(Almost) All of Machine Learning

Rayid Ghani

Slides liberally borrowed and customized from lots of excellent online sources

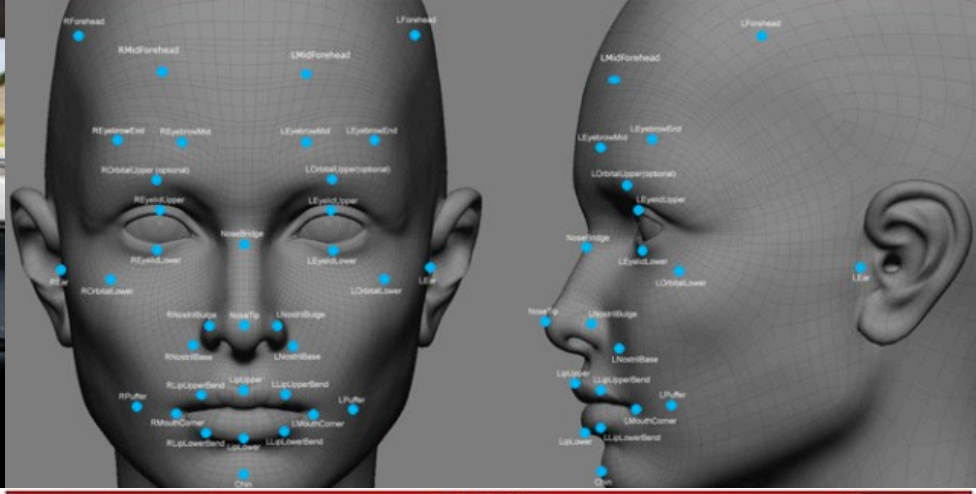
What we'll cover today

- What is Machine Learning
- Examples
- Process
- Methods
- Advanced Topics

After today, you should be able to

- Formulate a real problem as a machine learning problem

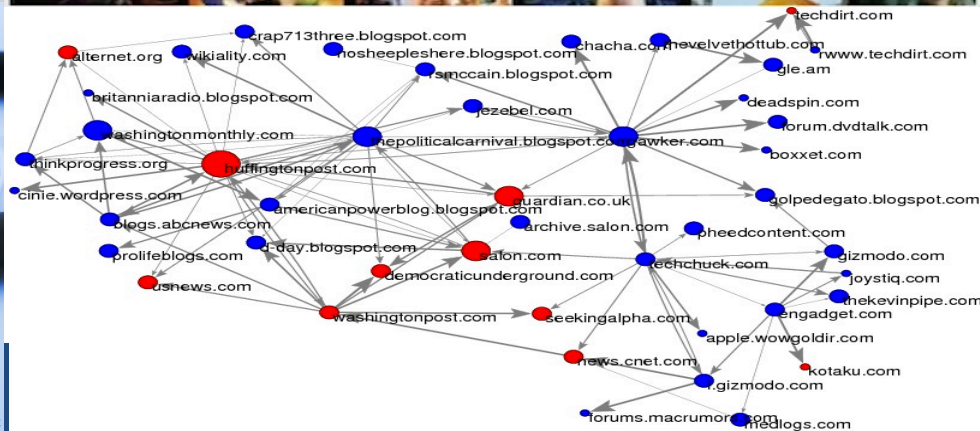
- Understand, use, and evaluate machine learning methods to solve them




fraud


Browse
Just for Kids
NETFLIX
Continue Watching
Rango
Melancholia
Lilyhammer
Top 10 for you
Breaking Bad
DAMAGES
MAD MEN
ABDUCTION
Instant Queue
CRAZY OUTSIDE
LILYHAMMER


EXPLANATION OF BENEFITS
THIS IS NOT A BILL
Billed Charges
Deductible
Copay


techdirt.com
rap713three.blogspot.com
alternet.org
wikiality.com
hosheepleshere.blogspot.com
chacha.com
nevelvethottub.com
www.techdirt.com
gle.am
smccain.blogspot.com
britanniaradio.blogspot.com
jezebel.com
deadspin.com
ashingtonmonthly.com
epoliticalcarnival.blogspot.com
gawker.com
forum.dvdtalk.com
boxxet.com
thinkprogress.org
huffingtonpost.com
cinie.wordpress.com
guardian.co.uk
golpedegato.blogspot.com
americanpowerblog.blogspot.com
blogs.abcnews.com
archive.salon.com
pheedcontent.com
prolifeblogs.com
f-day.blogspot.com
salon.com
techchuck.com
gizmodo.com
democraticunderground.com
joystiq.com
usnews.com
thekevinpipe.com
washingtonpost.com
seekingalpha.com
engadget.com
apple.wowgoldir.com
news.cnet.com
kotaku.com
gizmodo.com
forums.macrumors.com
medlogs.com

Machine Learning

"A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E."

Why Machine Learning?

- Goal: Adaptive, Scalable systems that are cost effective to build and maintain
- Rules-based systems are rigid and expensive
- Availability of lots of data

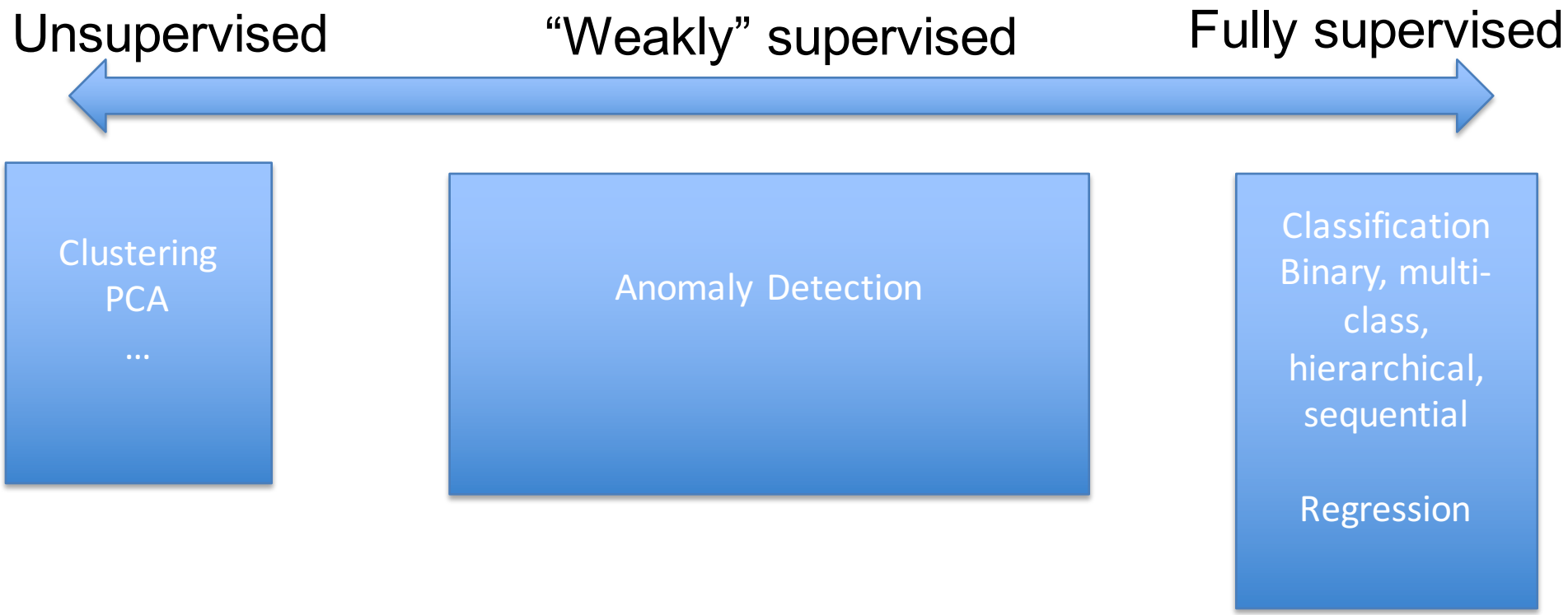
Types of ML tasks for Policy Problems

- Description (Understand the past)
- Prediction (Predict the Future)
- Detection (Anomalies, Events, Patterns)
- Behavior Change (Causal Inference)

Process

- Understand “Business” problem
- Map to Machine Learning problem
- Understand the data
- Explore and Prepare the data
- Feature Development
- Method Selection
- Evaluation
- Deployment

Types of Learning



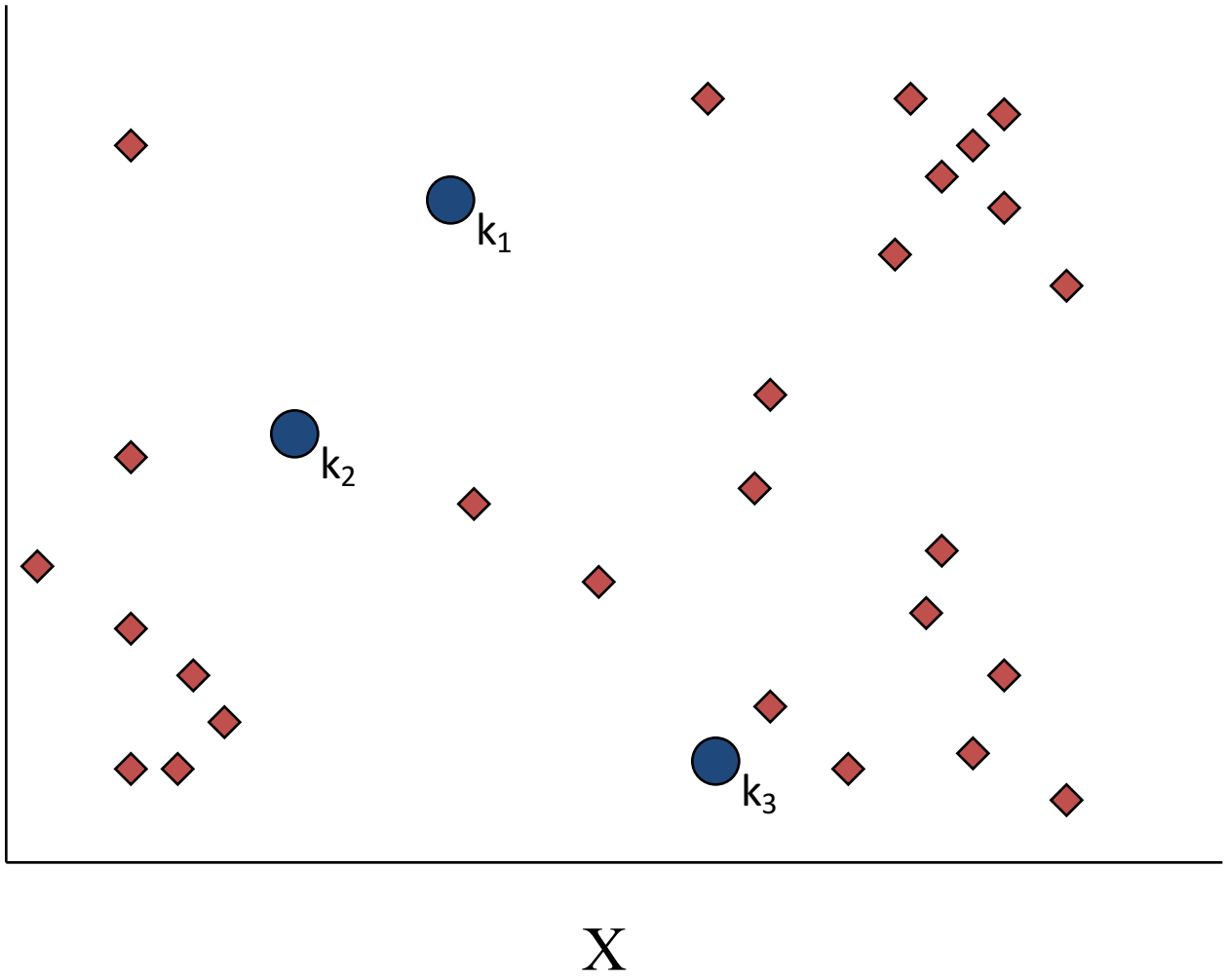
Unsupervised: Clustering

- A good clustering method will produce clusters with
  - High intra-cluster similarity
  - Low inter-cluster similarity

- K-Means is the simplest and the most common algorithm

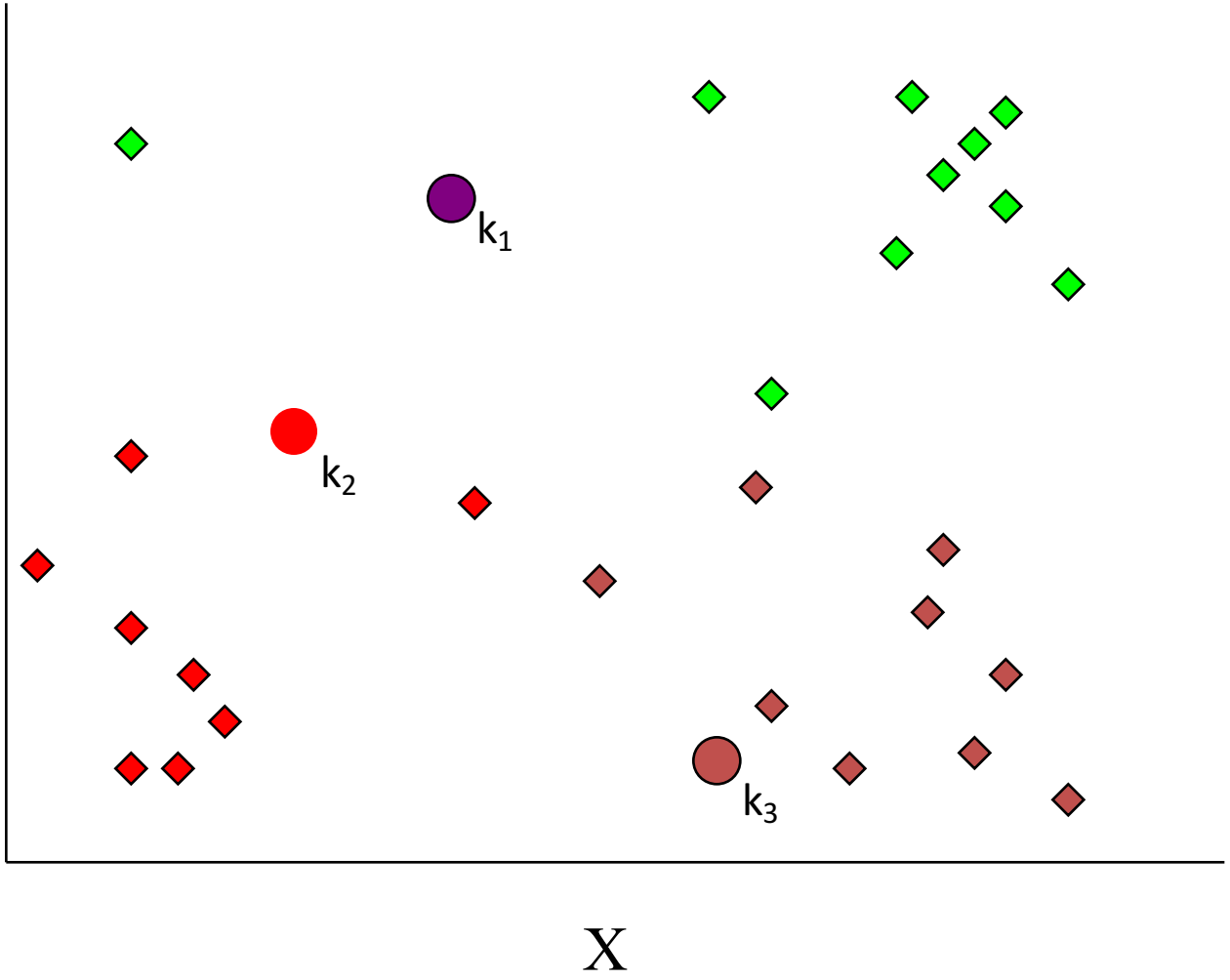
K-means example, step 1

Pick 3 initial cluster centers (randomly)



