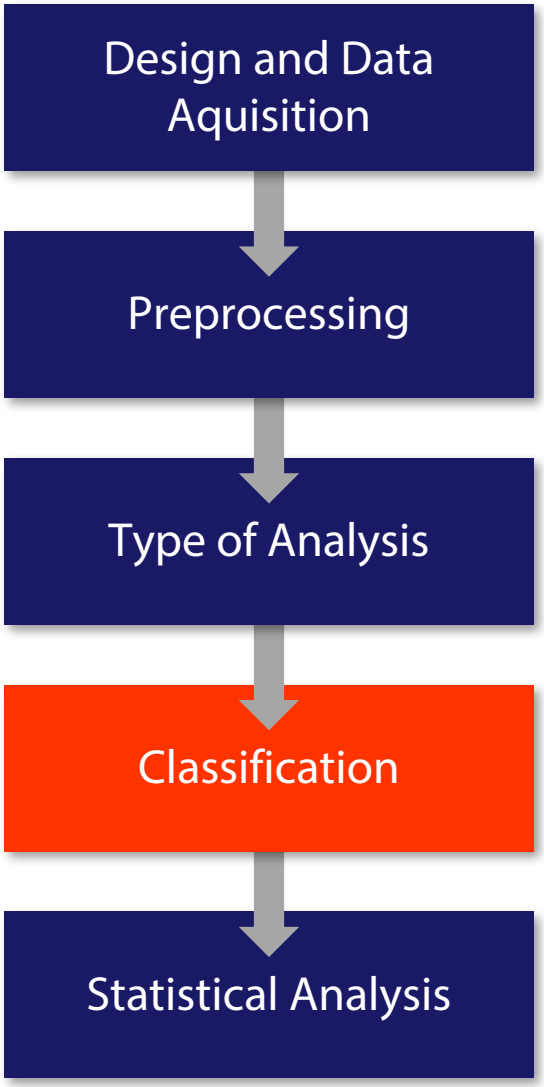
Session II: Introduction to Classification

Martin N. Hebart

Laboratory of Brain and Cognition

NIMH

Multivariate Decoding Workflow



Overview

The Foundations

- Crucial terminology (sample, feature, pattern, label, classifier)
- Basis of linear classification

Estimating classifier performance

- Cross-validation framework
- Classification measures (accuracy / AUC)

Bias-variance trade-off

- Overfitting and underfitting
- Regularization

Common classifiers

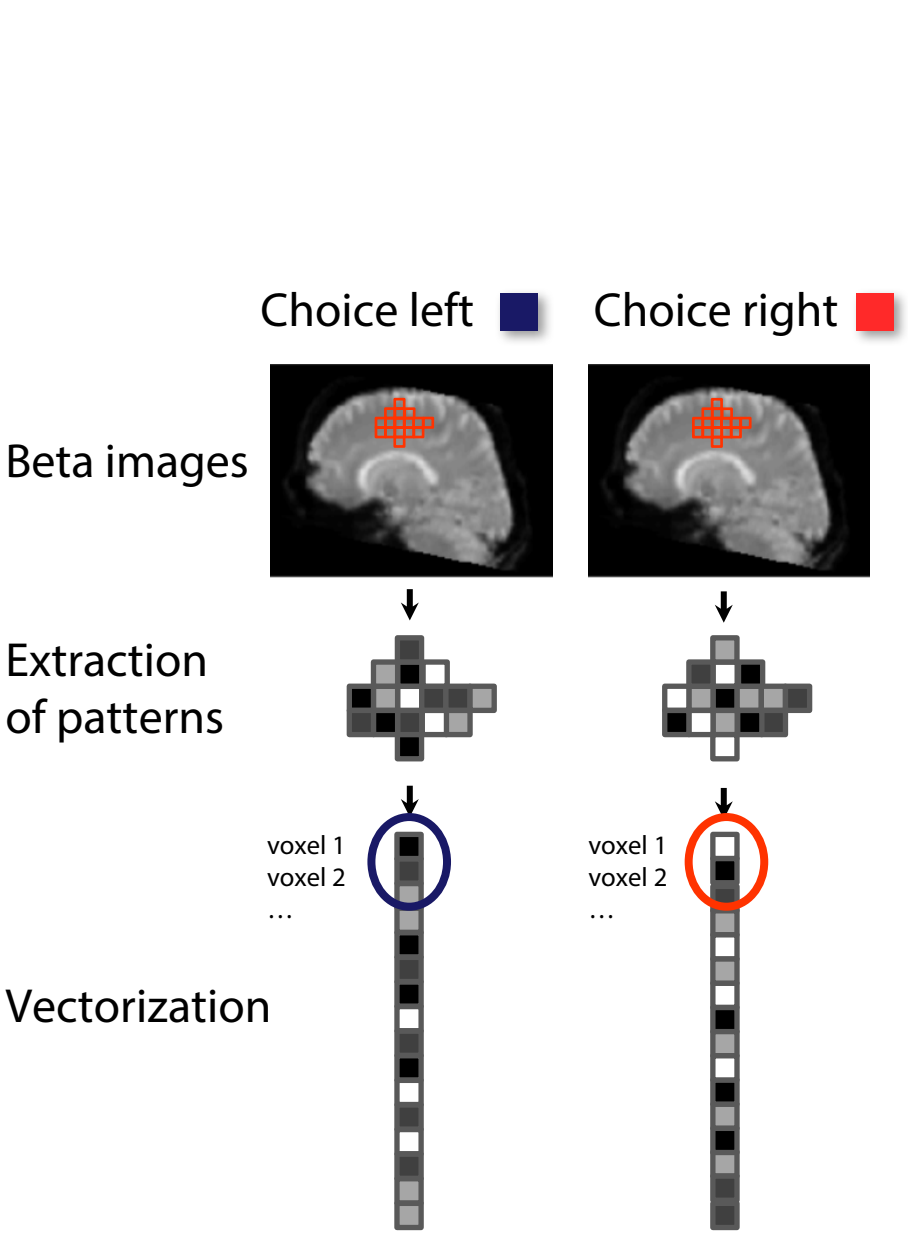
- Correlation classifier, Naïve Bayes, LDA, SVM

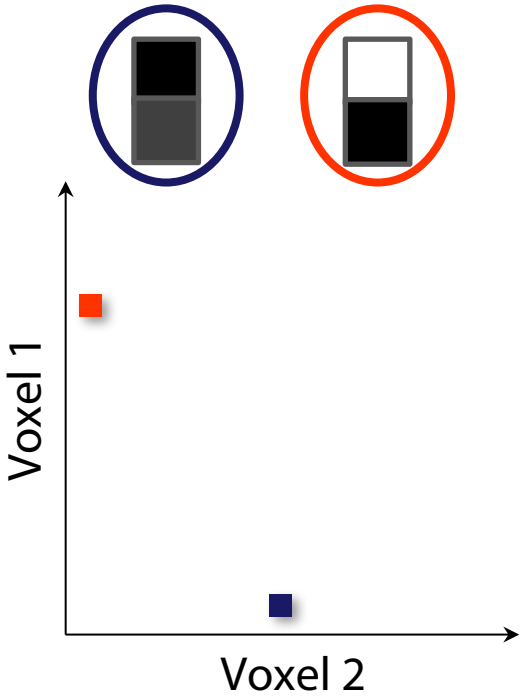
Non-independence and circular analysis

- Why “leave-one-run out” cross-validation?

