Classification and clustering

Neural data science with Python

Heike Stein
17/11/2025

Classification and Clustering: Overview

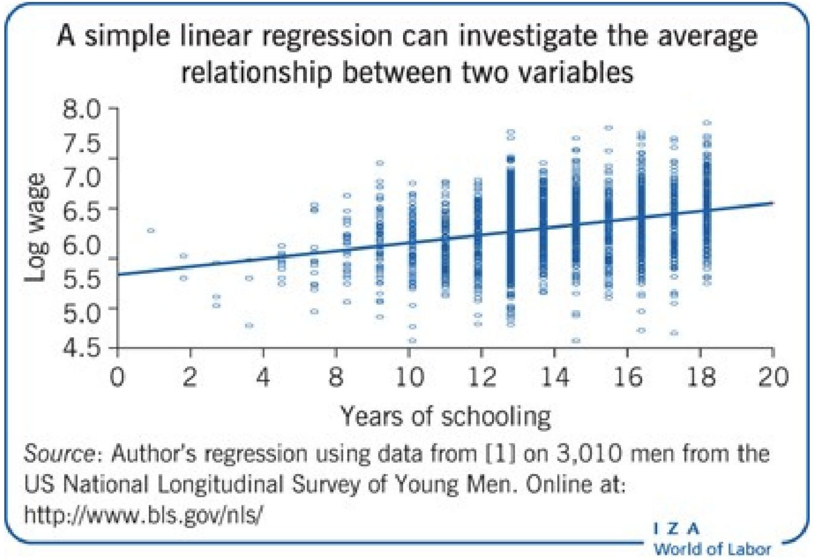
Classification

- Logistic regression
- Support vector machines

Clustering

- k-means

Regression analysis

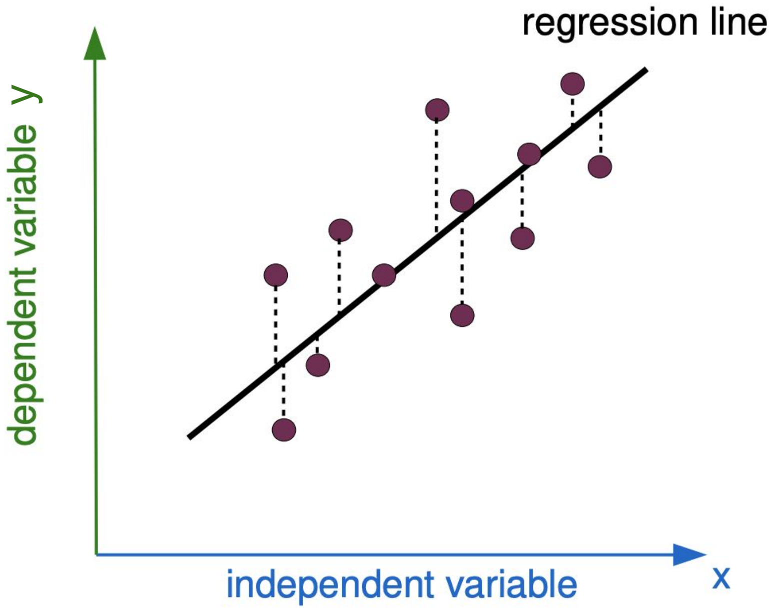

A simple linear regression can investigate the average relationship between two variables

Source: Author's regression using data from [1] on 3,010 men from the US National Longitudinal Survey of Young Men. Online at: http://www.bls.gov/nls/

I Z A World of Labor

→ linear mapping from predictor ("independent") variable(s) x (e.g. years of schooling) to outcome ("dependent") variable y (e.g. wage)

Regression analysis



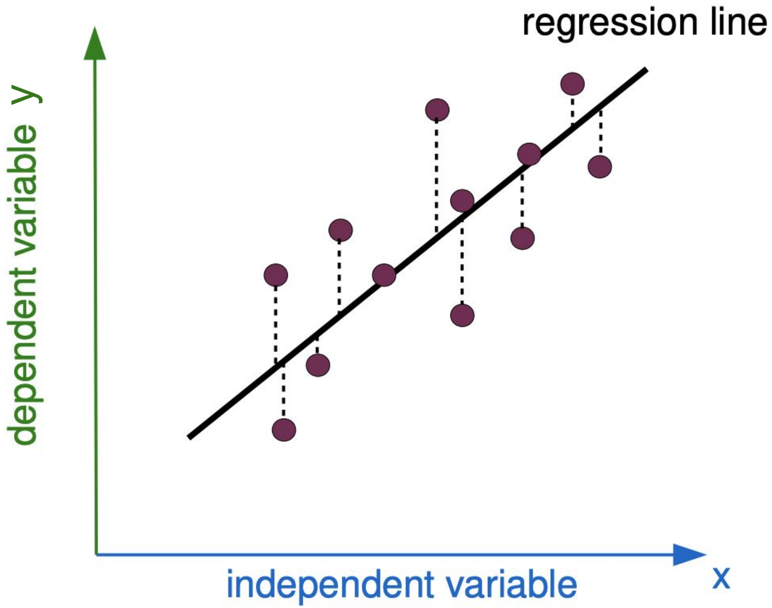
→ linear mapping from predictor (“independent”) variable(s) x (e.g. years of schooling) to outcome (“dependent”) variable y (e.g. wage)

$$y = \boxed{\beta_0} + \boxed{\beta_1} x_1 + \boxed{\varepsilon}$$

weights (“parameters”)

errors (“residuals”)

Regression analysis

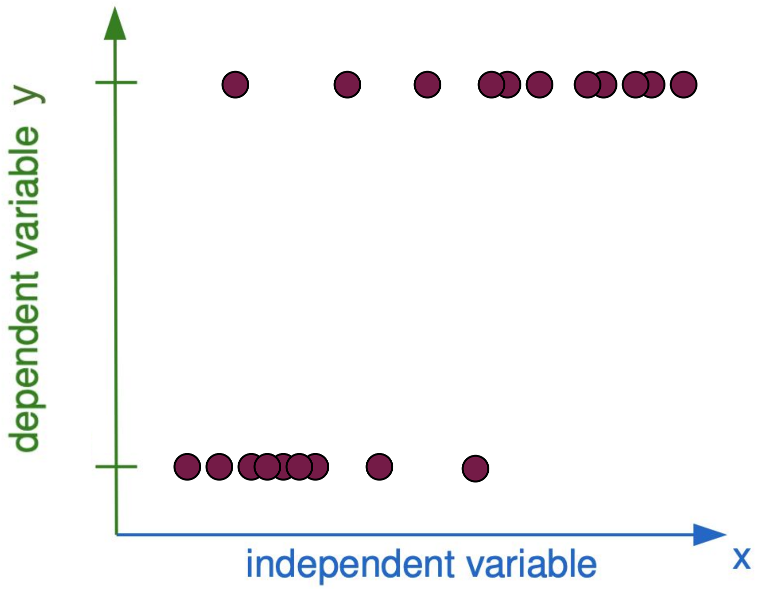


→ linear mapping from predictor (“independent”) variable(s) x (e.g. years of schooling) to outcome (“dependent”) variable y (e.g. wage)

$$y = \boxed{\beta_0} + \boxed{\beta_1} x_1 + \boxed{\varepsilon}$$

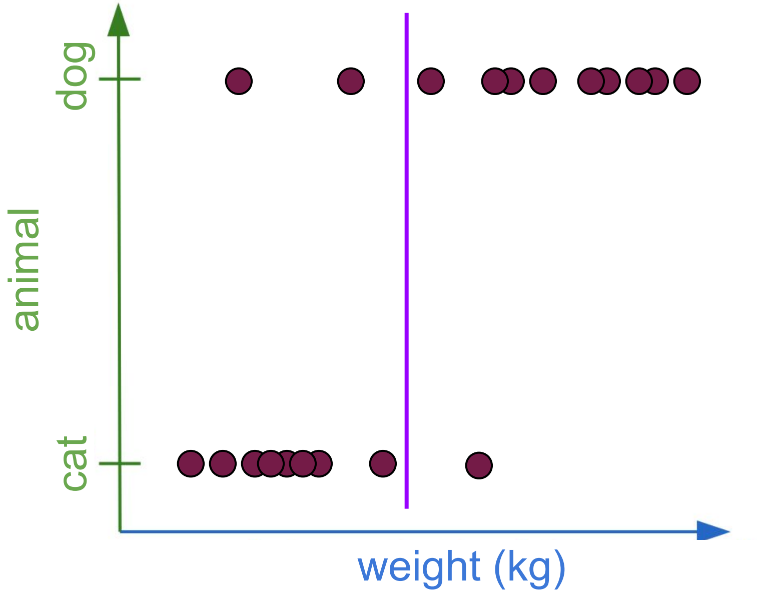
Weights are chosen so as to minimize errors (model fitting)

Sometimes, outcomes are categorical



Given knowledge of x, what is the most likely category of y?

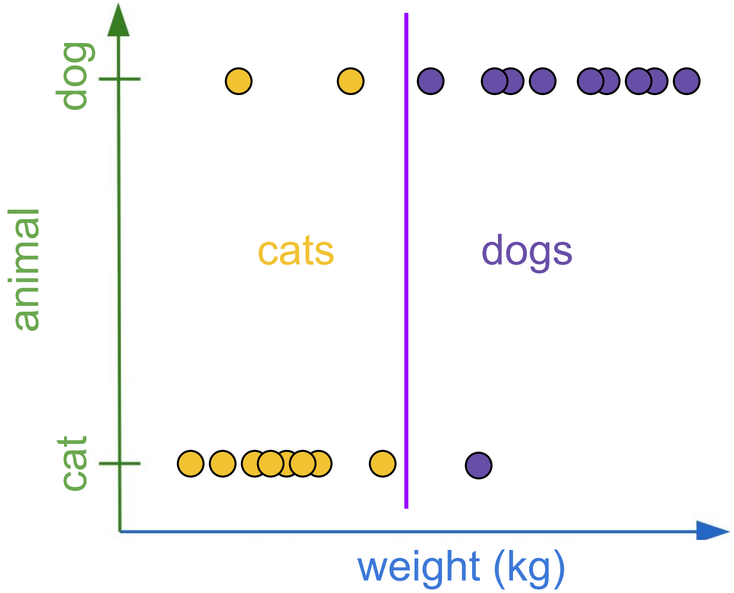
Sometimes, outcomes are categorical



Given knowledge of x, what is the most likely category of y?

We want to find the value of x that best separates cats vs. dogs:
The decision boundary

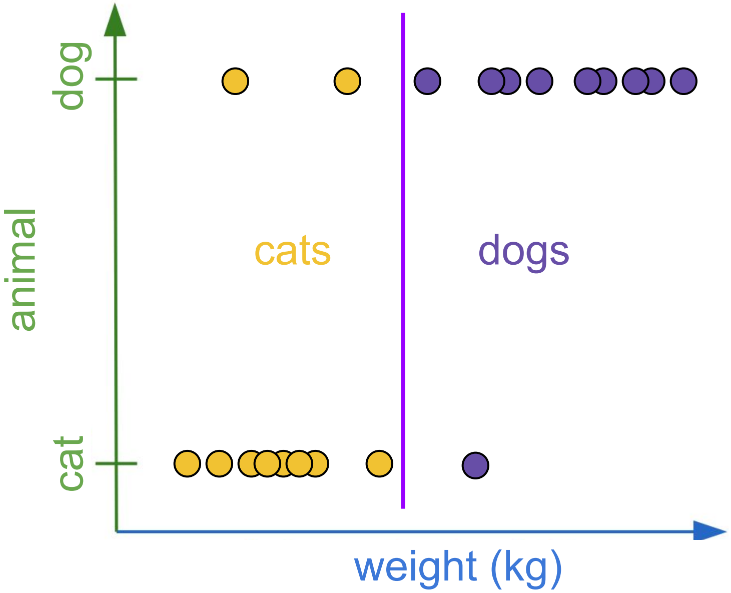
Classification: Predicting categorical values



Given knowledge of x, what is the most likely category of y?

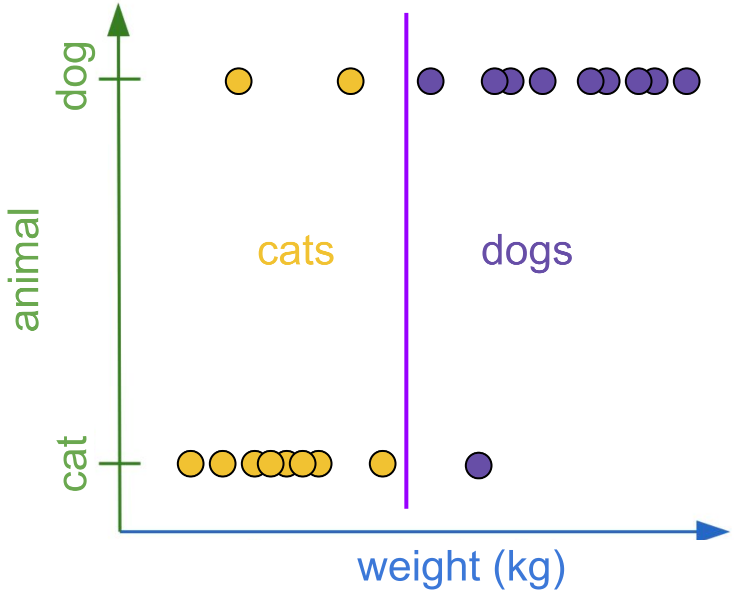
We want to find the value of x that best separates cats vs. dogs: The decision boundary

Classification: Predicting categorical values



How do we find a good decision boundary?

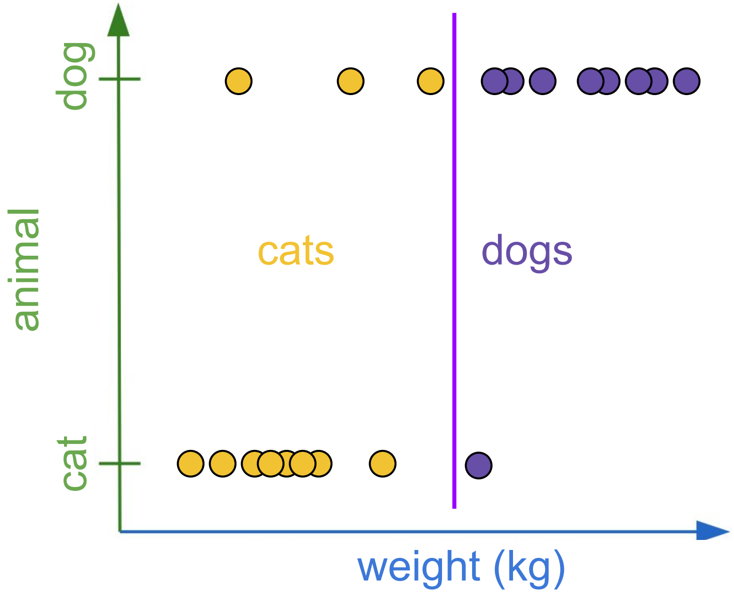
Classification: Predicting categorical values



How do we find a good decision boundary?

It should minimize the number of misclassifications.