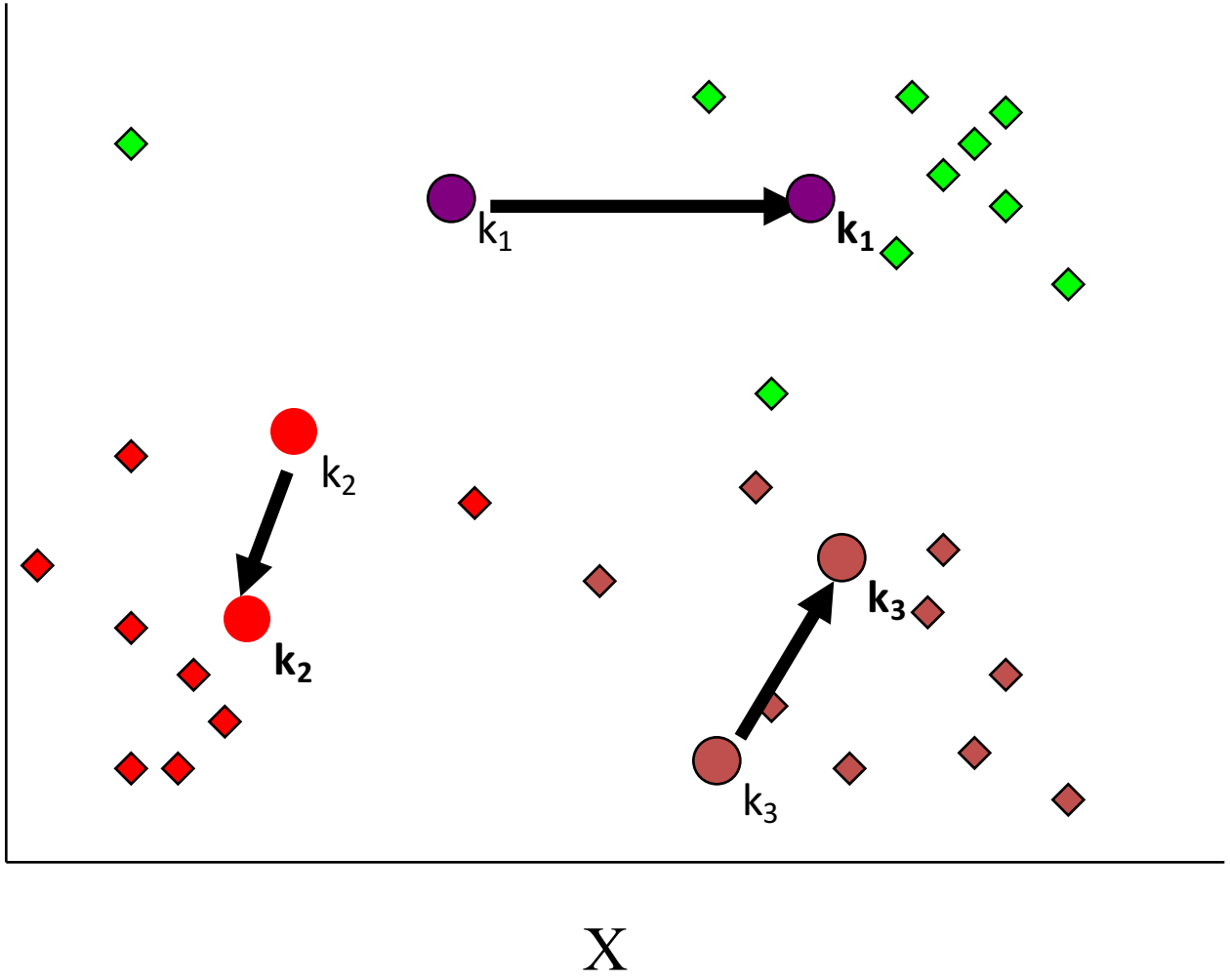
K-means example, step 2

Assign each point to the closest cluster center



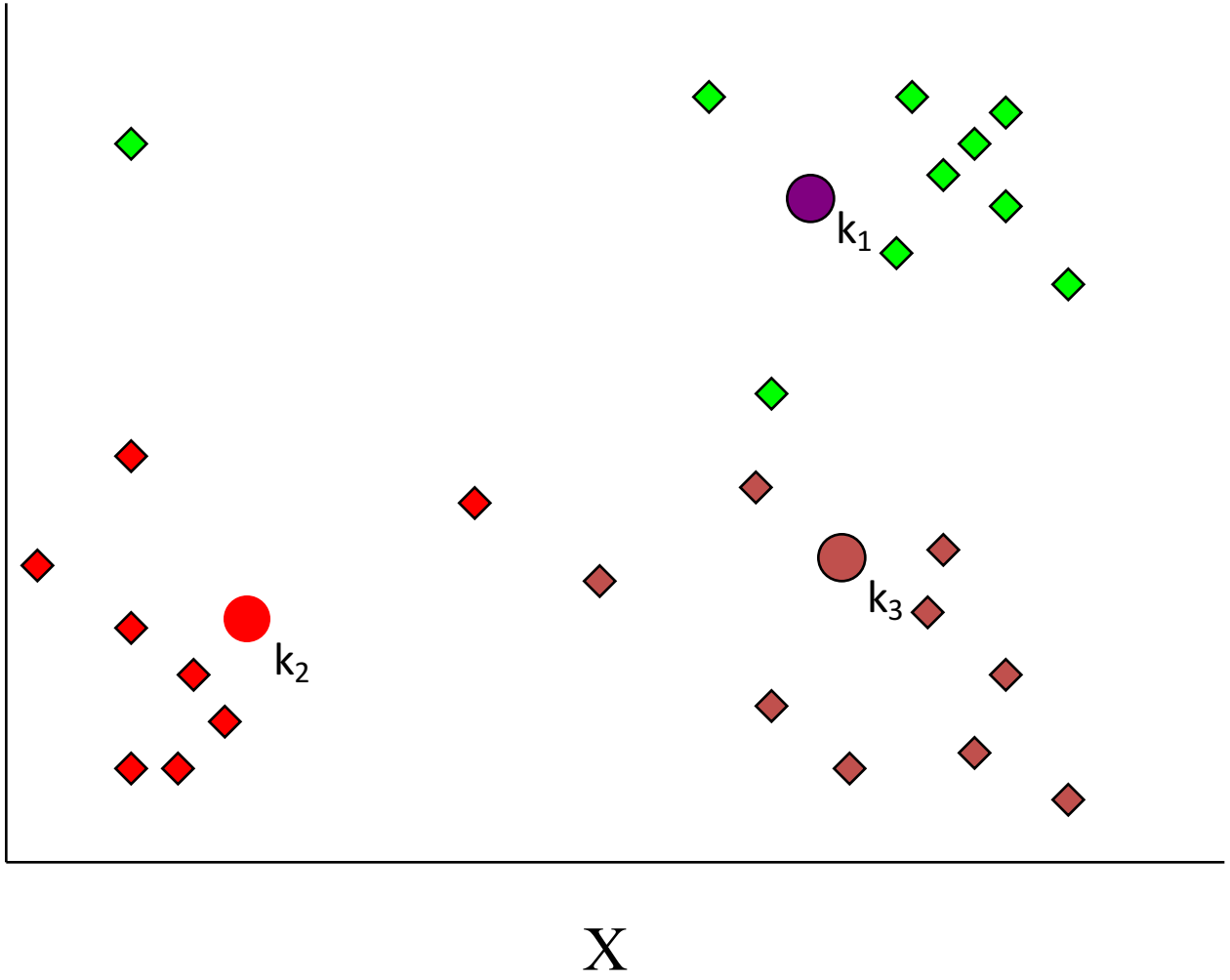
K-means example, step 3

Move
each cluster
center
to the mean
of each cluster

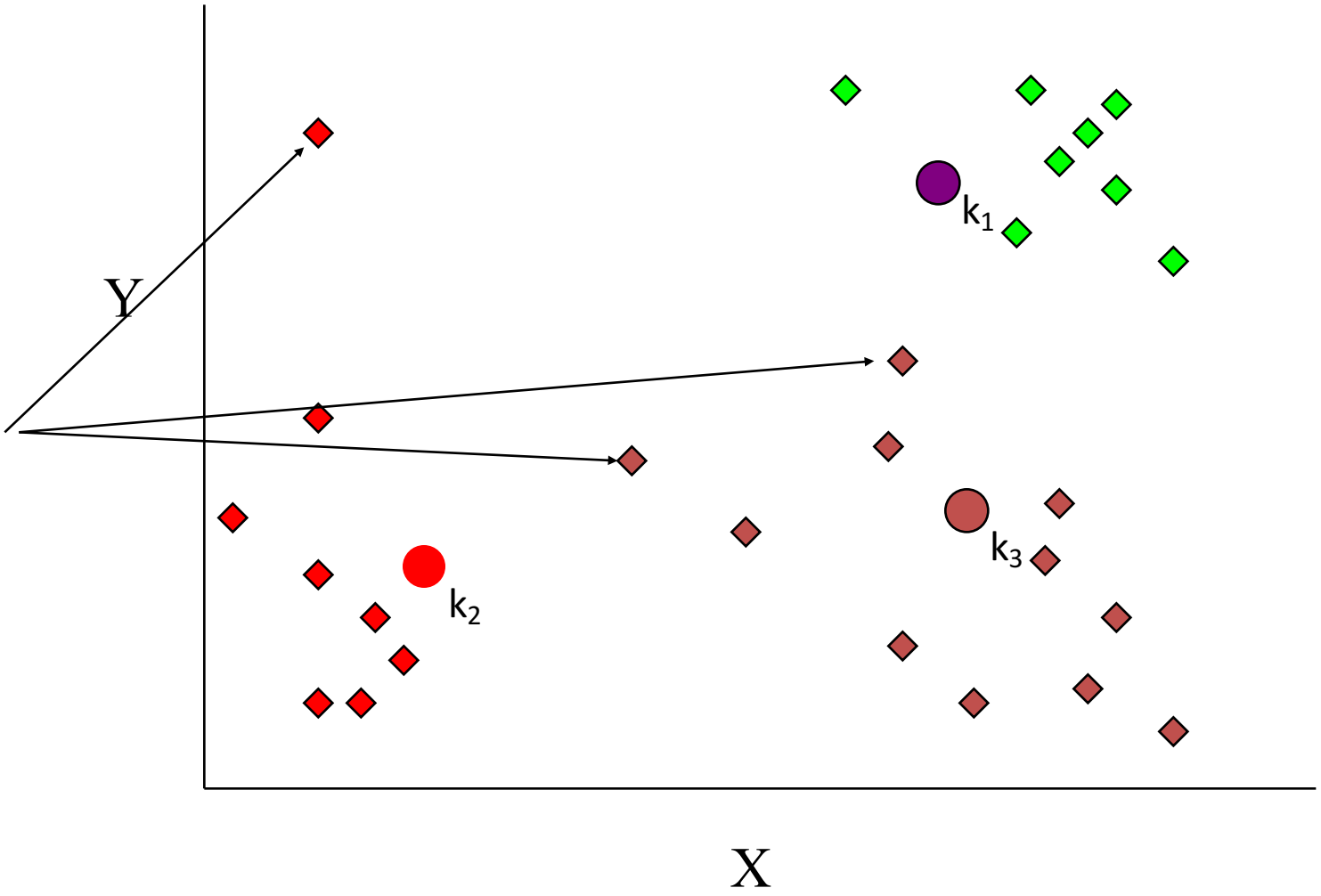


K-means example, step 4

Reassign points closest to a different new cluster center

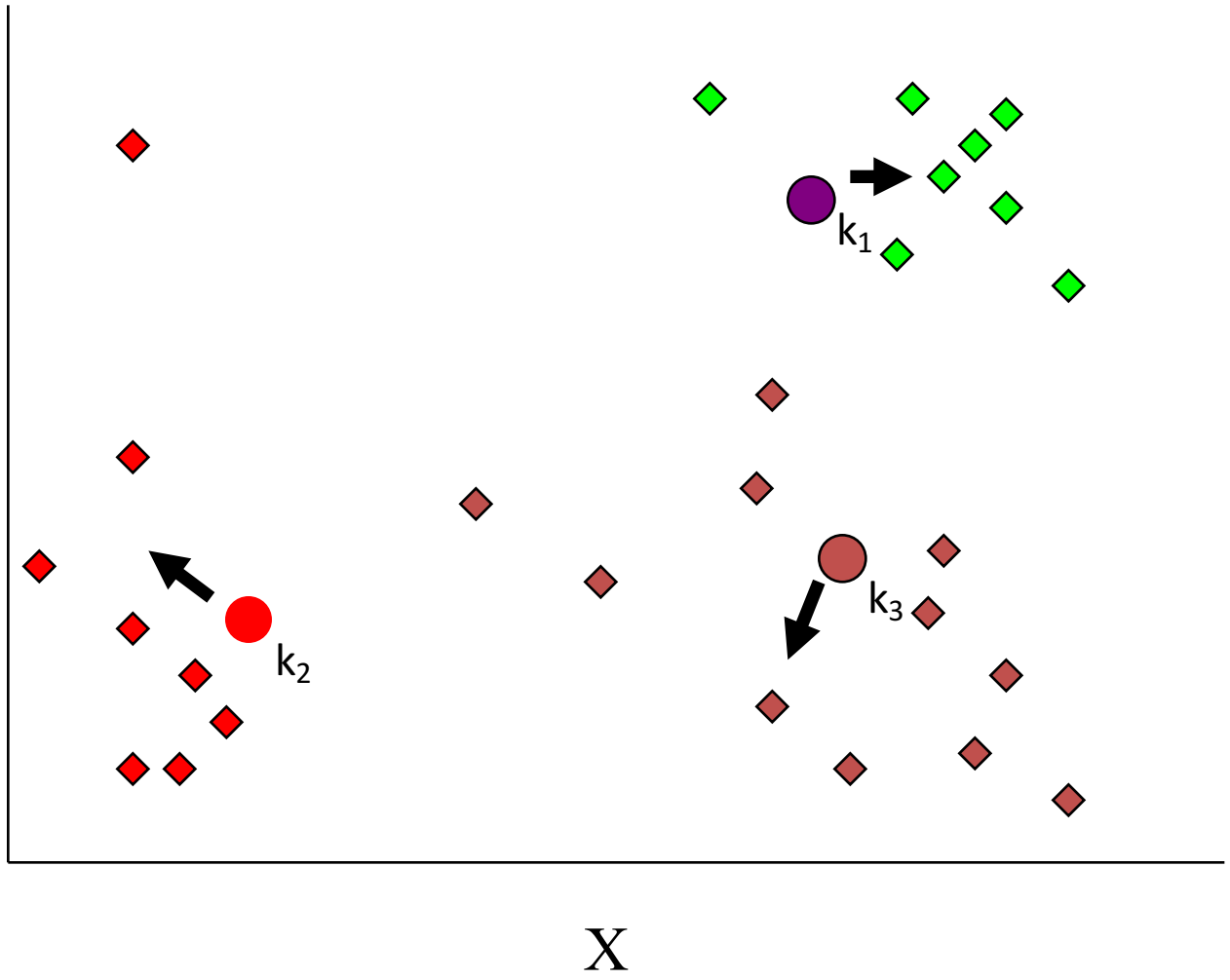


K-means example, step 4 …



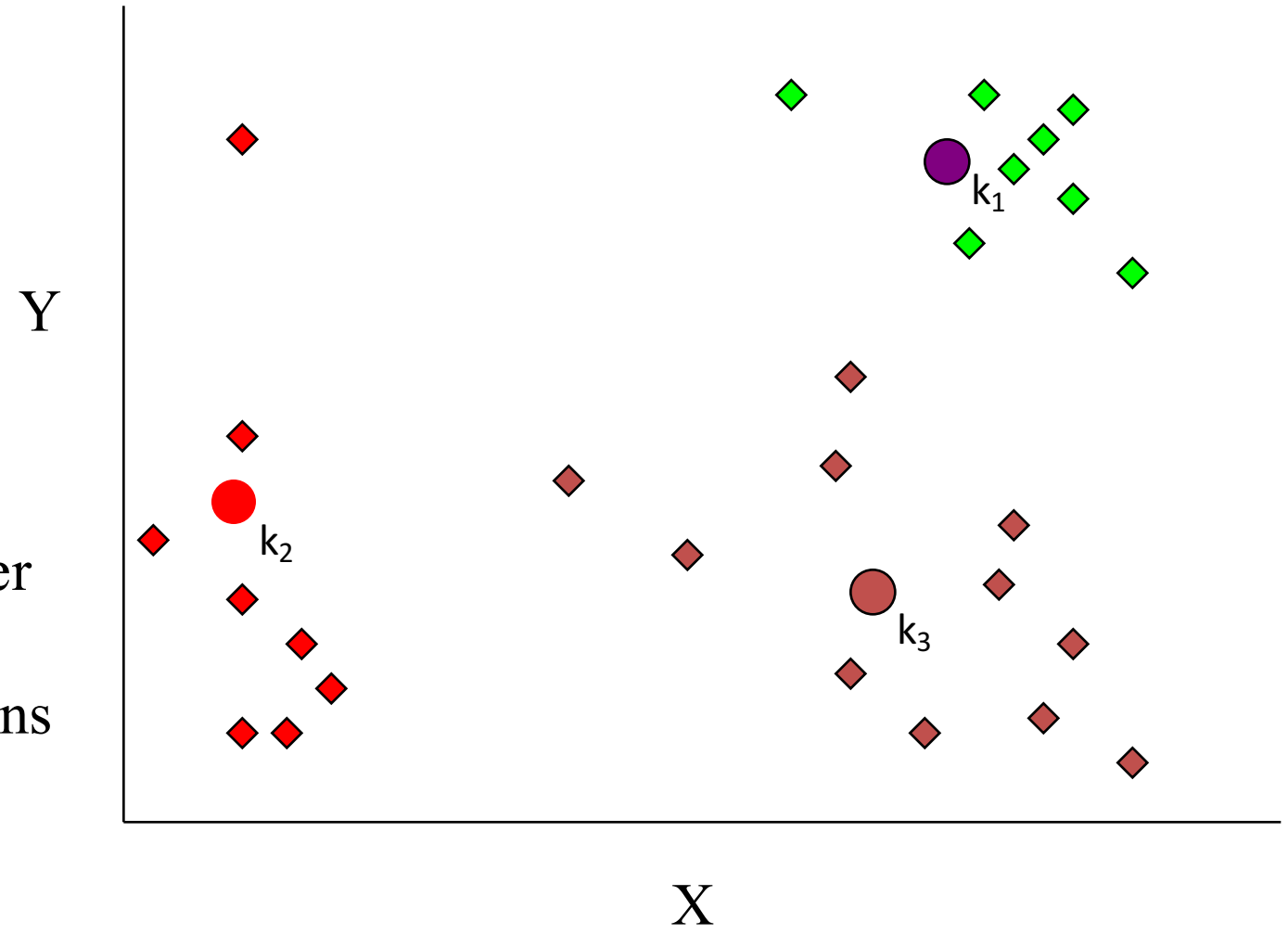
K-means example, step 4b

re-compute cluster means



K-means example, step 5

move cluster centers to cluster means