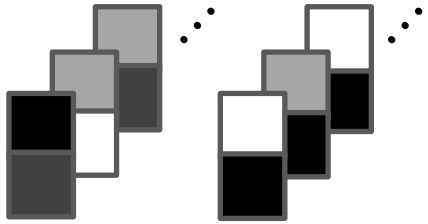
THE FOUNDATIONS

Classification Overview: Example



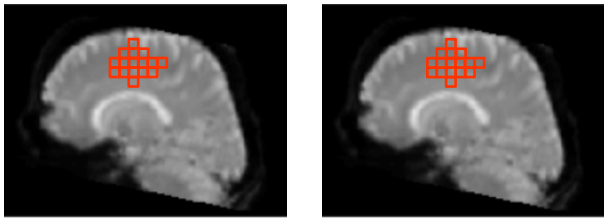


Classification Overview: Example



Choice left ■ Choice right ■

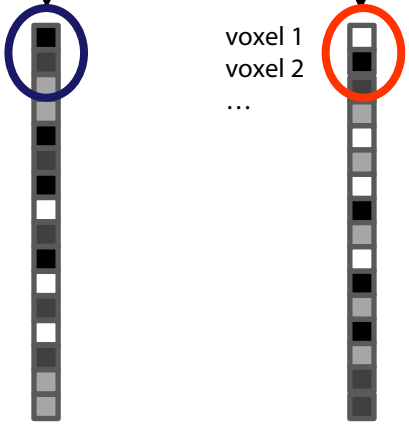
Beta images



Extraction of patterns

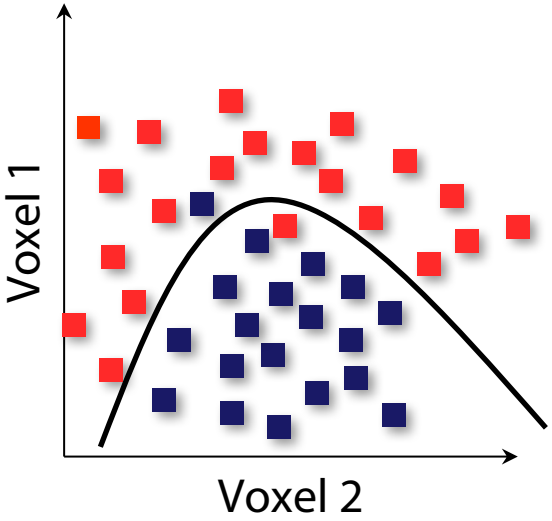
Vectorization

voxel 1
voxel 2
…

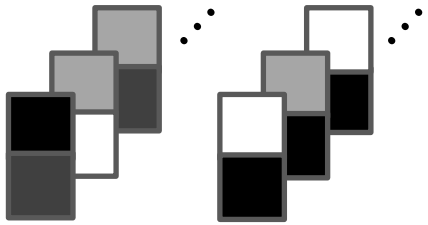


voxel 1
voxel 2
…

Train



Classification Overview: Example



Choice left

Choice right

Beta images

Extraction of patterns

Vectorization

voxel 1
voxel 2
…

voxel 1
voxel 2
…

Train

Predict

Voxel 1

Voxel 2

Voxel 1

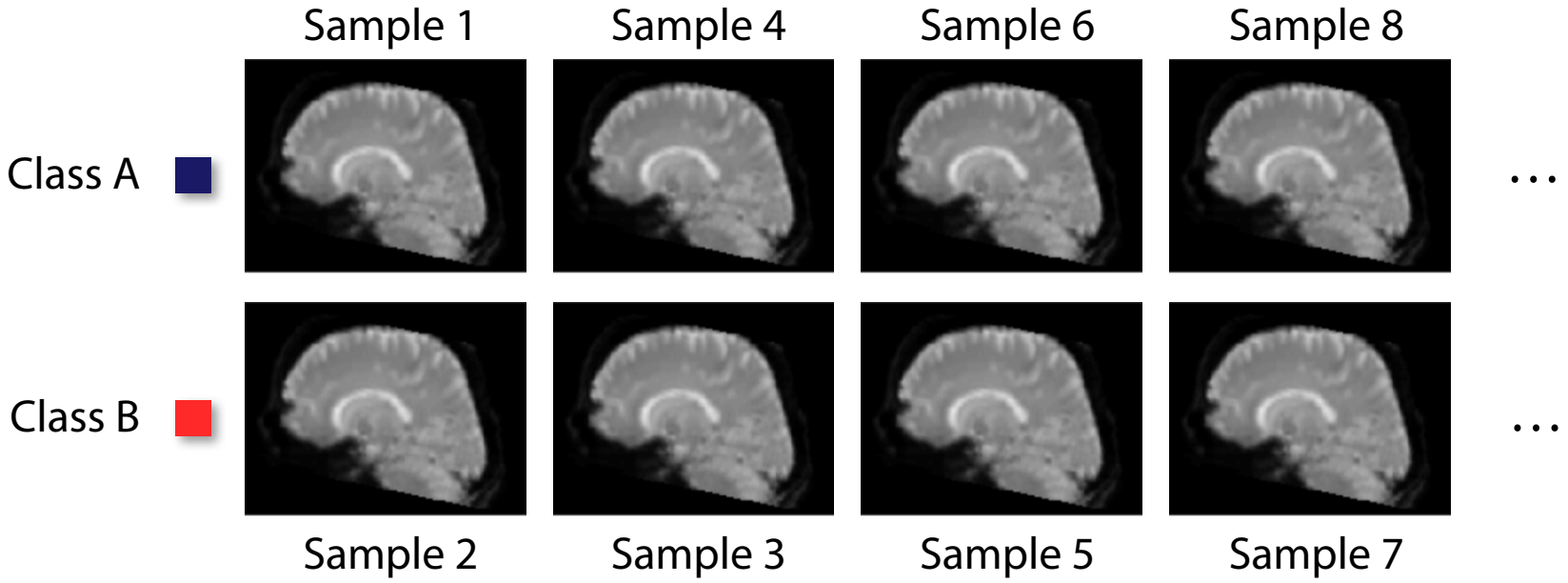
Voxel 2

Crucial Terminology

Sample

Samples are data that belong to a class

Examples: EPI volumes, beta volumes, VBM maps, EEG data



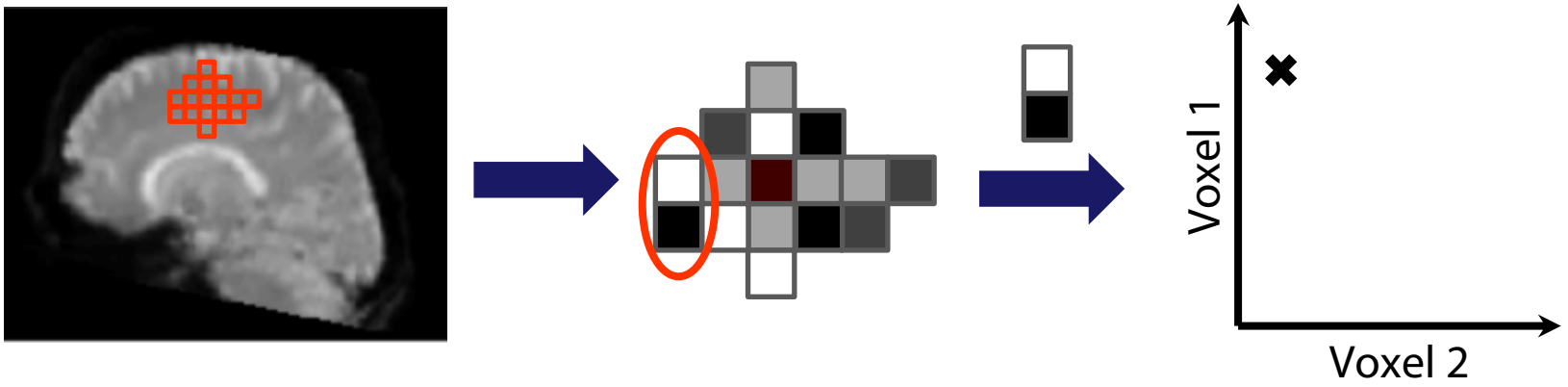
Crucial Terminology

Feature

Each feature is a measured variables that can be used for classification

- Each feature (hopefully) aids the classification process, by contributing signal and/or suppressing noise
- Each feature spans up a dimension → they build the feature space

Examples: A voxel, connectivity graph, EEG channel



Crucial Terminology

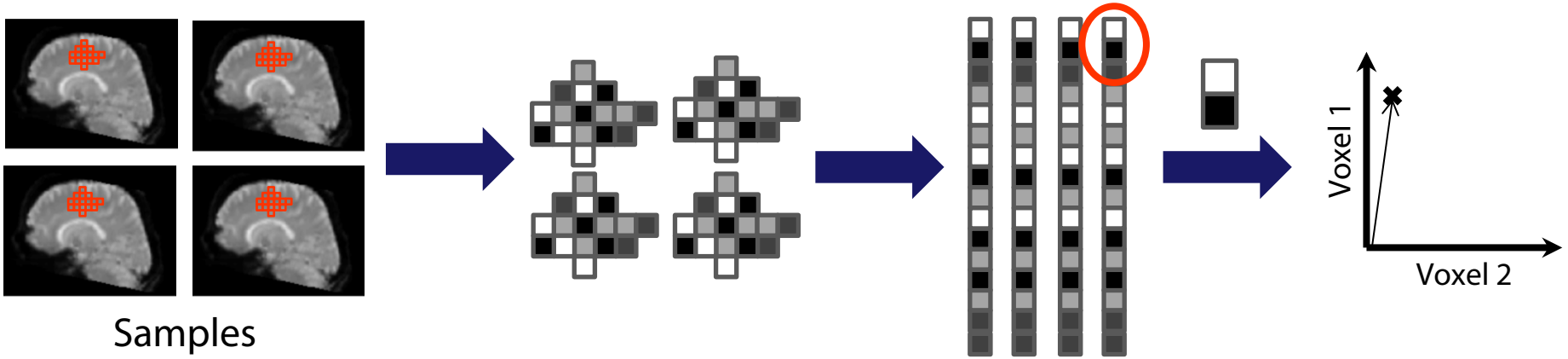
Pattern

A pattern is a sample for a set of features

A pattern is a point (or vector) in p-dimensional space (p is # of features)

Alternative uses of term "pattern" with different meaning:

- Prototypical pattern (i.e. the true class mean)
- Discriminating pattern (function that discriminates classes)



Crucial Terminology

Label

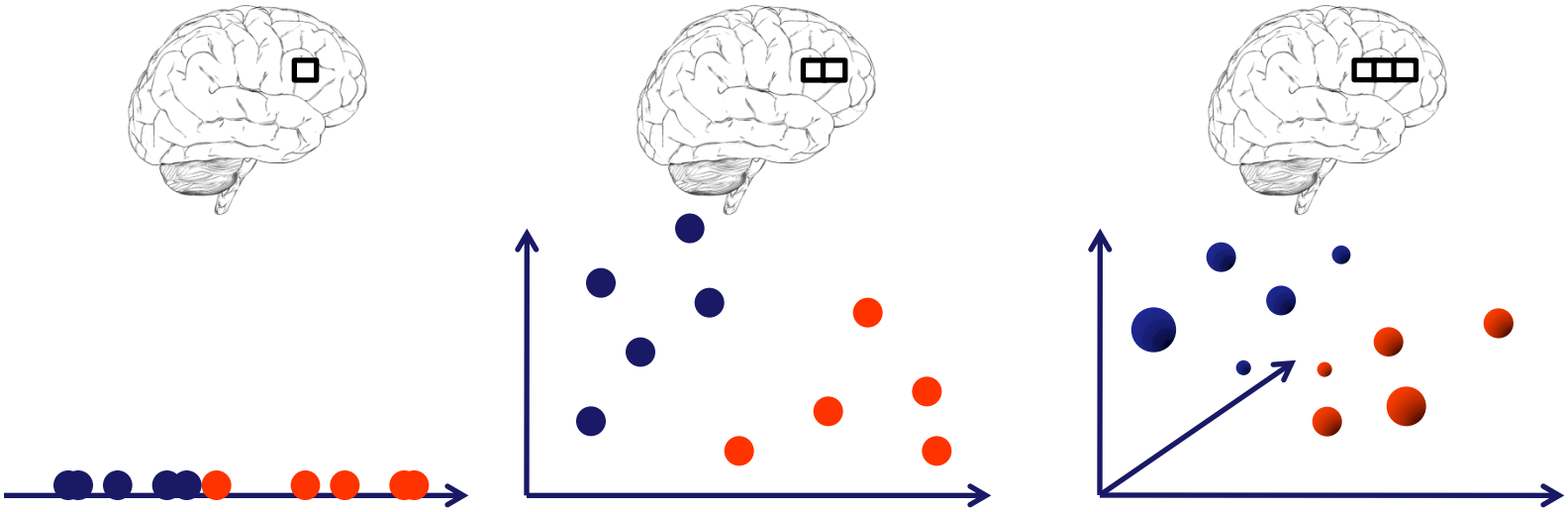
A label denotes the class membership of a pattern with a number