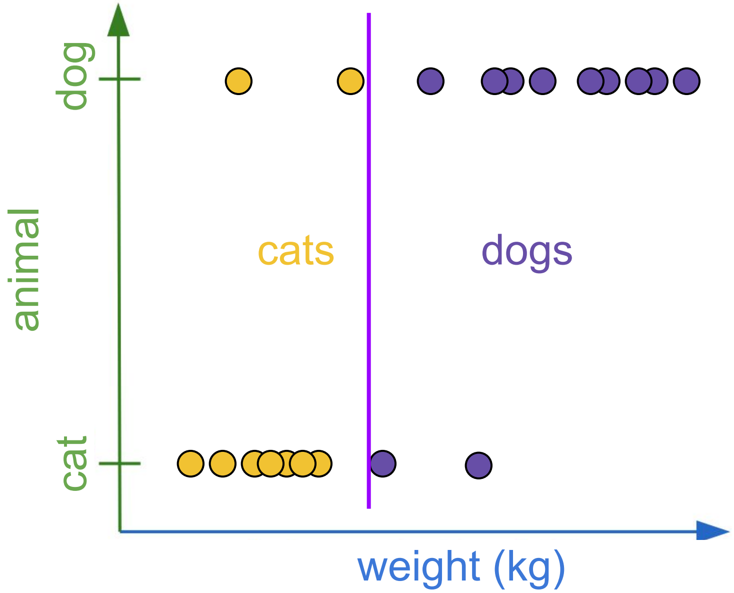
Classification: Predicting categorical values



How do we find a good decision boundary?

It should minimize the number of misclassifications.

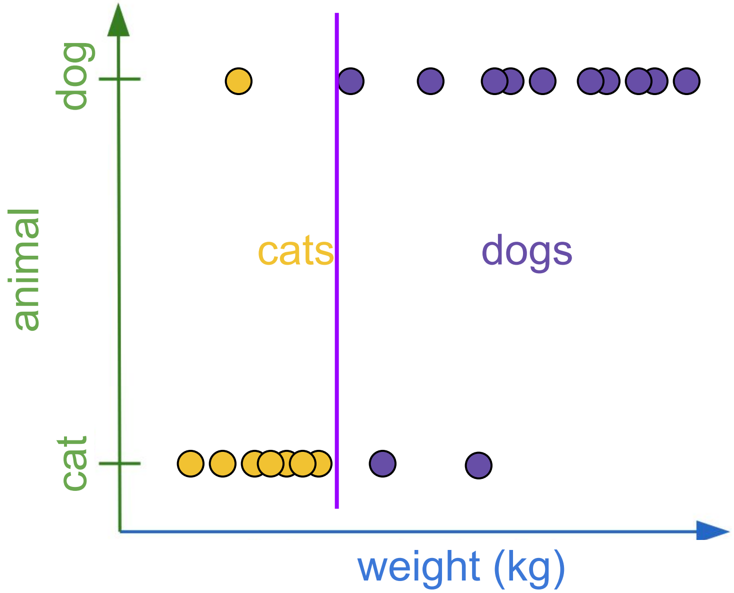
Classification: Predicting categorical values



How do we find a good decision boundary?

It should minimize the number of misclassifications.

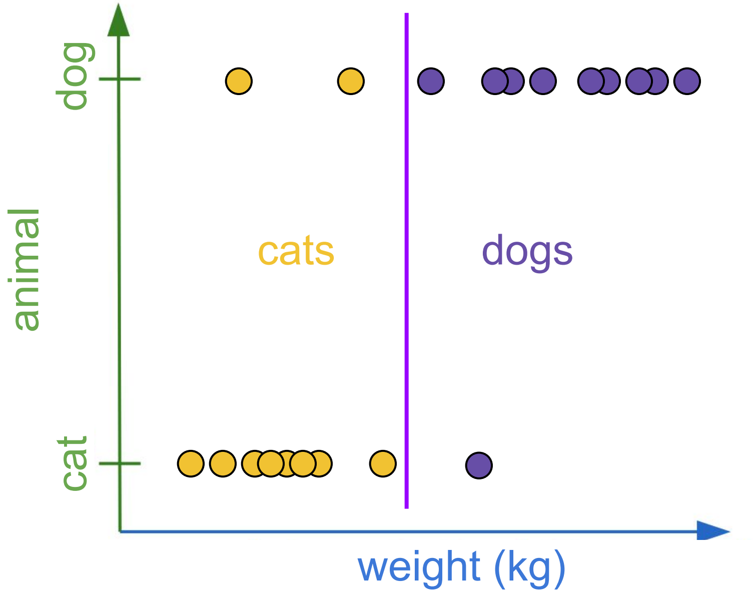
Classification: Predicting categorical values



How do we find a good decision boundary?

It should minimize the number of misclassifications.

Classification: Predicting categorical values



How do we find a good decision boundary?

It should minimize the number of misclassifications.

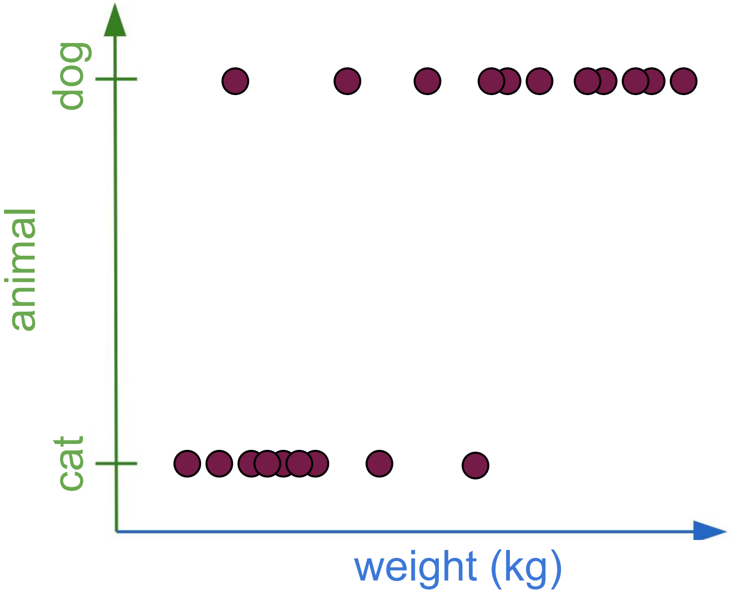
Which solution is found depends on the classification method.

Classification and Clustering: Overview

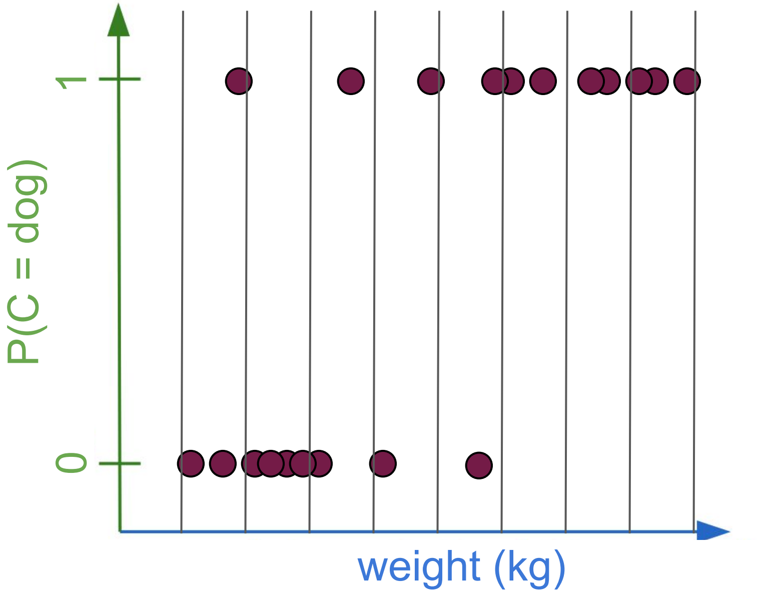
Classification

1. Logistic regression
2. Support vector machines

Classification: Logistic regression

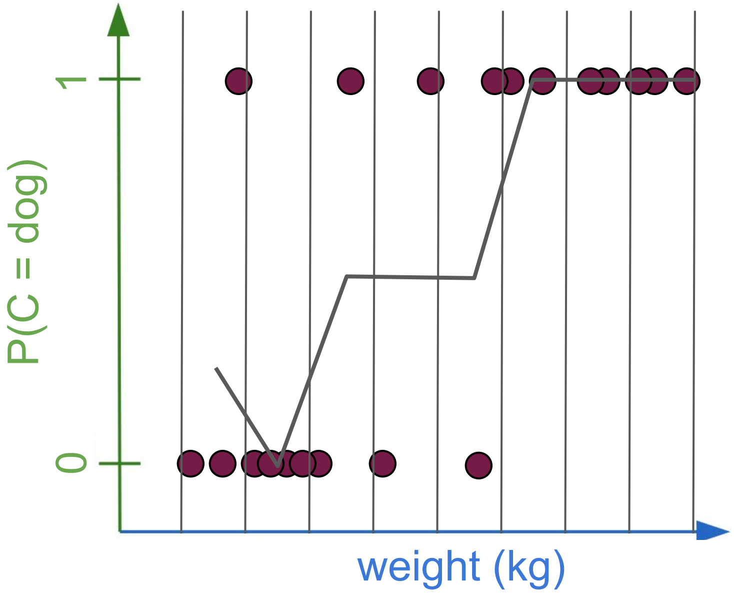


Classification: Logistic regression



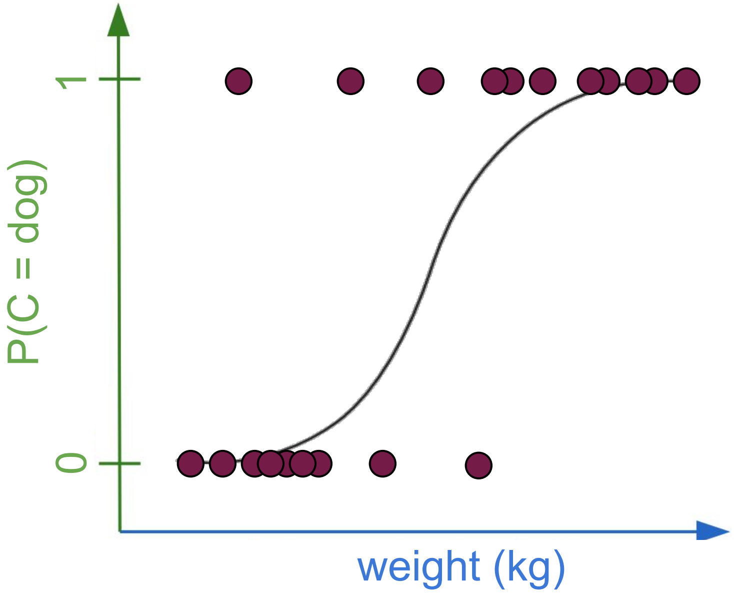
Idea: For each value on x, we can calculate the probability that the animal is a dog: $P(C = \text{"dog"})$

Classification: Logistic regression



Idea: For each value on x, we can calculate the probability that the animal is a dog: $P(C = \text{"dog"})$

Classification: Logistic regression

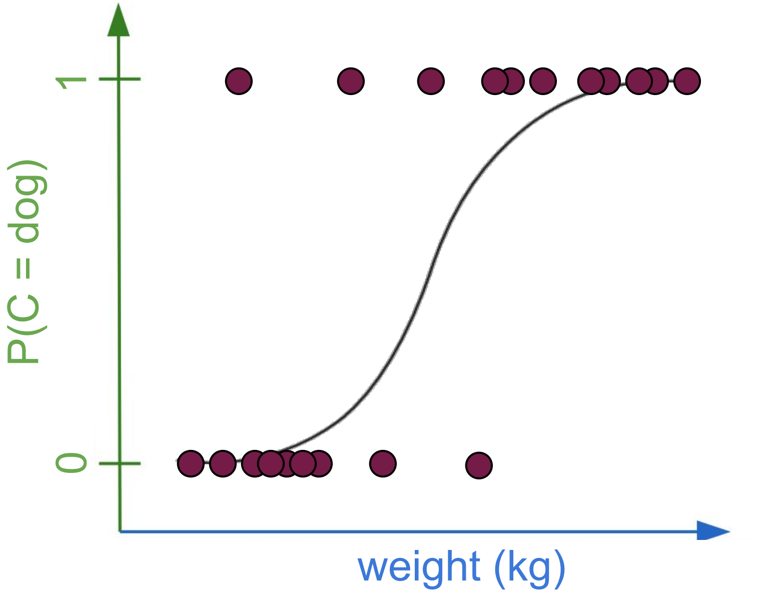


Logistic regression fits the weights of a generalized linear model:

$$P(C = \text{"dog"}) = f(\beta_0 + \beta_1 x_1)$$

with sigmoidal link function $f(x) = \dfrac{e^x}{1 + e^x}$

Classification: Logistic regression



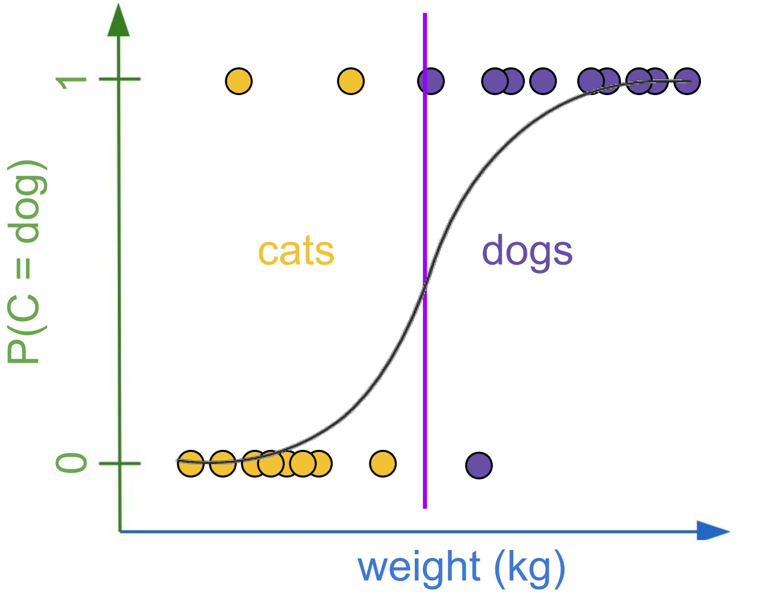
Logistic regression fits the weights of a generalized linear model:

$$P(C = \text{"dog"}) = f(\beta_0 + \beta_1 x_1)$$

with sigmoidal link function $f(x) = \dfrac{e^x}{1 + e^x}$

$$\Leftrightarrow \quad P(C = \text{"dog"}) = \frac{e^{(\beta_0 + \beta_1 x_1)}}{1 + e^{(\beta_0 + \beta_1 x_1)}}$$

Classification: Logistic regression



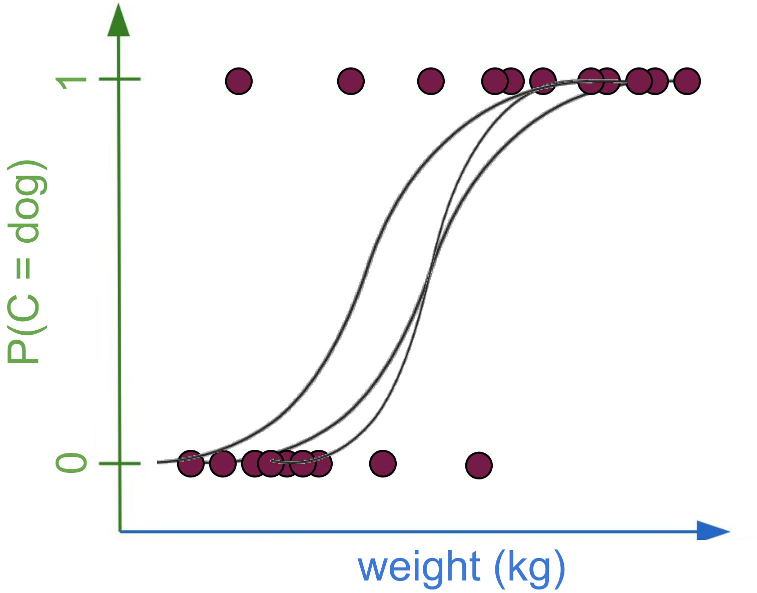
Logistic regression fits the weights of a generalized linear model:

$$P(C = \text{"dog"}) = f(\beta_0 + \beta_1 x_1)$$

with sigmoidal link function $f(x) = \dfrac{e^x}{1 + e^x}$

The decision boundary is the value of x for which P(C = “dog”) = 0.5

Classification: Logistic regression



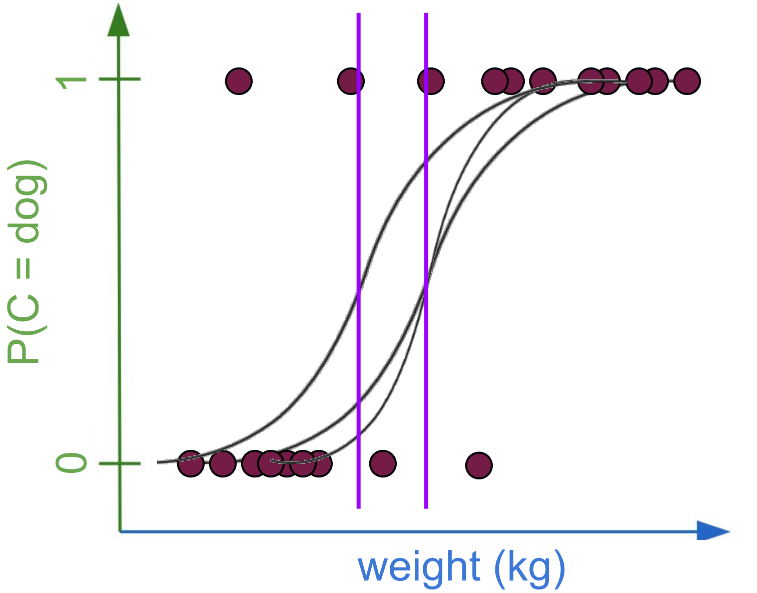
Logistic regression fits the weights of a generalized linear model:

$$P(C = \text{"dog"}) = f(\boxed{\beta_0} + \boxed{\beta_1} x_1)$$

with sigmoidal link function $f(x) = \dfrac{e^x}{1 + e^x}$

Weights affect slope and x-offset of the sigmoidal function.