K-means algorithm

- Given *k*, the *k-means* algorithm works as follows:
  1) Randomly choose *k* data points (seeds) to be the initial centroids, cluster centers
  2) Assign each data point to the closest centroid
  3) Re-compute the centroids using the current cluster memberships.
  4) If a convergence criterion is not met, go to 2).

Pros and Cons

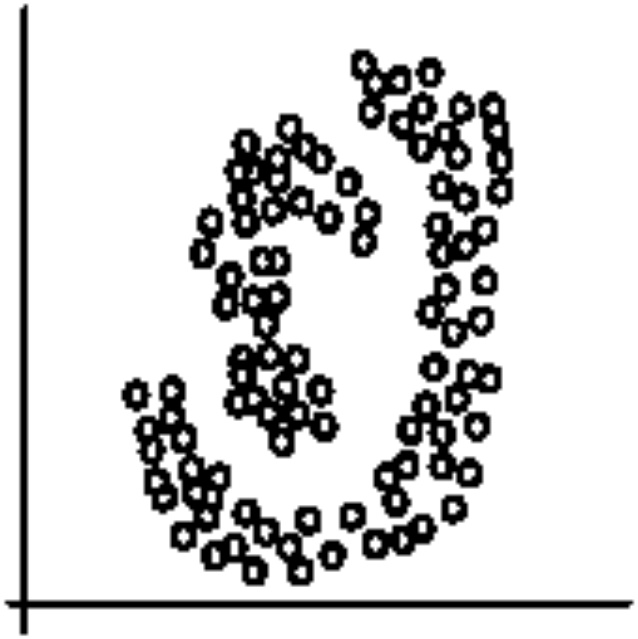
Pros

- Simple
- Efficient

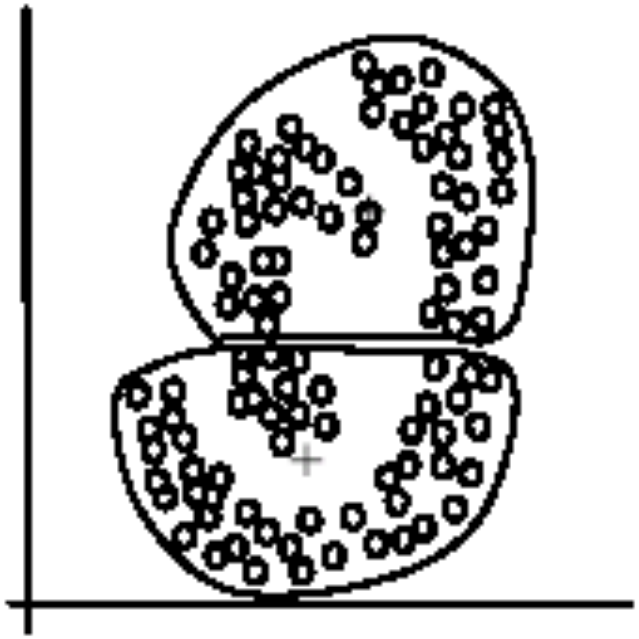
Cons

- Have to select k
- Local minima
- Sensitive to outliers
- Can only group points close in feature space