For classification the number is categorical and often arbitrary (some classifiers require 0 and 1 or -1 and 1)

For regression the number denotes a continuous number which is the regression target

High-dimensional Space

Textbook examples may be misleading



Real data: e.g. 200-D, but often fewer samples than features, i.e. p >> n

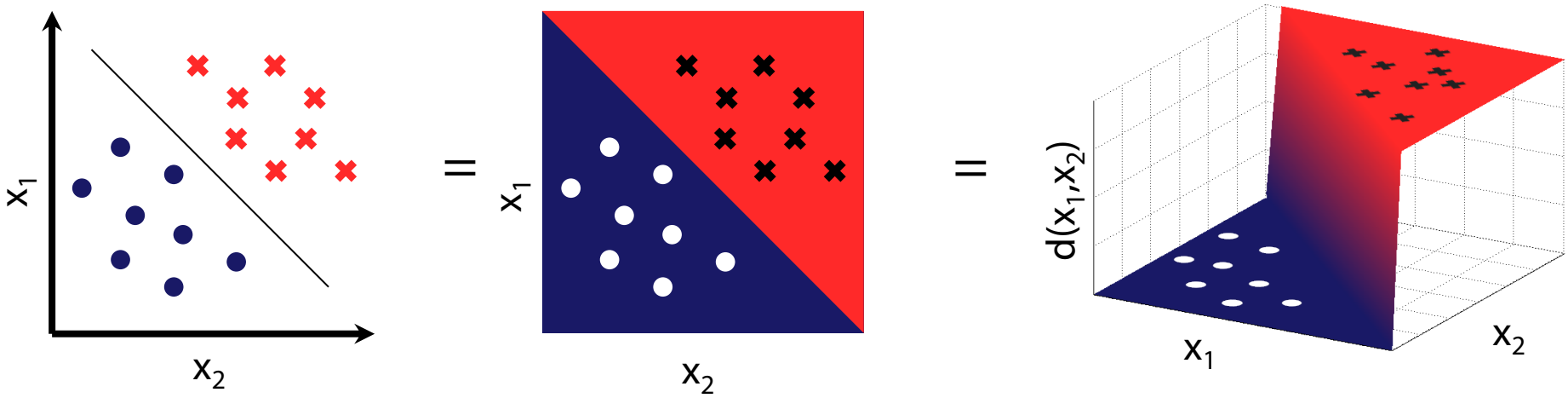
Crucial Terminology

Classifier

A function that separates feature space

Example for one sample with two features: $f(x_1, x_2) = -0.5$

This decision value f is then binarized in a decision function:
$if\ f(x_1, x_2) > 0$: $\boldsymbol{d(x_1, x_2) = 1}$; $if\ f(x_1, x_2) \leq 0$: $\boldsymbol{d(x_1, x_2) = -1}$



Basis of Linear Classification

The principle is always the same:

» Find a line/plane/hyperplane that separates data “optimally”«
Only difference between linear classifiers: the optimality criterion

General formula of all linear classifiers:

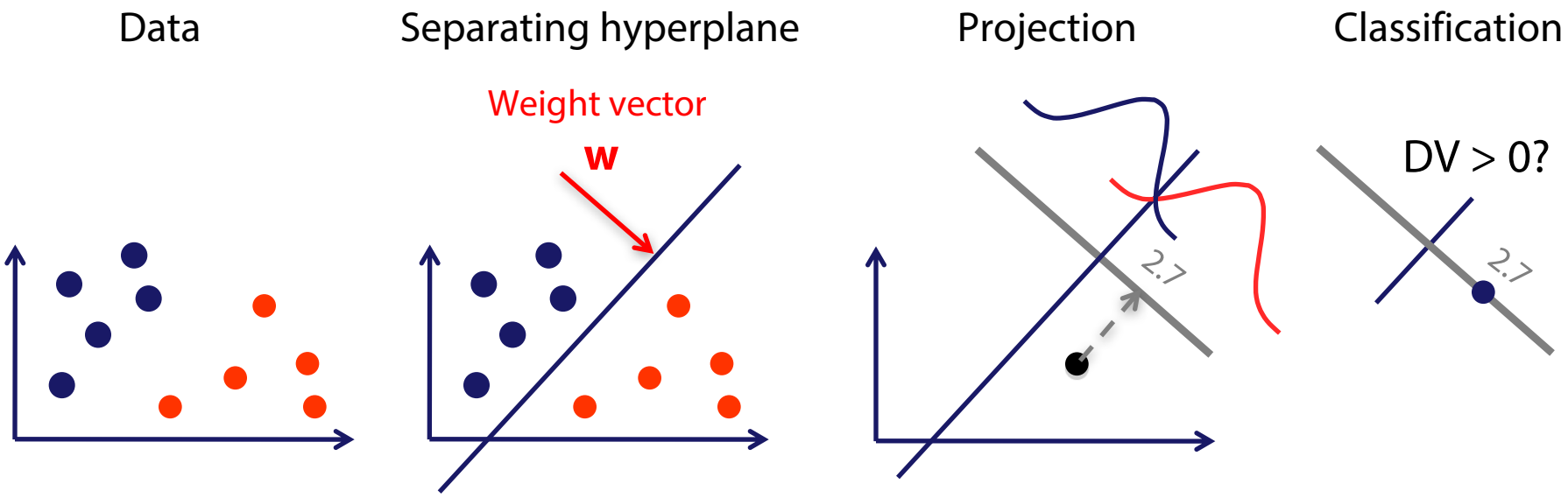
$$f(x) = w^T x + b$$

$$f(x) = \sum_{1}^{p} w_i x_i + b = w_1 x_1 + w_2 x_2 + \cdots + b$$

Linear classification is projection on weight vector!

Basis of Linear Classification

Geometric intuition

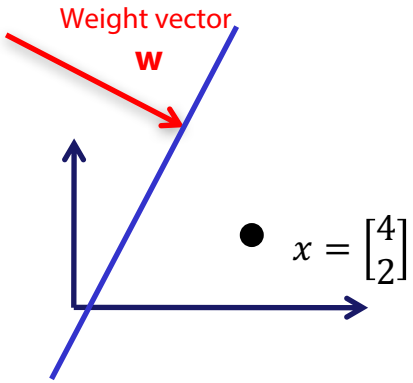


$$f(x) = w^T x + b$$

Basis of Linear Classification

Example

Hyperplane



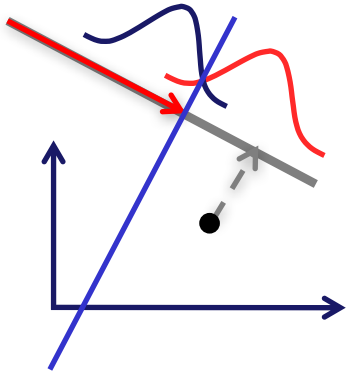
Given this weight vector

w_1=1.5

w_2=–0.7

$$\boldsymbol{w} = \begin{bmatrix} 1.5 \\ -0.7 \end{bmatrix}$$

Projection



Calculate decision value

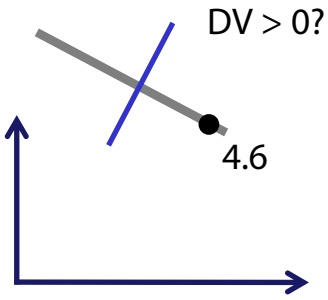
$DV = \mathbf{w}^T\mathbf{x}+b$

$DV = w_1x_1 + w_2x_2 =$

$= 1.5 \times 4 + -0.7 \times 2$

$= 6 - 1.4 = \underline{4.6}$

Classification



Decision rule

If DV < 0: Blue class
If DV > 0: Red class

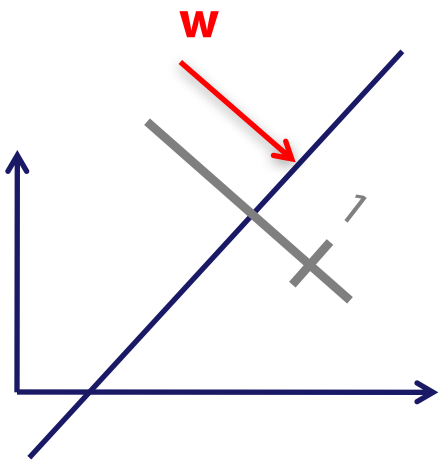
Here:
DV = 4.6 > 0: **Red class**

Basis of Linear Classification

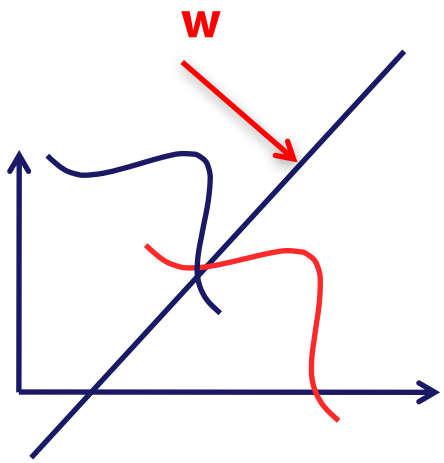
Quiz

Where else is DV = 1?

Where 0? –3.2?



