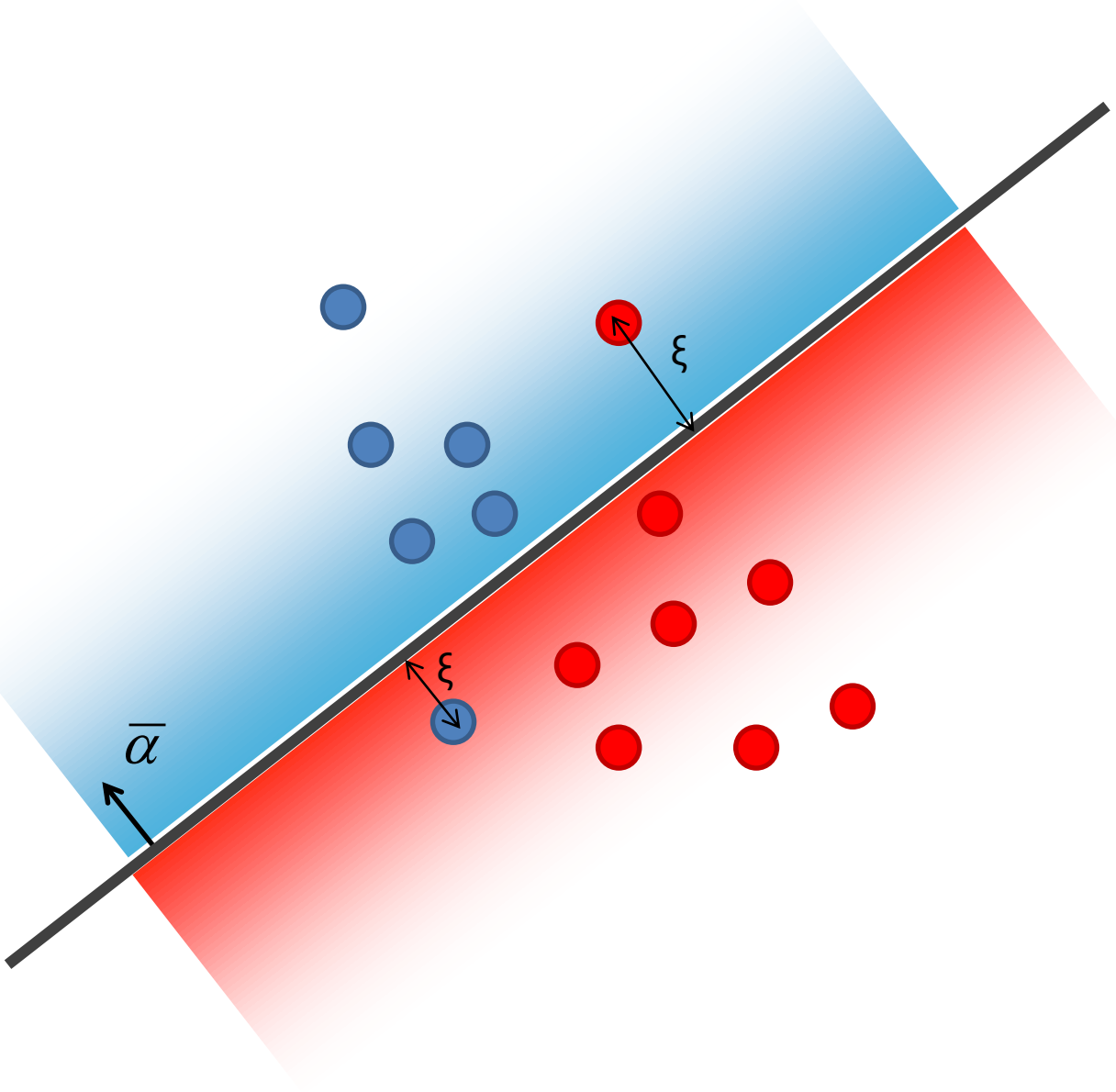


Basics of Machine Learning

Dmitry Ryabokon, github.com/dryabokon

Content

Part 1: Key concepts

Approaches for ML
Bayesian approach
Classifying the independent binary features

Approaches for ML

Approaches for ML

Maximum Likelihood

$x_1, x_2, \ldots, x_N$ features and

$k_1, k_2, \ldots, k_N$ states

$k \in \mathrm{K} = \{1, 2, \ldots, K\}$ domain for *k*

$p(x|k) = f(x, \mu_k)$ The model is parametrized by $\mu_1, \mu_2, \ldots, \mu_K$

input

$$(\mu_1, \mu_2, \ldots, \mu_K)^* = \arg\max_{\mu_1, \mu_2, \ldots, \mu_K} \prod_{i=1}^{N} p(x_i, k_i)$$

output

Approaches for ML

Maximum Likelihood

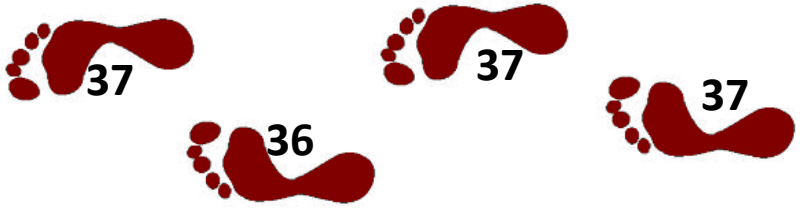
$$\arg\max_{\mu_1,\mu_2,\dots,\mu_K} \prod_{i=1}^{N} p(x_i, k_i) = \arg\max_{\mu_1,\mu_2,\dots,\mu_K} \left(\prod_{i=1}^{N} p(k_i) \cdot p(x_i|k_i) \right) = \arg\max_{\mu_1,\mu_2,\dots,\mu_K} \left(\prod_{i=1}^{N} p(x_i|k_i) \right) =$$

$$= \arg\max_{\mu_1,\mu_2,\dots,\mu_K} \left(\prod_{x \in X_1} p(x|1) \times \prod_{x \in X_2} p(x|2) \times \prod_{x \in X_K} p(x|K) \right)$$

$$\mu_k^* = \arg\max_{\mu} \prod_{x \in X_k} f(x, \mu)$$

Approaches for ML

Maximum Likelihood: example



$$p(x|k=1) = f(x,\mu_1) \cong \exp\left\{-(x-\mu_1)^2\right\}$$

$$p(x|k=2) = f(x,\mu_2) \cong \exp\left\{-(x-\mu_2)^2\right\}$$

$$(\mu_1,\mu_2) = \arg\max \prod_{i=1}^{7} p(x_i,k_i)$$

$$\mu_1 = \arg\max_{\mu} \quad \exp\left\{-(45-\mu)^2-(46-\mu)^2-(45-\mu)^2\right\}$$

Approaches for ML

Maximum Likelihood: example

$$\mu_1 = \arg\max_{\mu} \quad \exp\left\{-(45-\mu)^2 - (46-\mu)^2 - (45-\mu)^2\right\}$$

$$\mu_1 = \arg\min_{\mu} \left\{(45-\mu)^2 + (46-\mu)^2 + (45-\mu)^2\right\}$$

$$\mu_1 = \frac{45+46+45}{3} = \langle x \rangle_{X_1}$$

Approaches for ML

Bias in training data

$x_1, x_2, \ldots, x_N$ features and

$k_1, k_2, \ldots, k_N$ states

$k \in \mathrm{K} = \{1, 2, \ldots, K\}$ domain for *k*

$p(x|k) = f(x, \mu_k)$ The model is parametrized by $\mu_1, \mu_2, \ldots, \mu_K$

input

$$\mu_1^* = \arg\max_{\mu} \left\{ \min_{x \in X_1} p(x|k=1, \mu) \right\}$$

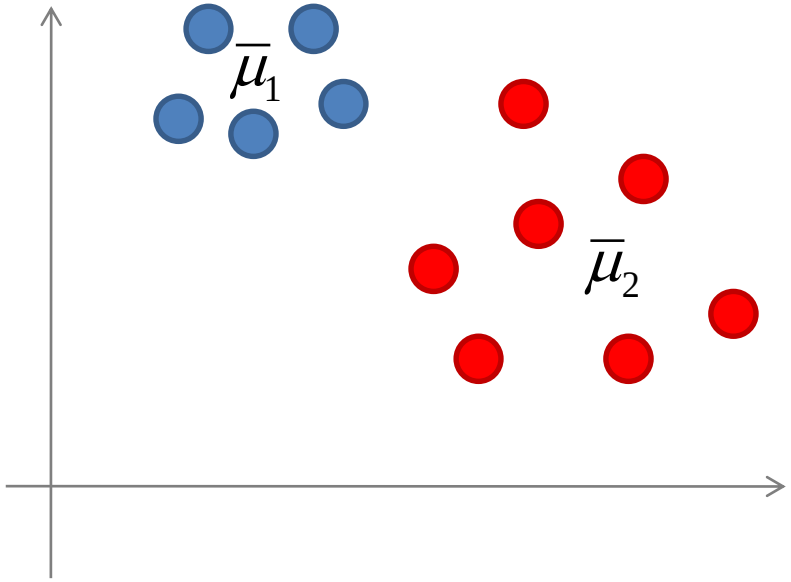
output

Approaches for ML

Bias in training data: example

$$p(x|k=1) = N(\bar{\mu}_1, 1)$$
$$p(x|k=2) = N(\bar{\mu}_2, 1)$$

$$\bar{\mu}_1^{\,*} = \arg\max_{\bar{\mu}} \left\{ \min_{x \in X_1} p(x|k=1, \bar{\mu}) \right\}$$



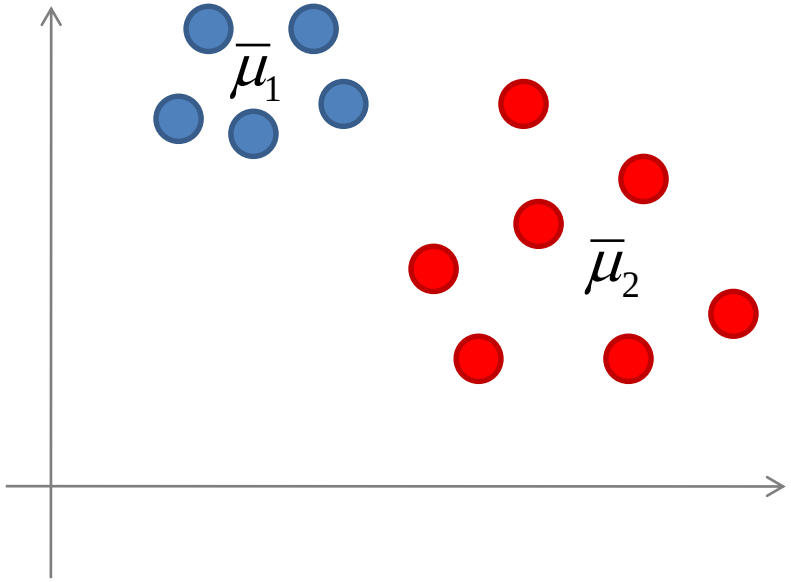
Approaches for ML

Bias in training data: example

$$p(x|k=1) = N(\bar{\mu}_1, 1)$$
$$p(x|k=2) = N(\bar{\mu}_2, 1)$$

$$\bar{\mu}_1^{\,*} = \arg\max_{\bar{\mu}} \left\{ \min_{x \in X_1} p(x|k=1, \bar{\mu}) \right\}$$