Can only group points close in feature space



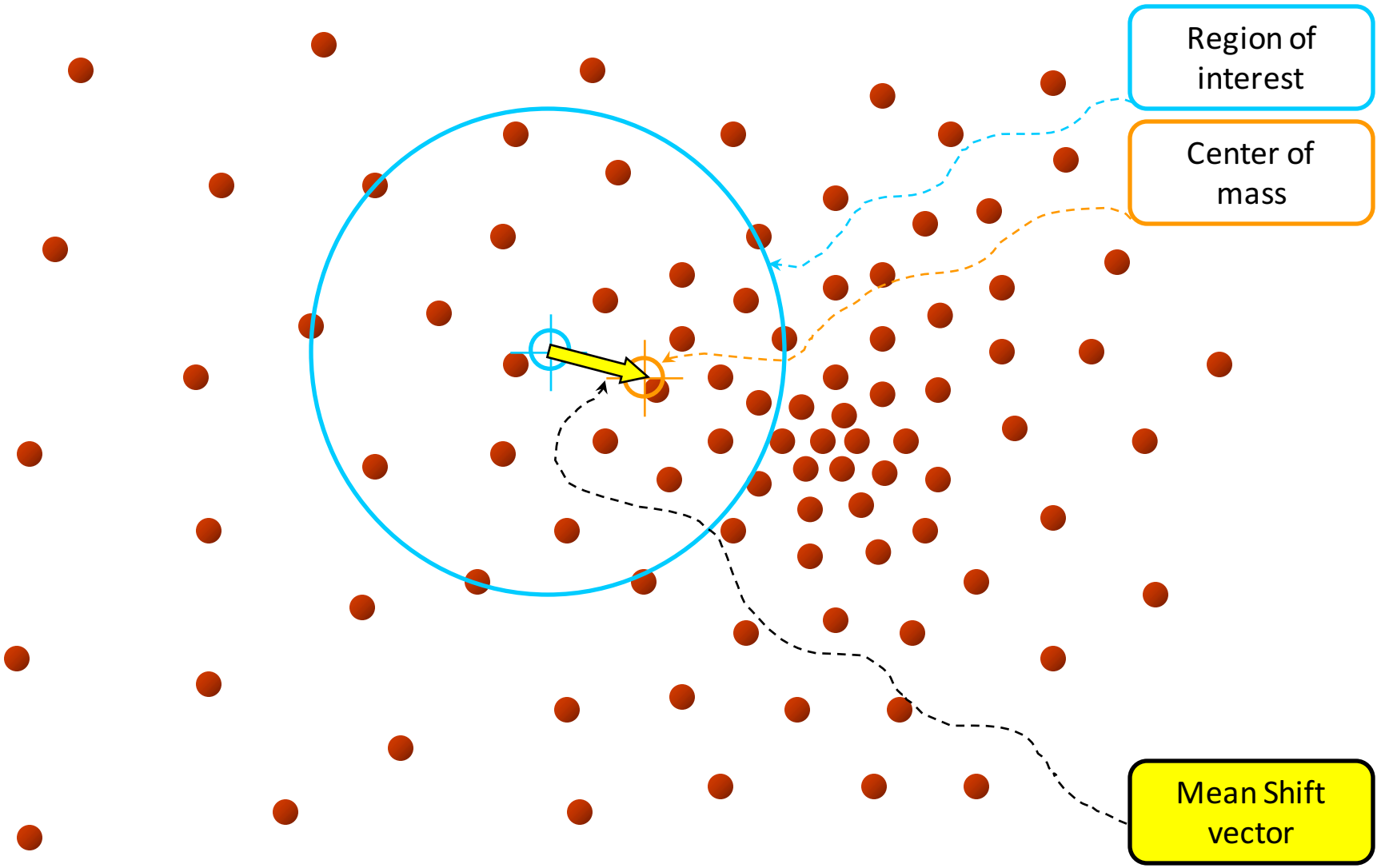
(A): Two natural clusters

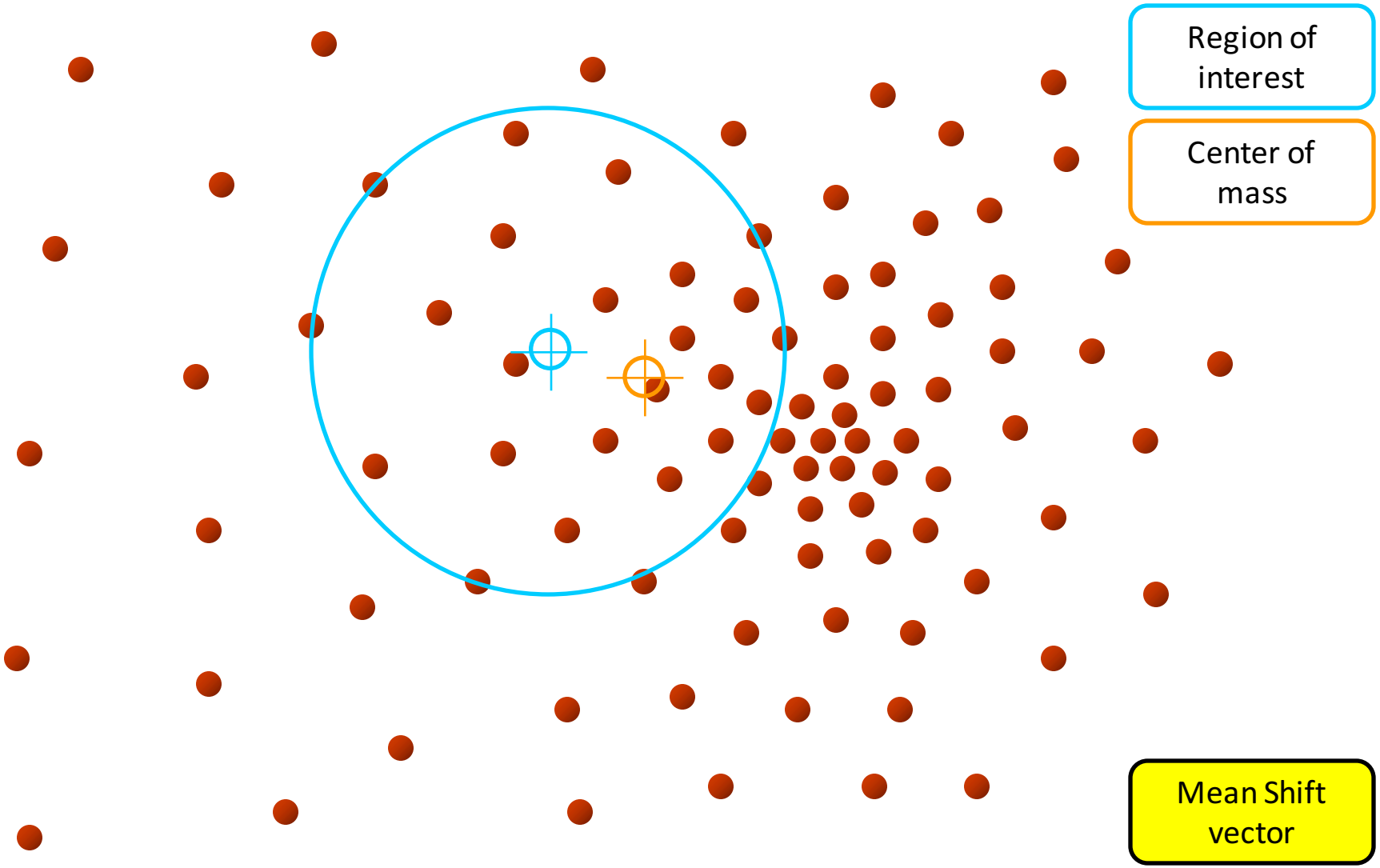


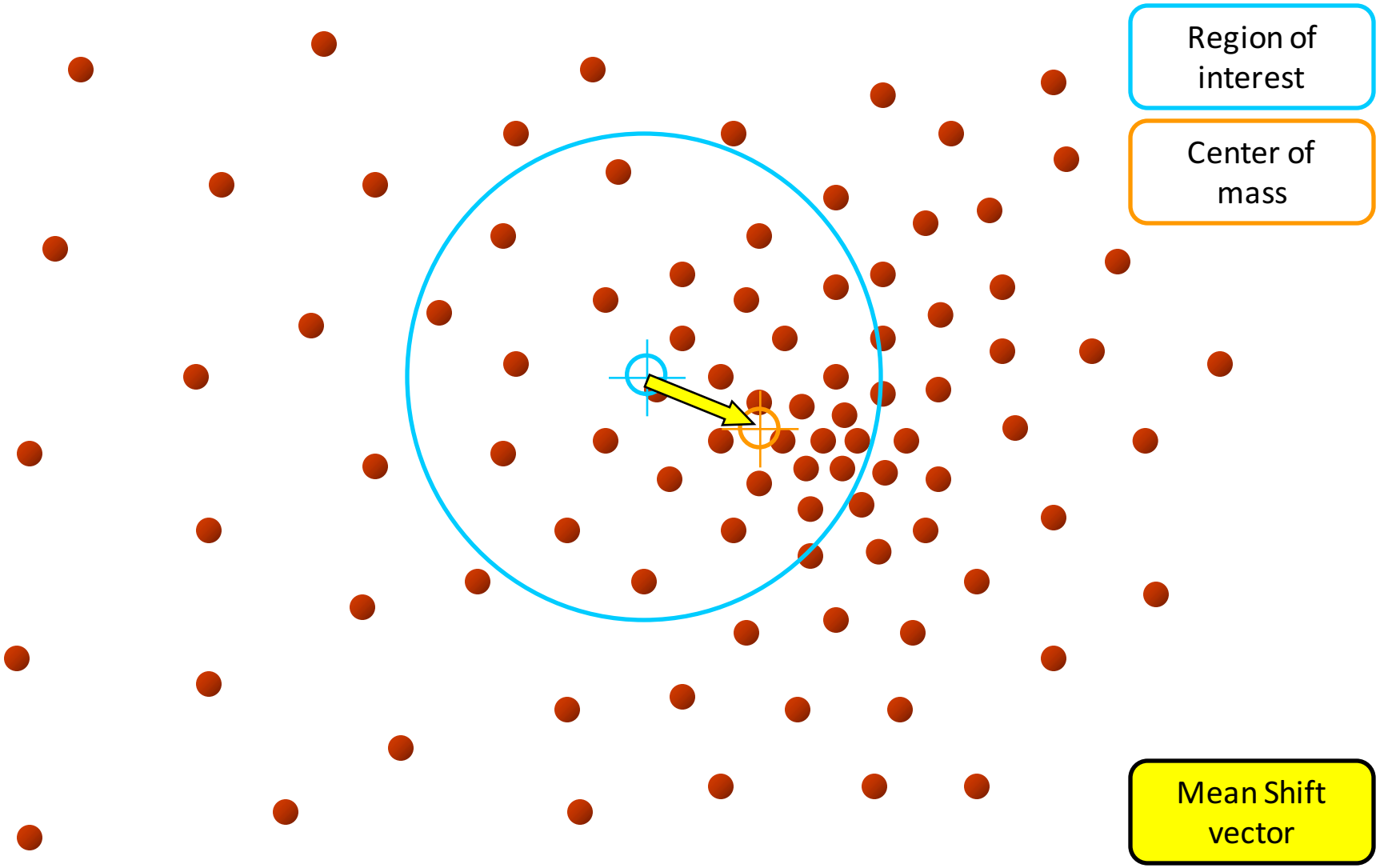
(B): k-means clusters

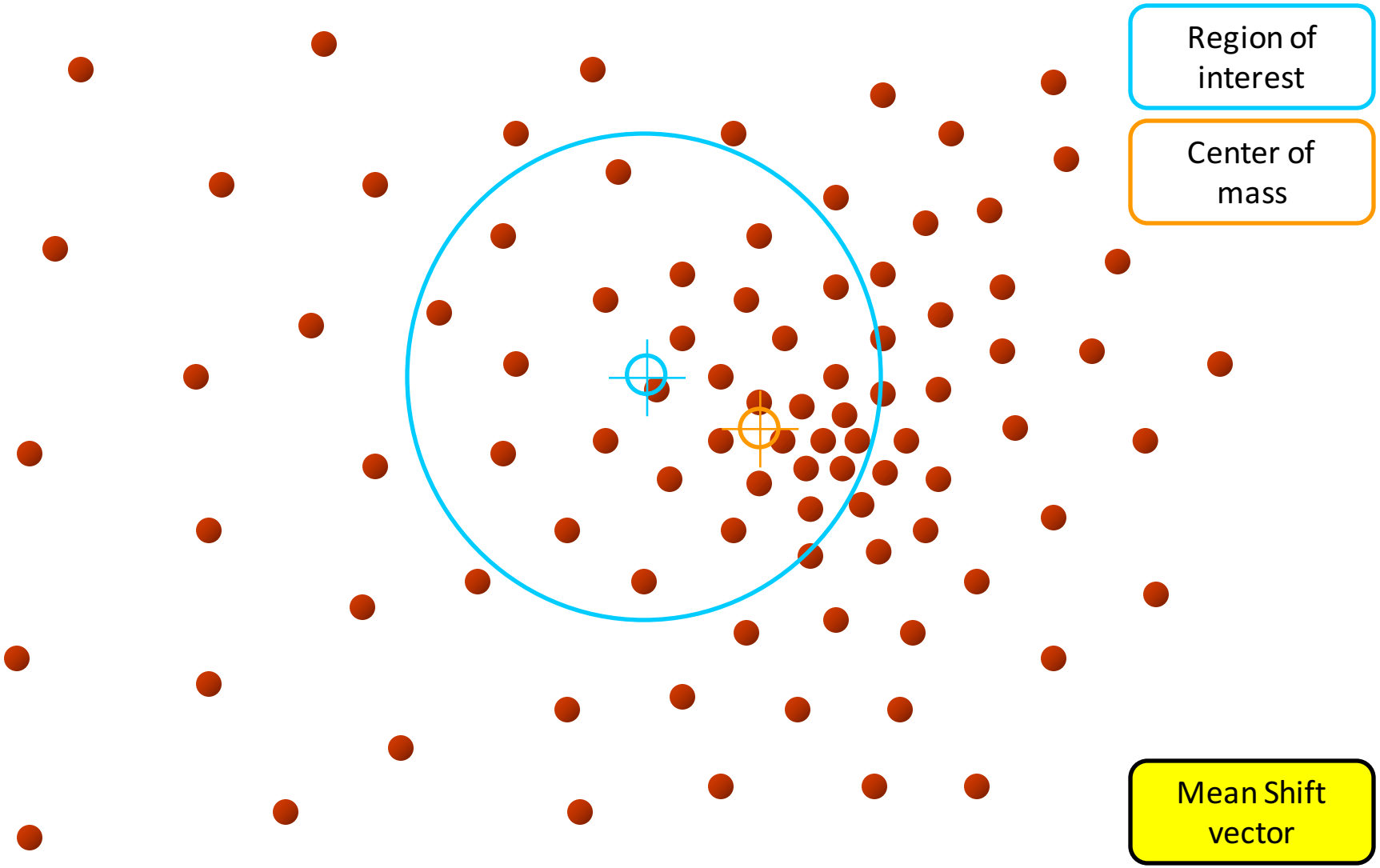
Mean shift algorithm

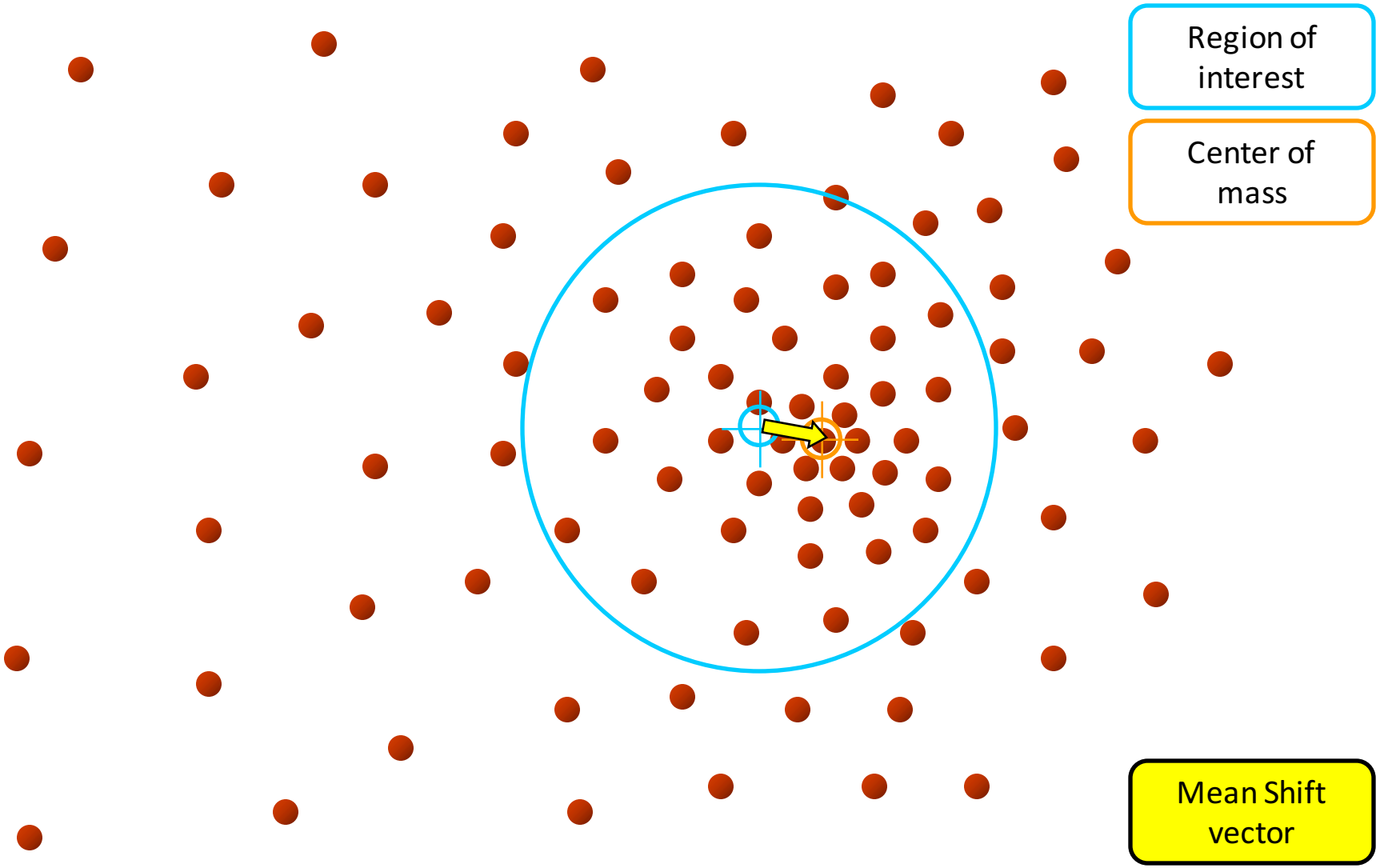
- The mean shift algorithm seeks a *mode* or local maximum of density of a given distribution
  - Choose a search window (width and location)
  - Compute the mean of the data in the search window
  - Center the search window at the new mean location
  - Repeat until convergence