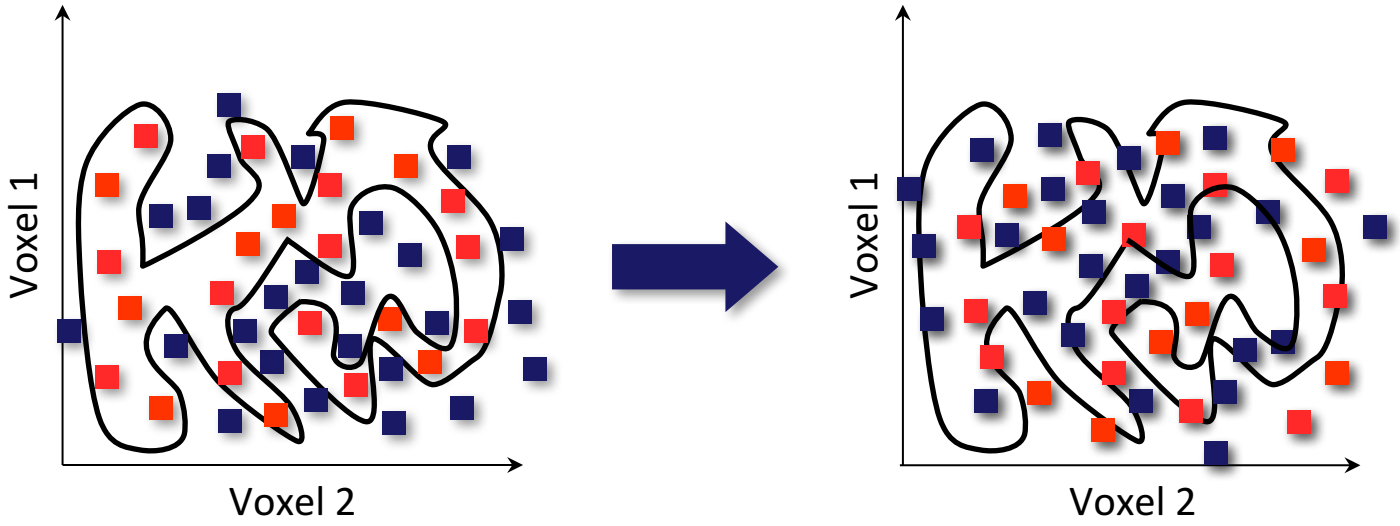
What does the constant b do to the separation bound?



$$f(x) = w^T x + b$$

ESTIMATING CLASSIFIER PERFORMANCE

Why Train and Test a Classifier?



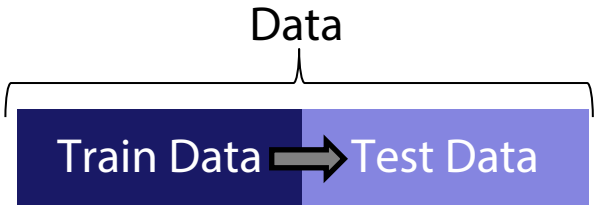
- Goal of classification: Finding a model that **generalizes** beyond noise in the data
- Way of testing generalization: Training and testing classifier

How to Split Data for Training and Testing?

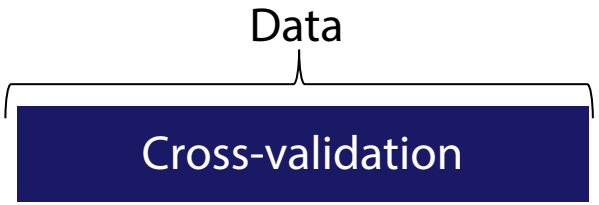
Problem: We need to both…

- …maximize size of training data for better model fit
- …maximize size of test data for precise generalization estimate

When data are not scarce not a problem:



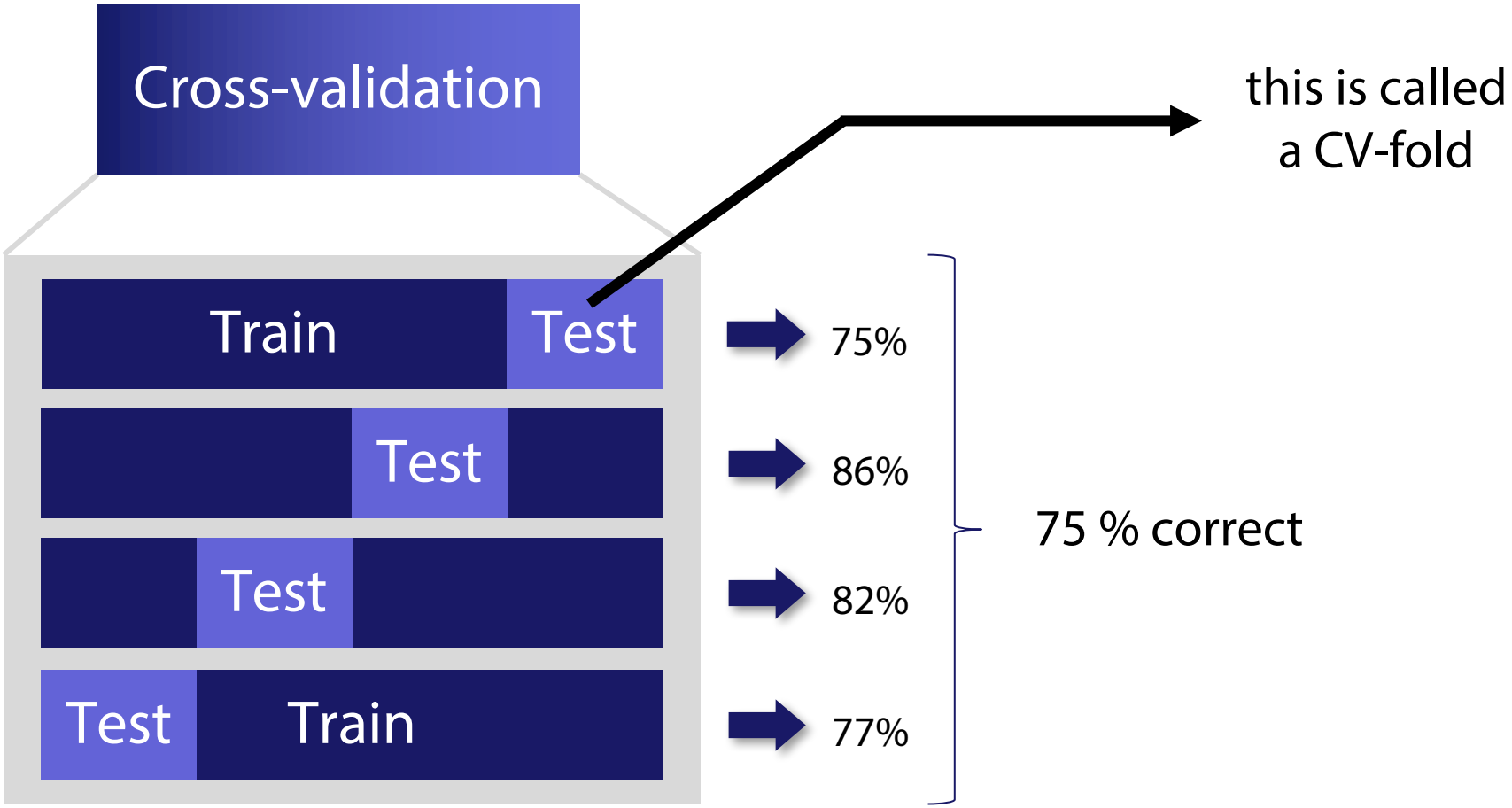
When data are scarce:



Most people in neuroimaging use cross-validation

Cross-validation

Efficient re-use of data for training and testing



Cross-validation

Advantages of cross-validation

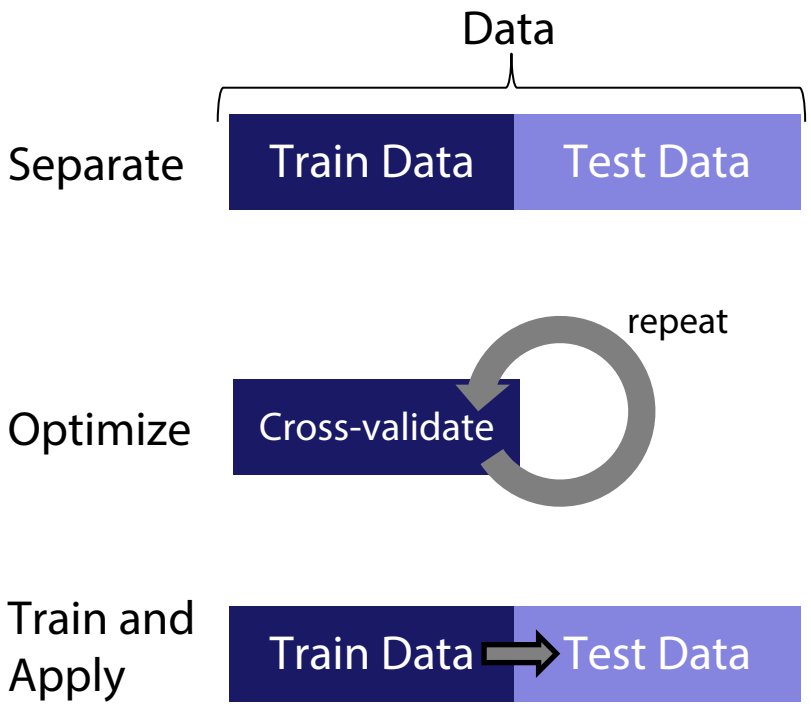
- Way of achieving non-optimistic estimate of information content
- Distances between classes are unbiased estimates

Disadvantages of cross-validation

- Re-use of training data increases the variance of accuracies → cannot run classical statistical test on cross-validation results
- Assumption of stationarity across folds

Prediction vs. Interpretation Revisited

Prediction



Use trained model in future for prediction

Interpretation (usual approach)

do not repeat!