center of the circle that
holds all the points from X1



Approaches for ML

ERM: Empirical risk minimization

$x_1, x_2, \ldots, x_N$ features and

$k_1, k_2, \ldots, k_N$ states

$k \in \mathrm{K} = \{1, 2, \ldots, K\}$ domain for *k*

$W(k, k')$ penalty function

$q(x, \bar{\mu}) \in K$ decision strategy is parametrized by μ

Approaches for ML

ERM: Empirical risk minimization

$q(x, \bar{\mu}) \in K$ decision strategy is parametrized by μ

$$Risk(q(\bar{\mu})) = \sum_{x \in X} \sum_{k \in K} p(x,k) \cdot W(q(x, \bar{\mu}), k)$$

$$\cong \sum_{i=1}^{N} p(x_i, k_i) \cdot W(q(x_i, \bar{\mu}), k_i) = \sum_{i=1}^{N} W(q(x_i, \bar{\mu}), k_i)$$

$$\mu^* = \arg\min_{\mu} Risk\, q(\mu)$$

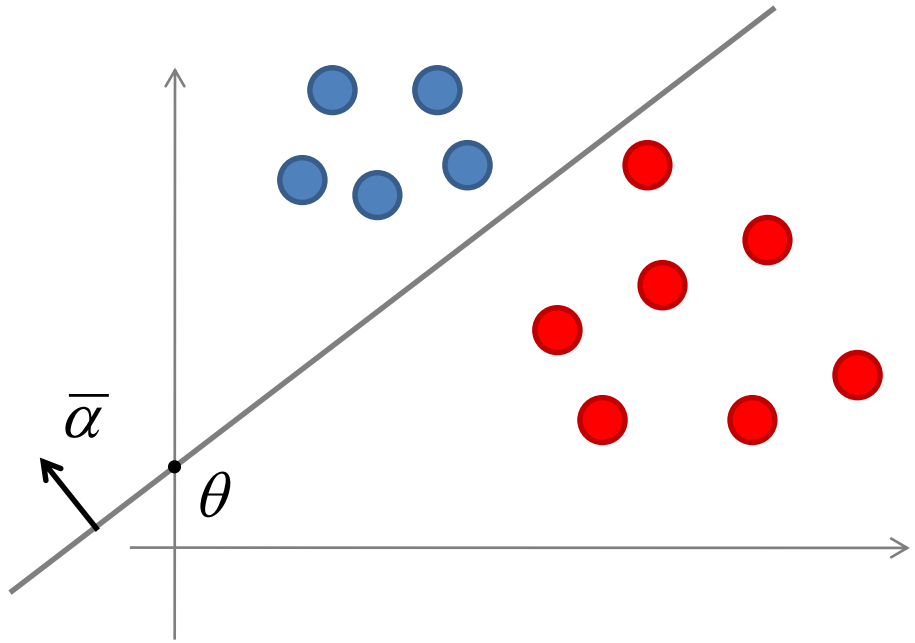
Approaches for ML

ERM: Empirical risk minimization

$$q(\bar{x}, \bar{\alpha}, \theta) = \begin{cases} -1 & when\ (\bar{x}, \bar{\alpha}) < \theta \\ +1 & when\ (\bar{x}, \bar{\alpha}) > \theta \end{cases}$$

$$W(k, k') = \begin{cases} 0 & k = k' \\ 1 & k \neq k' \end{cases}$$

$$Risk\left(q\left(\bar{\alpha}, \theta\right)\right) = \quad \text{Number of errors}$$



Bayesian approach

Bayesian approach

Definitions

$x \in X$ feature

$k \in \mathrm{K}$ state (hidden)

$p(x,k)$ joint probability

$W : \mathrm{K} \times \mathrm{K} \to R$ penalty function

$q : X \to \mathrm{K}$ decision strategy

$$Risk(q) = \sum_{k \in \mathrm{K}} \sum_{x \in X} p(x,k) \cdot W(k, q(x))$$

Bayesian approach

The problem

$x \in X$ feature

$k \in \mathrm{K}$ state (hidden)

$p(x,k)$ joint probability

$W : \mathrm{K} \times \mathrm{K} \to R$ penalty function

input

$q : X \to \mathrm{K}$ decision strategy

$$q^*(x) = \arg\min_{q(x)} Risk(q(x)) = \arg\min_{q(x)} \sum_{k \in \mathrm{K}} \sum_{x \in X} p(x,k) \cdot W(k, q(x))$$

output

Bayesian approach

Some basics

$$p(x,k) = p(k) \cdot p(x|k) = p(x) \cdot p(k|x)$$ joint probability

$$p(x) = \sum_k p(k) \cdot p(x|k)$$ law of total probability

$$p(k|x) = \frac{p(k) \cdot p(x|k)}{\sum_{k'} p(k') \cdot p(x|k')}$$ Bayes' theorem

---

p(k)	prior probability
p(k|x)	posterior probability
p(x|k)	conditional probability of *x*, assuming *k*
p(x)	probability of *x*

Bayesian approach

Example

$X = \{8, 9, 10, 11, 12\}$ time

K = {1-revision, 2-free_ride}

$p(k = 1 | x = 8) = 10\%$

$p(k = 2 | x = 8) = 90\%$

$W(k = 1, k' = 1) = 5$ $W(k = 2, k' = 1) = 100$

$W(k = 1, k' = 2) = 5$ $W(k = 2, k' = 2) = 0$

Bayesian approach

Flexibility to not make a decision

$$W(k,k') = \begin{cases} 0 & when\ k = k' \\ 1 & when\ k \neq k' \\ \varepsilon & when\ k = reject \end{cases}$$

$$Risk(reject) = \sum_{k \in K} p(k|x) \cdot W(k, reject) =$$

$$= \sum_{k \in K} p(k|x) \cdot \varepsilon = \varepsilon$$

Bayesian approach

Flexibility to not make a decision

$$\varepsilon = 0$$

if $\min_k Risk(k) < \varepsilon$

then $k^* = \arg\min_k \sum_{k' \in K} p(k|x) \cdot W(k,k')$

else reject

$$\varepsilon = \infty$$

if $\min_k Risk(k) < \varepsilon$

then $k^* = \arg\min_k \sum_{k' \in K} p(k|x) \cdot W(k,k')$

else reject

Bayesian approach

Decision strategies for different penalty functions

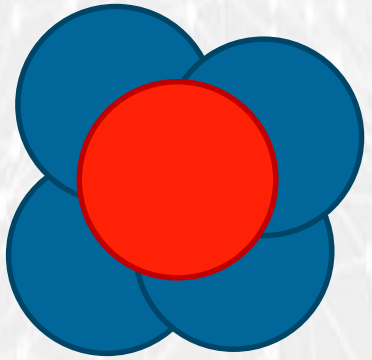
$$w(k,k') = |k - k'|$$ median

$$w(k,k') = (k - k')^2$$ average

$$w(k,k') = \begin{cases} 0 & k = k' \\ 1 & k \neq k' \end{cases}$$ the most probable sample

$$w(k,k') = \begin{cases} 0 & |k - k'| < \Delta \\ 1 & |k - k'| \geq \Delta \end{cases}$$ center of the most probable section Δ

Classifying the independent binary features



Classifying the independent binary features

The problem

$\bar{x} = (x_1, x_2, \ldots, x_N)$ feature – a set of independent binary values

$$p(\bar{x}|k) = p(x_1|k) \cdot p(x_2|k) \cdot \ldots \cdot p(x_N|k)$$

$k \in \mathrm{K} = \{1, 2\}$ Few states are possible

$$\frac{p(\bar{x}|k=1)}{p(\bar{x}|k=2)} \begin{matrix} 1 \\ \geq \\ \leq \\ 2 \end{matrix} \theta$$ Decision strategy

Classifying the independent binary features

The problem

$$\log \frac{p(\bar{x}|k=1)}{p(\bar{x}|k=2)} = \log \frac{p(x_1|k=1)\cdot \ldots \cdot p(x_n|k=1)}{p(x_1|k=2)\cdot \ldots \cdot p(x_n|k=2)} = \log \frac{p(x_1|k=1)}{p(x_1|k=2)} + \ldots + \log \frac{p(x_n|k=1)}{p(x_n|k=2)} =$$

$$x_1 \cdot \log \frac{p(x_1=1|k=1)\cdot p(x_1=0|k=2)}{p(x_1=1|k=2)\cdot p(x_1=0|k=1)} + \log \frac{p(x_1=0|k=1)}{p(x_1=0|k=2)} +$$

$$\vdots$$

$$x_N \cdot \log \frac{p(x_n=1|k=1)\cdot p(x_n=0|k=2)}{p(x_n=1|k=2)\cdot p(x_n=0|k=1)} + \log \frac{p(x_n=0|k=1)}{p(x_n=0|k=2)}$$

Classifying the independent binary features

Decision rule

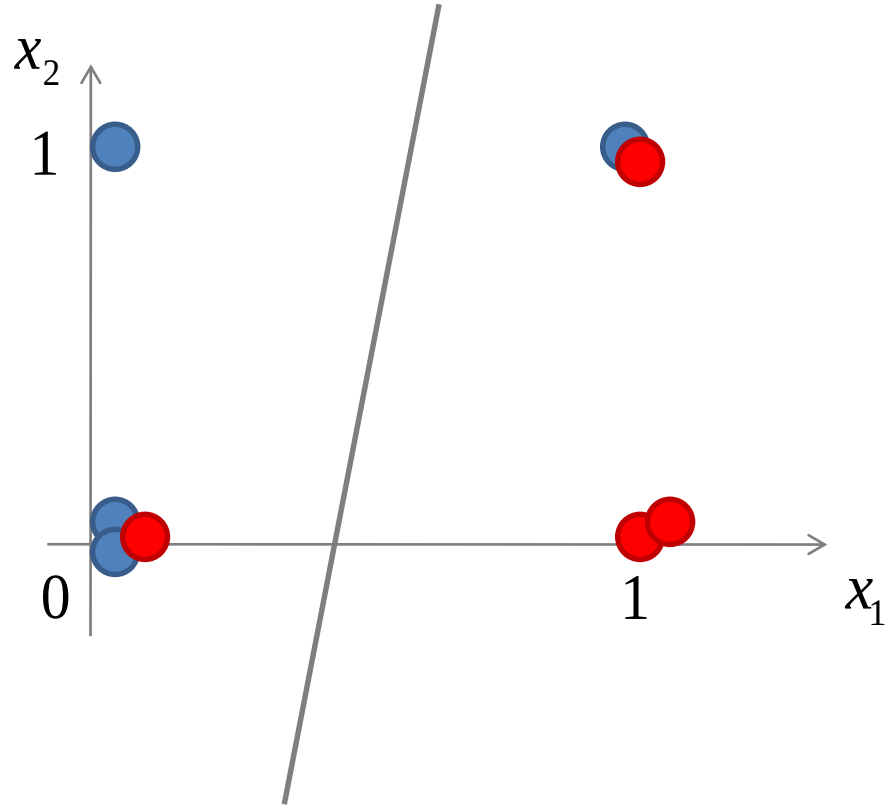
$$\sum_{i=1}^{N} x_i \cdot \log \frac{p(x_i = 1 | k = 1) \cdot p(x_i = 0 | k = 2)}{p(x_i = 1 | k = 2) \cdot p(x_i = 0 | k = 1)} + \log \frac{p(x_i = 0 | k = 1)}{p(x_i = 0 | k = 2)} \begin{matrix} 1 \\ \geq \\ \leq \\ 2 \end{matrix} \log \theta$$

$$\sum_{i=1}^{N} x_i \cdot a_i \begin{matrix} 1 \\ \geq \\ \leq \\ 2 \end{matrix} \theta'$$