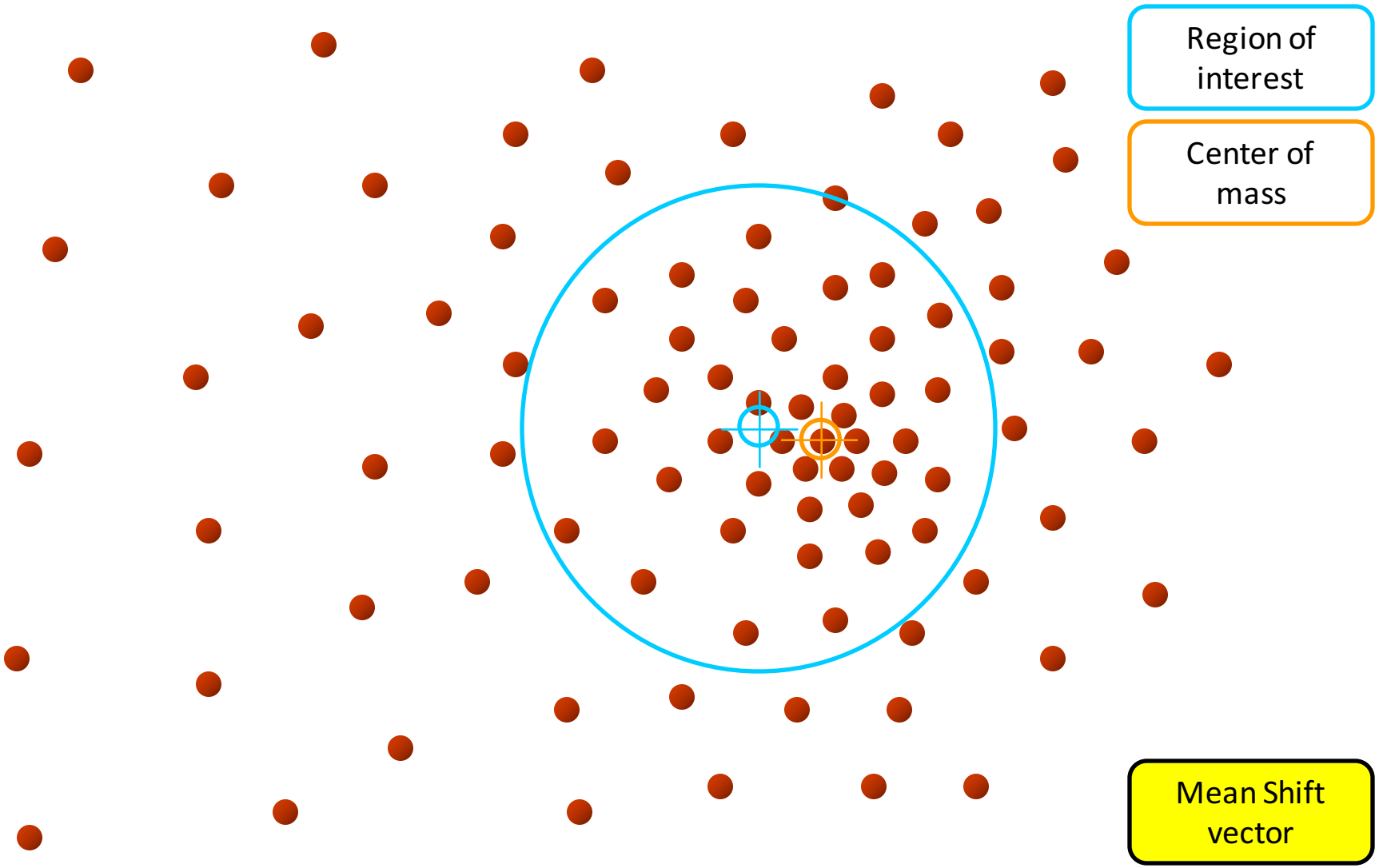

Region of interest
Center of mass
Mean Shift vector

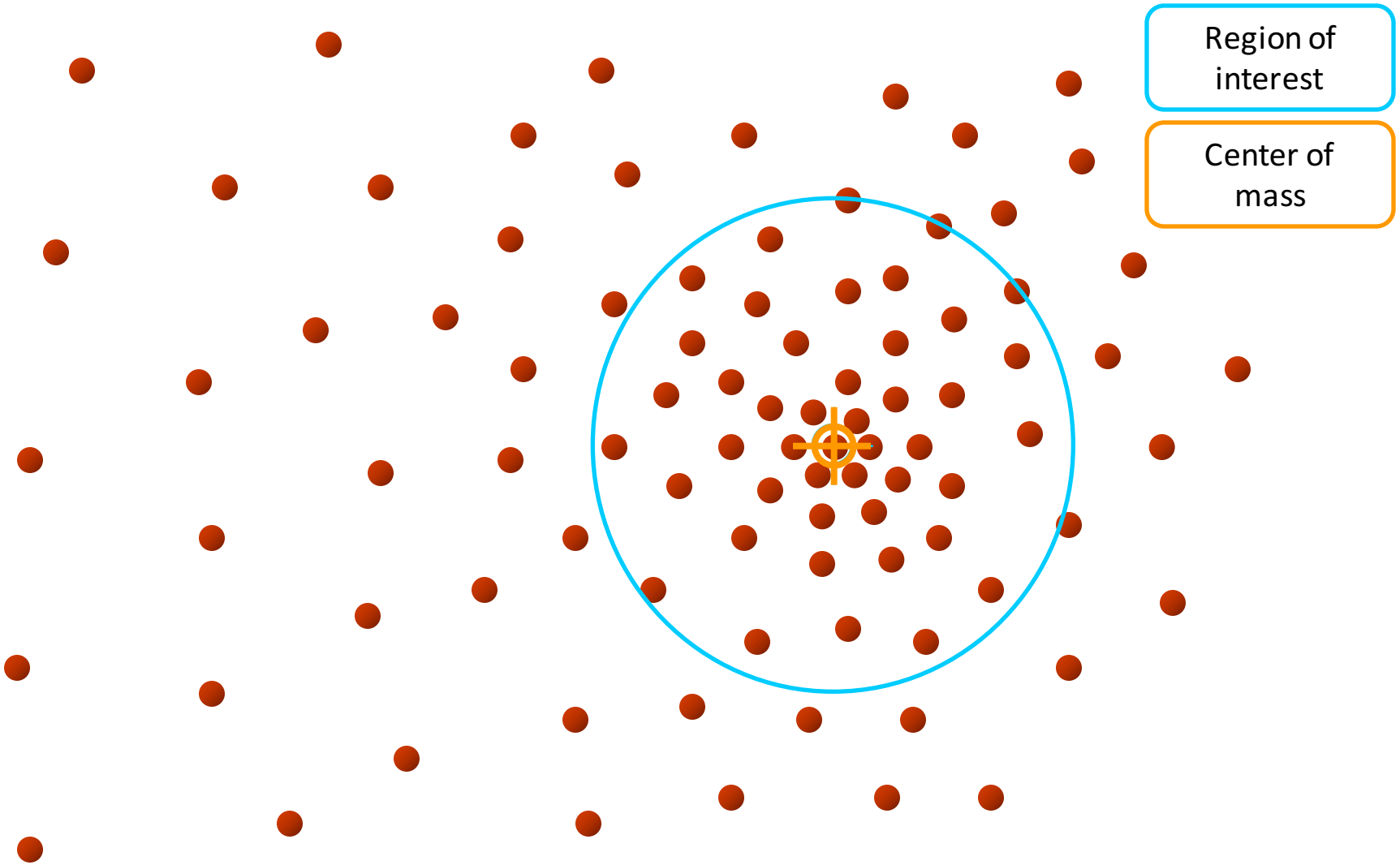

Region of interest
Center of mass
Mean Shift vector


Region of interest
Center of mass
Mean Shift vector


Region of interest
Center of mass
Mean Shift vector


Region of interest
Center of mass
Mean Shift vector


Region of interest
Center of mass
Mean Shift vector


Region of interest
Center of mass

Mean shift pros and cons

- Pros
  - Does not assume spherical clusters
  - Just a single parameter (window size)
  - Finds variable number of modes
  - Robust to outliers
- Cons
  - Output depends on window size
  - Computationally expensive
  - Does not scale well with dimension of feature space

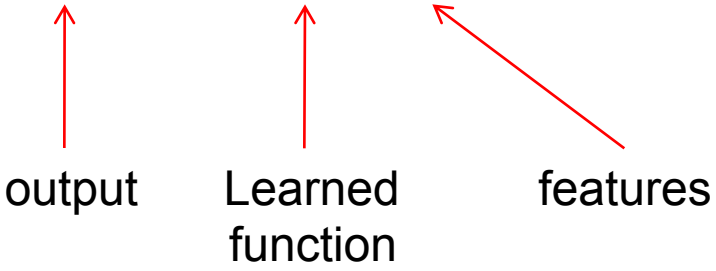
More Clustering Methods

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point in a separate cluster
  - At each iteration, merge two of the "closest" clusters
- Divisive clustering
  - Start with all points grouped into a single cluster
  - At each iteration, split the "largest" cluster
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

As we go down this chart, the clustering strategies have more tendency to transitively group points even if they are not nearby in feature space

Supervised learning framework

$$y = f(\mathbf{x})$$

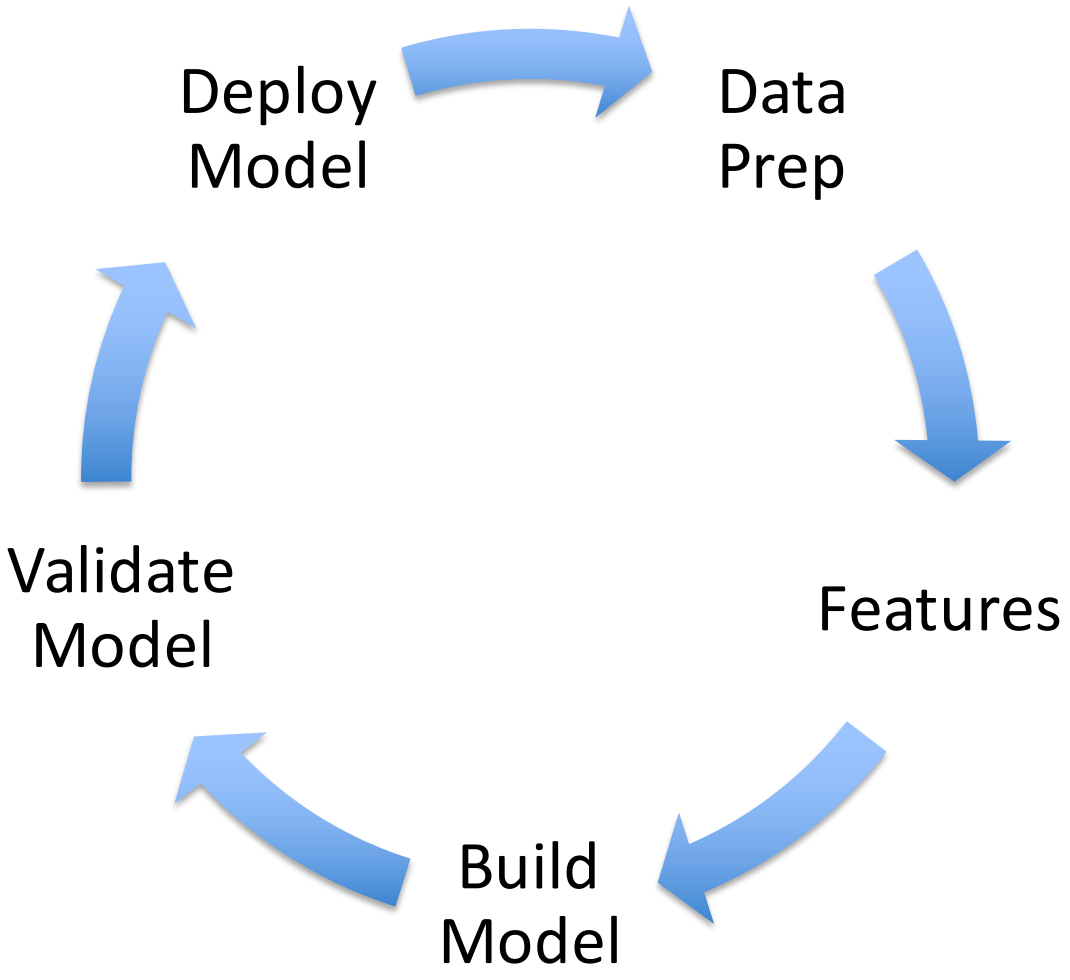


- **Training:** given a *training set* of labeled examples $\{(\mathbf{x}_1,y_1), \ldots, (\mathbf{x}_N,y_N)\}$, estimate the prediction function f that minimizes future generalization error

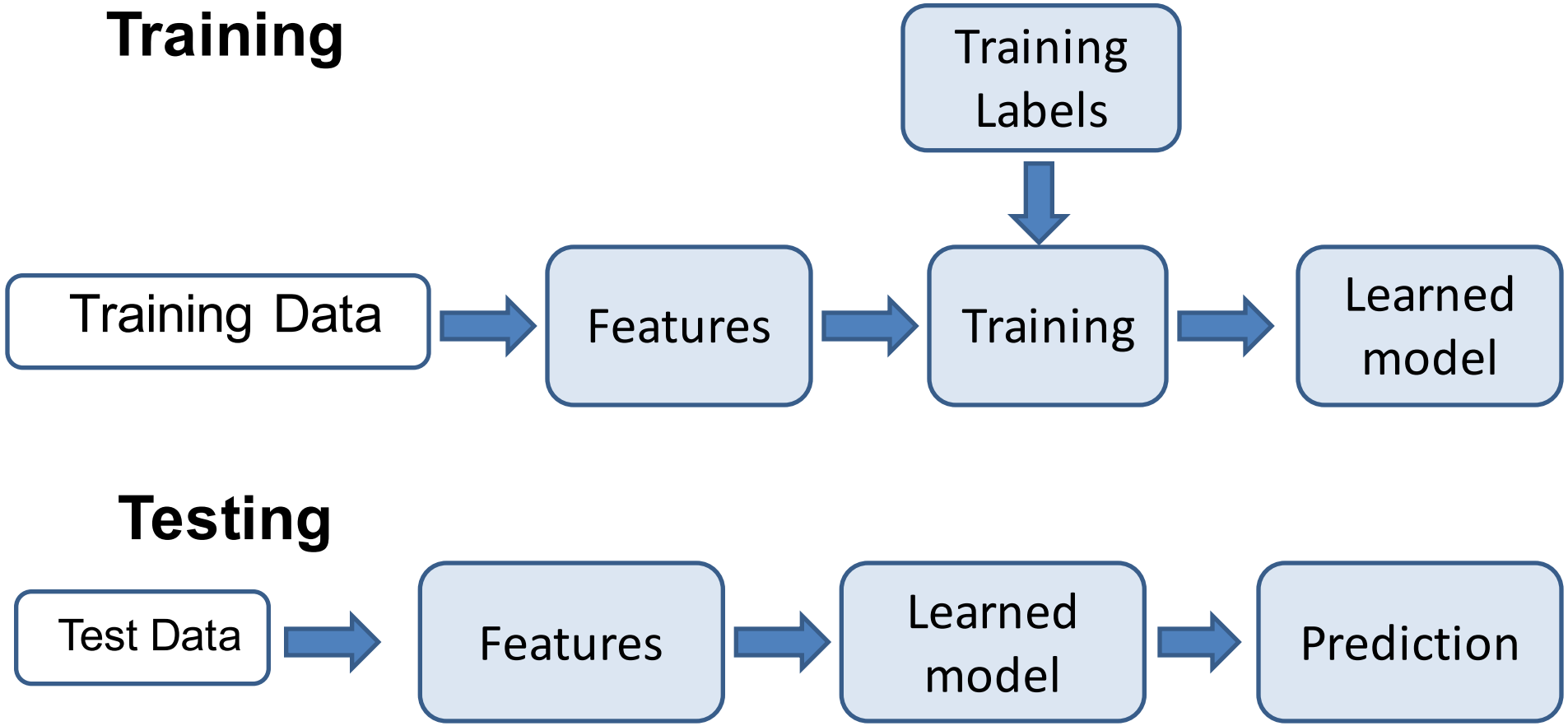
- **Testing:** apply f to a new *test example* $\mathbf{x}$ and output the predicted value $y = f(\mathbf{x})$

Slide credit: L. Lazebnik

Steps


Deploy Model
Data Prep
Features
Build Model
Validate Model

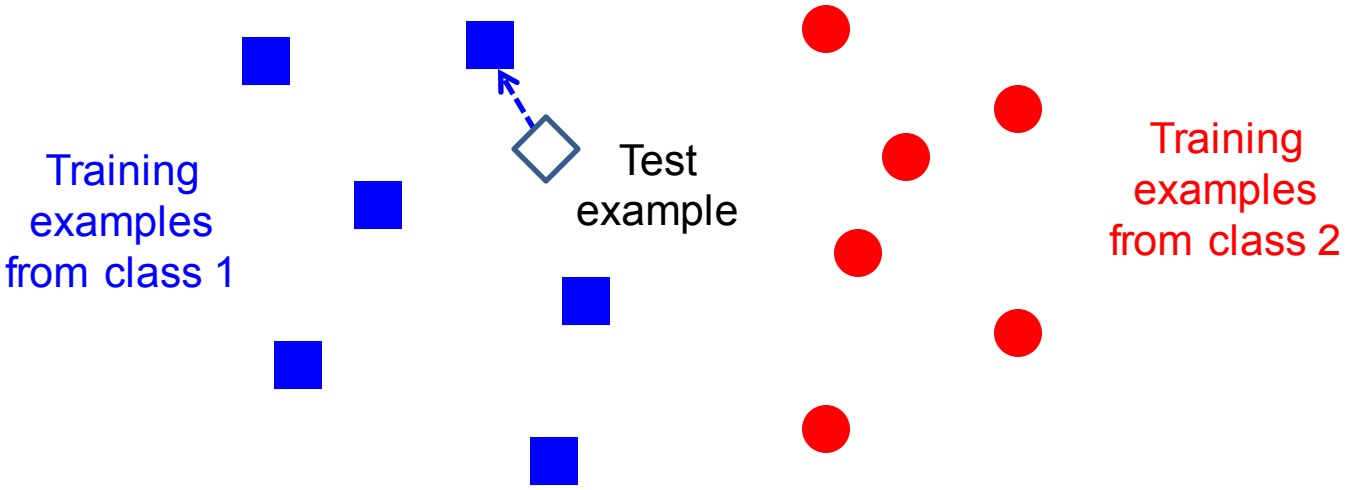
Modeling Steps



Classification Methods

- Regression
- Nearest neighbor
- Decision Trees
- Bayes Classifier
- Support Vector Machines
- Ensembles
  - Bagging
  - Boosting
  - Random Forests