Classification: Logistic regression



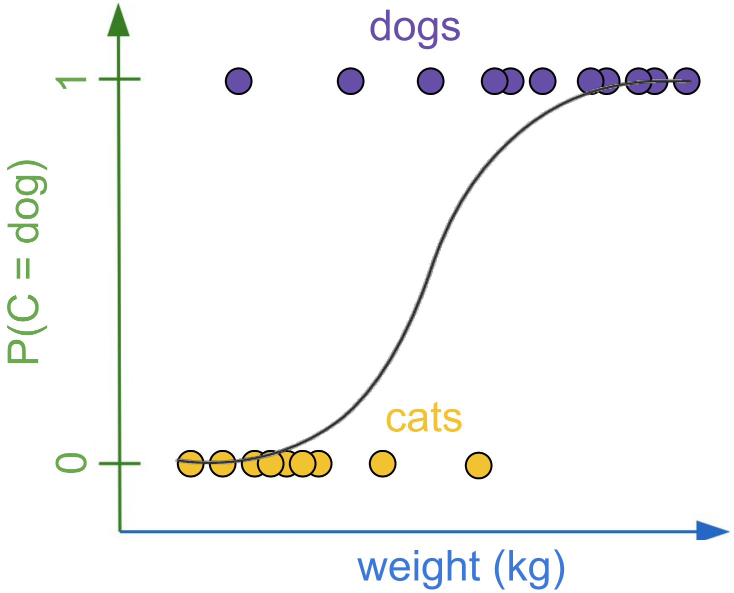
Logistic regression fits the weights of a generalized linear model:

$$P(C = \text{"dog"}) = f(\boxed{\beta_0} + \boxed{\beta_1} x_1)$$

with sigmoidal link function $f(x) = \dfrac{e^x}{1 + e^x}$

Weights affect slope and x-offset of the sigmoidal function. They thereby affect the decision boundary.

Classification: Logistic regression



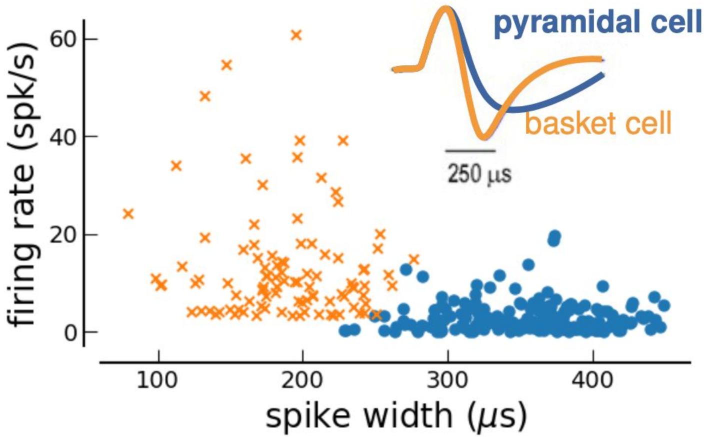
Logistic regression fits the weights of a generalized linear model:

$$P(C = \text{"dog"}) = f(\boxed{\beta_0} + \boxed{\beta_1} x_1)$$

with sigmoidal link function $f(x) = \dfrac{e^x}{1+e^x}$

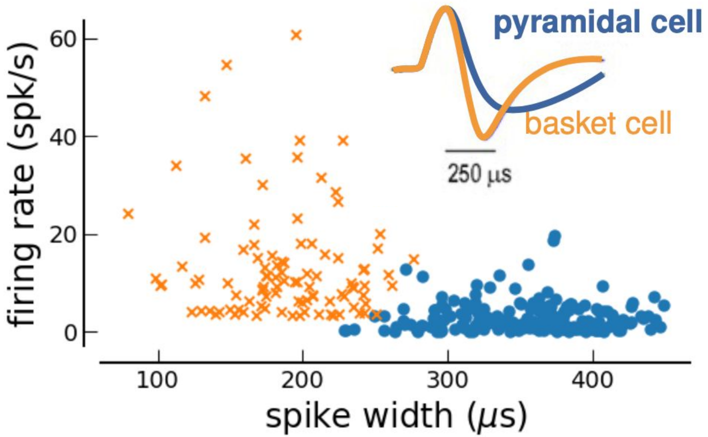
Weights are fitted by minimizing an error function between P(C = dog) and true labels: t = 1 if dog, t = 0 if cat

Classification: Logistic regression in 2D



Given the predictor variables x_1: spike width and x_2 : firing rate, what is the probability that we are looking at a basket cell (y: P(C = “basket cell”))?

Classification: Logistic regression in 2D



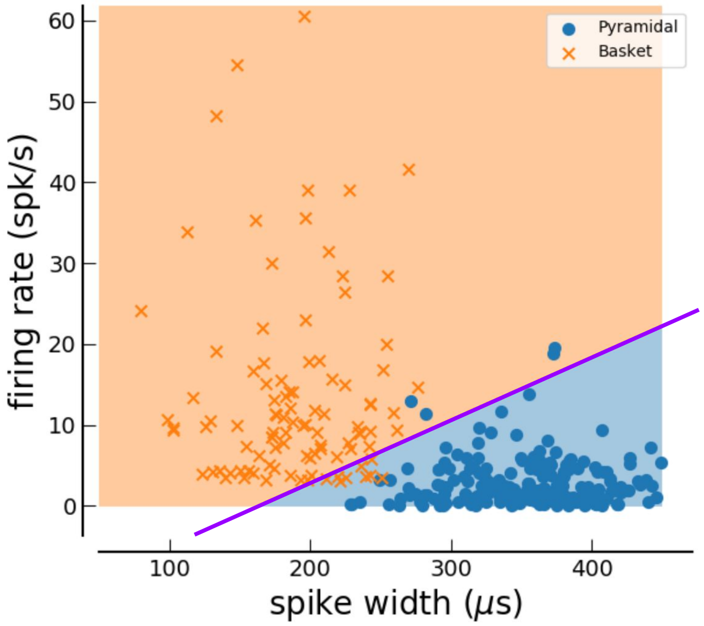
Given the predictor variables *x_1*: spike width and *x_2* : firing rate, what is the probability that we are looking at a basket cell (*y*: P(C = "basket cell"))?

Logistic regression fits the weights of a generalized linear model:

$$P(C = \text{"basket cell"}) = f(\boxed{\beta_0} + \boxed{\beta_1} x_1 + \boxed{\beta_2} x_2)$$

with sigmoidal function $f(x) = \dfrac{e^x}{1 + e^x}$

Classification: Logistic regression in 2D



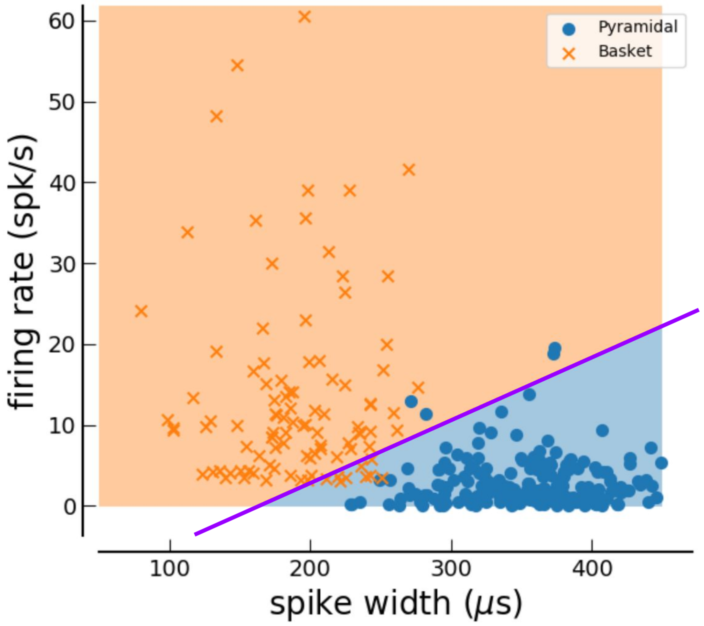
Logistic regression fits the weights of a generalized linear model:

$$P(C = \text{"basket cell"}) = f(\boxed{\beta_0} + \boxed{\beta_1} x_1 + \boxed{\beta_2} x_2)$$

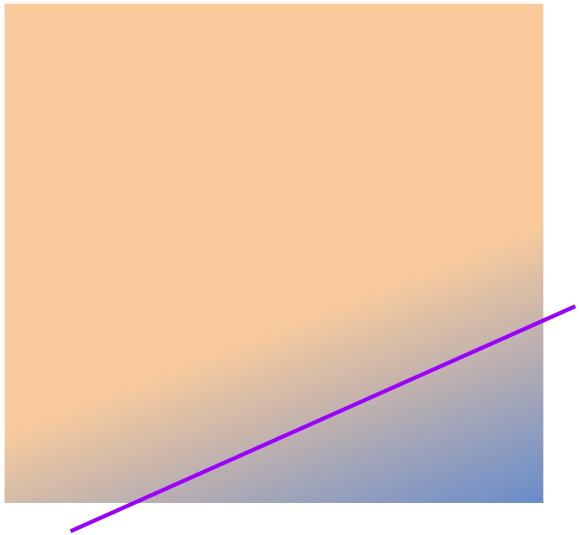
with sigmoidal function $f(x) = \dfrac{e^x}{1 + e^x}$

The decision boundary is now a 1D line in the 2D space for which P(C = “basket cell”) = 0.5

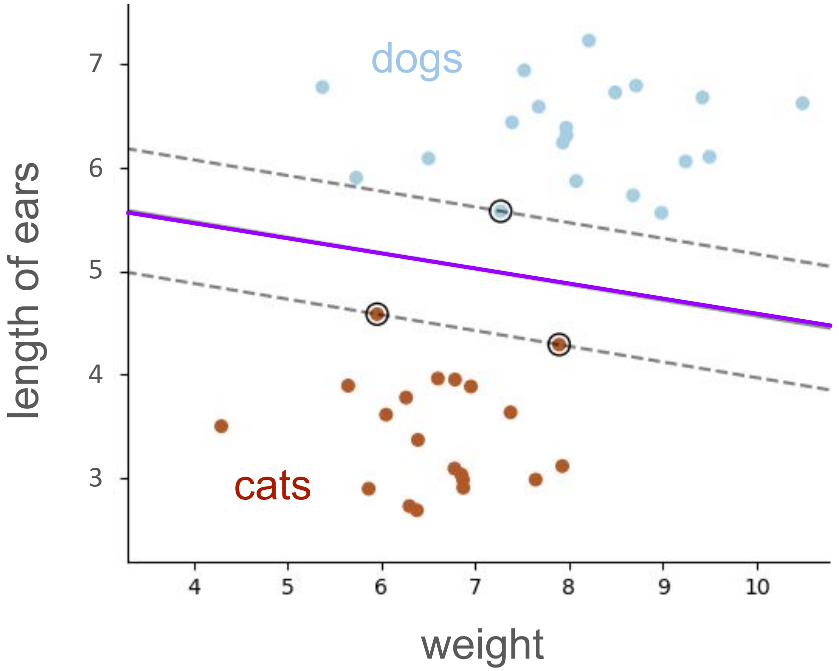
Classification: Logistic regression in 2D



Sigmoidal function in 2D

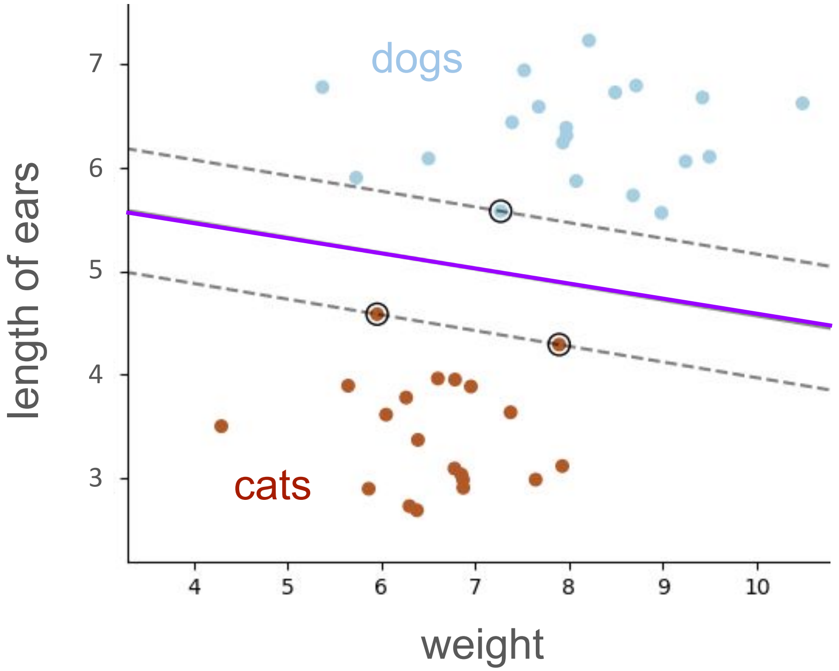


Classification: Support vector machines (SVM)