Classifying the independent binary features

Example

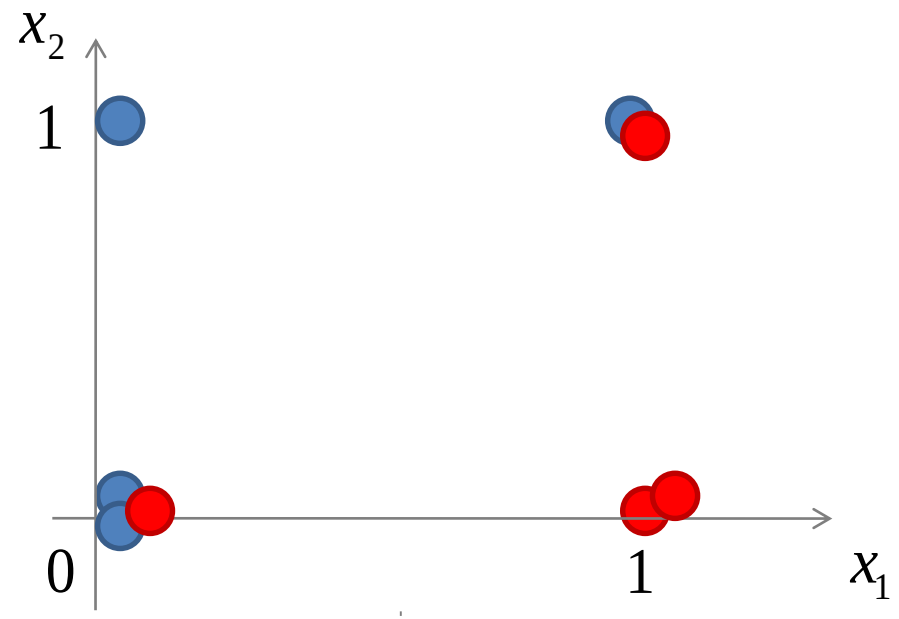
$$\sum_{i=1}^{N} x_i \cdot a_i \quad \begin{matrix} 1 \\ \geq \\ \leq \\ 2 \end{matrix} \quad \theta'$$



Classifying the independent binary features

Example

$p(x_2 \mid \bullet)$	$p(x_2 \mid \bullet)$
1/4	2/4
3/4	2/4



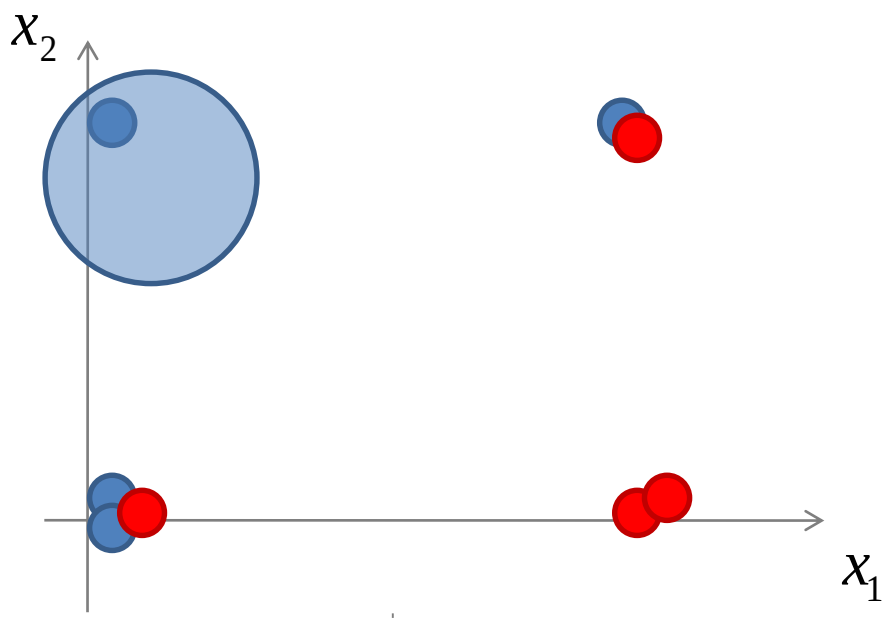
1/4	3/4	$p(x_1 \mid \bullet)$
3/4	1/4	$p(x_1 \mid \bullet)$

Classifying the independent binary features

Example

$p(x_2|\bullet)$ $p(x_2|\bullet)$

1/4	2/4
3/4	2/4

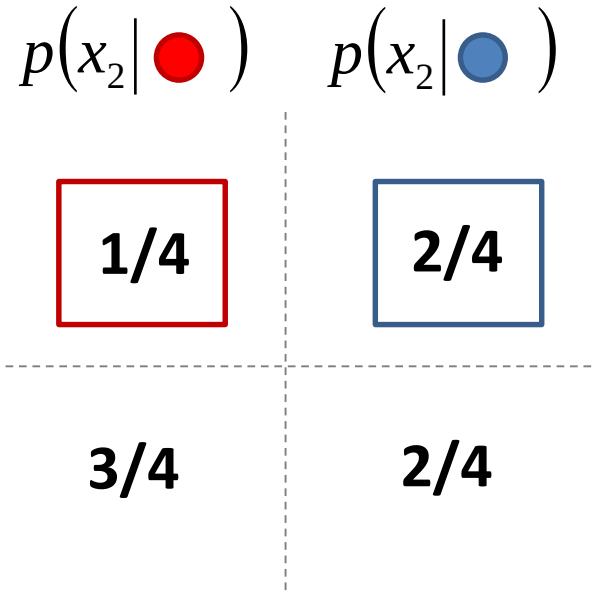


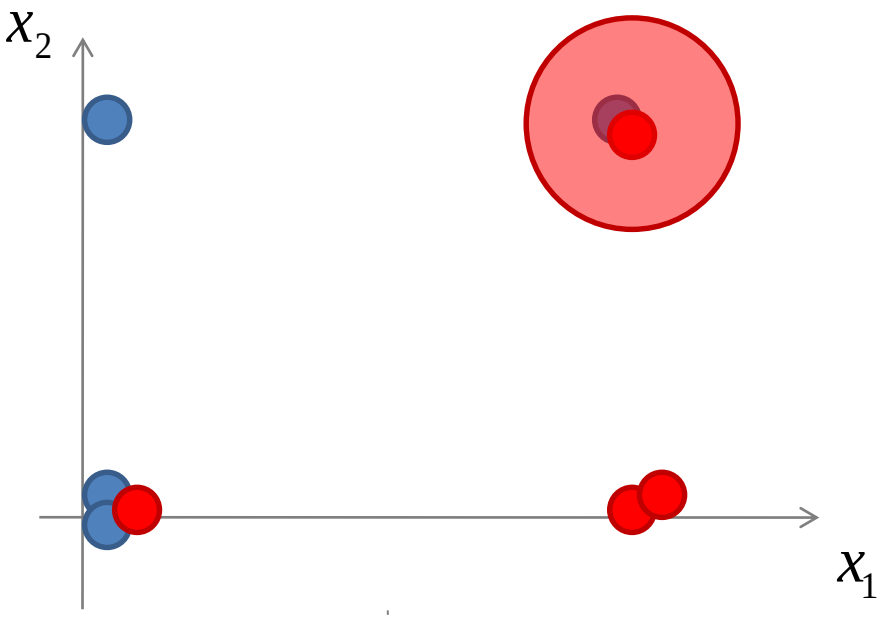
1/4	3/4	$p(x_1|\bullet)$
3/4	1/4	$p(x_1|\bullet)$

(1/4)x(1/4) < (2/4)x(3/4)

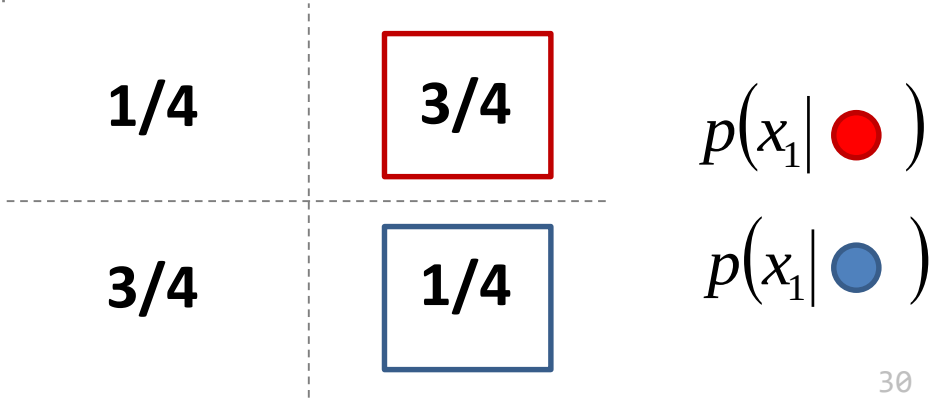
Classifying the independent binary features

Example





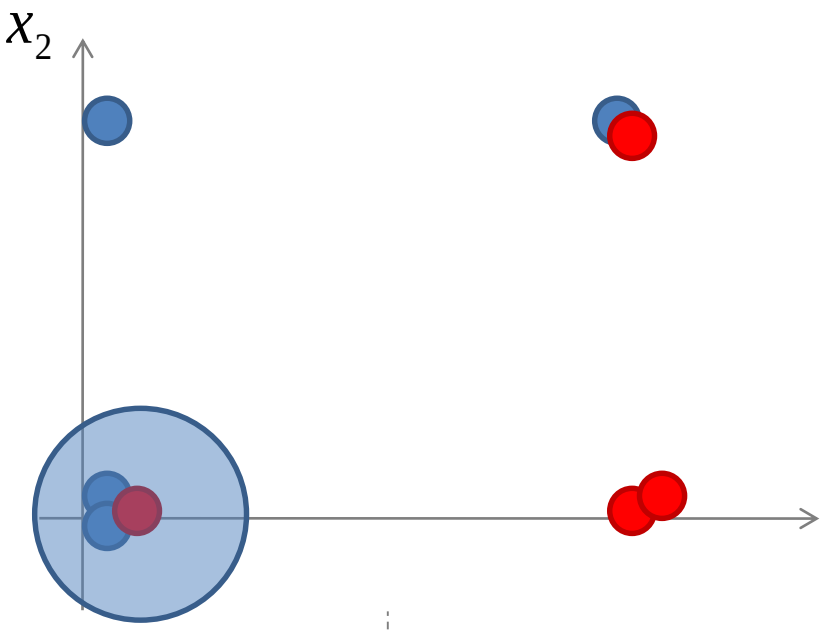
(1/4)x(3/4) > (2/4)x(1/4)