Classifiers: Nearest neighbor

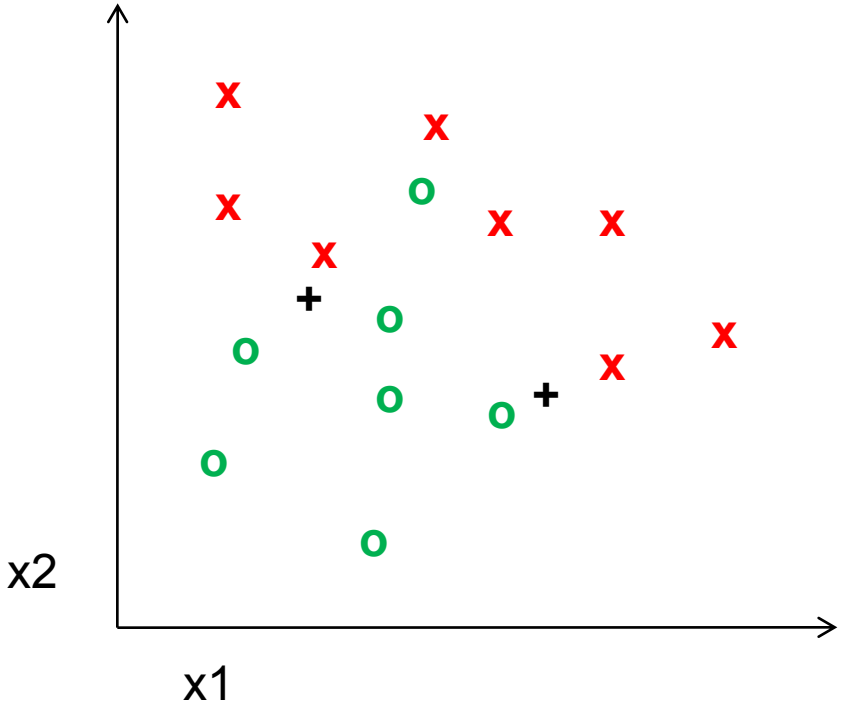


f(**x**) = label of the training example nearest to **x**

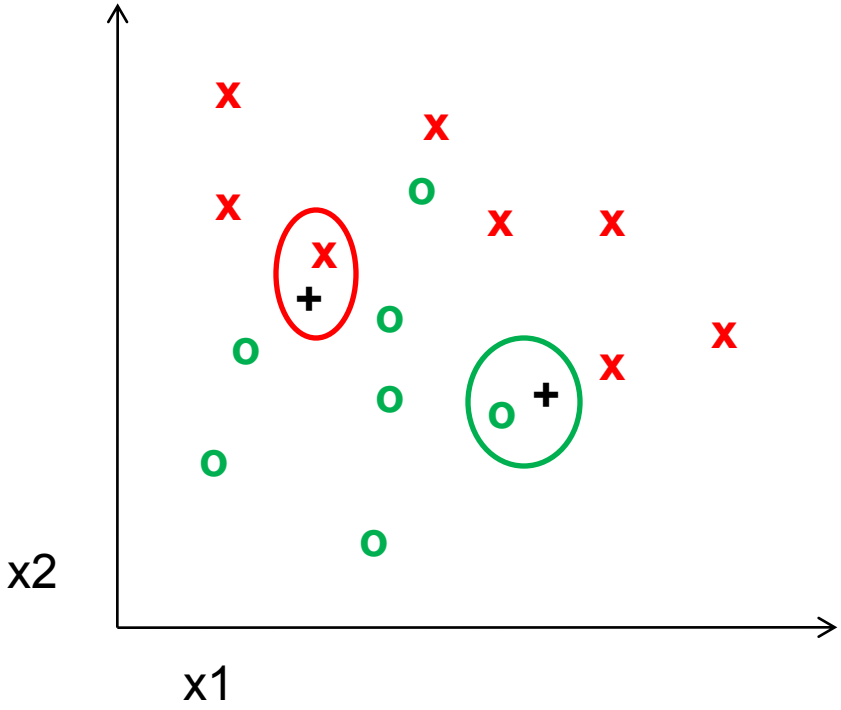
- All we need is a distance function for our inputs
- No training required!

Slide credit: L. Lazebnik

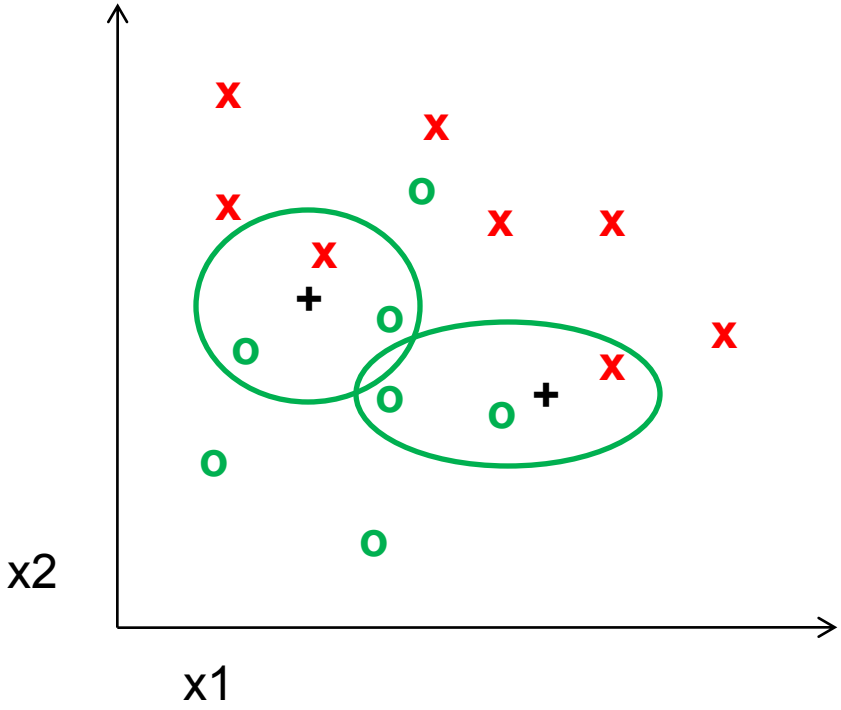
K-nearest neighbor



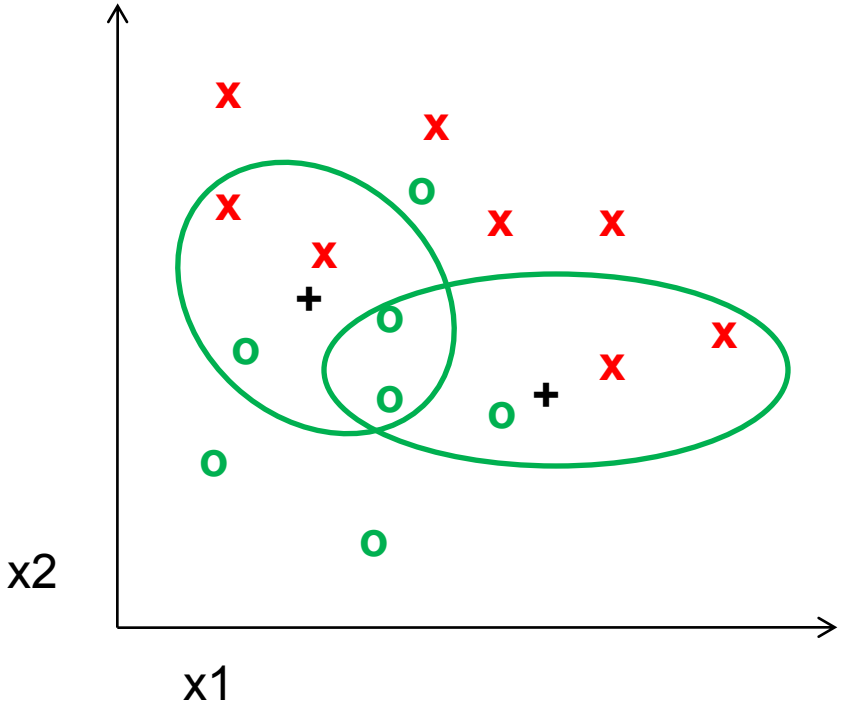
1-nearest neighbor



3-nearest neighbor

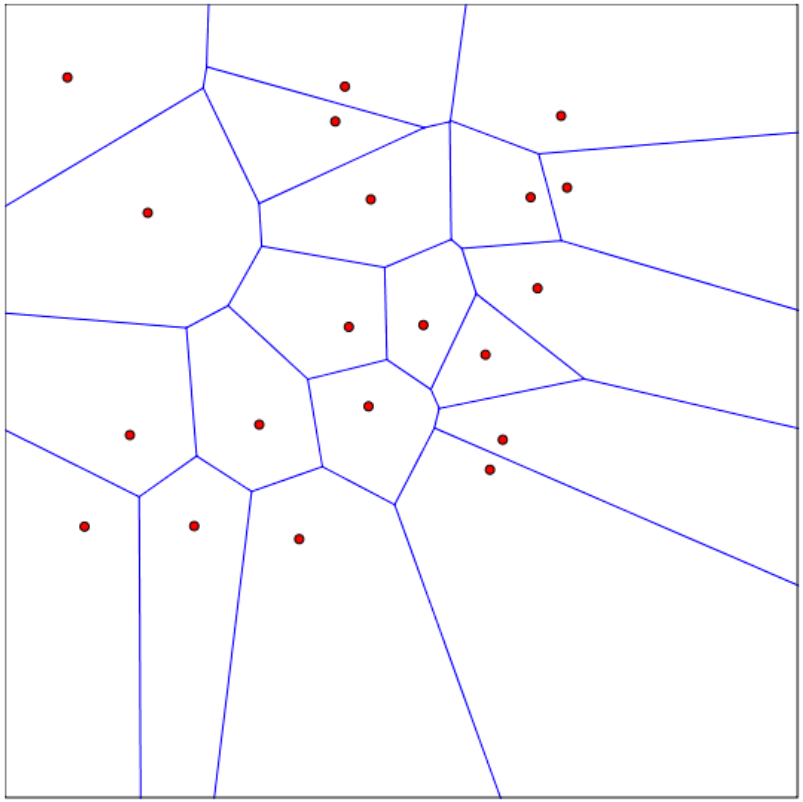


5-nearest neighbor



Voronoi Diagrams

- Given a set of points, a Voronoi diagram describes the areas that are nearest to any given point.



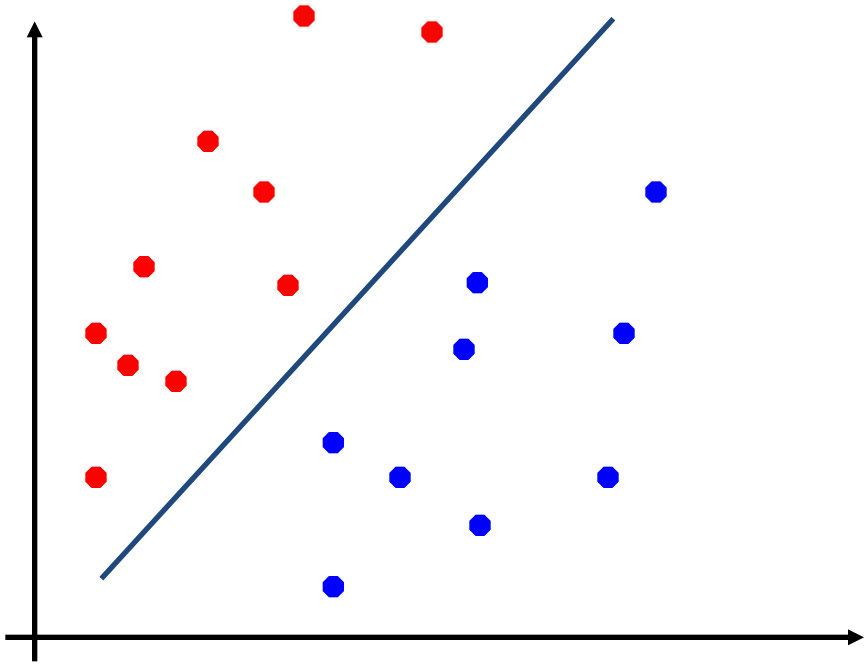
Using K-NN

- Simple
- “Cheap” to train
- With infinite examples, 1-NN provably has error that is at most twice Bayes optimal error

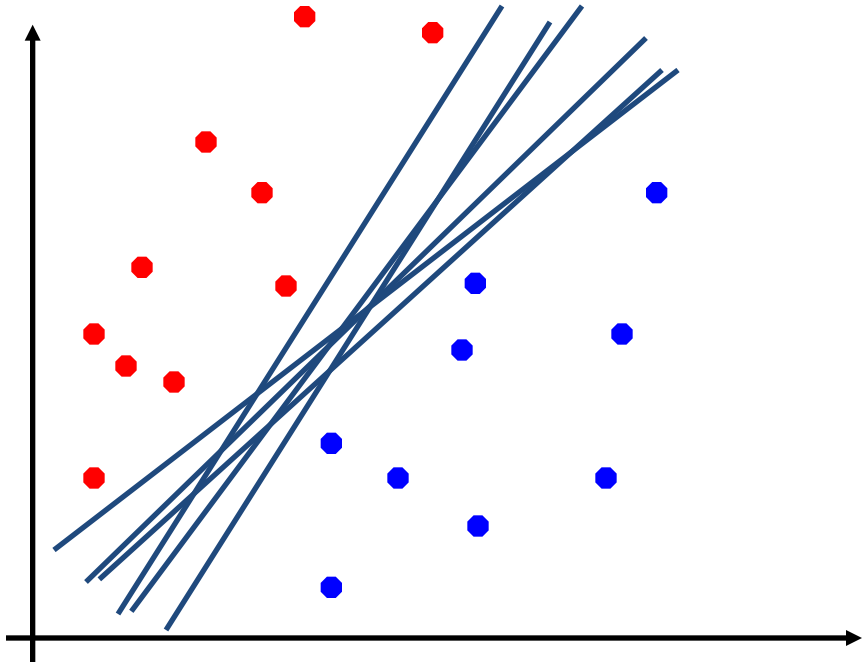
K-NN: Problems

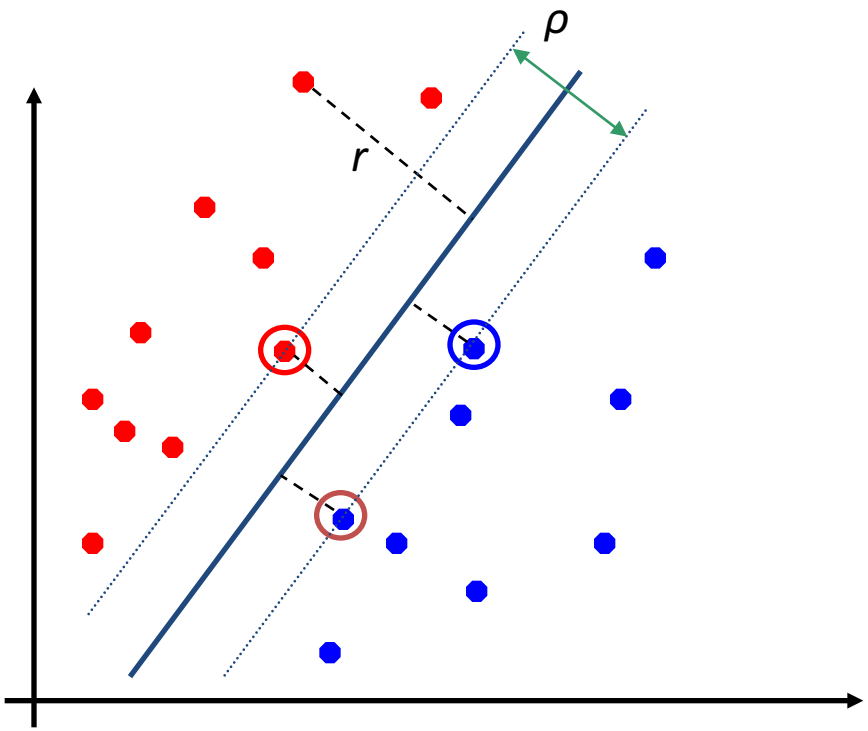
- Need to figure out the right distance function
- Slow at scoring/prediction time
- Need to store all the data
- Extra variables can hurt

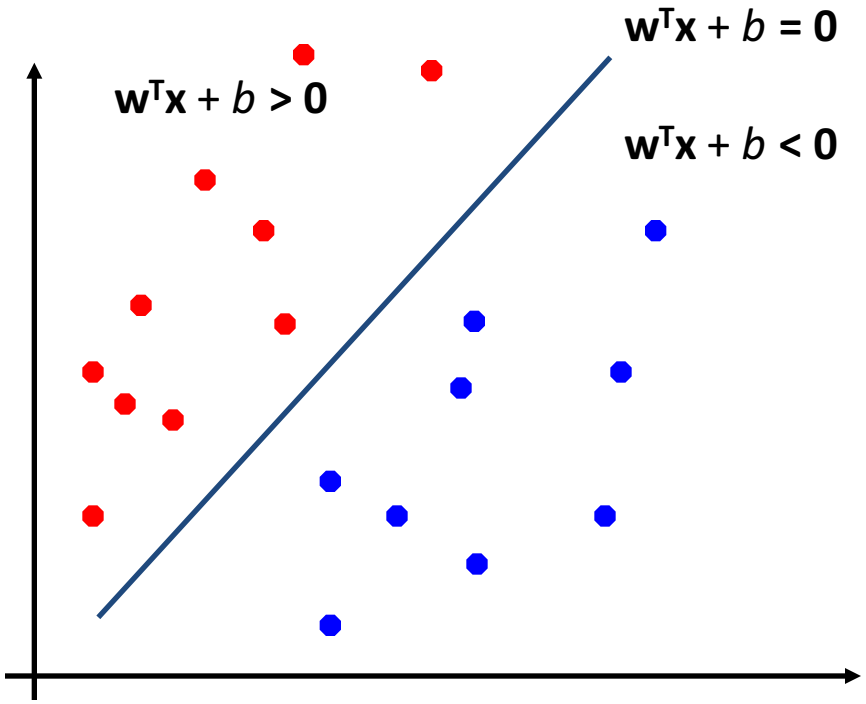
Linear Classifiers



Which is the “best” separator?