Again, we're looking for a decision boundary

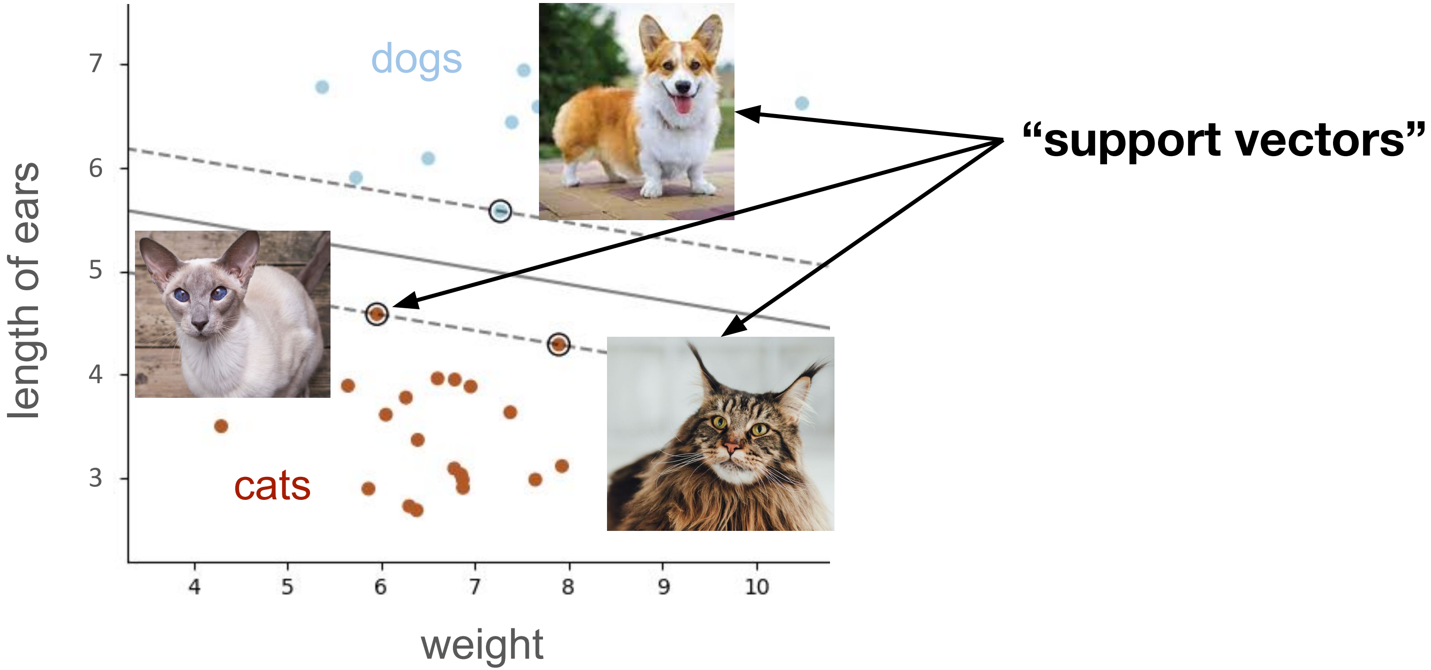
Classification: Support vector machines (SVM)



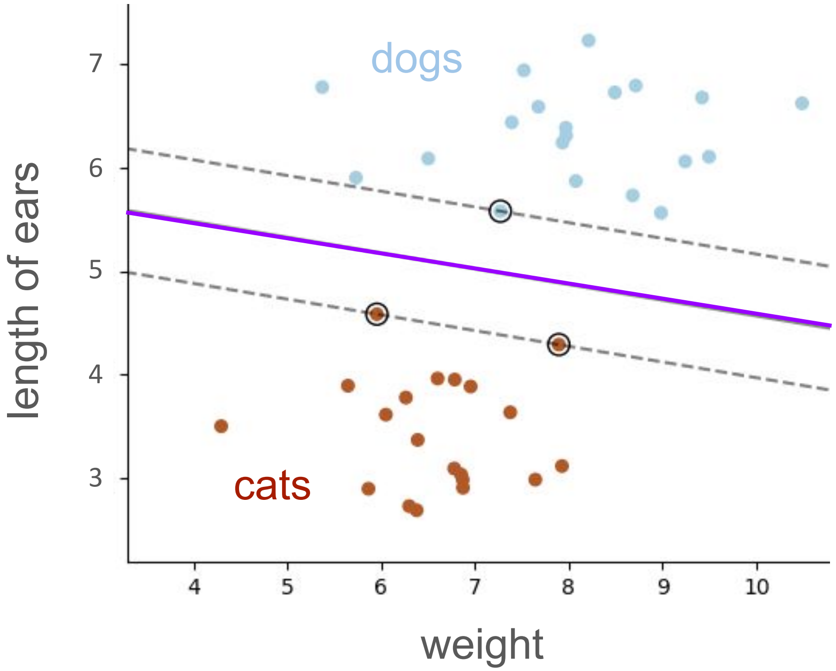
Again, we're looking for a decision boundary

This time, we define it as the line that is equally far away from the nearest data point of each class:
the **"support vectors"**

Classification: Support vector machines (SVM)

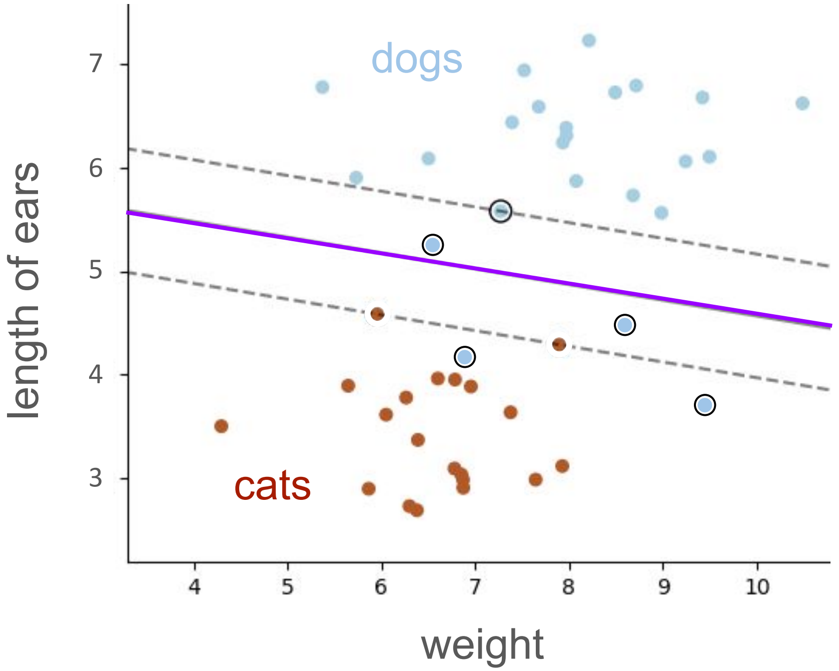


Classification: Support vector machines (SVM)



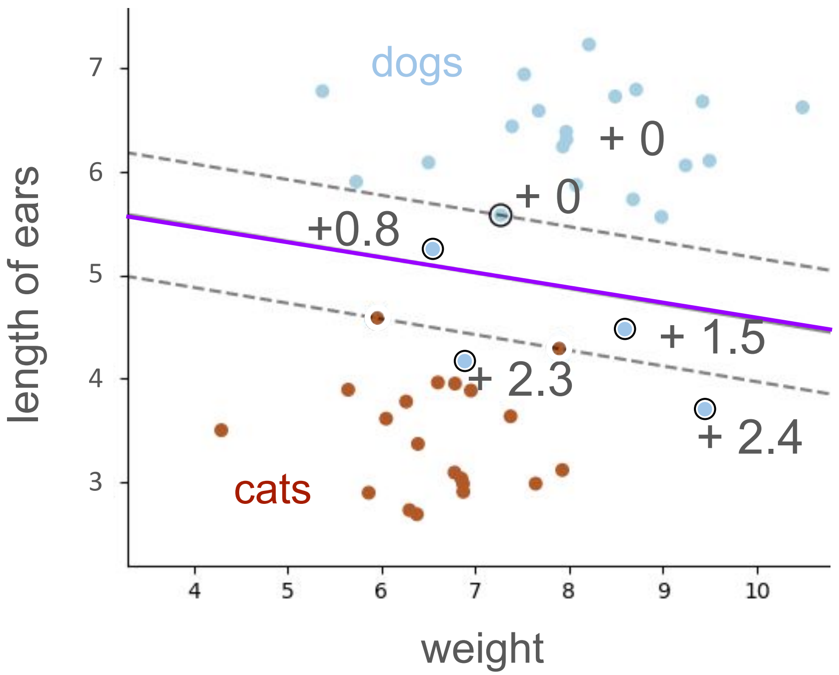
The decision boundary is the line that is equally far away from “support vectors” of each class

Classification: Support vector machines (SVM)



What to do if classes are not separable?

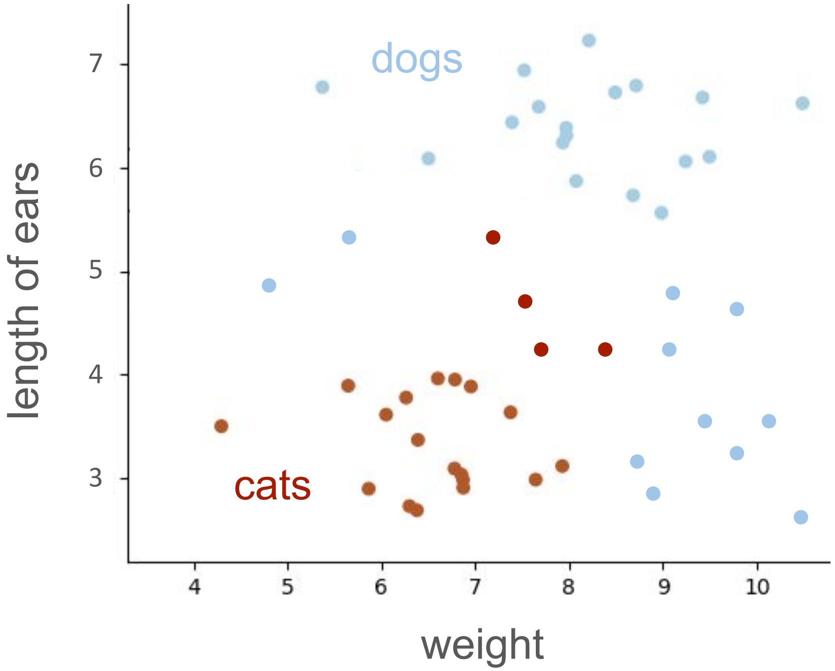
Classification: Support vector machines (SVM)



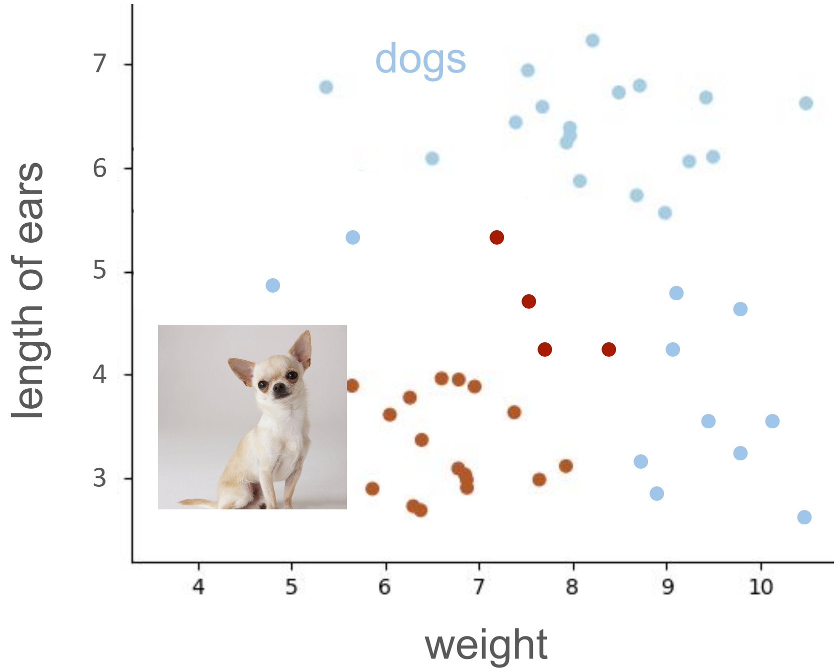
What to do if classes are not separable?
→ We penalize misclassifications with a point system that depends on the distance from boundary and margin.

Fitting an SVM means maximizing the margin while minimizing penalties.

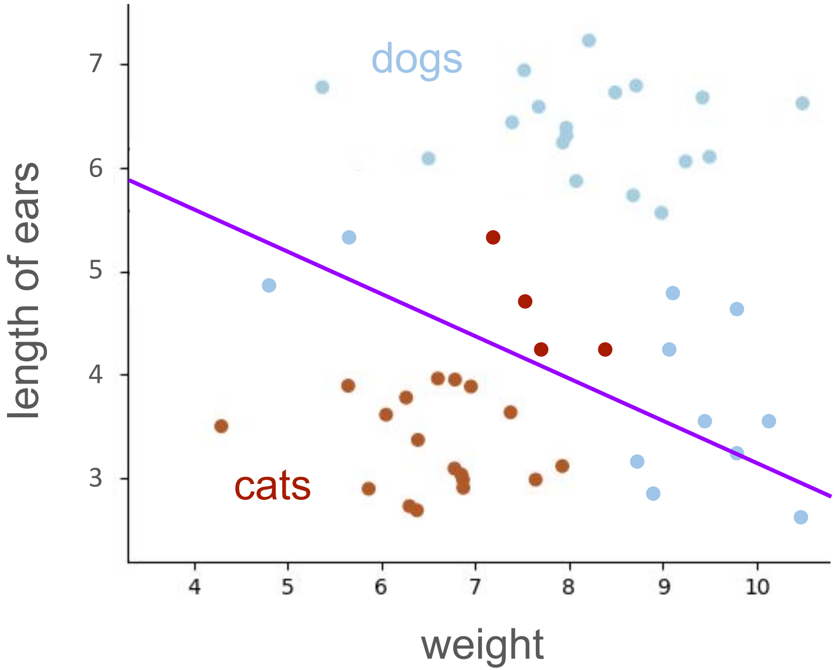
Classification: Support vector machines (SVM)



Classification: Support vector machines (SVM)

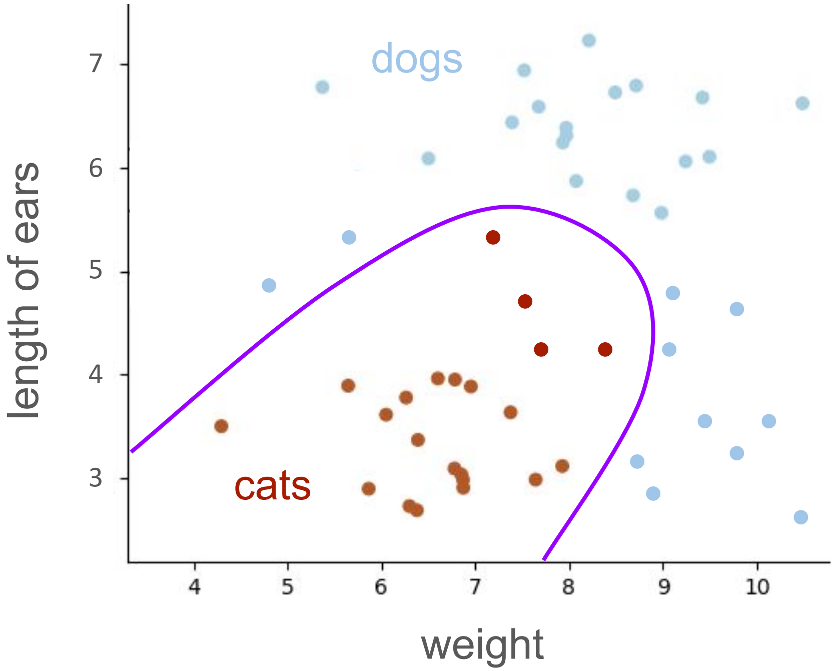


Classification: Support vector machines (SVM)



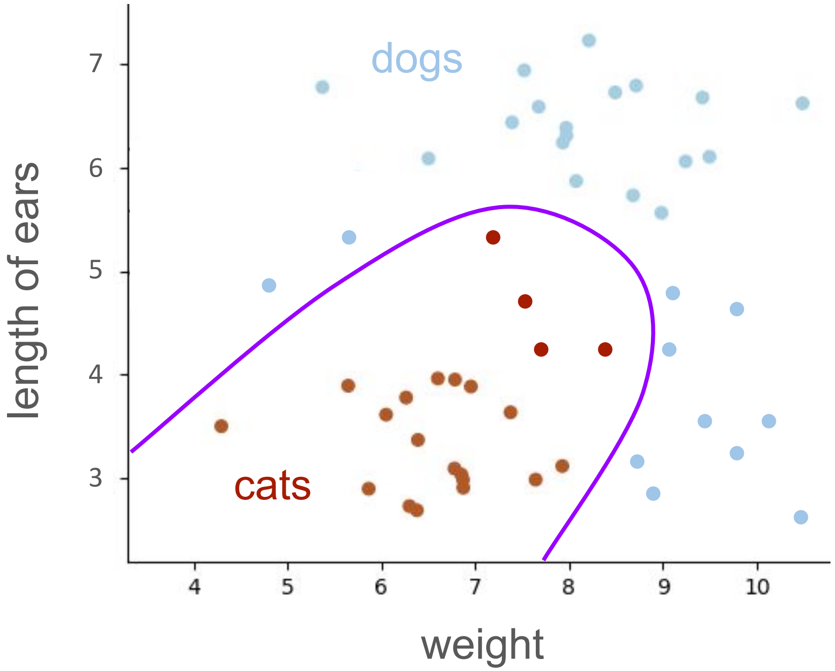
Some problems are not linearly separable

Classification: Support vector machines (SVM)



Some problems are not linearly separable, but they might be nonlinearly separable.