Classifying the independent binary features

Example

$p(x_2 \mid \bullet)$	$p(x_2 \mid \bullet)$
1/4	2/4
3/4	2/4

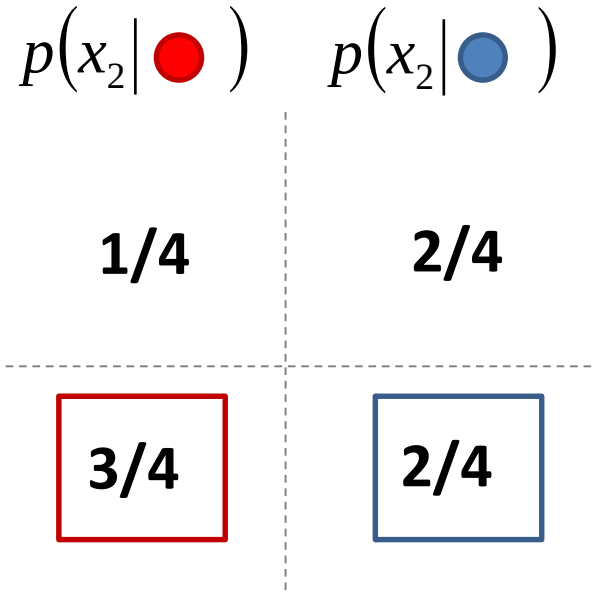


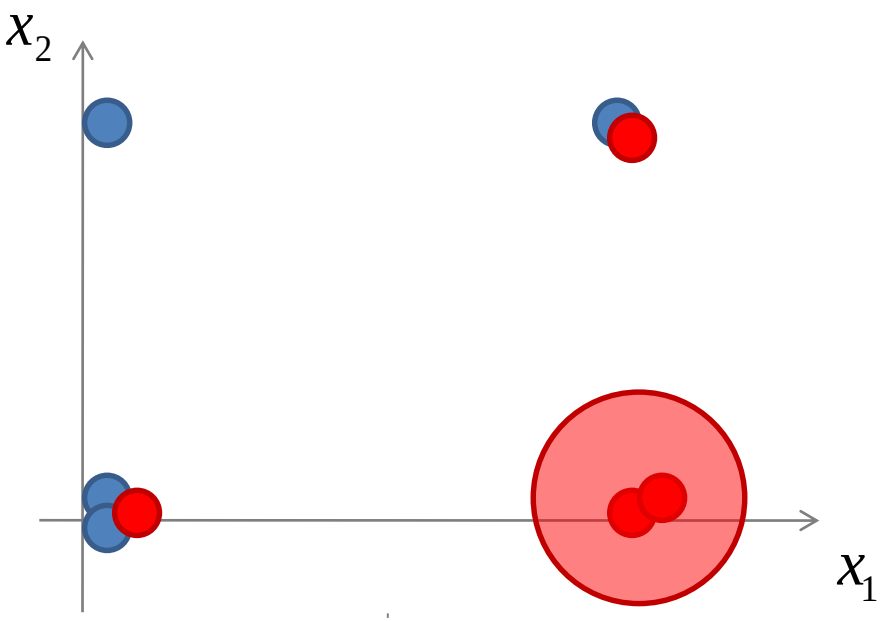
$p(x_1 \mid \bullet)$	$p(x_1 \mid \bullet)$
1/4	3/4
3/4	1/4

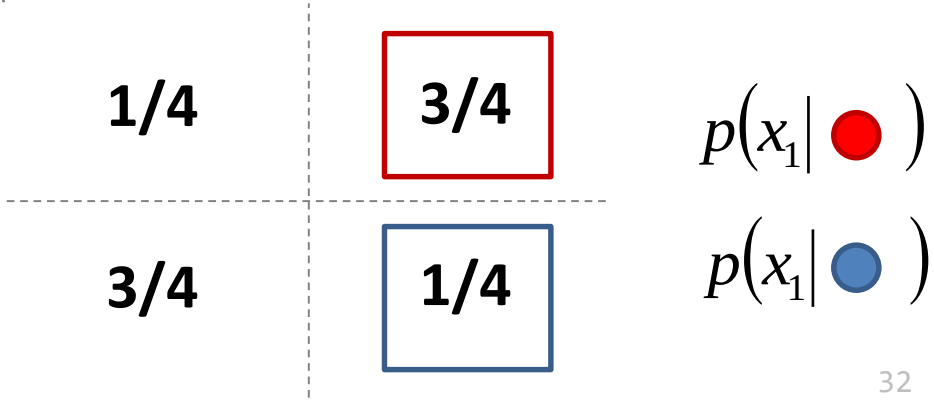
(3/4)x(1/4) < (2/4)x(3/4)

Classifying the independent binary features

Example



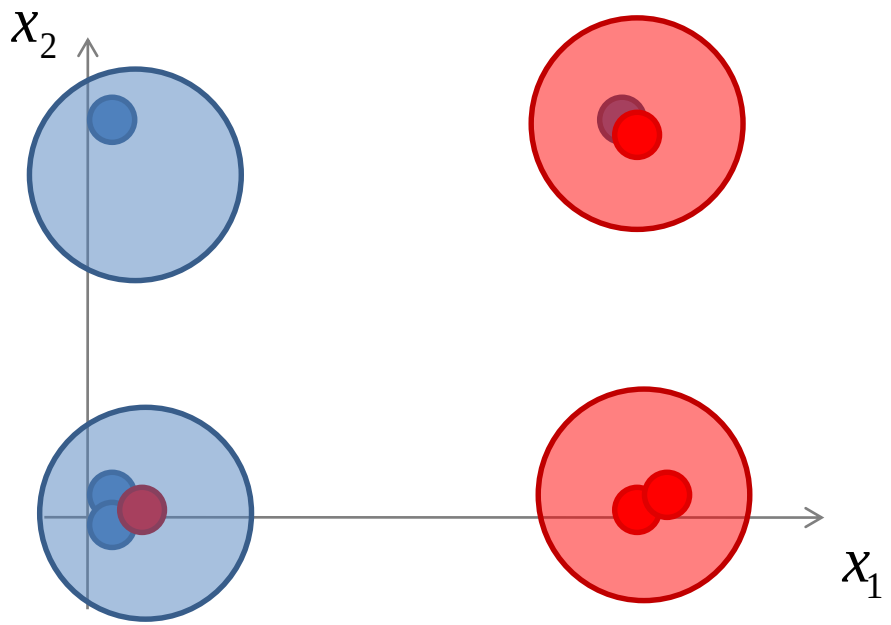




(3/4)x(3/4) > (2/4)x(1/4)

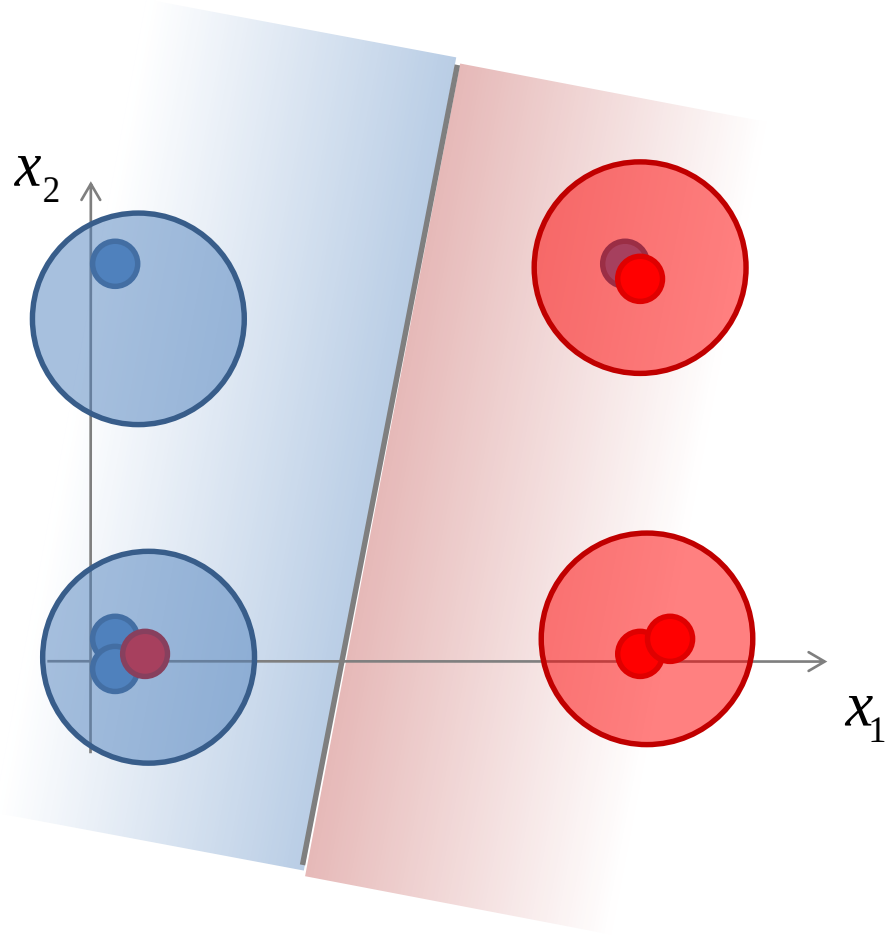
Classifying the independent binary features

Example



Classifying the independent binary features

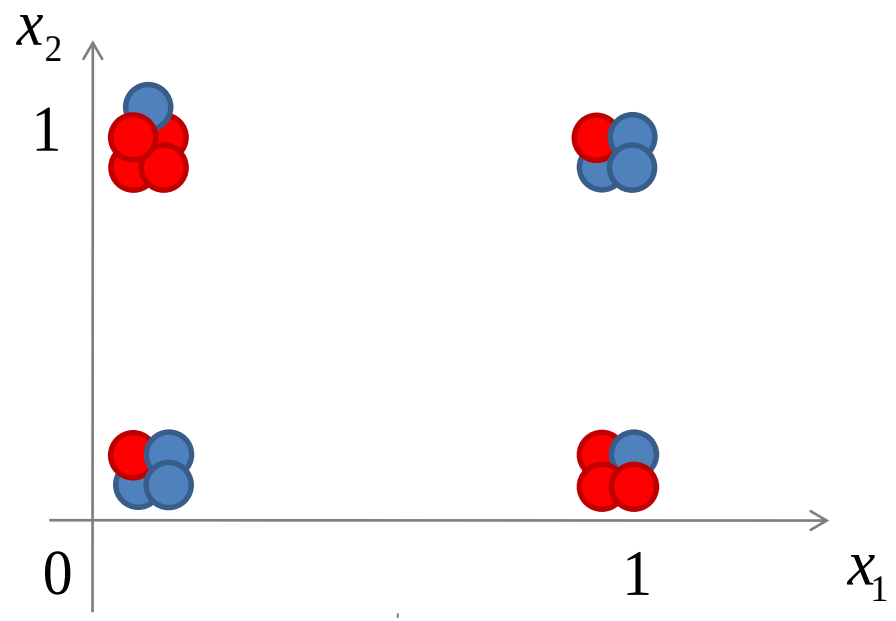
Example



Classifying the independent binary features

Another example

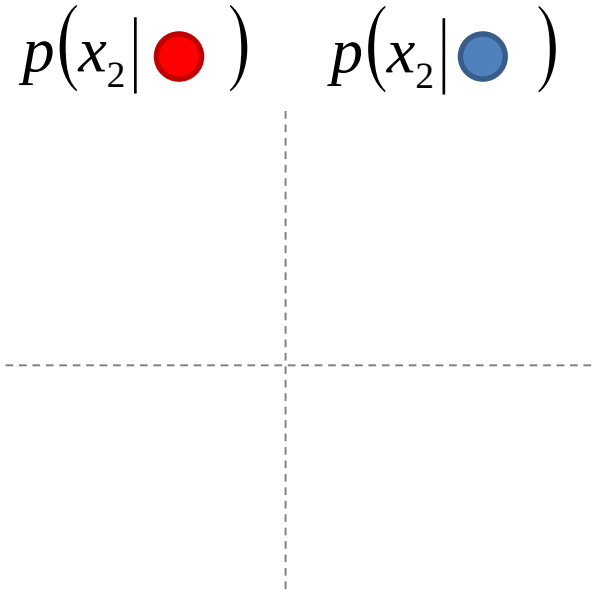
$p(x_2 \mid \bullet)$	$p(x_2 \mid \bullet)$
5/9	4/8
4/9	4/8