Don't use trained model in future

Prediction: Use cross-validation for optimization

Interpretation: Use cross-validation for "data augmentation"

Classification Measures

Most typical measure: Classification accuracy

- Useful in many cases
- Not so useful when classes have different sizes
- Discrete results

More sophisticated measure: AUC

- Calculates information content irrespective of classifier's preference for one class
- Looks like continuous results but discrete as well (rank-based)

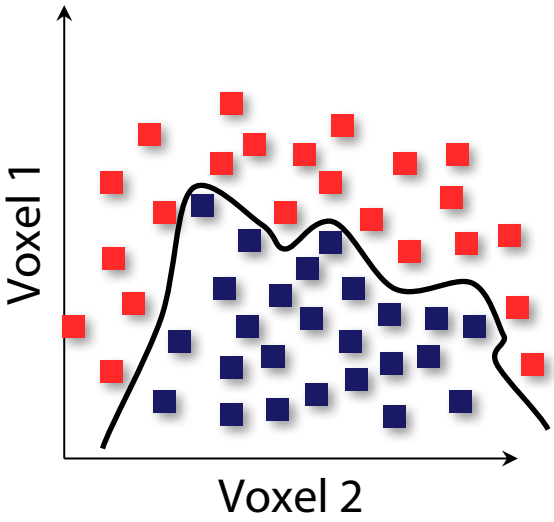
For unbalanced test data: Balanced accuracy

- Calculates accuracy of each class separately
- Combines accuracies together afterwards

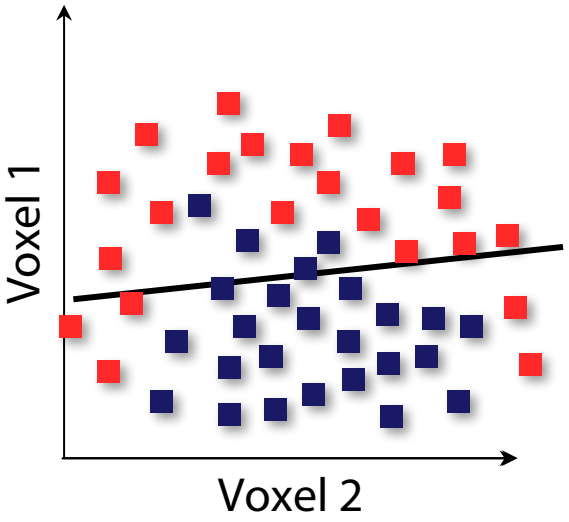
BIAS-VARIANCE TRADE-OFF

Bias-Variance Trade-Off

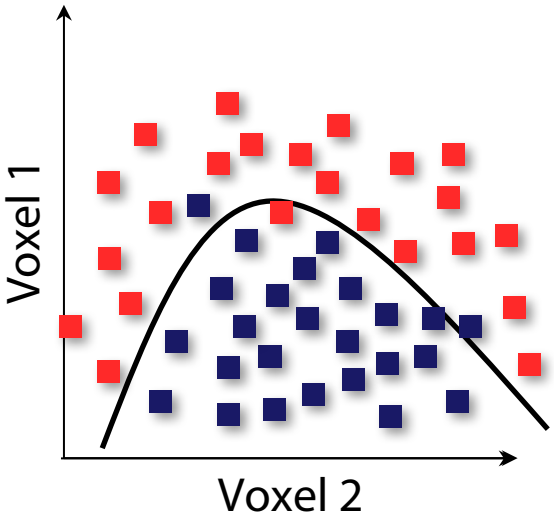
What is the best classifier for this data?



Overfitting



Underfitting



Good Fit

Goal: Best possible generalization to new data

Bias-Variance Trade-Off

Two goals in machine learning / statistics:

1. Accurately describe structure in data with model
2. Find model that generalize to the population

→ **Problem:** We always have only limited data and don't know what is structure in data and what is noise

→ Bias-variance trade-off matters when:

- there are many different variables (e.g. features in classification, regressors in GLM)
- there is limited data
- the variables (e.g. features, regressors) are correlated

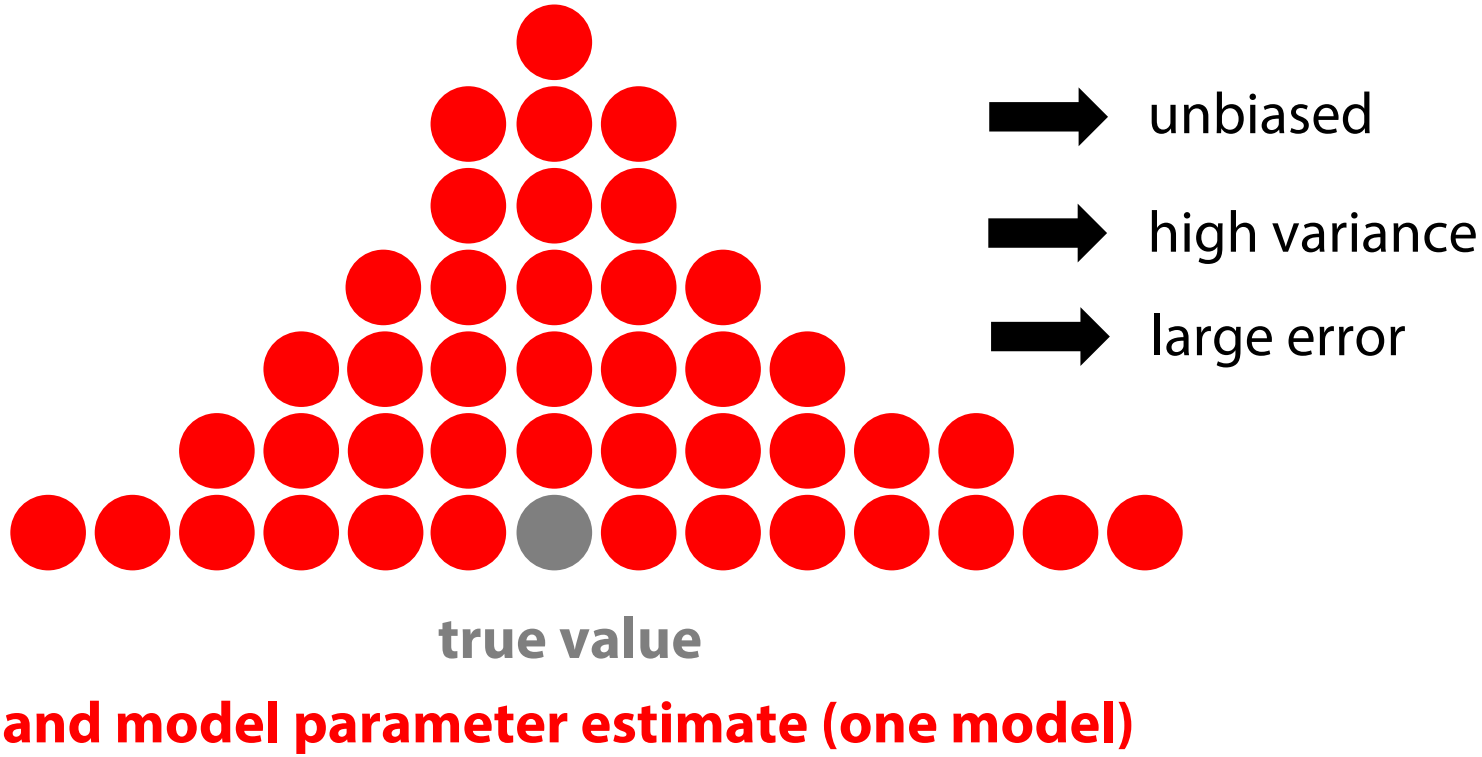
Bias-Variance Trade-Off

Thought experiment: We know the true state of the world but still run lots of experiments to see if our statistical model captures it

true value

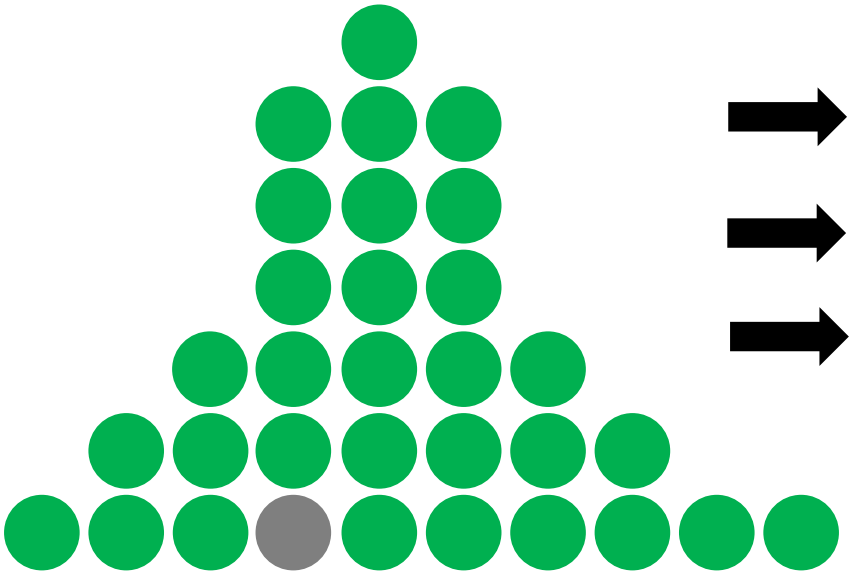
Bias-Variance Trade-Off

Thought experiment: We know the true state of the world but still run lots of experiments to see if our statistical model captures it



Bias-Variance Trade-Off

Thought experiment: We know the true state of the world but still run lots of experiments to see if our statistical model captures it



slight bias

lower variance

smaller error

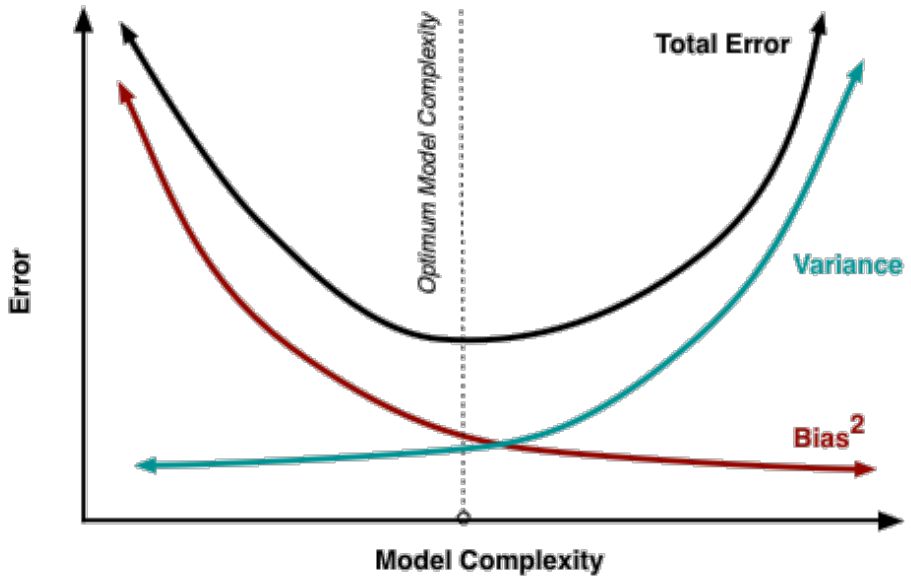
true value

and model parameter estimate (different model)

Bias-Variance Trade-Off

Bias-variance trade-off: Trade-off of model complexity

- Goal: Add some bias and give up some interpretability for much lower variance and lower prediction error