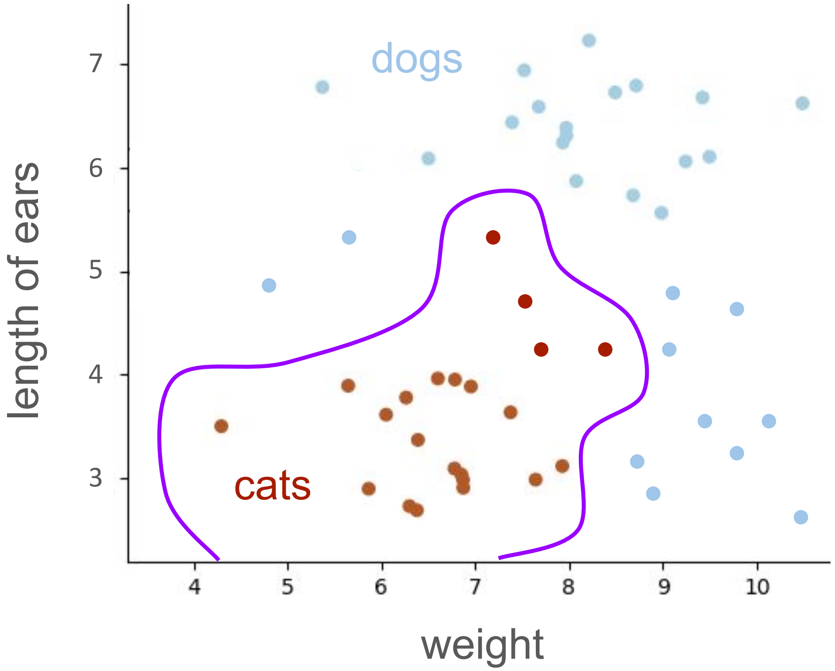
Classification: Support vector machines (SVM)



Some problems are not linearly separable, but they might be nonlinearly separable.

SVMs use **“kernels”** to transform predictor variables nonlinearly. This can give us nonlinear decision boundaries.

Classification: Support vector machines (SVM)

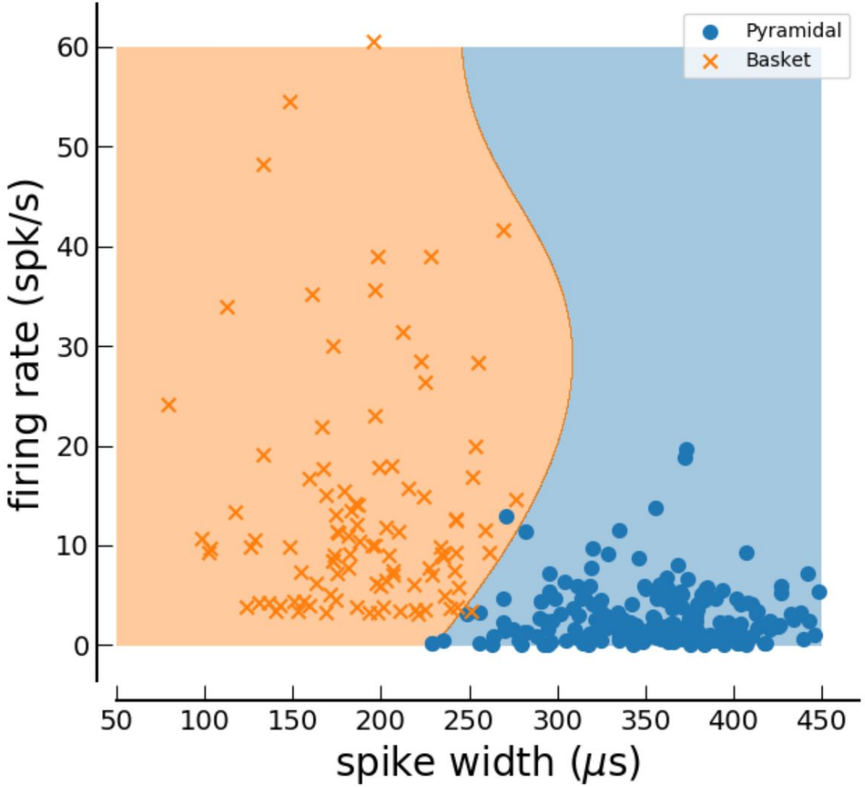


Some problems are not linearly separable, but they might be nonlinearly separable.

SVMs use **"kernels"** to transform predictor variables nonlinearly. This can give us nonlinear decision boundaries.

E.g. polynomial kernels of increasing degree create increasing nonlinearity.

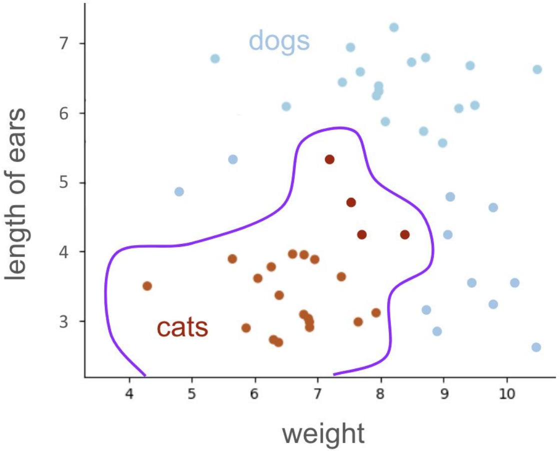
Classification: Support vector machines (SVM)



Classification: Overfitting and classification performance

The more flexible our boundary, the more likely we are to “**overfit**”: Classification performance is good on the original dataset, but bad on a new sample

Train set: data used to fit the classifier

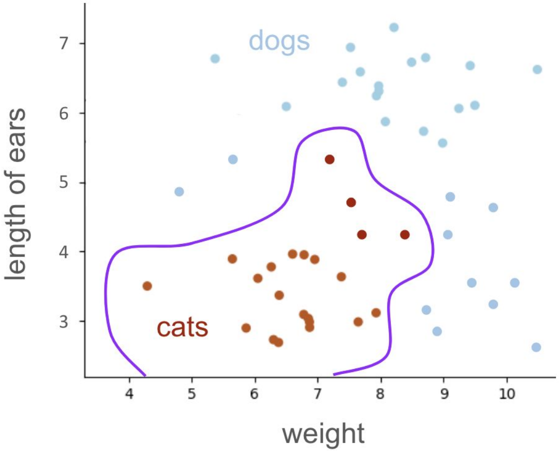


100 % classification performance

Classification: Overfitting and classification performance

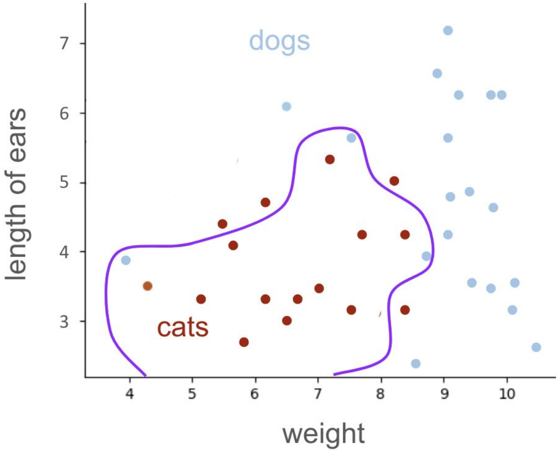
The more flexible our boundary, the more likely we are to “**overfit**”: Classification performance is good on the original dataset, but bad on a new sample

Train set: data used to fit the classifier



100 % classification performance

Test set: new data, “old” model.

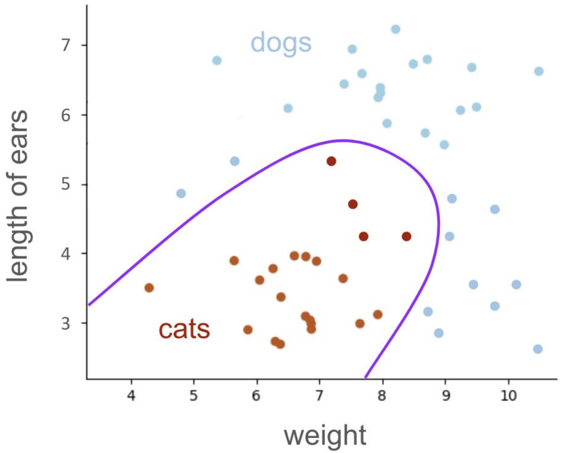


~90 % classification performance

Classification: Overfitting and classification performance

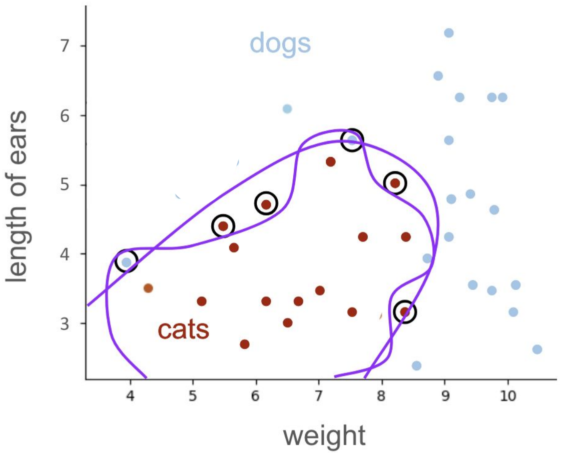
Simpler models tend to **generalize** better (= less overfitting): Less discrepancy between performance on train and test set.

Train set: data used to fit the classifier



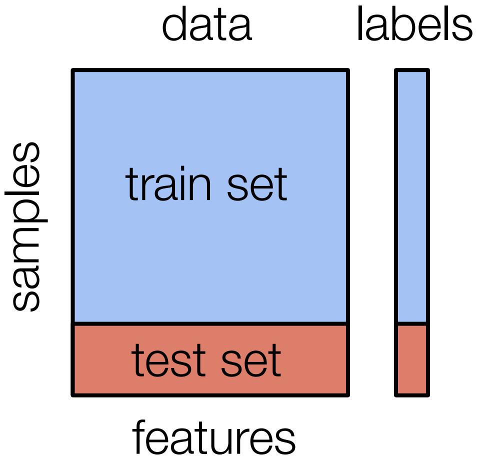
100 % classification performance

Test set: new data, “old” model.