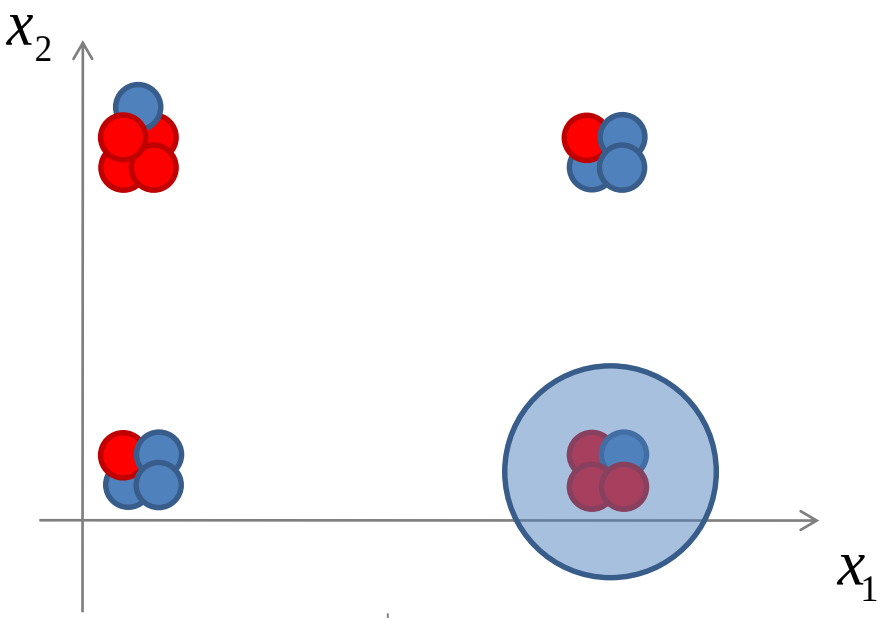


5/9	4/8	$p(x_1 \mid \bullet)$
4/8	4/8	$p(x_1 \mid \bullet)$

Classifying the independent binary features

Another example

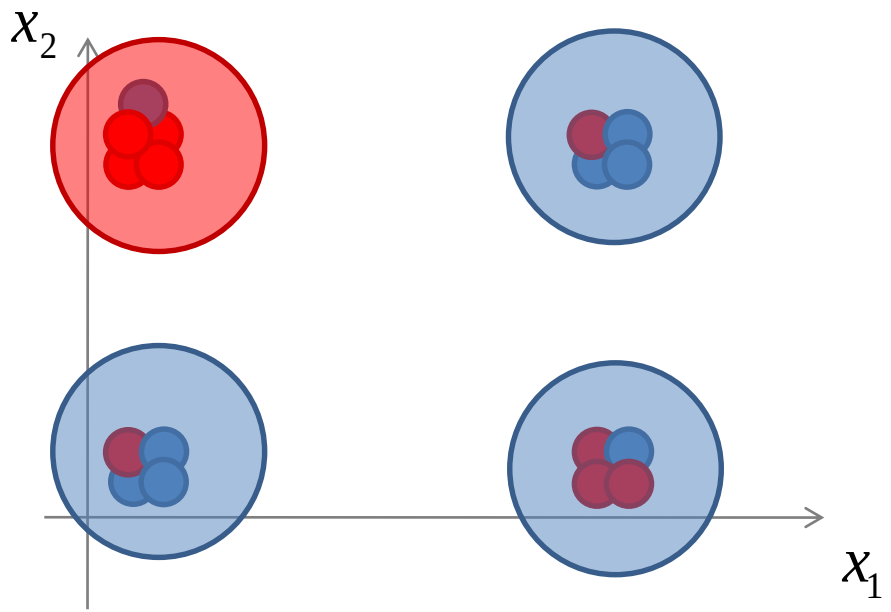




(4/9)x(4/9) < (4/8)x(4/8)

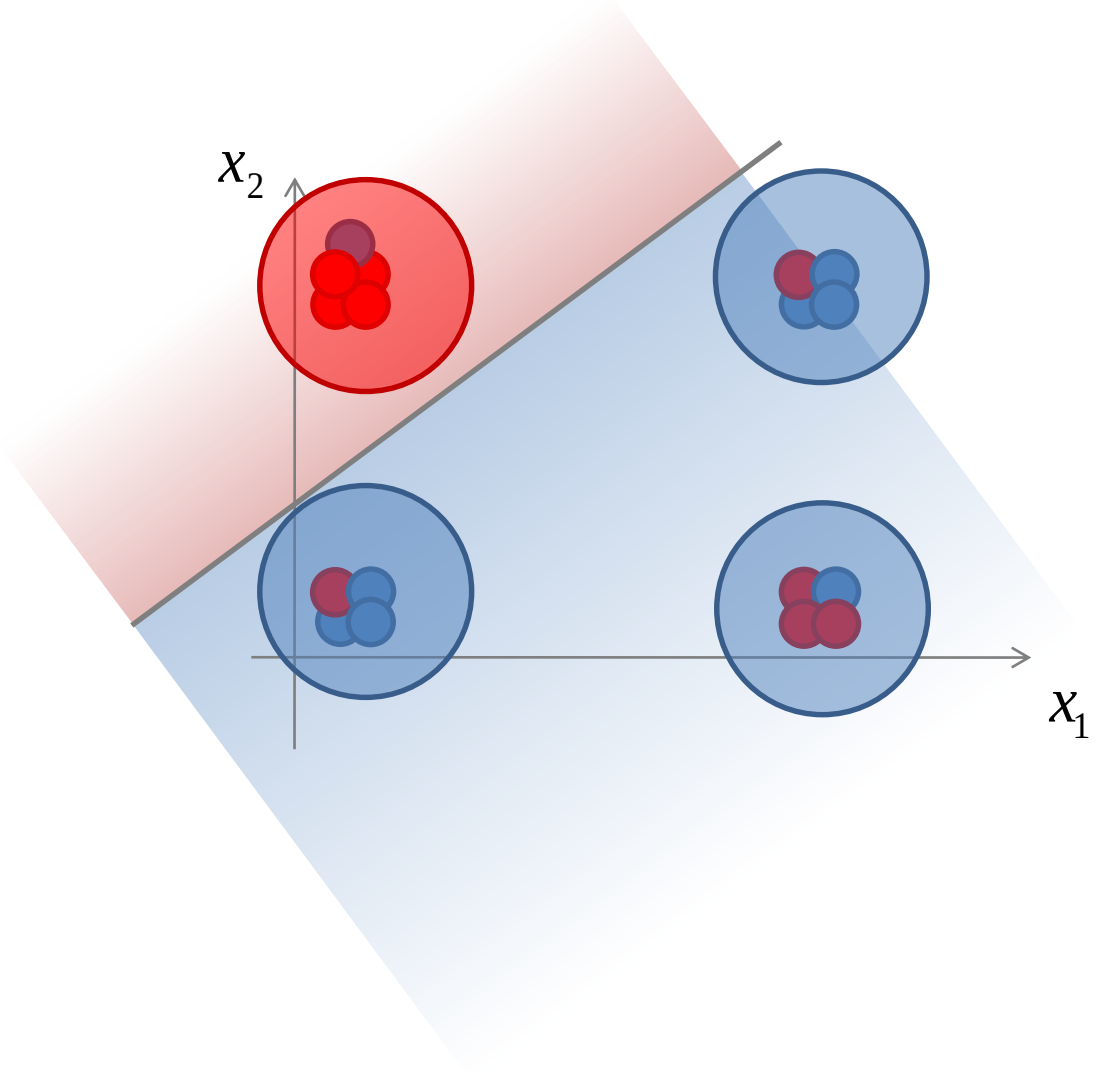
Classifying the independent binary features

Another example



Classifying the independent binary features

Another example



Part 2: Supervised Learning

Naive Bayesian classifier
Gaussian classifier
Linear discrimination
Logistic regression
Ensemble learning
Decision Tree
Random Forest
Adaboost

Naive Bayesian classifier

Naive Bayesian classifier

The problem

$\bar{x} = (x_1, x_2, \ldots, x_N)$ feature

$$p(\bar{x}|k) = p(x_1|k) \cdot p(x_2|k) \cdot \ldots \cdot p(x_N|k)$$

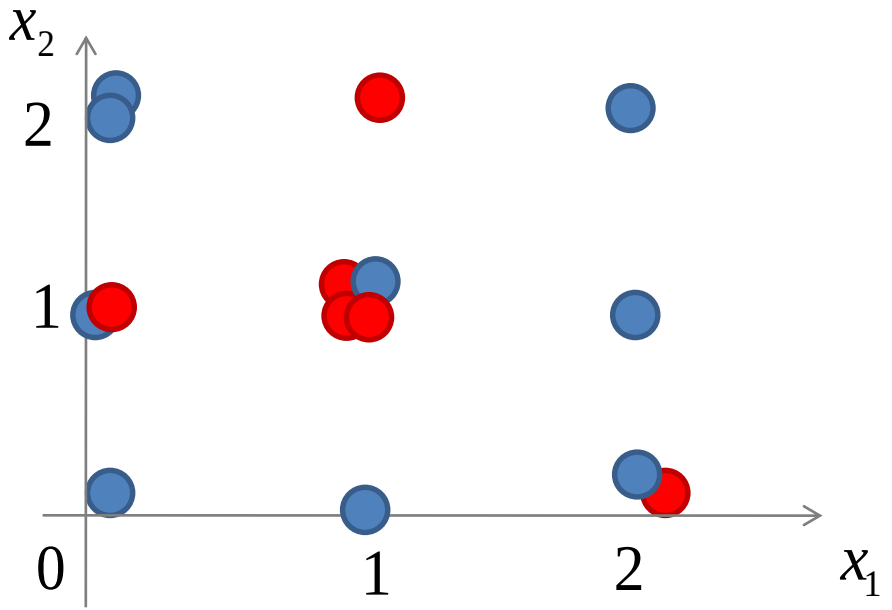
$k \in \mathrm{K} = \{1, 2\}$ few states are possible

$$\frac{p(\bar{x}|k=1)}{p(\bar{x}|k=2)} \begin{matrix} \geq \\ \theta \\ \leq \end{matrix} \begin{matrix} 1 \\ \\ 2 \end{matrix}$$ decision strategy

Naive Bayesian classifier

Example

$p(x_2 \mid \bullet)$	$p(x_2 \mid \bullet)$
1/6	3/9
4/6	3/9
1/6	3/9

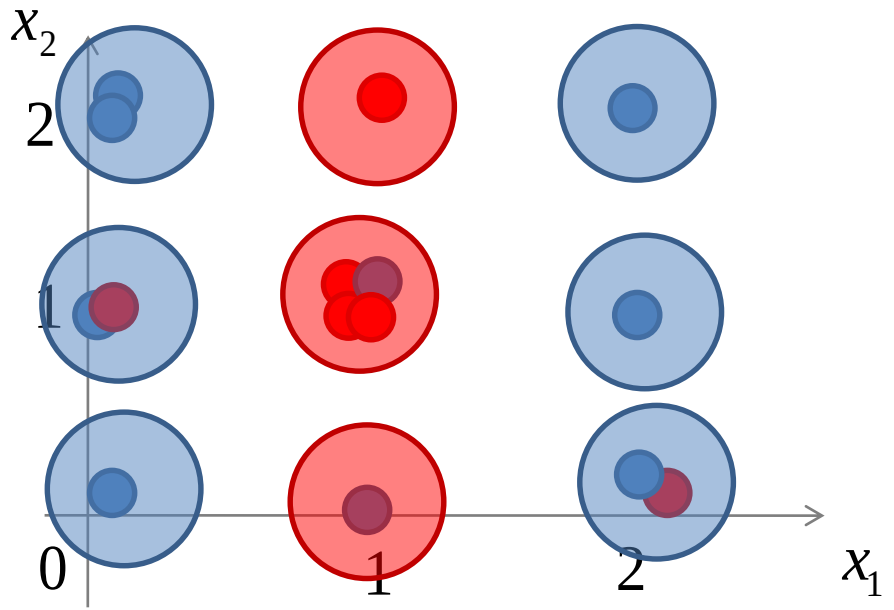


1/6	4/6	1/6	$p(x_1 \mid \bullet)$
4/9	2/9	3/9	$p(x_1 \mid \bullet)$

Naive Bayesian classifier

Example

$p(x_2 \mid \bullet)$	$p(x_2 \mid \bullet)$
1/6	3/9
4/6	3/9
1/6	3/9



1/6	4/6	1/6	$p(x_1 \mid \bullet)$
4/9	2/9	3/9	$p(x_1 \mid \bullet)$

Gaussian classifier

Gaussian classifier

Definitions

$$\bar{x} = \left(x_1, x_2, \ldots, x_N\right)$$ feature

$$p\left(\bar{x}\middle|k\right) \cong \exp\left(-0.5 \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} a_{i,j}^{[k]} \left(x_i - \mu_i^{[k]}\right)\left(x_j - \mu_j^{[k]}\right) \right)$$