ρ
r



$$y = f(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} + b)$$

Linear SVMs Mathematically

- Then we can formulate the *quadratic optimization problem:*

Find $\mathbf{w}$ and b such that

$\rho = \frac{2}{\|\mathbf{w}\|}$ is maximized

and for all $(\mathbf{x}_i, y_i)$, i=1..n : $y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$

Which can be reformulated as:

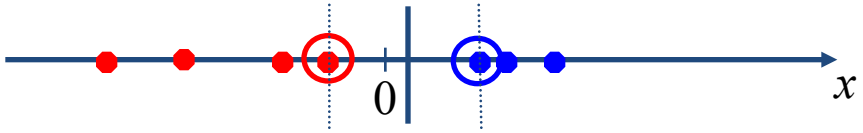
Find $\mathbf{w}$ and b such that

$\Phi(\mathbf{w}) = ||\mathbf{w}||^2 = \mathbf{w}^T\mathbf{w}$ is minimized

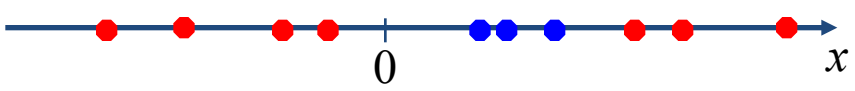
and for all $(\mathbf{x}_i, y_i)$, i=1..n : $y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$

Nonlinear SVMs

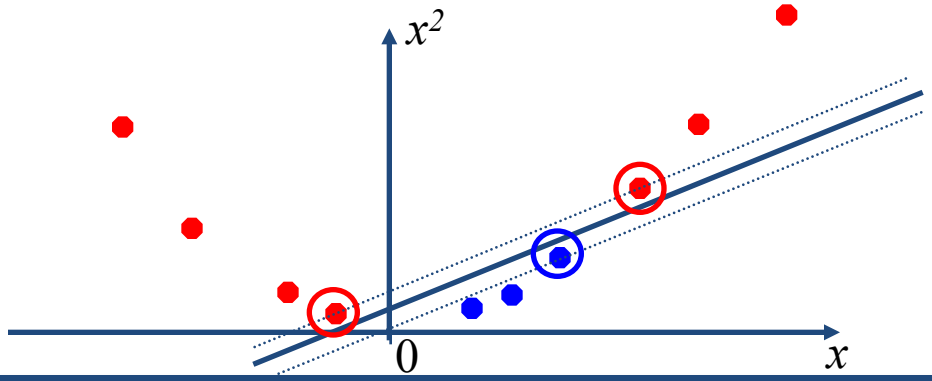
- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?



- We can map it to a higher-dimensional space:



Logit vs Linear SVMs

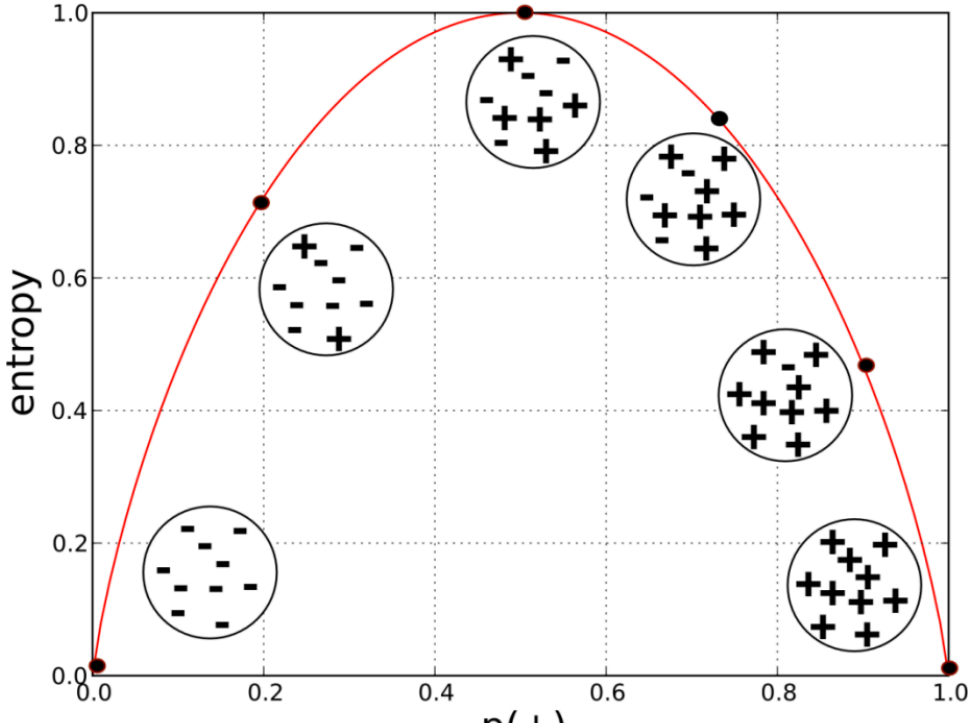
- Logistic regression maximizes the conditional likelihood of the training data

- Linear SVMs maximize the margin

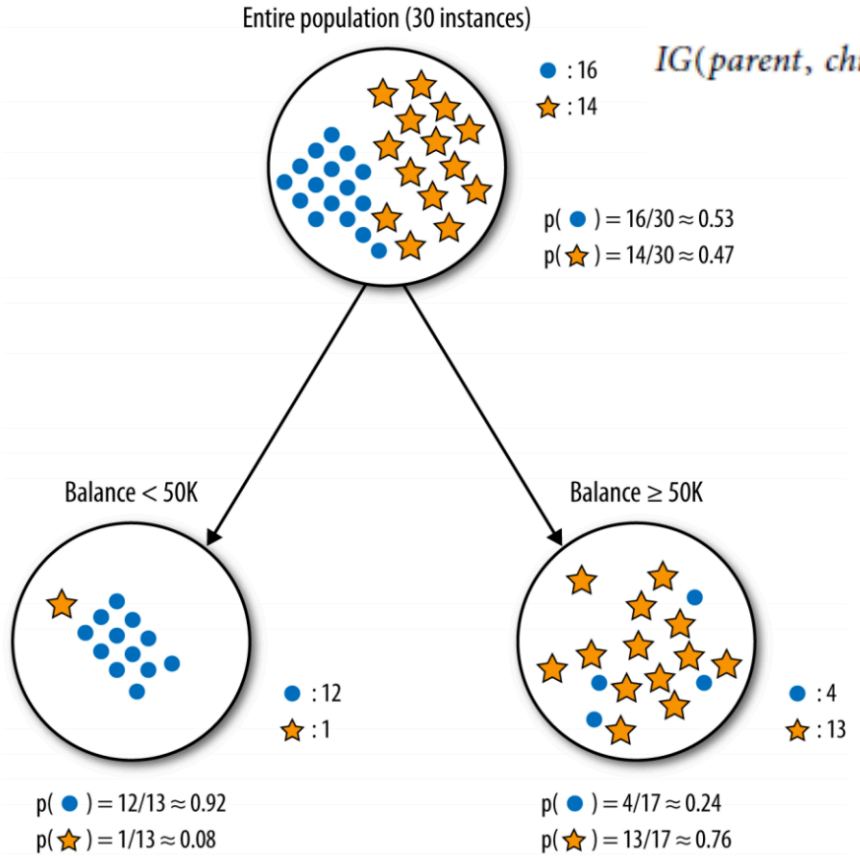
Classifiers: Decision Trees

Information Gain

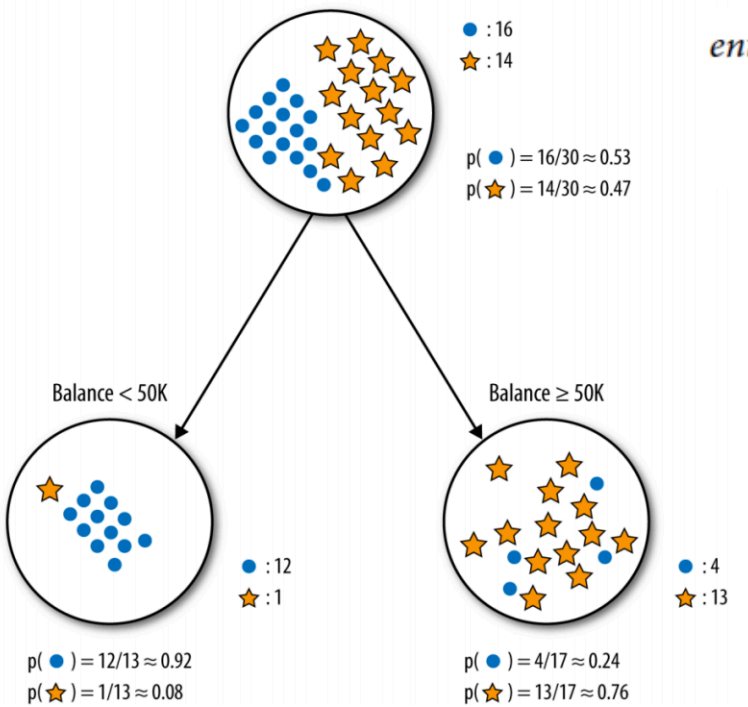
- The most common splitting criterion is called **information gain** (IG)
  - It is based on a **purity measure** called **entropy**
    - $entropy = -\mathrm{p}_1 \log_2(p_1) - p_2 \log_2(p_2) - ..$
    - Measures the general disorder of a set



- Information gain measures the *change* in entropy due to any amount of new information being added



$$IG(parent, children) = entropy(parent) - [p(c_1) \times entropy(c_1) + p(c_2) \times entropy(c_2) + \cdots]$$



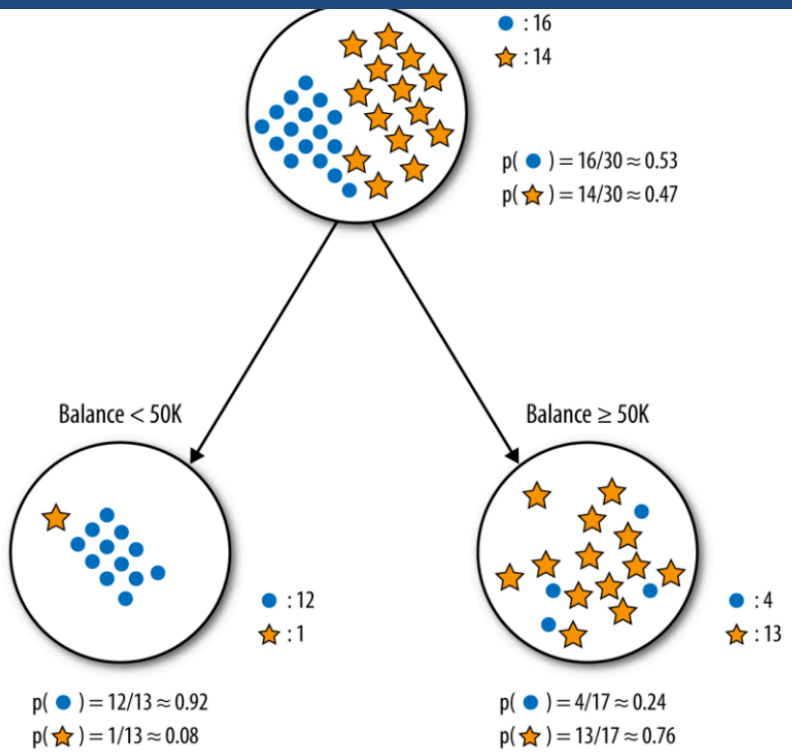
$$\begin{aligned} entropy(parent) &= -[p(\bullet) \times \log_2\ p(\bullet) + p(\star) \times \log_2\ p(\star)] \\ &\approx -[0.53 \times -0.9 + 0.47 \times -1.1] \\ &\approx 0.99 \quad \text{(very impure)} \end{aligned}$$

The entropy of the *left* child is:

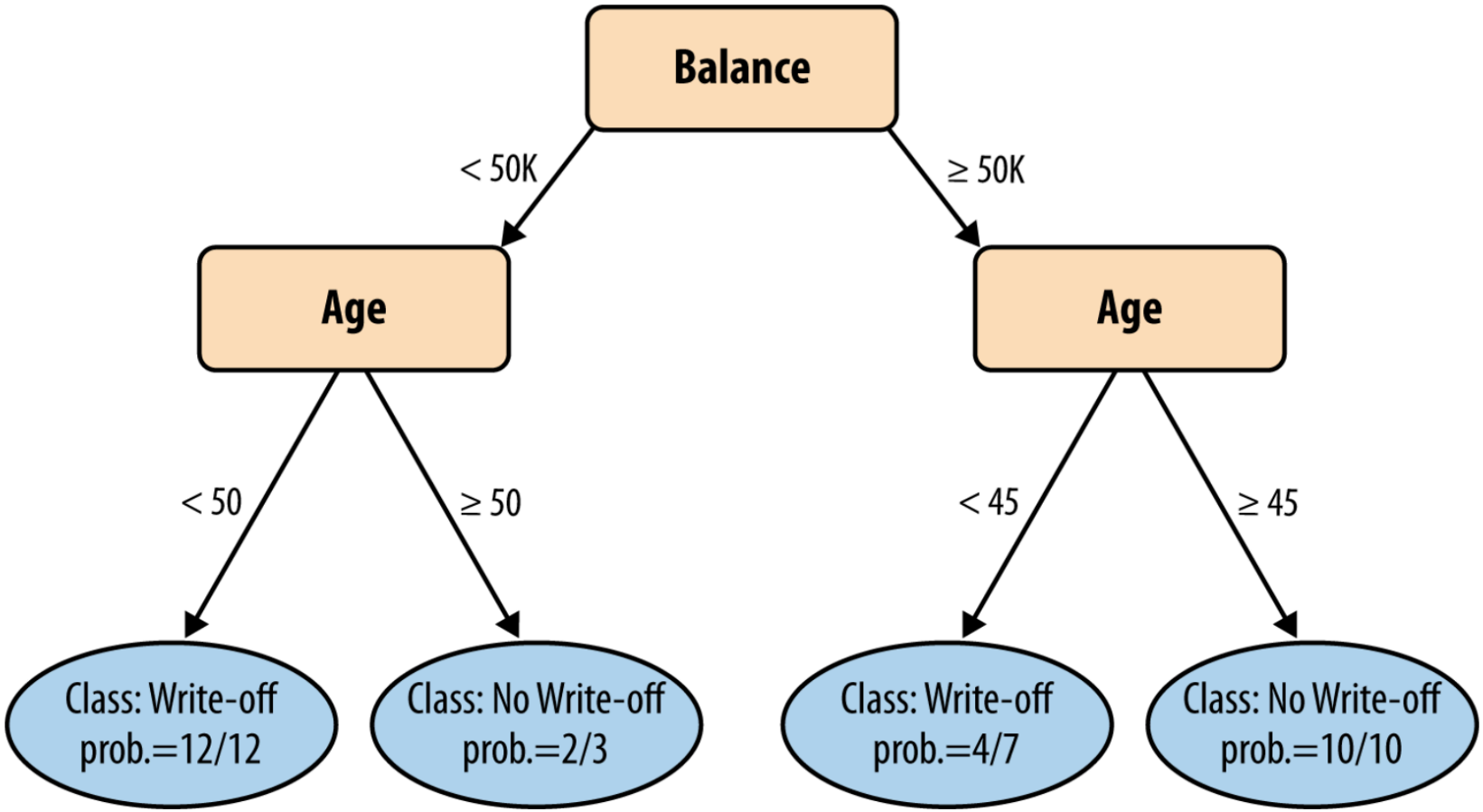
$$\begin{aligned} entropy(Balance < 50K) &= -[p(\bullet) \times \log_2\ p(\bullet) + p(\star) \times \log_2\ p(\star)] \\ &\approx -[0.92 \times (-0.12) + 0.08 \times (-3.7)] \\ &\approx 0.39 \end{aligned}$$