Bias-Variance Trade-Off

Bias-variance trade-off: Trade-off of model complexity

Model prediction error: $E[(\mathrm{y} - \hat{f}(x))^2]$

- y is true state plus noise, $\hat{f}(x)$ is our estimate based on the chosen model (which may be a bad model) based on data x

Prediction error can be rewritten as: $\boldsymbol{\sigma^2 + \mathbf{Bias}[\hat{f}(x)]^2 + Var[\hat{f}(x)]}$

σ^2 ← irreducible error, caused by noise in the data

$\mathrm{Bias}[\hat{f}(x)] = E[\hat{f}(x) - f(x)]$ ← expected difference between our estimated model and the true model

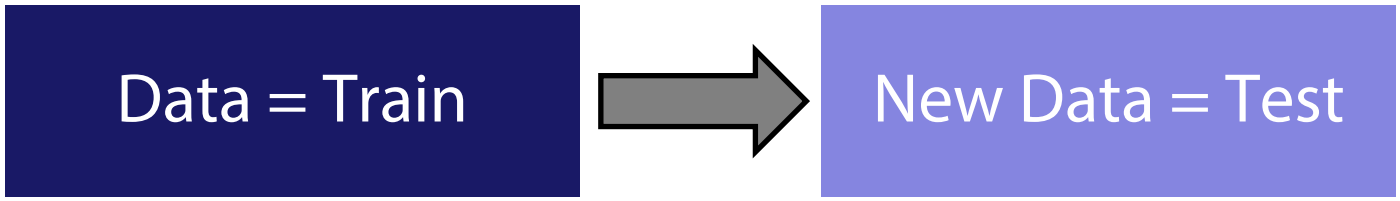
$\mathrm{Var}[\hat{f}(x)] = E[(\hat{f}(x) - E[\hat{f}(x)])^2]$ ← expected variance of our estimated model, equivalent to the squared difference between the estimated model and the mean of all estimated models

Bias-Variance Trade-Off

Bias-variance trade-off: Find a good compromise

Underfitting: Model doesn't fit training data and doesn't predict well
Overfitting: Model fits training data *too* well and doesn't predict well
Good fit: Model fits training data ok but predicts new data well

Question: How can we know that we are underfitting or overfitting?



Regularization

Adjust model complexity

More regularization: Lower complexity, i.e. more bias, less variance

Less regularization: Higher complexity, i.e. less bias, more variance

Example: Linear regression vs. ridge regression

Linear regression error: $\sum(y - \hat{y})^2 = \sum(y - x^T ß)^2$

Ridge regression error: $\sum(y - x^T ß)^2 + \lambda_r ||ß||^2$

LASSO error: $\sum(y - x^T ß)^2 + \lambda_l ||ß||$

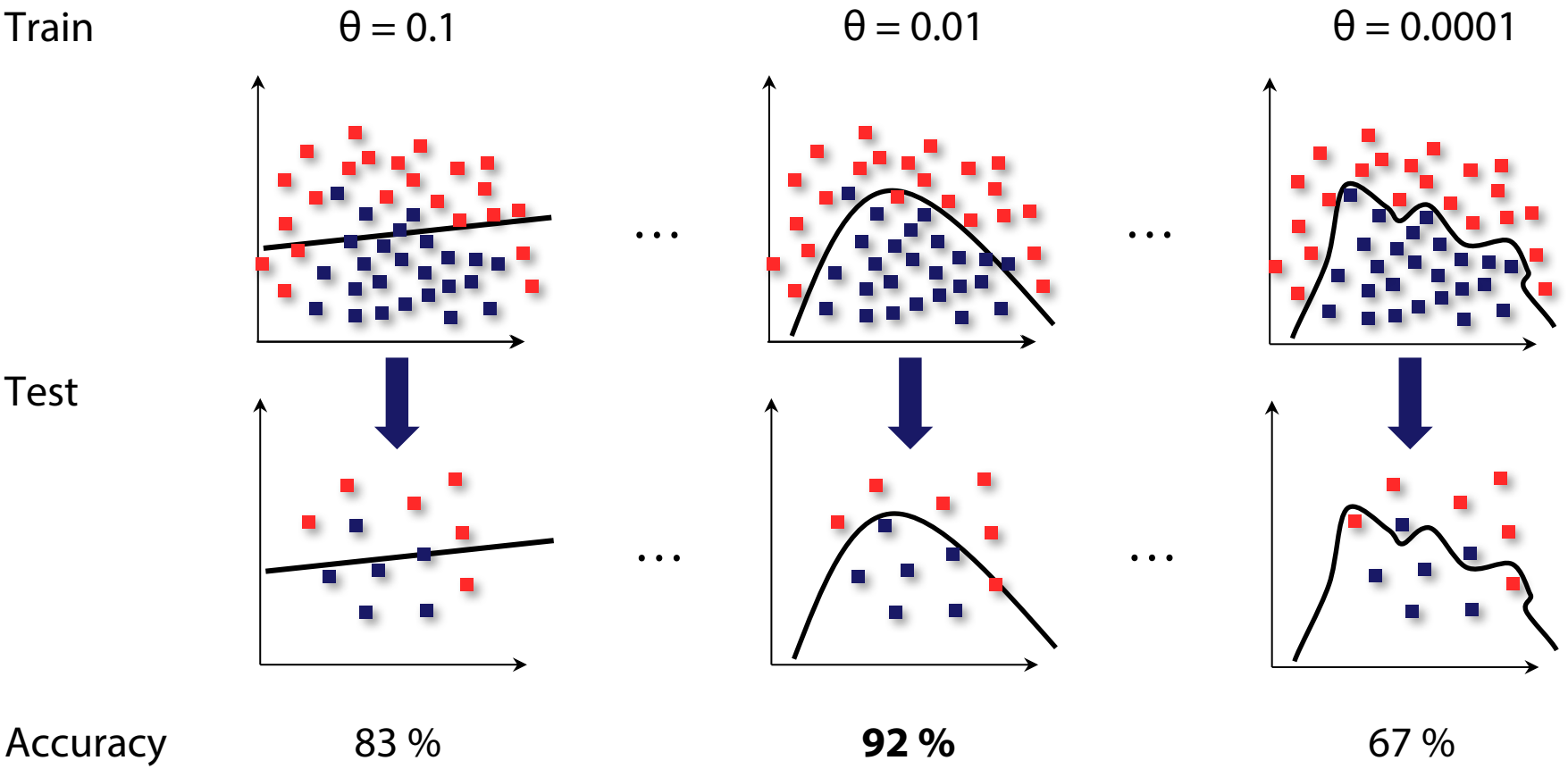
Elastic Net error: $\sum(y - x^T ß)^2 + \lambda_r ||ß||^2 + \lambda_l ||ß||$

→ hyperparameter λ downweights large betas = shrinkage

→ model fit to training data is worse, but possibly better generalization to test data

Training and Testing Classifier

Example

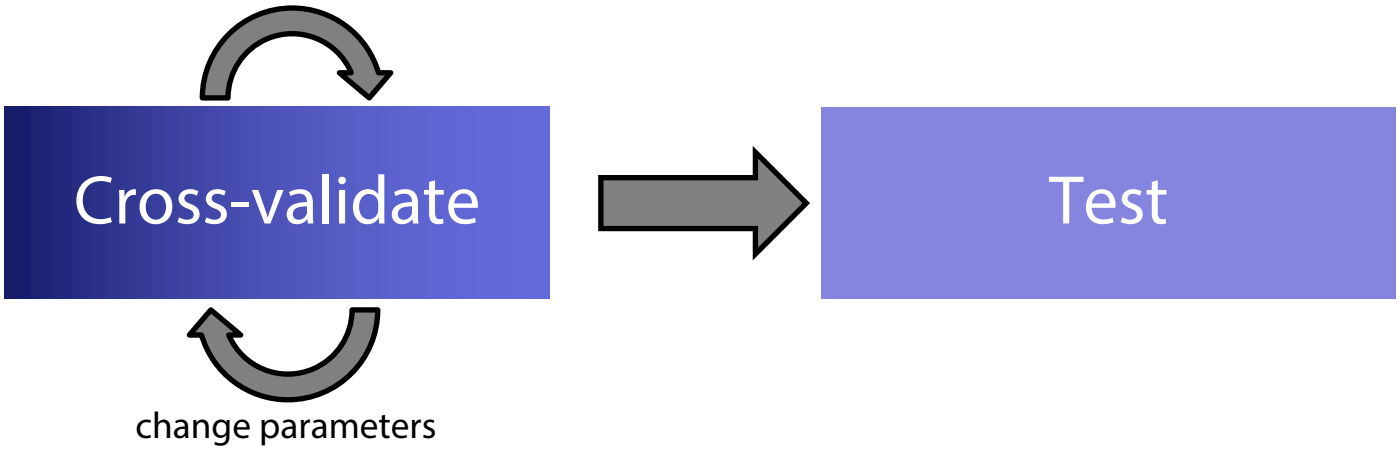


Training and Testing Classifier

Problem: Repeating training and testing is overfitting

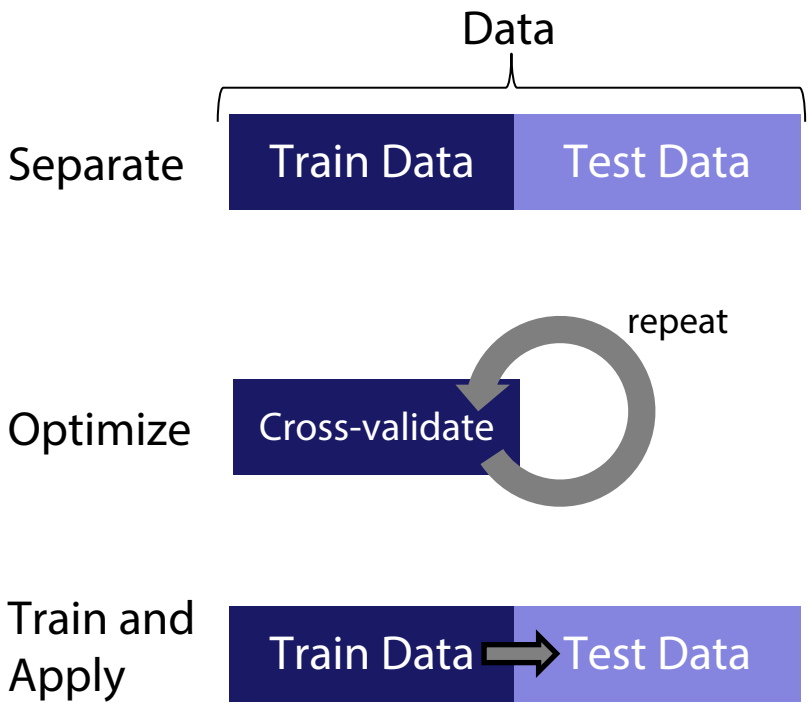
Imagine you try all possible hyperparameters, some will fit test data well by chance, but will not generalize well to even newer data

Solution: Cross-validation on training data only



Prediction vs. Interpretation Revisited

Prediction



Use trained model in future for prediction

Interpretation (usual approach)

do not repeat!