$$A^{[k]} = \left(B^{[k]}\right)^{-1}$$

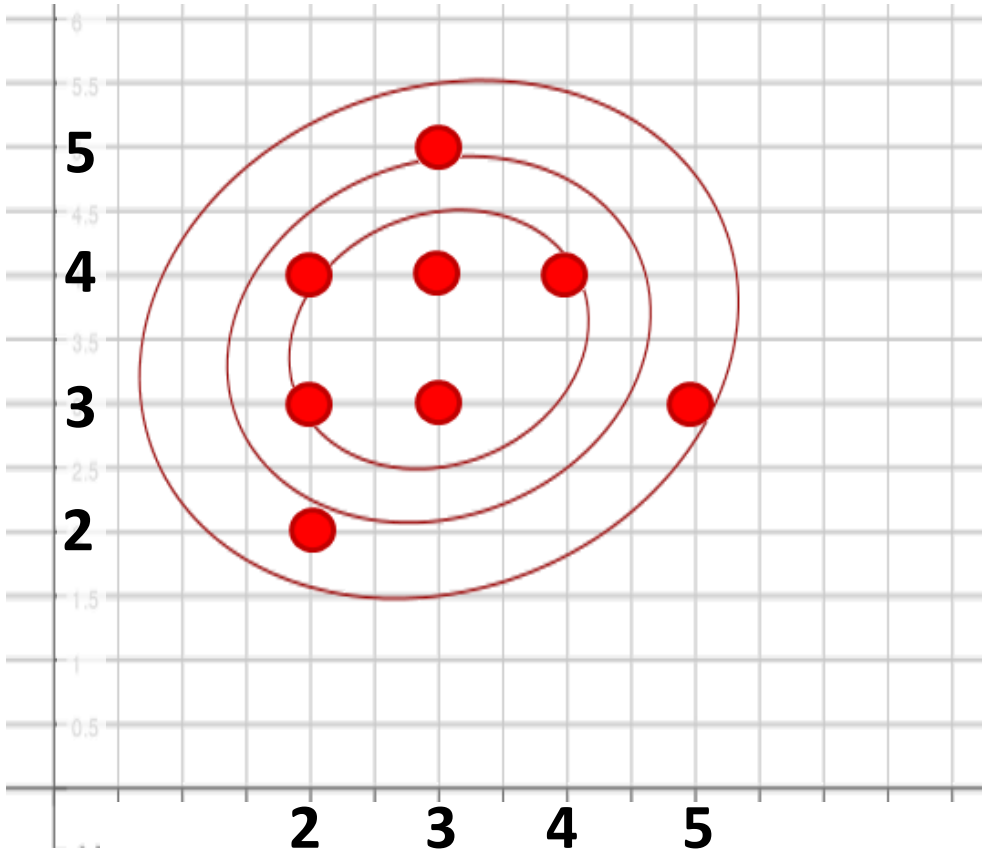
$$\mu_i^{[k]} = M.O.\left(x_i\middle|k\right)$$

$$b_{ij}^{[k]} = M.O.\left(x_i - \mu_i^{[k]}\right) \cdot \left(x_j - \mu_j^{[k]}\right)$$

Gaussian classifier

Example

$$p(\bar{x}|\bullet) \cong \exp\left(-0.5 \cdot \sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j}(x_i - \mu_i)(x_j - \mu_j)\right)$$



Gaussian classifier

$$\mu_1 = MO(x_1|\bullet) = 3$$

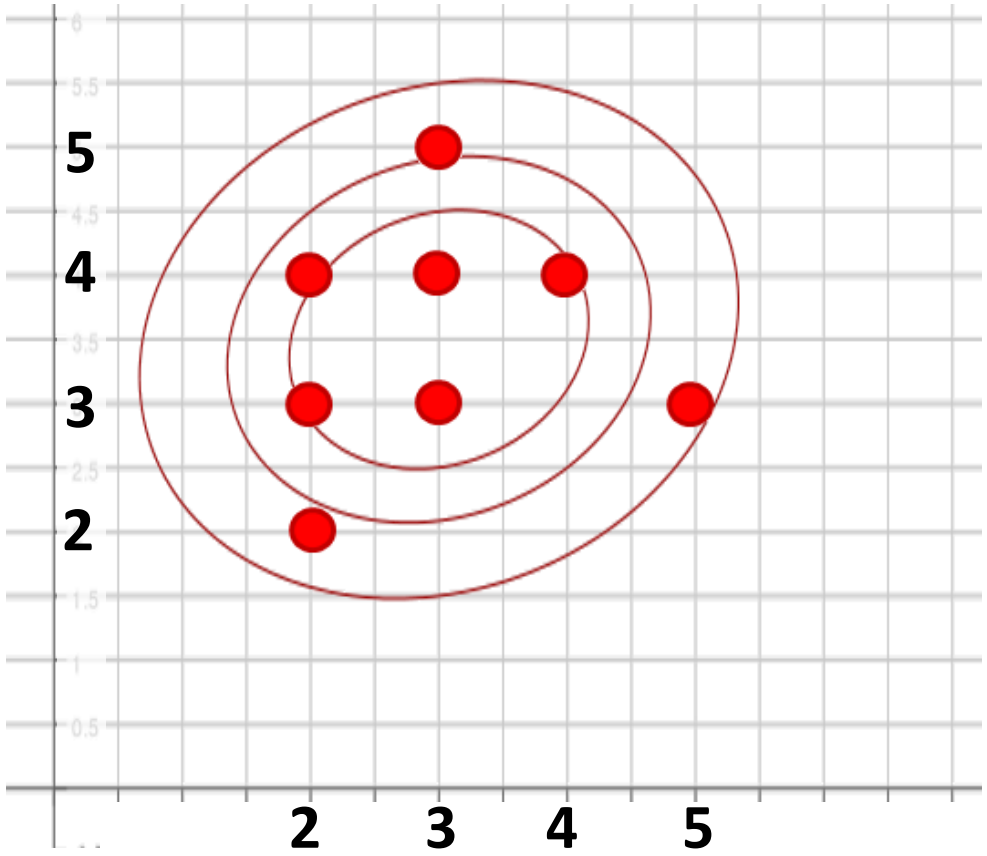
$$\mu_2 = MO(x_2|\bullet) = 3{,}5$$

$$b_{11} = \underset{(1+1+1+0+0+0+1+4)/8}{MO}(x_1-\mu_1)\cdot(x_1-\mu_1) = 1$$

$$b_{22} = MO(x_2-\mu_2)\cdot(x_2-\mu_2) = 3/4$$

$$b_{12} = b_{21} = MO(x_1-\mu_1)\cdot(x_2-\mu_2) = 1/8$$

$$B = \begin{bmatrix} 1 & 1/8 \\ 1/8 & 3/4 \end{bmatrix}^{-1} = \frac{64}{47}\cdot\begin{bmatrix} 3/4 & -1/8 \\ -1/8 & 1 \end{bmatrix} = A$$



$$p(\bar{x}|\bullet) \cong \exp\left(\frac{64}{47}\cdot\left(-\frac{3}{4}\cdot(x_1-3)^2+\frac{1}{4}(x_1-3)(x_2-3{,}5)-\frac{1}{1}(x_2-3{,}5)^2\right)\right)$$

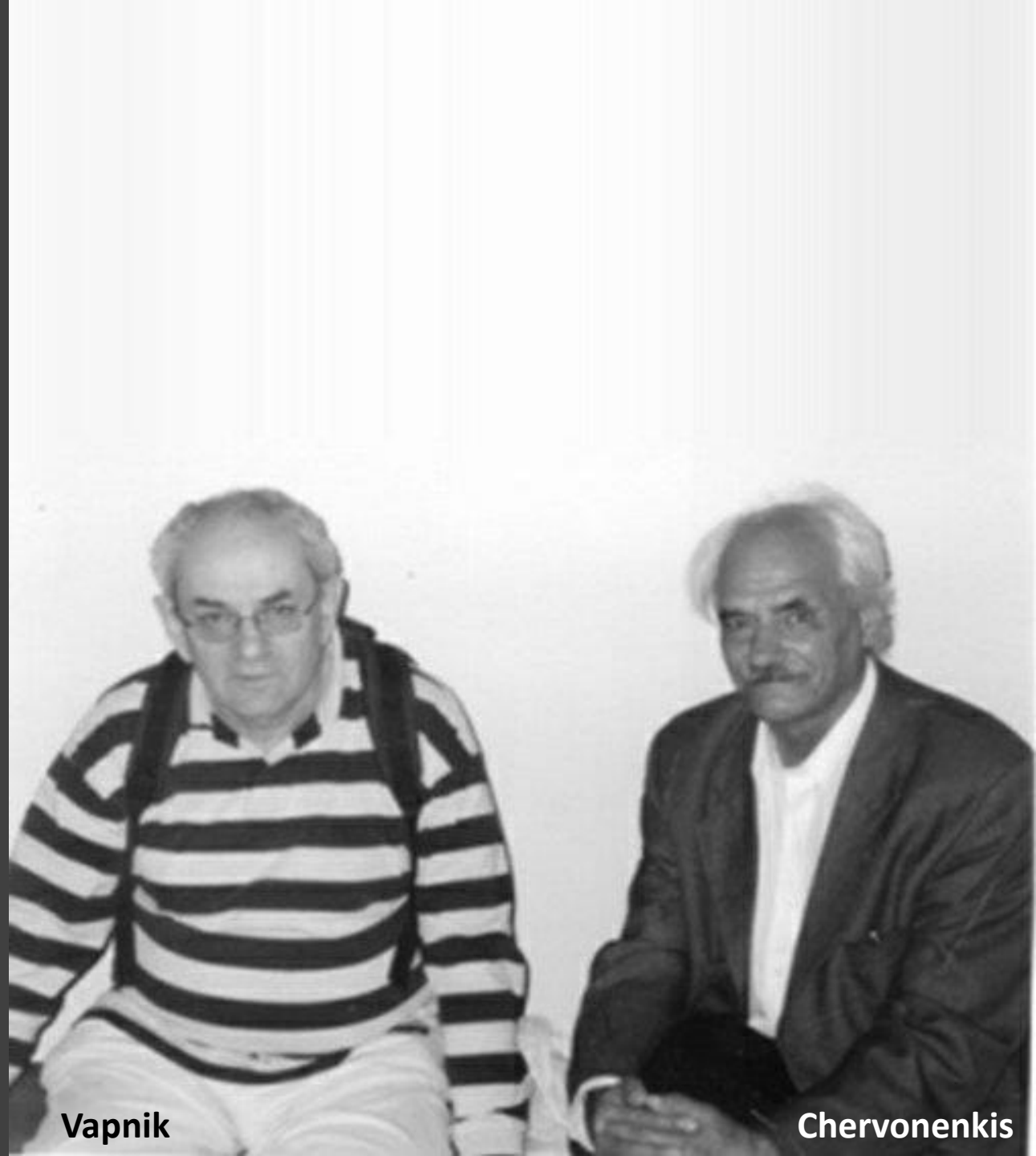
Gaussian classifier

Decision strategy

$$\log \frac{p(\bar{x}|k=1)}{p(\bar{x}|k=2)} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j}^{[1]}\left(x_i-\mu_i^{[1]}\right)\left(x_j-\mu_j^{[1]}\right)}{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j}^{[2]}\left(x_i-\mu_i^{[2]}\right)\left(x_j-\mu_j^{[2]}\right)}$$

$$\sum_i \sum_j \alpha_{ij} \cdot x_i x_j + \sum_i \beta_i \cdot x_i \begin{matrix} 1 \\ \geq \\ \leq \\ 2 \end{matrix} \theta$$