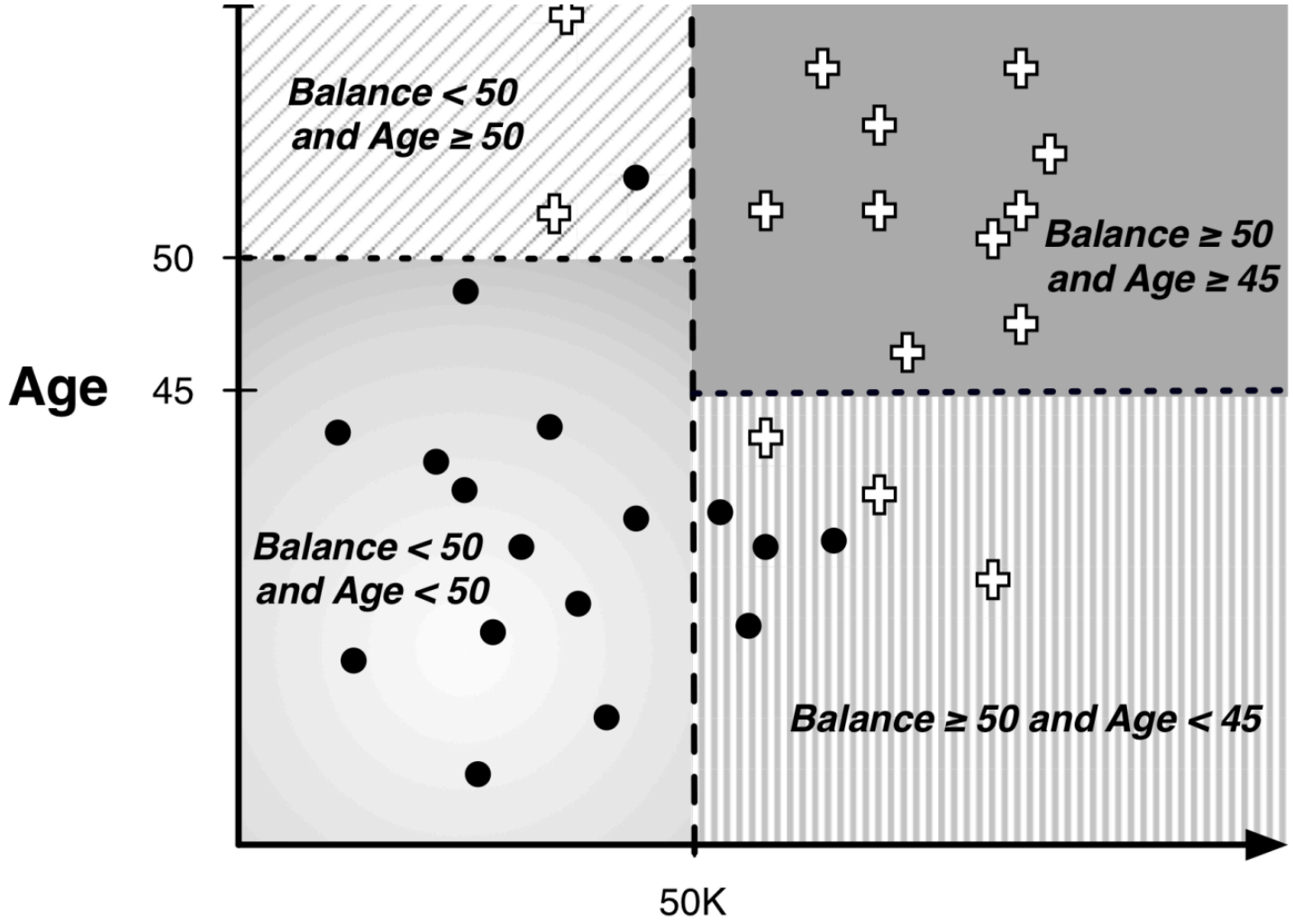
The entropy of the *right* child is:

$$\begin{aligned} entropy(Balance \geq 50K) &= -[p(\bullet) \times \log_2\ p(\bullet) + p(\star) \times \log_2\ p(\star)] \\ &\approx -[0.24 \times (-2.1) + 0.76 \times (-0.39)] \end{aligned}$$

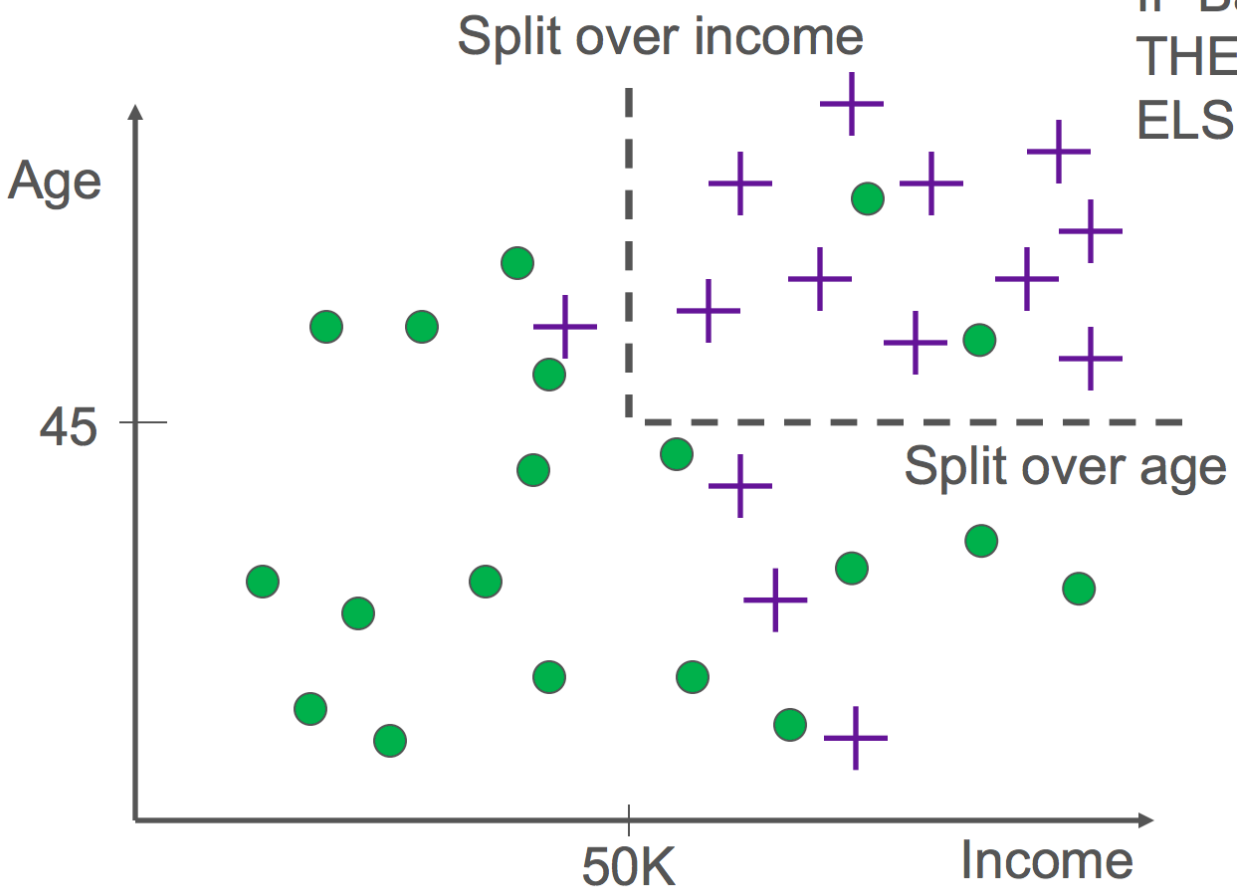


$$
\begin{aligned}
IG &= entropy(parent) - [p(\text{Balance} < 50\text{K}) \times entropy(\text{Balance} < 50\text{K}) \\
&\quad + p(\text{Balance} \geq 50\text{K}) \times entropy(\text{Balance} \geq 50\text{K})] \\
&\approx 0.99 - [0.43 \times 0.39 + 0.57 \times 0.79]
\end{aligned}
$$


Balance
< 50K
≥ 50K
Age
Age
< 50
≥ 50
< 45
≥ 45
Class: Write-off
prob.=12/12
Class: No Write-off
prob.=2/3
Class: Write-off
prob.=4/7
Class: No Write-off
prob.=10/10


Balance < 50
and Age ≥ 50
Balance ≥ 50
and Age ≥ 45
Balance < 50
and Age < 50
Balance ≥ 50 and Age < 45
Age
50
45
50K

geflip | Trello
os://trello.com/board/edgeflip/
266e631391d05c63000bac



Pattern:

IF Balance >= 50K & Age > 45
THEN Default = 'no'
ELSE Default = 'yes'

Bayes' Rule

- For a data point *d* and a class *c*

$$P(c \mid d) = \frac{P(d \mid c)P(c)}{P(d)}$$

Naïve Bayes Classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

MAP is "maximum a posteriori" = most likely class

$$= \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c)P(c)}{P(d)}$$

Bayes Rule

$$= \underset{c \in C}{\operatorname{argmax}} P(d \mid c)P(c)$$

Dropping the denominator

Naïve Bayes Classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(d \mid c) P(c)$$

$$= \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

Data point d represented as features x1..xn

Naïve Bayes Classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

$O(|X|^n \bullet |C|)$ parameters

How often does this class occur?

Could only be estimated if a very, very large number of training examples was available.

We can just count the relative frequencies in a corpus

Naïve Bayes - Independence Assumptions

$$P(x_1, x_2, \ldots, x_n \mid c)$$

- **Conditional Independence**: Assume the feature probabilities $P(x_i \mid c_j)$ are independent given the class c.

$$P(x_1, \ldots, x_n \mid c) = P(x_1 \mid c) \bullet P(x_2 \mid c) \bullet P(x_3 \mid c) \bullet \ldots \bullet P(x_n \mid c)$$

Naïve Bayes Classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c_j) \prod_{x \in X} P(x \mid c)$$

Ensembles

How can we create ensembles?

- Different learning algorithms
- Algorithms with different choice for parameters
- Data set with different features (e.g. random subspace)
- Data set = different subsets (e.g. bagging, boosting)

Ensemble Methods

- Bagging (Bootstrap Aggregation)
- Boosting
- Random Forests
- Stacking

Bagging

- For i = 1 .. M
  - Draw bootstrap samples with replacement
  - Learn classifier C_i
- Final classifier is a vote of C_1 .. C_M
- Why does it work?
  - Increases classifier stability/reduces variance
  - Works better with unstable classifiers (Decision Trees)

figure from Friedman et al. 2000

Ensemble Methods: Boosting

- Improves classification accuracy
- Can be used with many different types of classifiers
- Not prone to overfitting

figure from Friedman et al. 2000

Boosting

- General Loop:

  Set all examples to have equal uniform weights.

  For t from 1 to T do:

  Learn a classifier, C_t, from the weighted examples

  Increase the weights of examples C_t classifies incorrectly

- Base (weak) learner must focus on correctly classifying the most highly weighted examples while strongly avoiding over-fitting.

- During testing, each of the T hypotheses get a weighted vote proportional to their accuracy on the training data.

Boosting

- Examples are given weights.
- At each iteration, a new model is learned and the examples are reweighted to focus the system on examples that the most recently learned classifier got wrong.

Boosting

- Improves classification accuracy
- Can be used with many different types of classifiers

figure from Friedman et al. 2000

Using Boosted Decision Trees

- Flexible: can deal with both continuous and categorical variables
- How to control bias/variance trade-off
  - Size of trees
  - Number of trees
- Boosting trees often works best with a small number of well-designed features
- Boosting “stubs” can give a fast classifier

Random Forests

- Motivation: reduce error correlation between classifiers
- Main idea: build a larger number of un-pruned decision trees
- Key: using a random selection of features to split on at each node

Random Forest

- Each tree is grown on a bootstrap sample of the training set of **N** cases.
- A number **m** is specified much smaller than the total number of variables **M** (e.g. m = sqrt(M)).
- At each node, m variables are selected at random out of the M.
- The split used is the best split on these **m** variables.
- Final classification is done by majority vote across trees.

Advantages of random forest

- Error rates compare favorably to Adaboost
- More robust with respect to noise.
- More efficient on large data
- Provides an estimation of the importance of features in determining classification
- More info at: http://stat-www.berkeley.edu/users/breiman/RandomForests/cc_home.htm

What to remember about classifiers

- Generally better to have smart features and simple classifiers than simple features and smart classifiers
- Use increasingly powerful classifiers with more training data (bias-variance tradeoff)

Slide credit: D. Hoiem

Factors to consider

- Complexity
  - Model
  - Tuning
  - Pre-processing (scaling, missing values, discrete/continuous etc.)
- Overfitting
- Robustness
- Interpretability
- Training Time
- Update Time
- Scoring Time

Complexity

- Linear vs Non-Linear
- Fewer parameters vs more parameters

Overfitting

- Reduce Features
  - Force less complexity
- Regularization
  - Penalize for complexity

Interpretability

- To a ML expert?
- To a domain expert?

- Interpretability of the model vs Interpretability of the predictions

Time

- Training
- Scoring new data
- Updating when you get more labels back

Classifier Comparisons

- Azure ML Blog Post
- Another google doc done by Data School (not all the information in there is “correct”)
- Sklearn map
- Comparison Methodology
  - http://jmlr.org/papers/volume15/delgado14a/delgado14a.pdf
  - http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.122.5901

Evaluation - Methodology

- In-sample
- Out of sample
- Splits to deal with variance
- Cross Validation
  - Leave one out (LOO)
  - K fold
- Temporal
- Baseline?

Evaluation - Metrics

- Accuracy Threshold?
- Precision
- Recall
- AUC
- …

		Predicted Class: Yes	Predicted Class: No
Actual Class	Yes	TP	FN
	No	FP	TN

Measure	Formula
precision	$tp/(tp + fp)$
recall	$tp/(tp + fn)$
f-score	$2pr * re/(pr + re)$
accuracy	$(tp + tn)/(tp + tn + fp + fn)$

Features

- Detailed session later

Other tools

- Free
  - Knime
  - Rapidminer
  - Weka (mostly research)
- Commercial
  - Sas enterprise miner
  - Ibm/spss
  - Kxen
  - Skytree
  - megaputer

Advanced Topics

- Semi-supervised learning
- Active learning
- Reinforcement learning
- Streaming data

Contact Information

Rayid Ghani

rayid@uchicago.edu