Don't use trained model in future

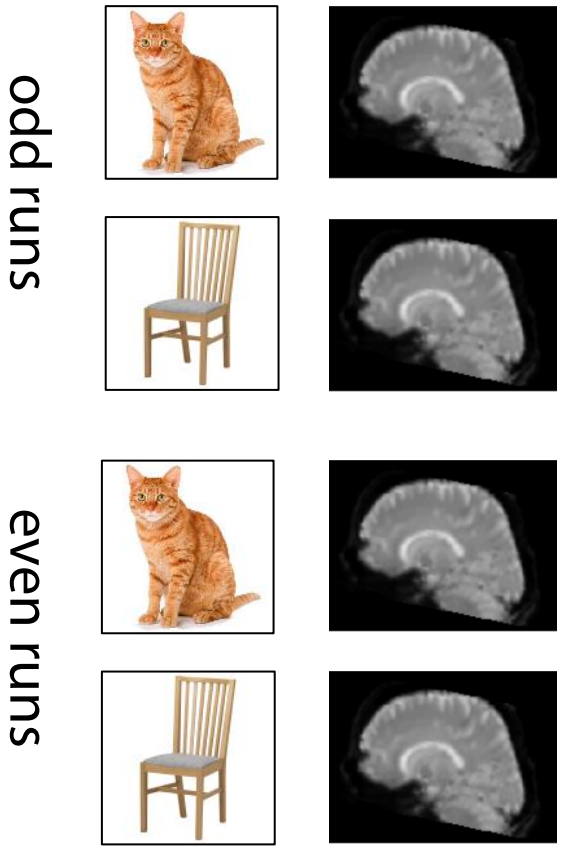
→ hyperparameter optimization possible within cross-validation

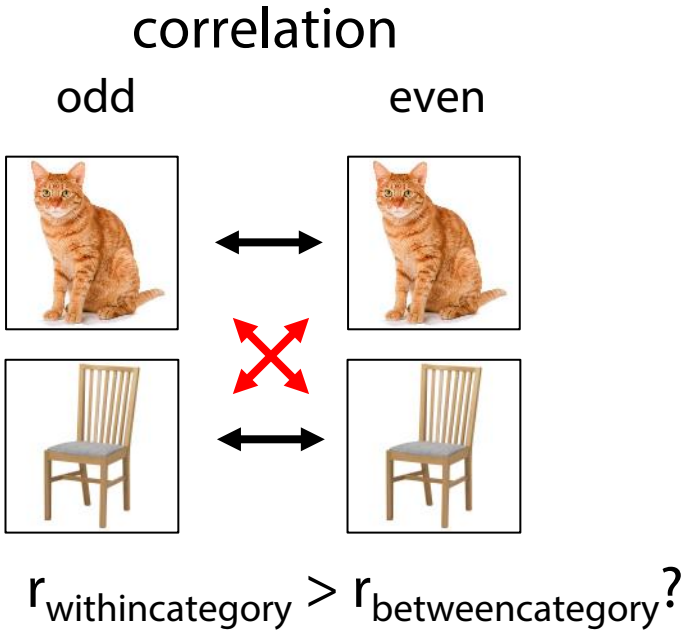
→ this is called nested cross-validation

COMMON CLASSIFIERS

Correlation Classifier

Very simple classifier: find maximal pattern correlation

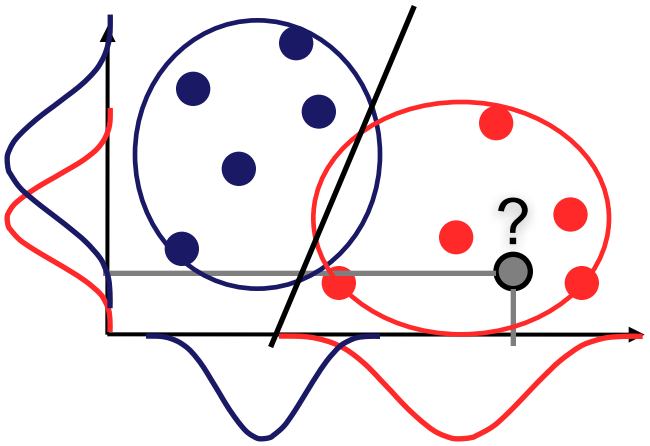




Geometric interpretation: smallest angular distance from centroid

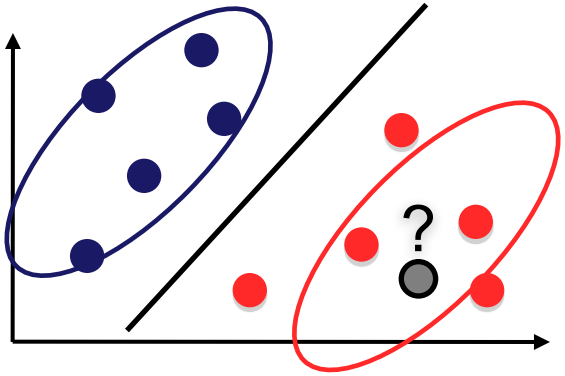
Linear Classifiers

Gaussian Naïve Bayes



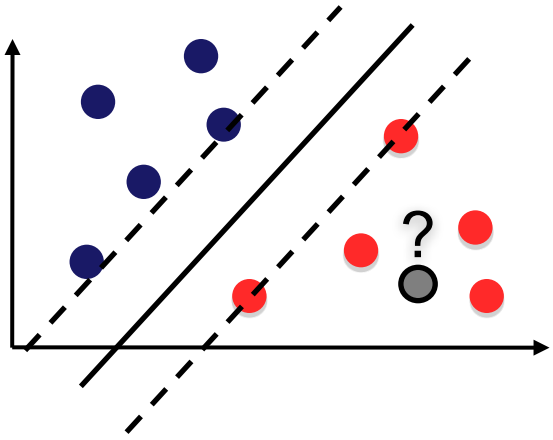
Ignores covariance between voxels

Linear Discriminant Analysis



Considers covariance between voxels

Support Vector Machine



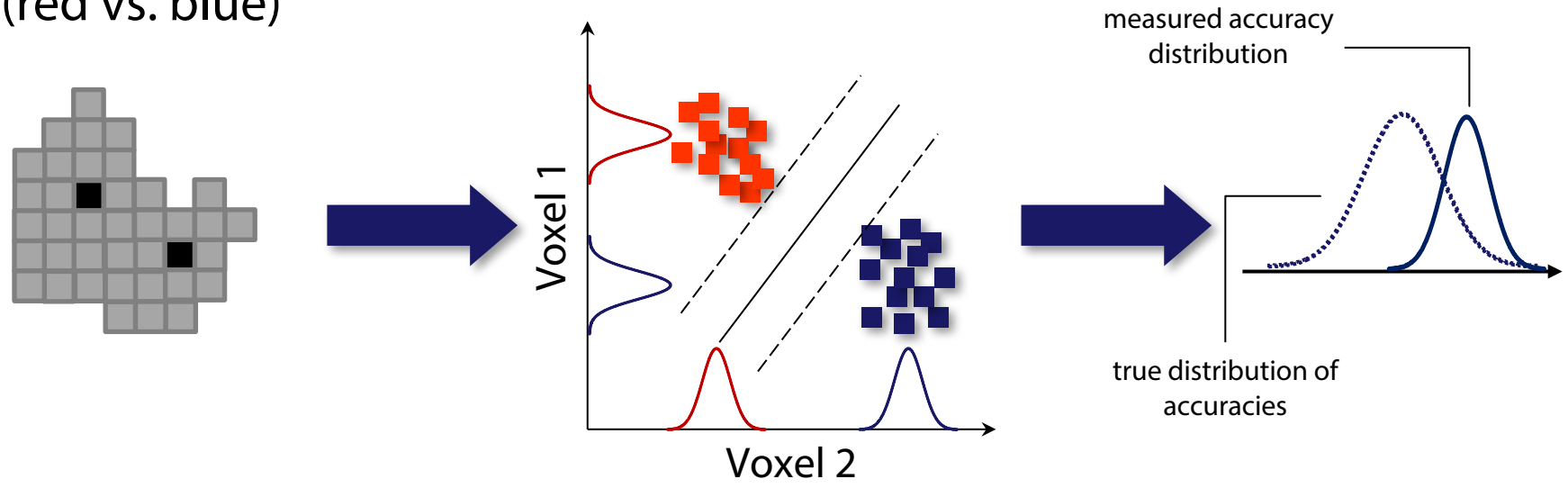
Maximizes margin (distance between closest points of different classes)

NON-INDEPENDENCE AND CIRCULAR ANALYSIS

Non-independence and Circular Analysis

For classification: Information about class label of test set **leaks** to training set (in machine learning: leakage)

Example: Feature selection on all data before classification using label (red vs. blue)



Kriegeskorte et al. (2009) – Nat Neurosci

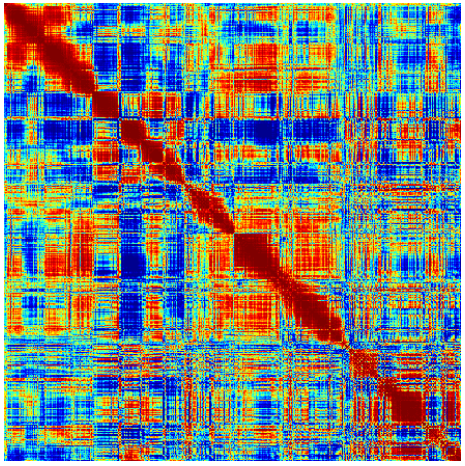
Less Obvious Non-Independence: FMRI Runs