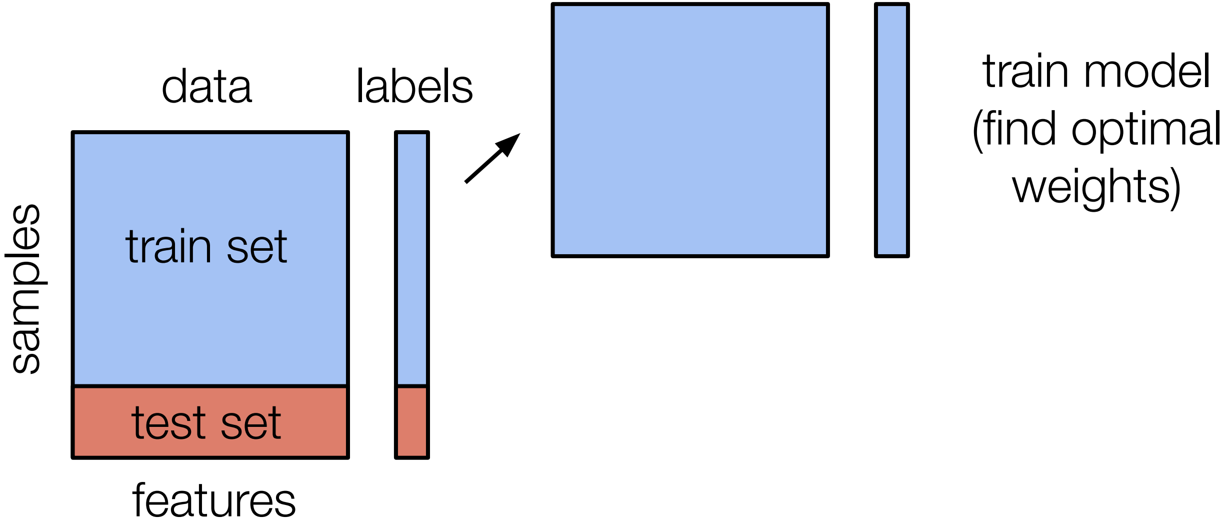


~98 % classification performance

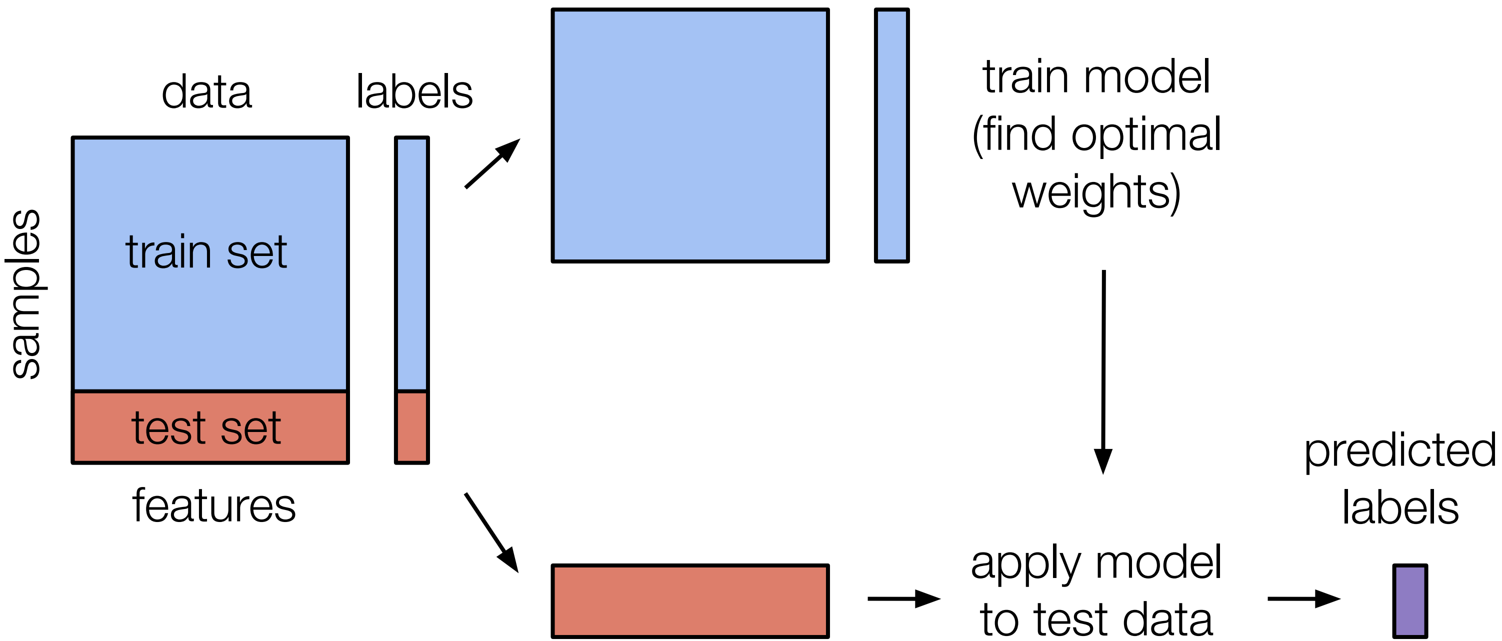
Classification: Cross-validation



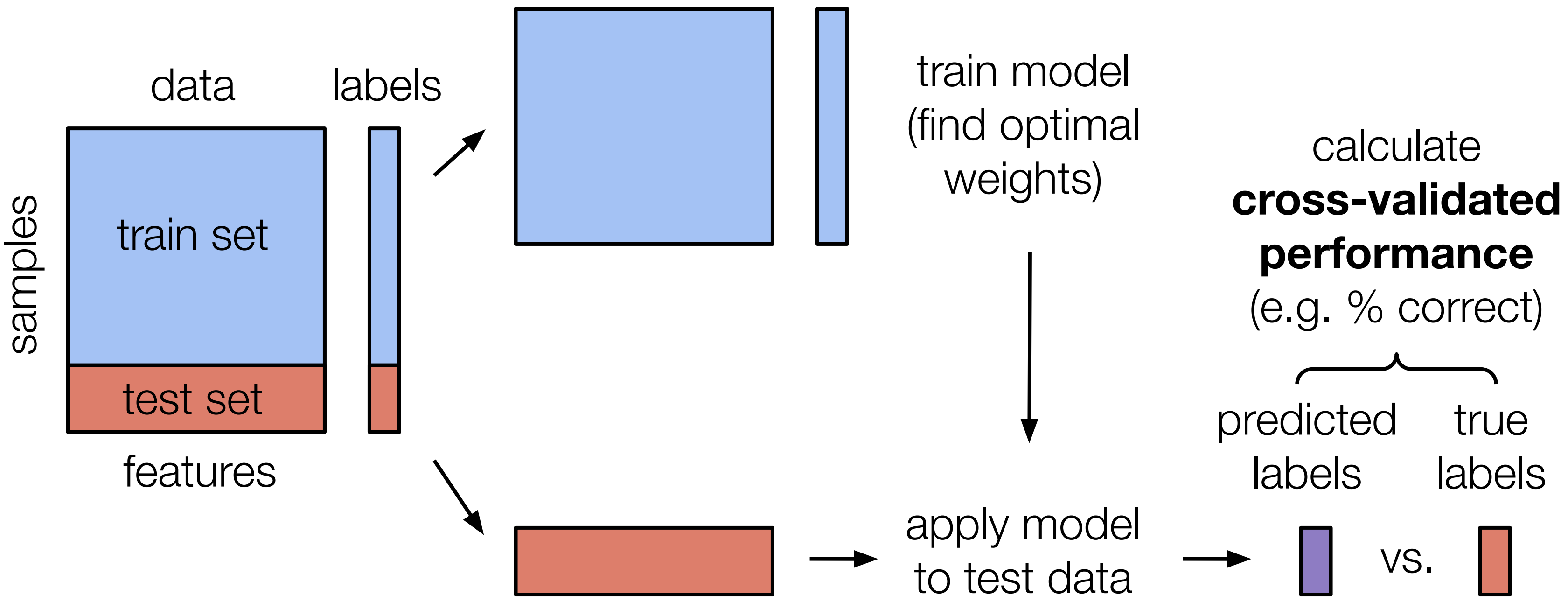
Classification: Cross-validation



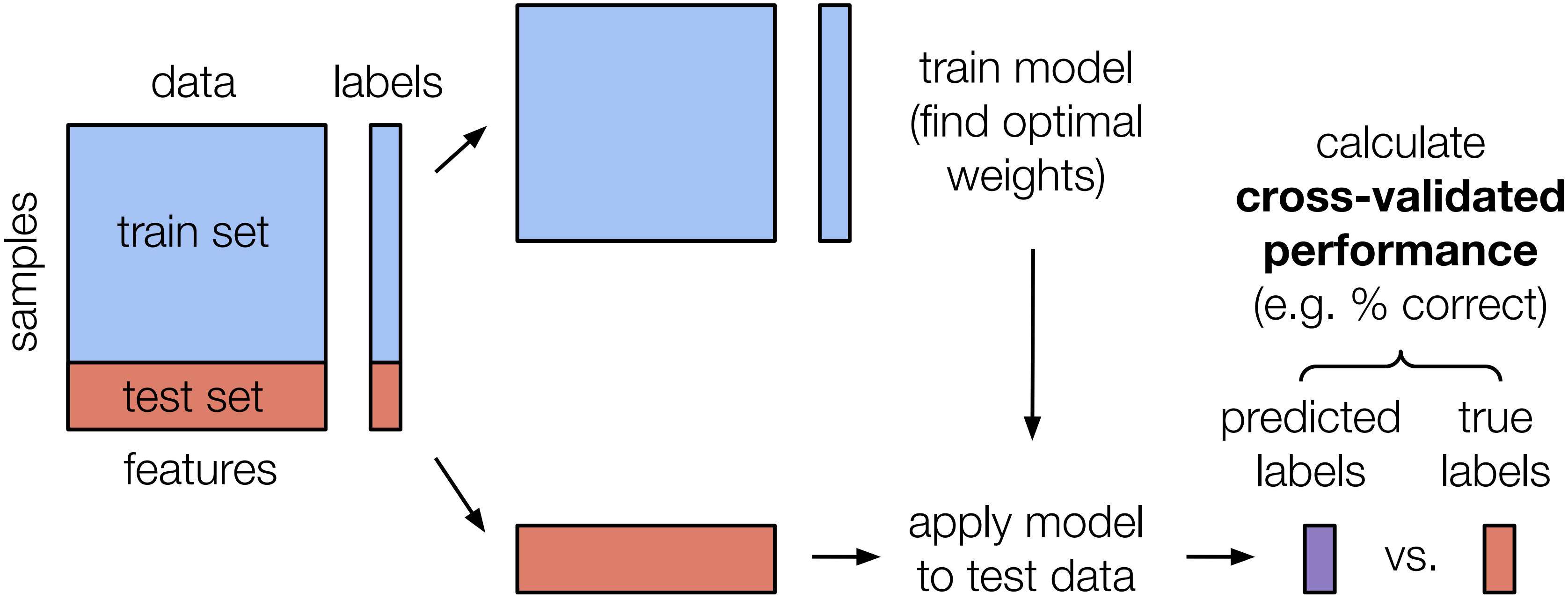
Classification: Cross-validation



Classification: Cross-validation



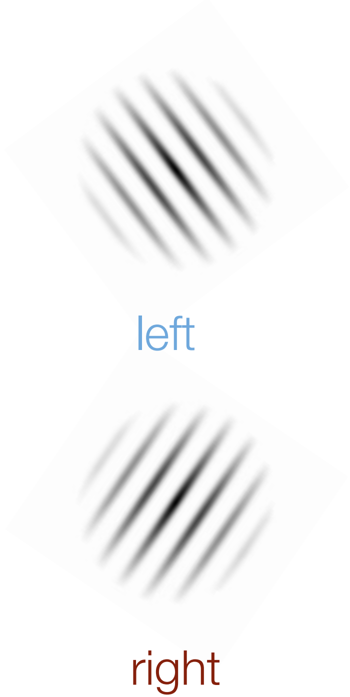
Classification: Cross-validation

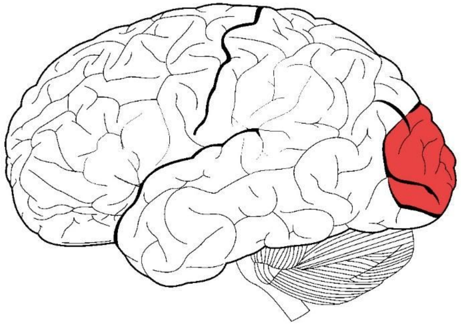


→ Repeat with many train/test splits. Is **perf. better than expected by chance**?

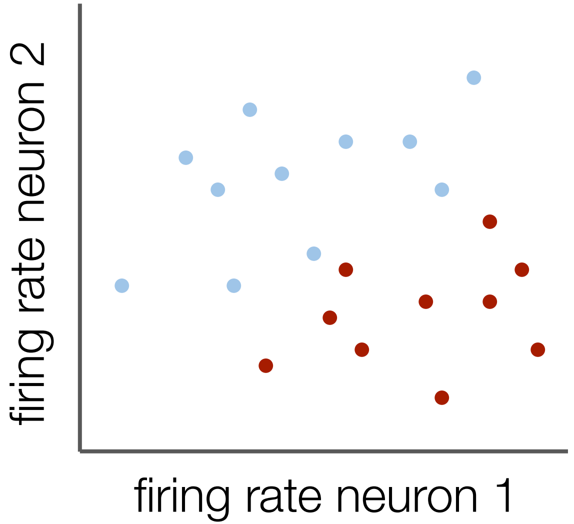
Classification: What can we learn about the brain?

We can test whether a group of neurons encodes stimulus information in its activity



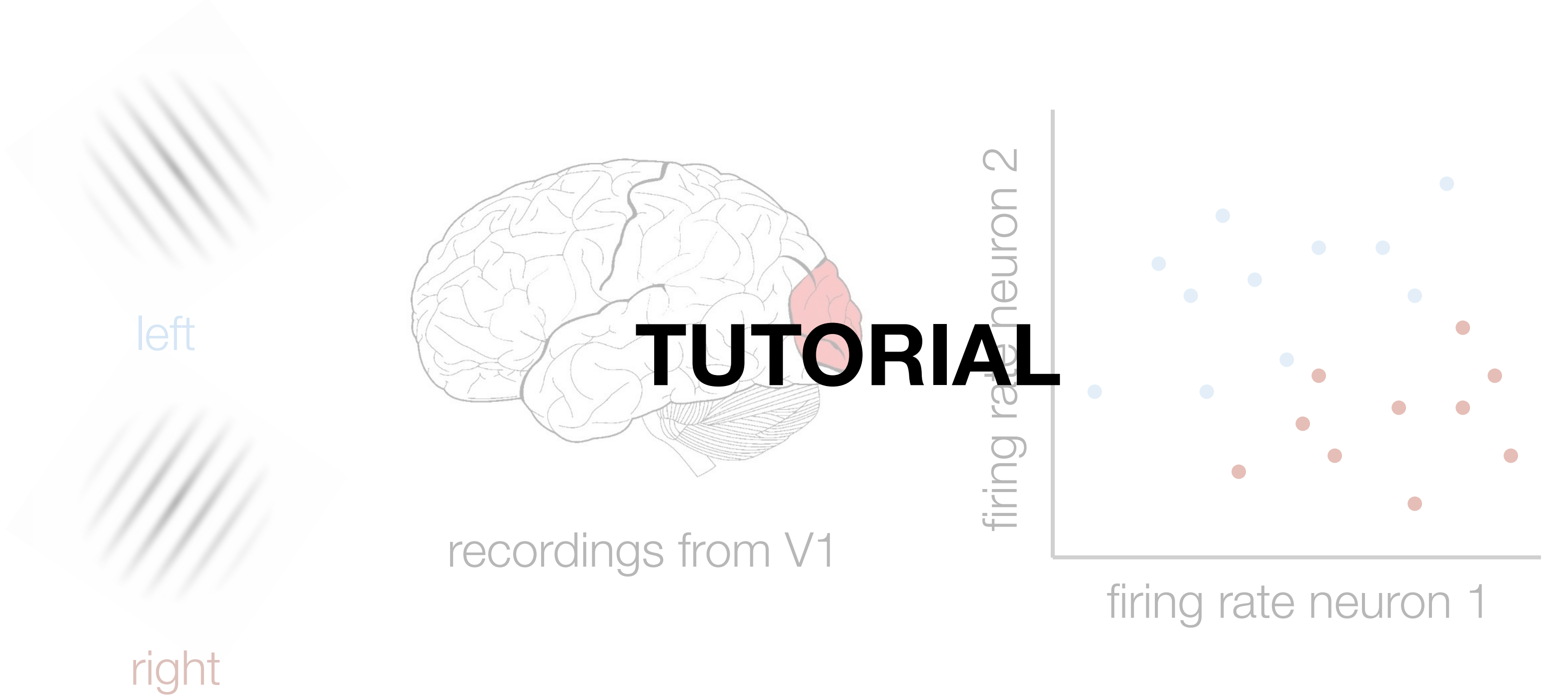


recordings from V1



Classification: What can we learn about the brain?

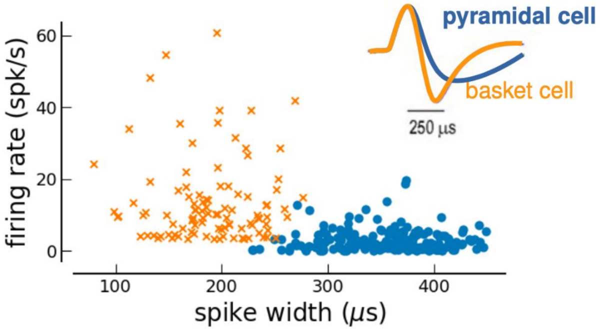
We can test whether a group of neurons encodes stimulus information in its activity



TUTORIAL

Classification and Clustering: Overview

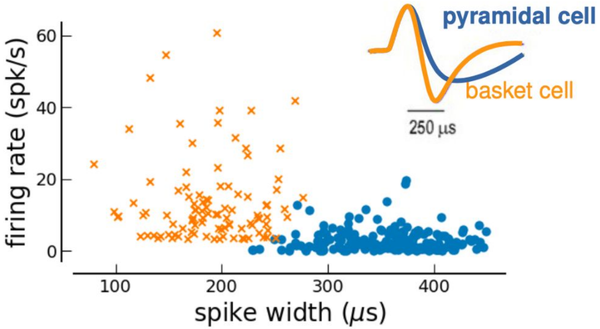
Classification: We **know the true categories** and want to know whether there is a reliable relationship between data and categories**.** Classification methods are also called “**supervised**” (known ground truth).

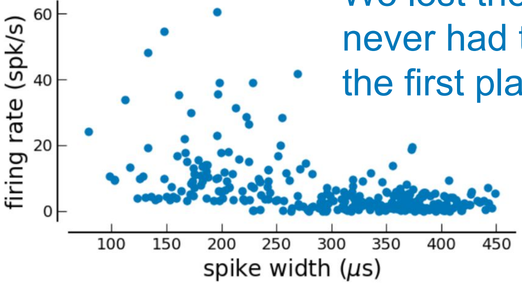


Classification and Clustering: Overview

Classification: We **know the true categories** and want to know whether there is a reliable relationship between data and categories. Classification methods are also called "**supervised**" (known ground truth).

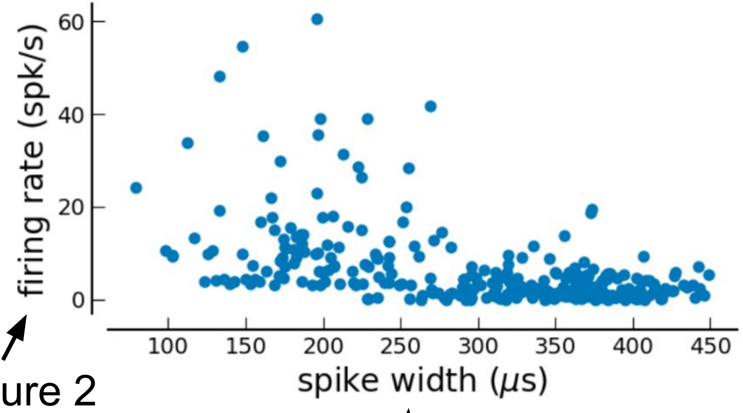
Clustering: We see **patterns** in the data that **suggest multiple categories**, but we don't know which data point belongs to which category. Clustering is an "**unsupervised**" method (unknown ground truth).





