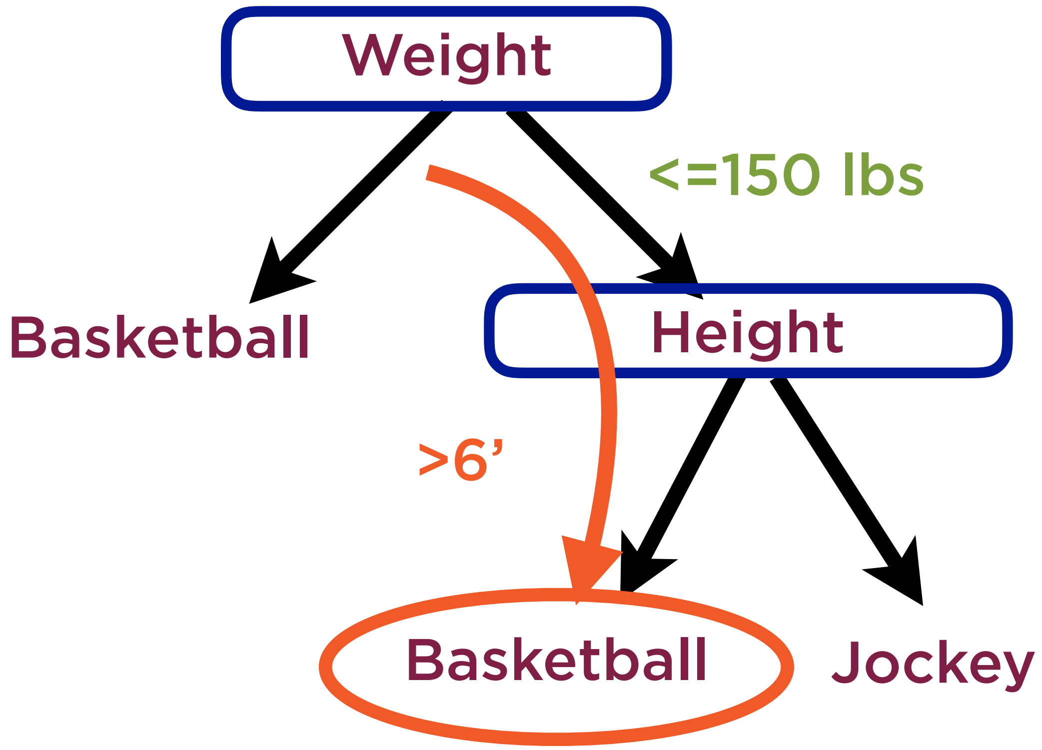
Classification And Regression Tree



Decision Trees for Classification



Decision Trees for Classification



Traverse tree to find right node

Return most frequent label of all training data points in that node

Demo

Classification using Decision Trees

Naive Bayes' for Classification Problems

Swoosh as a Binary Classification Problem

Runner

Police Officer

Classify a person who jogs past you on the street

A Priori Probabilities

Items	Occurence
Runners	9
Police officers	1
Total	10

Observation 1: Today is the city marathon, more runners than police officers out on the streets

A Priori Probabilities

P(Runner) = 9/10

P(Police Officer) = 1/10

These are ***a priori* probabilities:** before anything specific about the person is known

Conditional Probabilities

Handcuffs

Walkie-Talkie

Running Shoes

Observation 2: Specific items appear more often with one category than with the other

Conditional Probabilities

Item	Occurrences with Police Officers	Occurrences with Runners
Handcuffs	6	0
Running Shoes	2	8
Gun	9	0
Badge	8	0
Walkie-Talkie	8	3

Upon Closer Examination

Handcuffs

Badge

The person that zipped past carried these two items

Applying Bayes' Theorem

P(Runner/ Handcuffs,Badge)	=	Probability that a person carrying handcuffs and a badge is a runner

Step 1: Find probability that this person is a runner

Applying Bayes' Theorem

P(Police Officer/ Handcuffs,Badge) = Probability that a person carrying handcuffs and a badge is a police officer

Step 2: Find probability that this person is a police officer

Applying Bayes' Theorem

Compare

P(Police Officer/
Handcuffs,Badge)

and

P(Runner/
Handcuffs,Badge) =

Step 3: Pick the label with the higher probability

Jogger Is a Police Officer

P(Police Officer/ Handcuffs,Badge) > **P(Runner/ Handcuffs,Badge) =**

Jogger Is a Marathon Runner

P(Police Officer/ Handcuffs,Badge) < P(Runner/ Handcuffs,Badge) =

Naive Bayes' makes naive (strong) assumptions about independence of features

Demo

Classification using Naive Bayes

Summary

scikit-learn support for classification models

Discriminant Analysis

Stochastic Gradient Descent

Support Vector Machines

Nearest Neighbors

Decision Trees

Naive Bayes