Data in training and test set need to be sampled independently

Two problems for fMRI

- FMRI data even without effect are autocorrelated, i.e. classifier can pick up noise from neighboring samples / trials
- Overlapping fMRI regressors are correlated, i.e. their parameter estimates will be correlated even for large ISI (e.g. 15s)

visual cortex: 4 runs
640 regressors spaced 4 s apart



Mumford et al. (2014) – Neuroimage

Less Obvious Non-Independence: FMRI Runs

Data in training and test set need to be sampled independently

Possible solutions

- Carry out leave-one-run out cross-validation (safest approach)
- Use better autocorrelation models
- Make sure regressors don't overlap
- Make sure the non-independence is the same across all classes
- Use alternative within-run permutation approaches (currently being developed, see Allefeld et al., 2017 – OHBM poster)

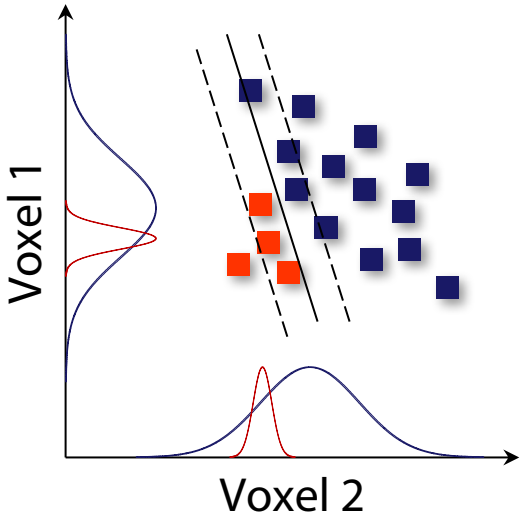
Always ask yourself: If the data are not independent, is the dependence the same across all classes?

Mumford et al. (2014) – Neuroimage

UNBALANCED DATA

Unbalanced Training Data

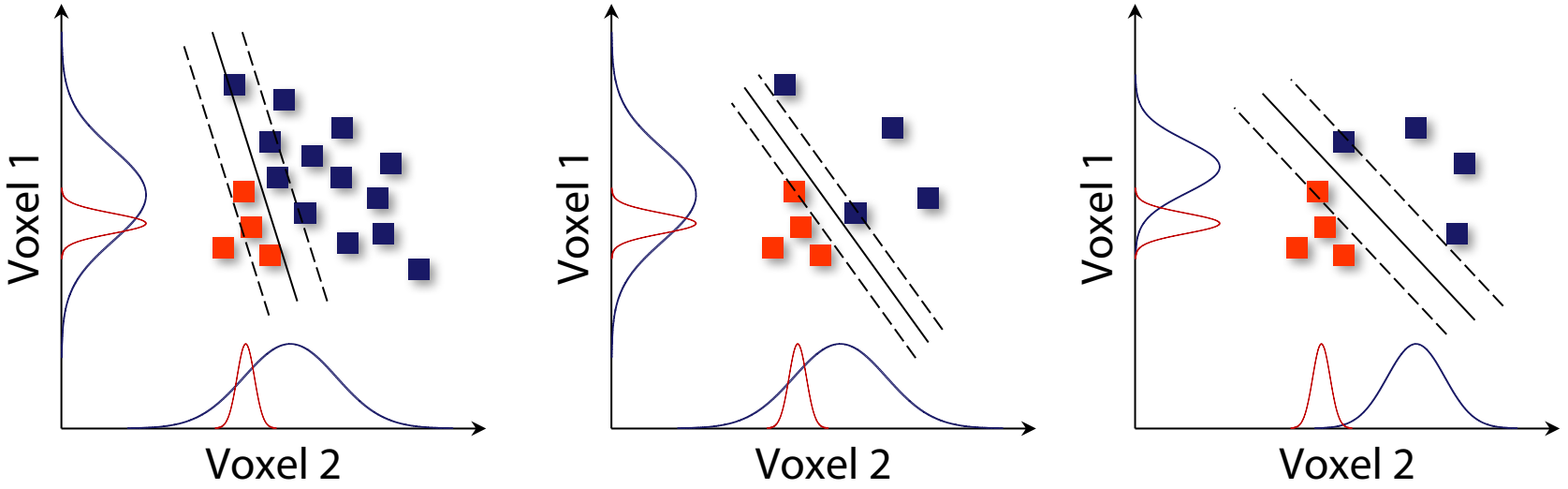
Most classifiers (e.g. soft-margin SVM) prefer the more frequent class



Unbalanced Training Data

Solution (1): Repeated subsampling

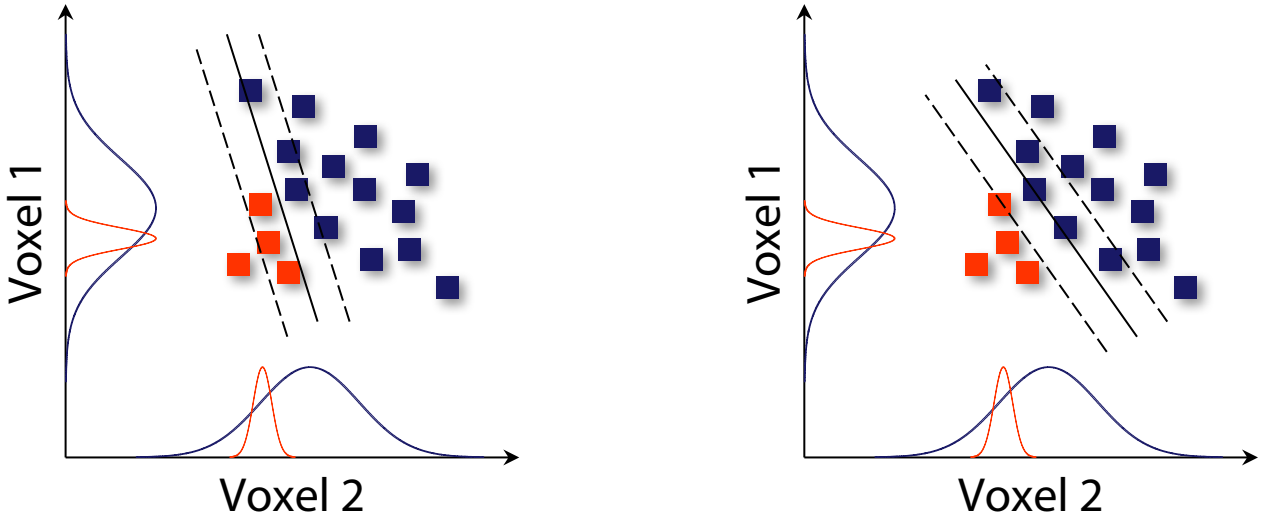
but: computationally intense, uses only part of information



Unbalanced Training Data

Solution (2): Weighted margin

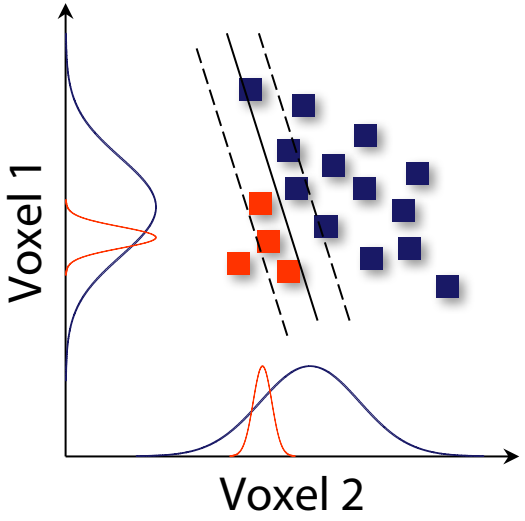
<u>but</u>: only of limited use for $n_{dimensions} >> n_{samples}$

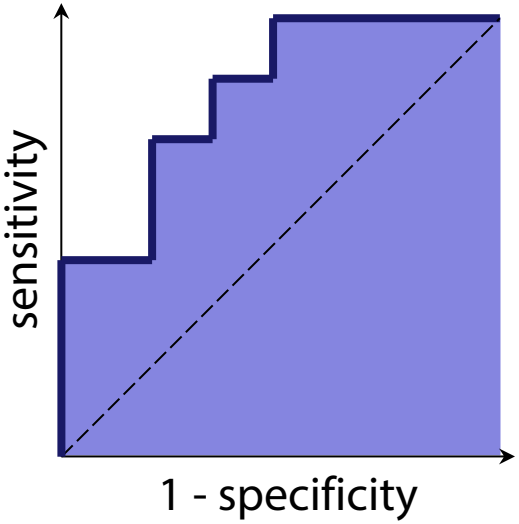


Unbalanced Training Data

Best solution (3): Area under the Curve (AUC)

but: only of practical use when goal presence of information, not prediction as such; might not work for strong imbalance





Summary

- Important terminology: Features, samples, labels, patterns, classifier
- All linear classifiers work the same way
- The bias-variance trade-off optimizes the balance between overfitting and underfitting to training data for good generalization
- Machine learning people use cross-validation for model optimization
- MVPA users use cross-validation mainly to measure information content

Good Textbooks

Hastie et al: Elements of statistical learning

- Good and very deep introduction
- Weak on some topics (e.g. SVM)

James et al: Introduction to statistical learning

- Simpler version of Hastie
- Very good for beginners, but requires some math

Bishop: Pattern Recognition

- Some parts very intuitive
- Other parts quite technical, strong Bayesian focus
- Good coverage of SVMs

Study Questions

Question 1: A colleague comes to you who would like to do between-subject classification (patients vs. controls). What is the assumption that needs to be fulfilled (hint: think of the features…)

Question 2: Can you think of an alternative analysis that avoids this assumption?

Question 3: Your colleague wants to run repeated cross-validation on all of their data to find the best hyperparameters, to avoid overfitting and underfitting. Is this approach valid? If yes, why? If no, why not?

Question 4: Complete this sentence: In bias-variance trade-off we sacrifice _____ of parameters for _____ of the model.