We lost the labels or never had them in the first place.

Clustering: Guessing categories from patterns in the feature space



feature 2

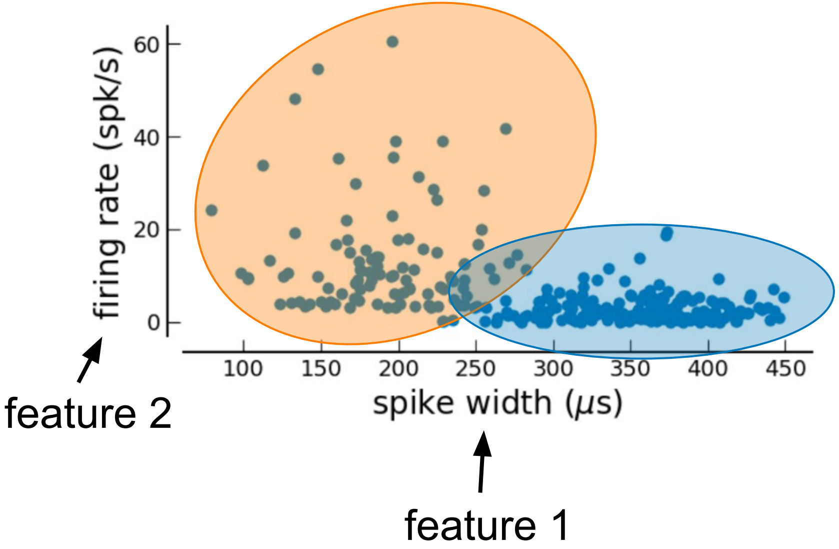
feature 1

Let’s assume that we know that there are two different cell types.

However, we don’t have the label for each cell (we cannot classify).

Can we **guess which are the points belonging to one vs. the other class**?

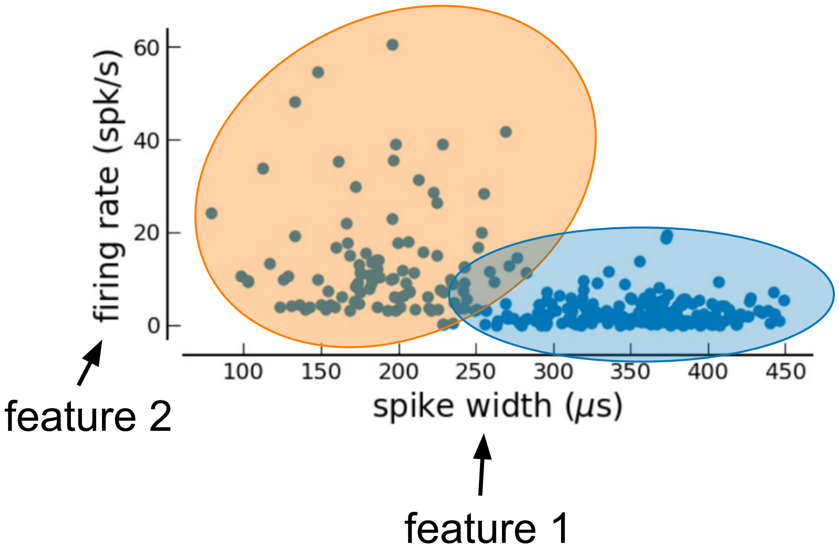
Clustering: Guessing categories from patterns in the feature space



Can we guess which are the points belonging to one vs. the other class?

We do this by defining **regions of the feature space**.

Clustering: Guessing categories from patterns in the feature space

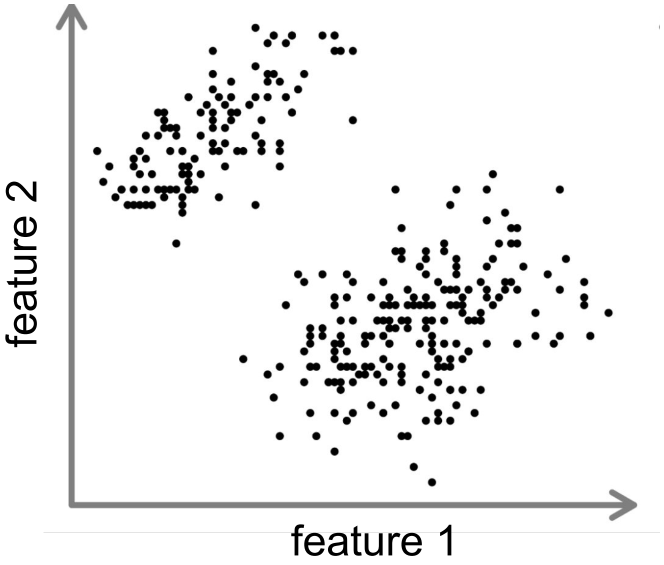


Can we guess which are the points belonging to one vs. the other class?

We do this by defining **regions of the feature space**.

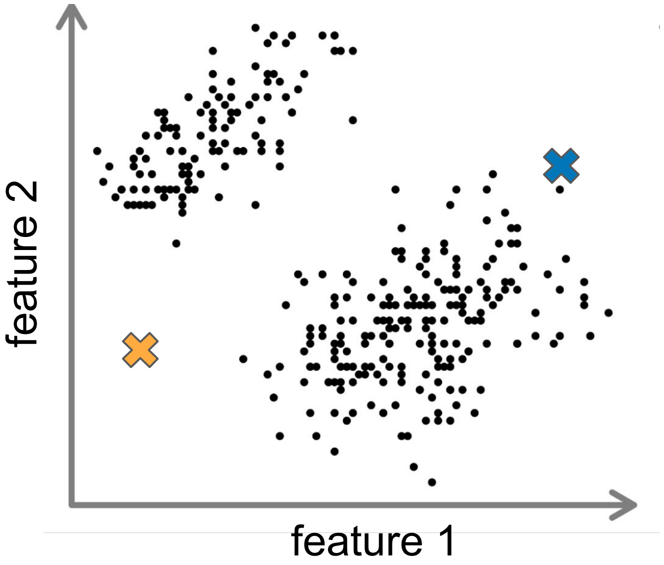
But how?

Clustering: k-means



Assuming two clusters, we try to find k = 2 cluster means (“centroids”).

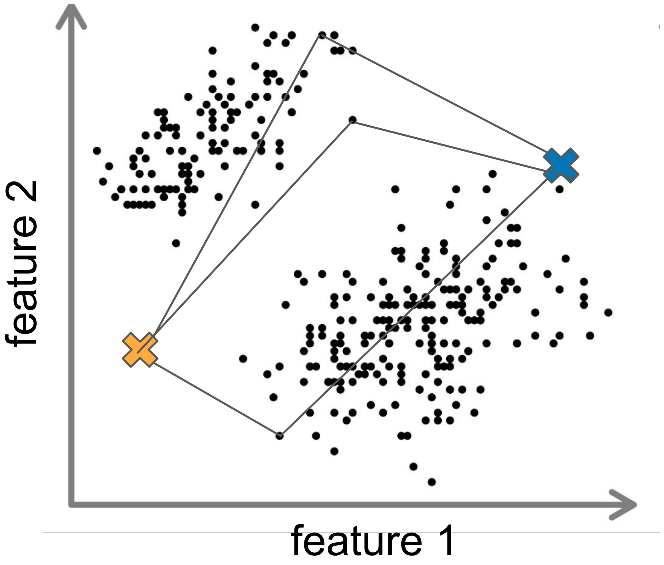
Clustering: k-means



Assuming two clusters, we try to find k = 2 cluster means (“centroids”).

1. Pick two random values (“initialization”).

Clustering: k-means

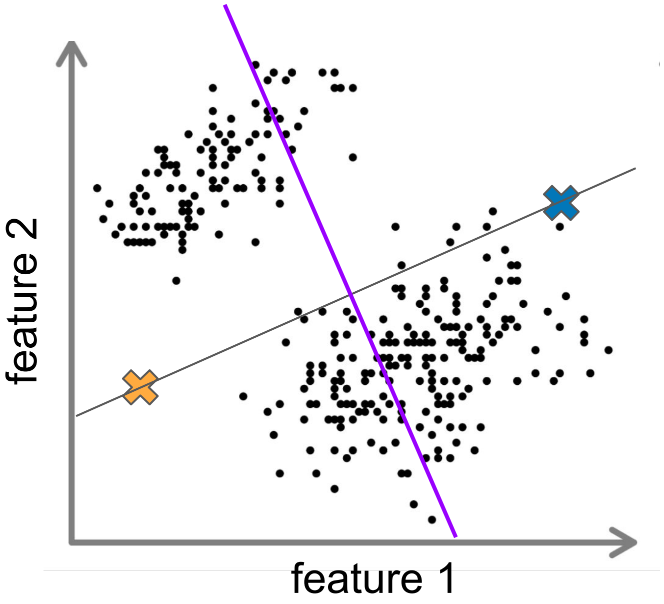


Assuming two clusters, we try to find
k = 2 cluster means (“centroids”).

1. Pick two random values (“initialization”).

2. Assign each data point to closest cluster.

Clustering: k-means

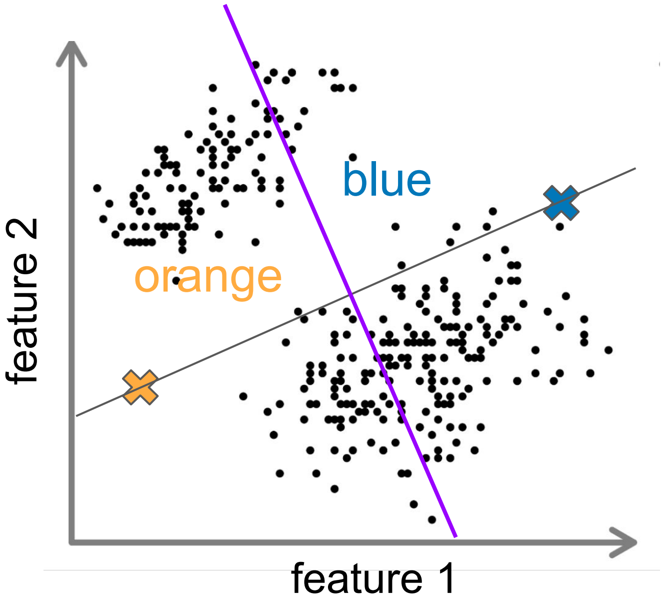


Assuming two clusters, we try to find
k = 2 cluster means (“centroids”).

1. Pick two random values (“initialization”).

2. Assign each data point to closest cluster (note: there is effectively a decision boundary!).

Clustering: k-means

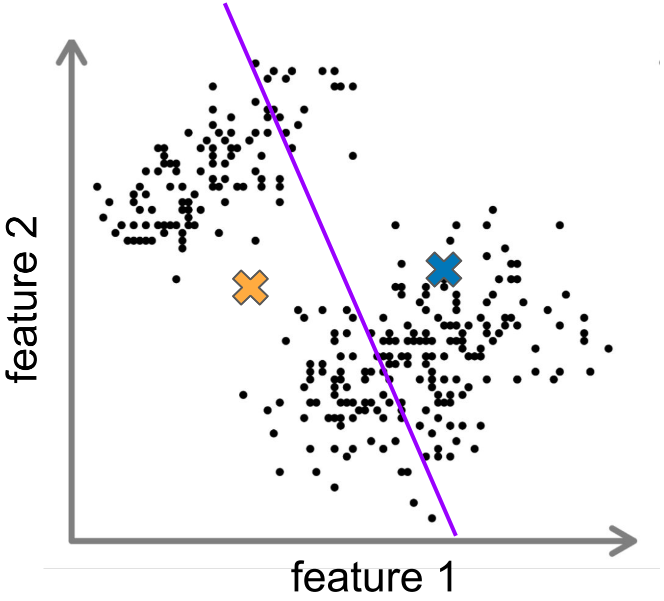


Assuming two clusters, we try to find k = 2 cluster means ("centroids").

1. Pick two random values ("initialization").

2. Assign each data point to closest cluster (note: there is effectively a decision boundary!).

Clustering: k-means



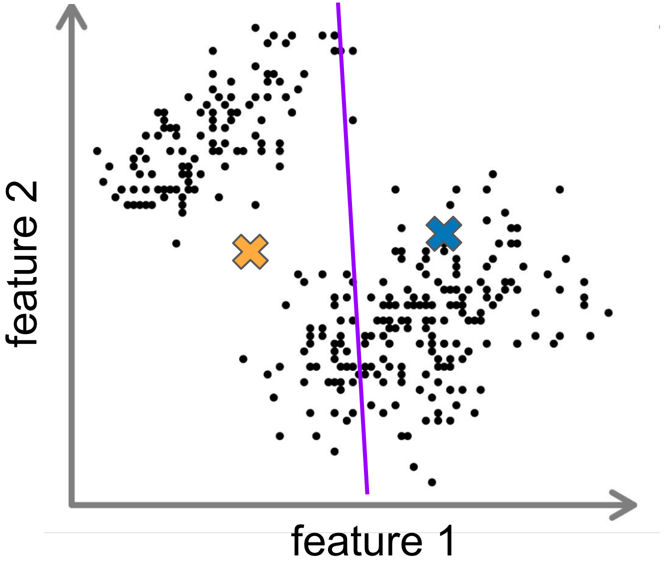
Assuming two clusters, we try to find
k = 2 cluster means (“centroids”).

1. Pick two random values (“initialization”).

2. Assign each data point to closest cluster.

3. Recalculate cluster means

Clustering: k-means

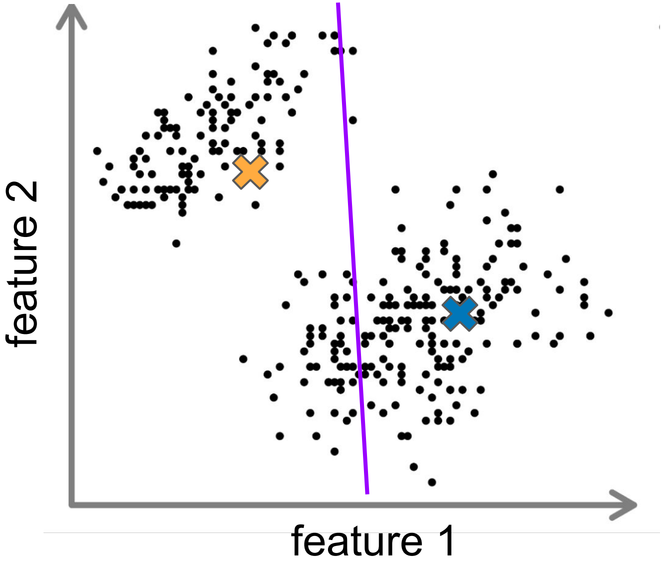


Assuming two clusters, we try to find k = 2 cluster means (“centroids”).

1. Pick two random values (“initialization”).

2. Assign each data point to closest cluster.

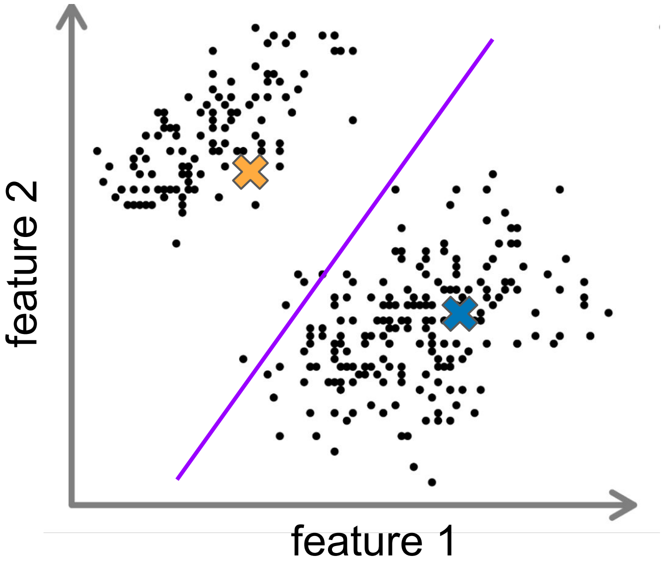
Clustering: k-means



Assuming two clusters, we try to find
k = 2 cluster means ("centroids").