Linear discrimination



Linear discrimination

Perceptron

input

$$\begin{cases} X = \{x_1, x_2, \ldots, x_r\} \\ X' = \{x'_1, x'_2, \ldots, x'_s\} \end{cases}$$

output

$\alpha \in R^n$

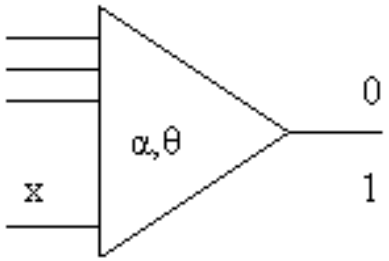
θ

$$\begin{cases} \forall x \in X \quad (x, \alpha) = \sum_{i=1}^{n} x_i \cdot \alpha_i > \theta \\ \forall x' \in X' \quad (x', \alpha) = \sum_{i=1}^{n} x'_i \cdot \alpha_i < \theta \end{cases}$$


Rosenblatt

Linear discrimination

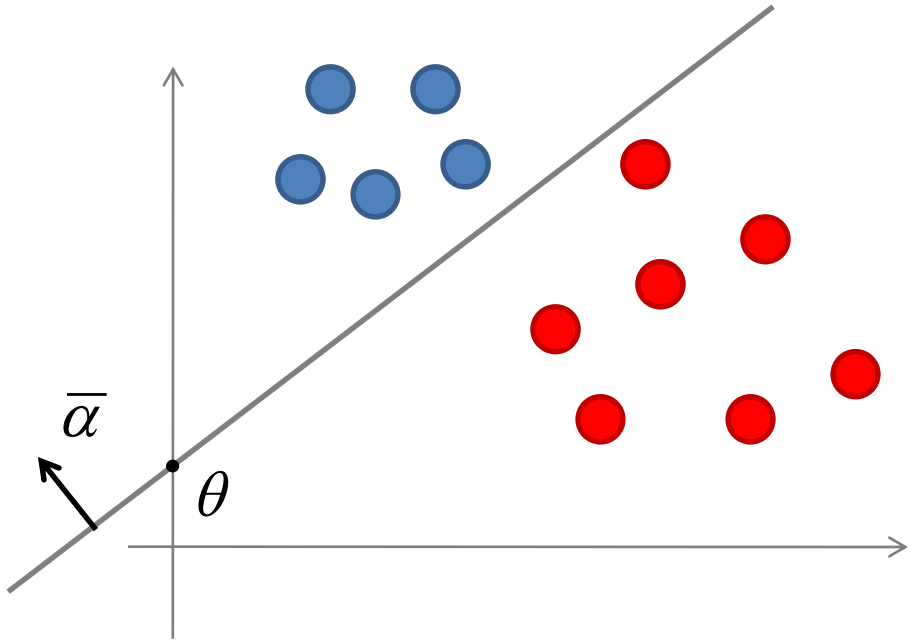
Perceptron



$$\text{output}\begin{cases}\forall x \in X & (x,\alpha)=\sum_{i=1}^{n} x_i \cdot \alpha_i > \theta \\ \forall x' \in X' & (x',\alpha)=\sum_{i=1}^{n} x'_i \cdot \alpha_i < \theta\end{cases}$$

$\alpha \in R^n$

θ



Linear discrimination

Perceptron

$$\begin{cases} \forall x \in X & \sum_{i=1}^{n} x_i \cdot \alpha_i > \theta \\ \forall x' \in X' & \sum_{i=1}^{n} x'_i \cdot \alpha_i < \theta \end{cases} = \begin{cases} \forall x \in X & \sum_{i=1}^{n} x_i \cdot \alpha_i + 1 \cdot \alpha_{n+1} > 0 \\ \forall x' \in X' & \sum_{i=1}^{n} x'_i \cdot \alpha_i + 1 \cdot \alpha_{n+1} < 0 \end{cases}$$

Linear discrimination

Perceptron: tuning

$t = 0$

$\alpha_t = 0$

while (sets are not separated by hyperplane)

$\{$

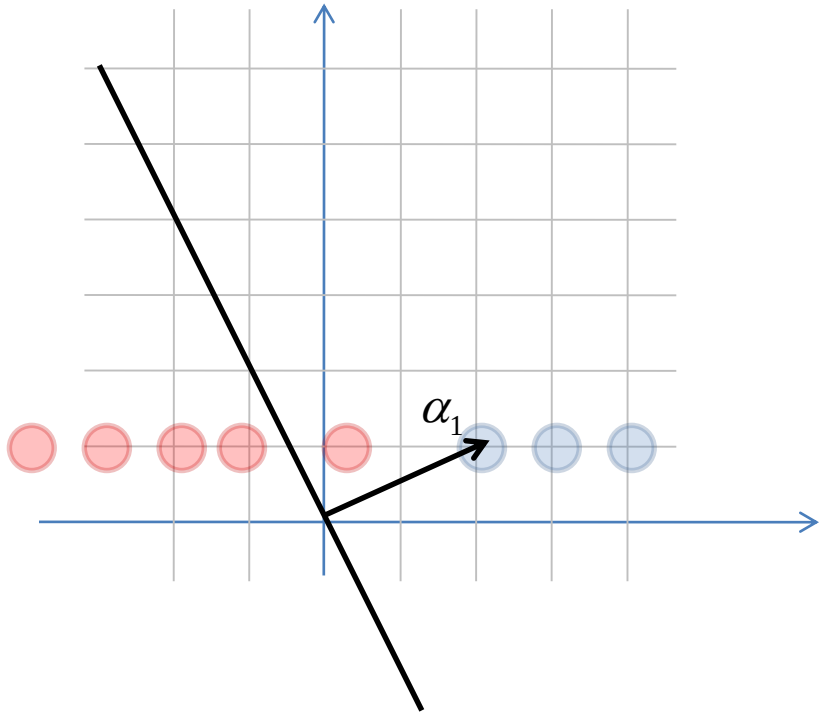
$\quad if\left(\exists x \in X \middle| (x, \alpha_t) < 0\right) \quad \alpha_{t+1} = a_t + x;$

$\quad if\left(\exists x' \in X' \middle| (x', \alpha_t) > 0\right) \quad \alpha_{t+1} = a_t - x';$