1. Pick two random values ("initialization").
2. Assign each data point to closest cluster.
3. Recalculate cluster means

Clustering: k-means

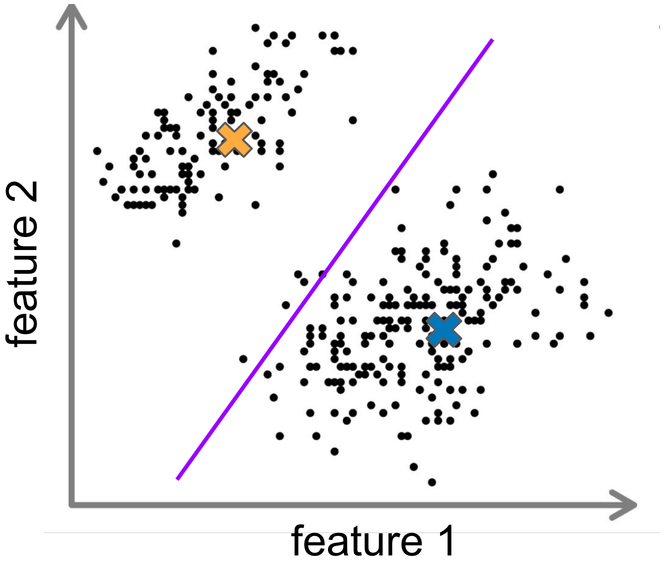


Assuming two clusters, we try to find
k = 2 cluster means (“centroids”).

1. Pick two random values (“initialization”).

2. Assign each data point to closest cluster.

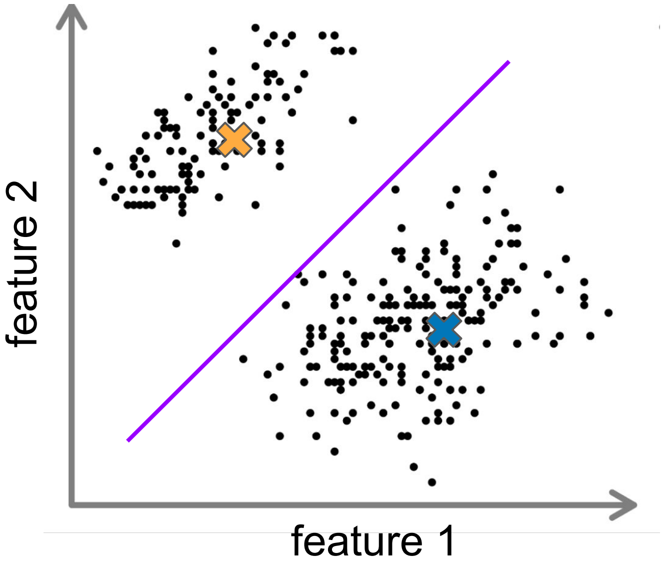
Clustering: k-means



Assuming two clusters, we try to find
k = 2 cluster means (“centroids”).

1. Pick two random values (“initialization”).
2. Assign each data point to closest cluster.
3. Recalculate cluster means

Clustering: k-means



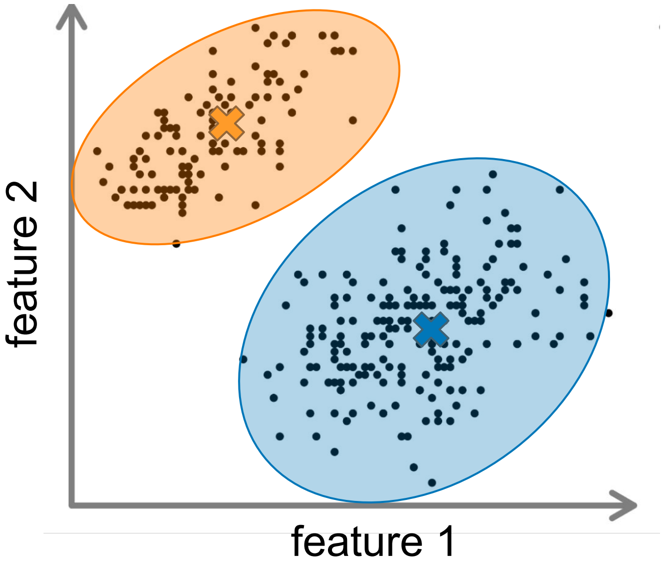
Assuming two clusters, we try to find
k = 2 cluster means (“centroids”).

1. Pick two random values (“initialization”).

2. Assign each data point to closest cluster.

3. Recalculate cluster means

Clustering: k-means

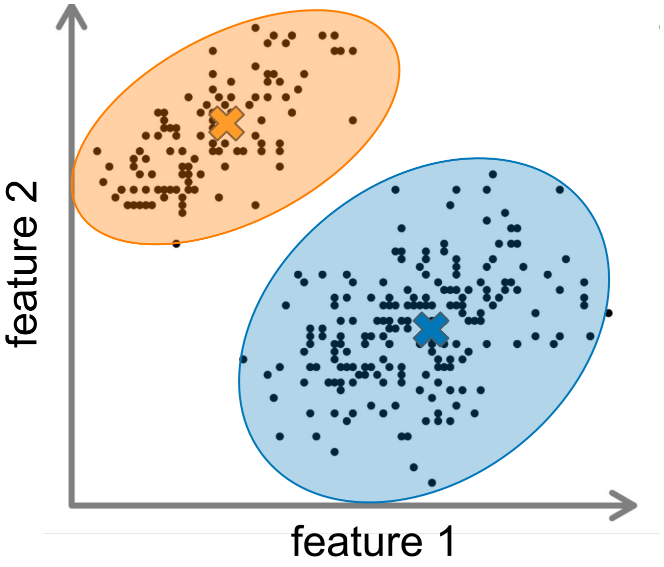


Assuming two clusters, we try to find
k = 2 cluster means (“centroids”).

1. Pick two random values (“initialization”).
2. Assign each data point to closest cluster.
3. Recalculate cluster means.

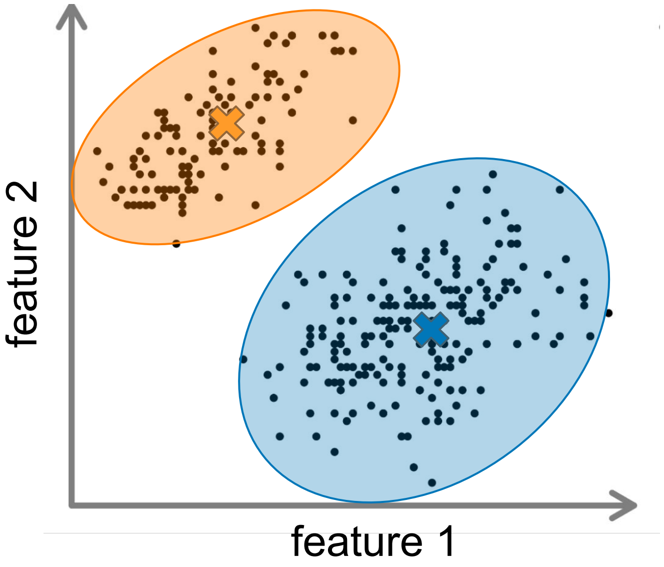
Repeat 2.-3. until cluster means are stable (they “converged”).

Clustering: k-means



Clustering is **iterative** (repeat steps until convergence).

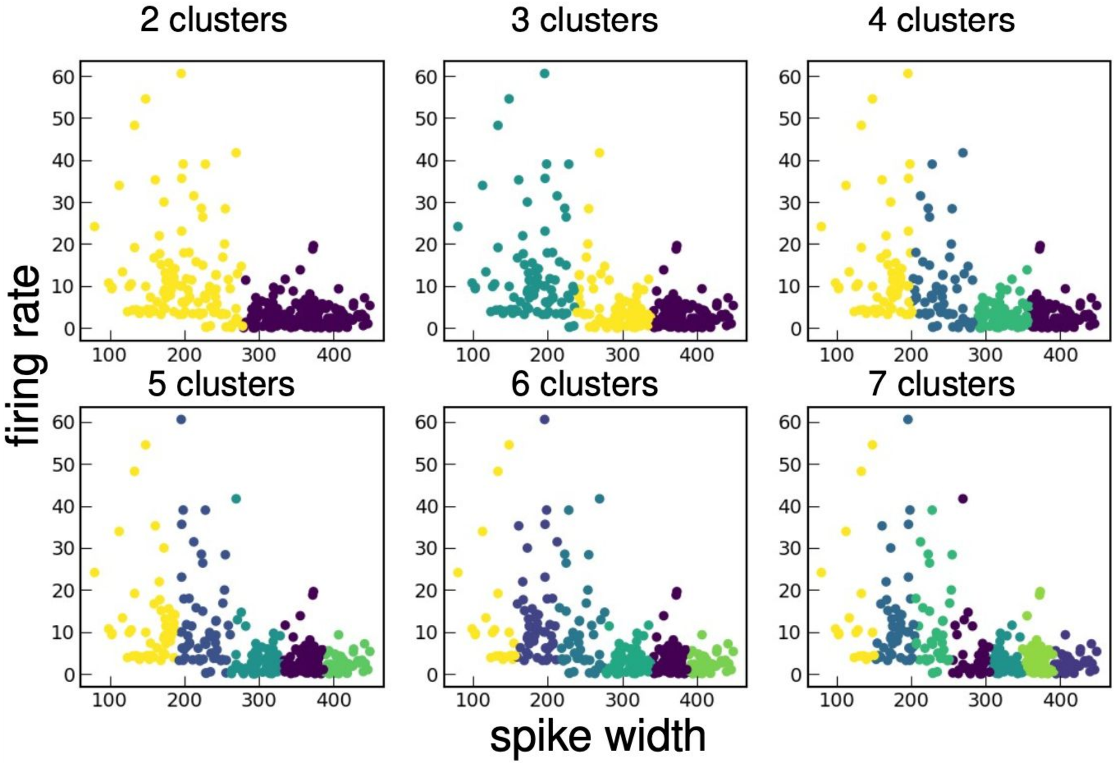
Clustering: k-means



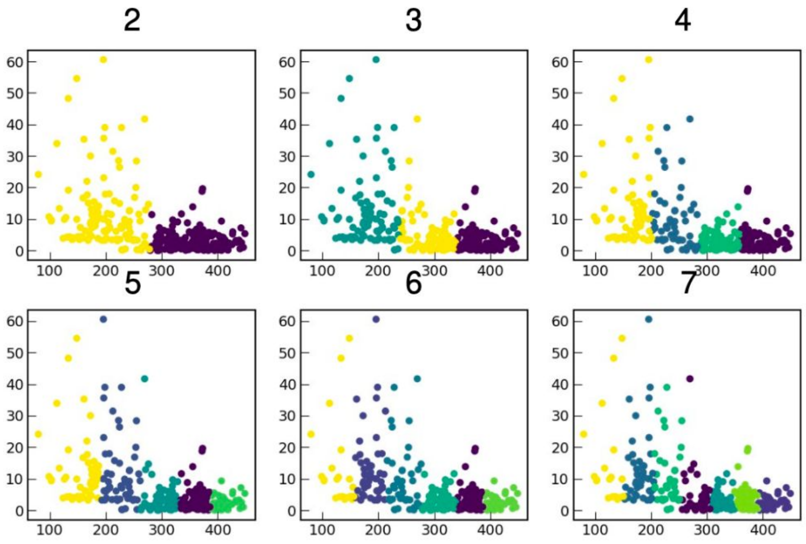
Clustering is **iterative** (repeat steps until convergence).

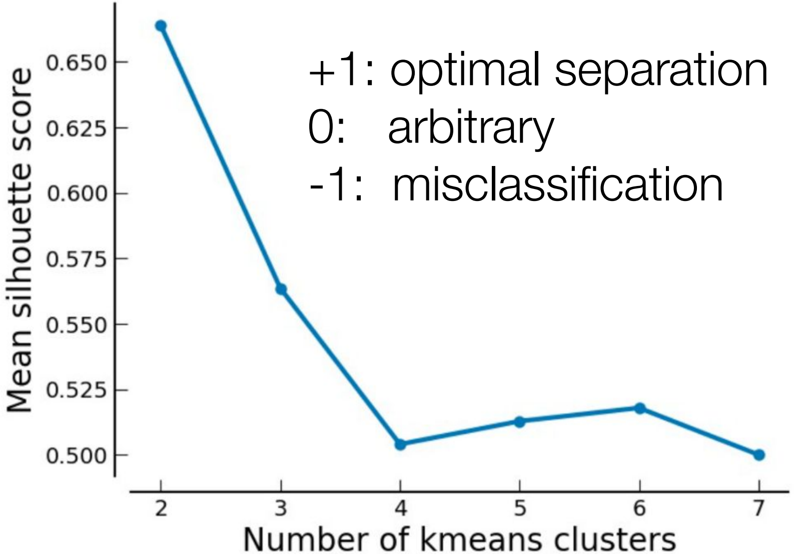
The **optimal number of clusters is not always obvious**: We therefore often repeat clustering for different values of “hyperparameter” k and compare a “goodness of fit” measure
(in k-means: “Silhouette score”).

Clustering: hyperparameter selection (k = ?)



Clustering: hyperparameter selection (k = ?)





+1: optimal separation

0: arbitrary

-1: misclassification

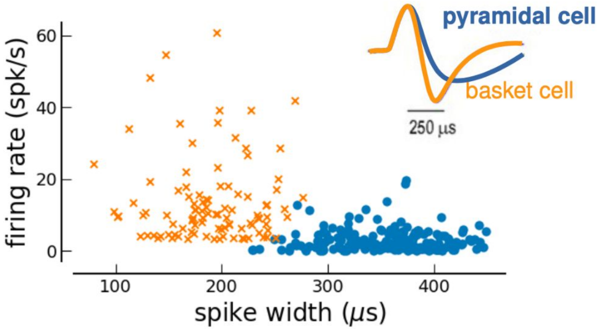
The **Silhouette score** for each point measures the distance to points in the same cluster vs. distance to points in the neighboring cluster

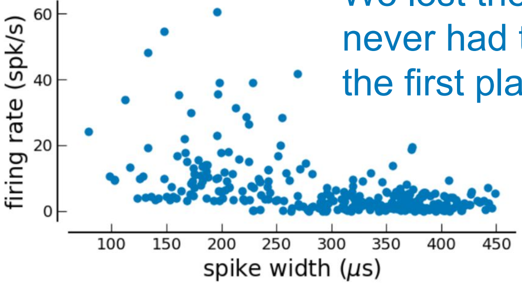
$$S = \frac{1}{N} \boxed{\sum_{i=1}^{N}} \frac{\boxed{b(i)} - \boxed{a(i)}}{\max(a(i),\, b(i))}$$

Classification and Clustering: Overview

Classification: We **know the true categories** and want to know whether there is a reliable relationship between data and categories. Classification methods are also called "**supervised**" (known ground truth).

Clustering: We see **patterns** in the data that **suggest multiple categories**, but we don't know which data point belongs to which category. Clustering is an "**unsupervised**" method (unknown ground truth).





We lost the labels or never had them in the first place.

Classification and Clustering: Overview

Classification

- Logistic regression
- Support vector machines

Clustering

- k-means