$\}$

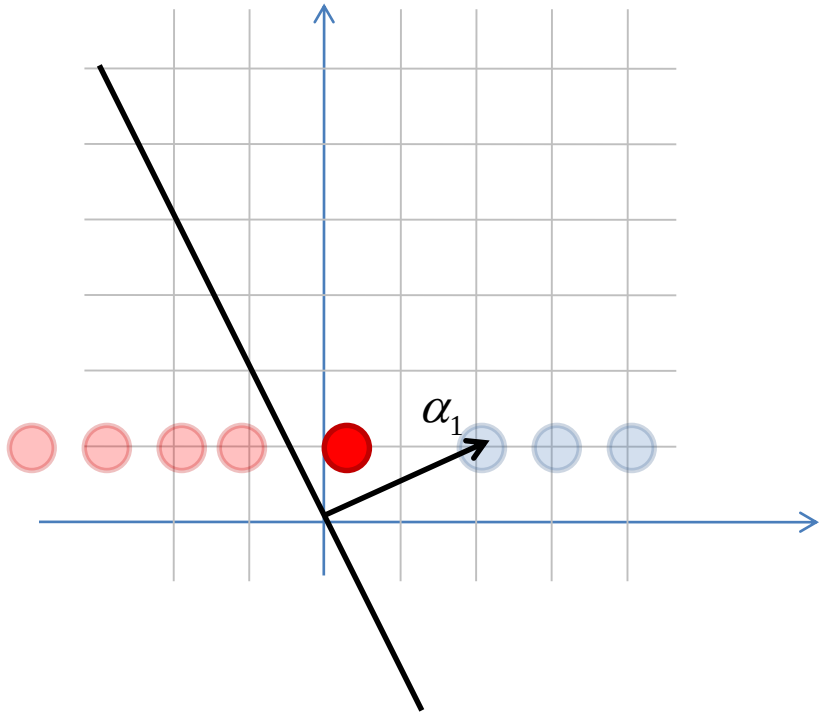
Linear discrimination

Perceptron: tuning



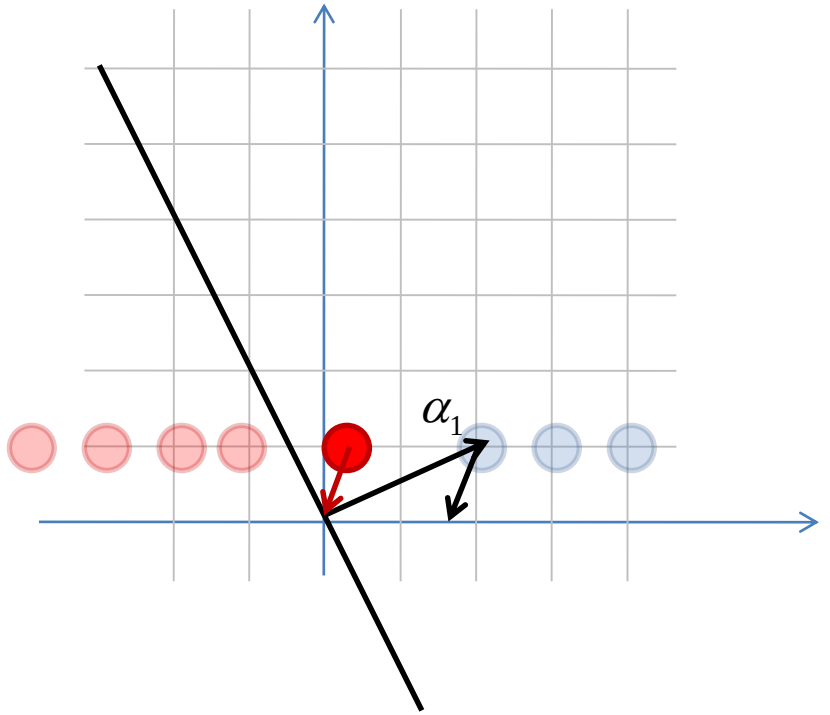
Linear discrimination

Perceptron: tuning



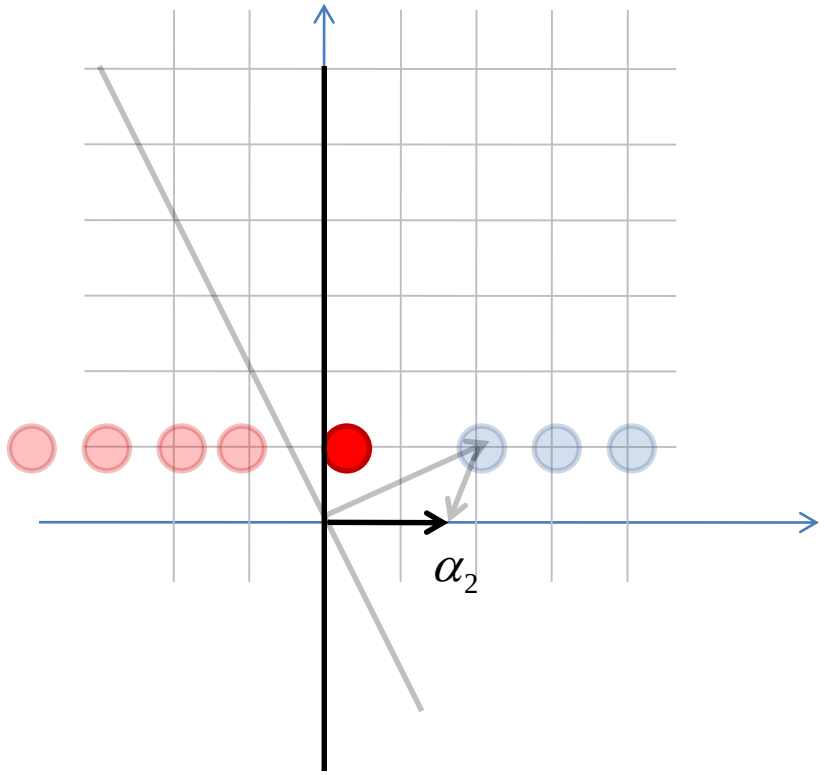
Linear discrimination

Perceptron: tuning



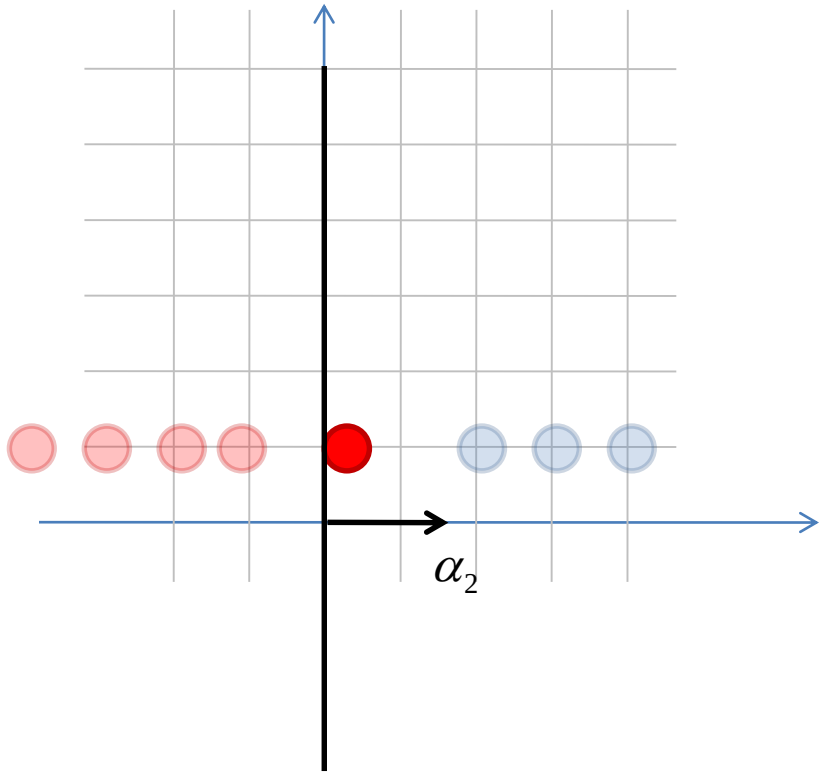
Linear discrimination

Perceptron: tuning



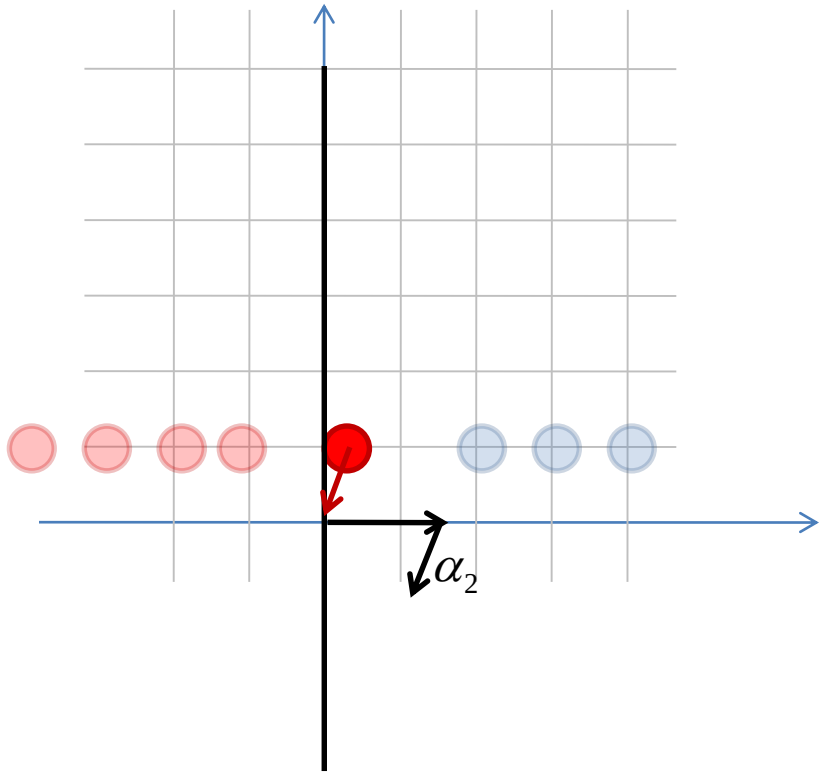
Linear discrimination

Perceptron: tuning



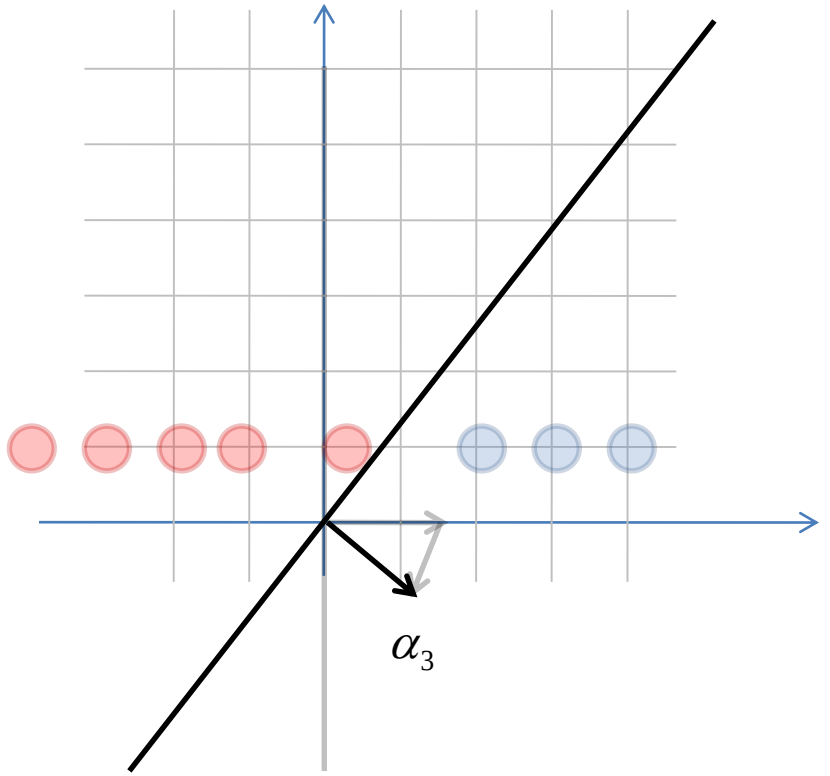
Linear discrimination

Perceptron: tuning



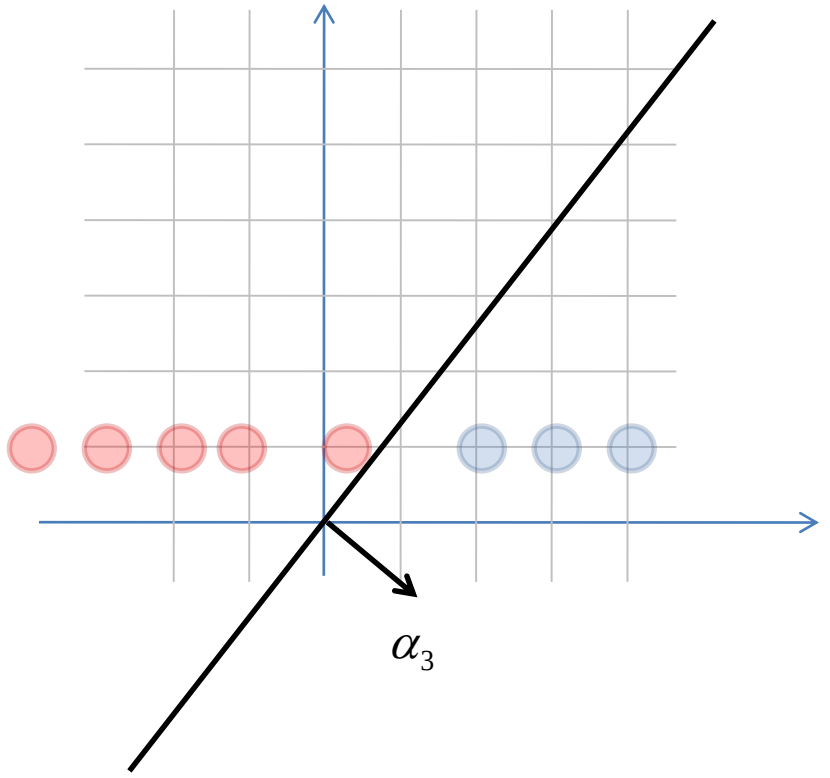
Linear discrimination

Perceptron: tuning



Linear discrimination

Perceptron: tuning



Linear discrimination

Perceptron: convergence

$$\max_{\substack{x \in X \\ x \in X'}} \|x\| = D \qquad \|\alpha^*\| = 1$$

$$\begin{cases} \forall x \in X \quad (x, \alpha^*) \geq \varepsilon > 0 \\ \forall x \in X' \quad (x', \alpha^*) \leq -\varepsilon < 0 \end{cases}$$

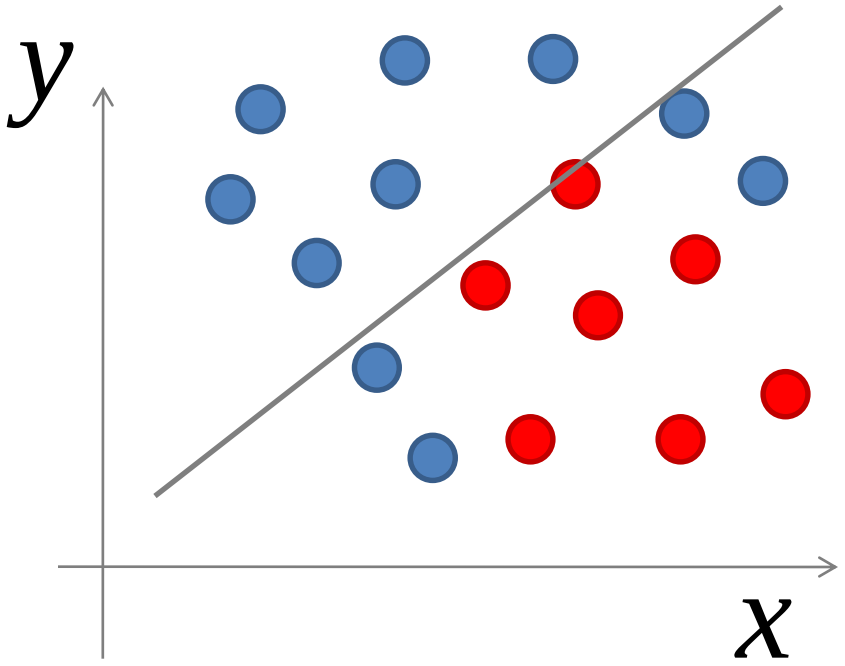
$$t \leq \frac{D^2}{\varepsilon}$$

Linear discrimination

Transformation of the feature space

$$Ax + By \gtrless C$$

a line in R^2



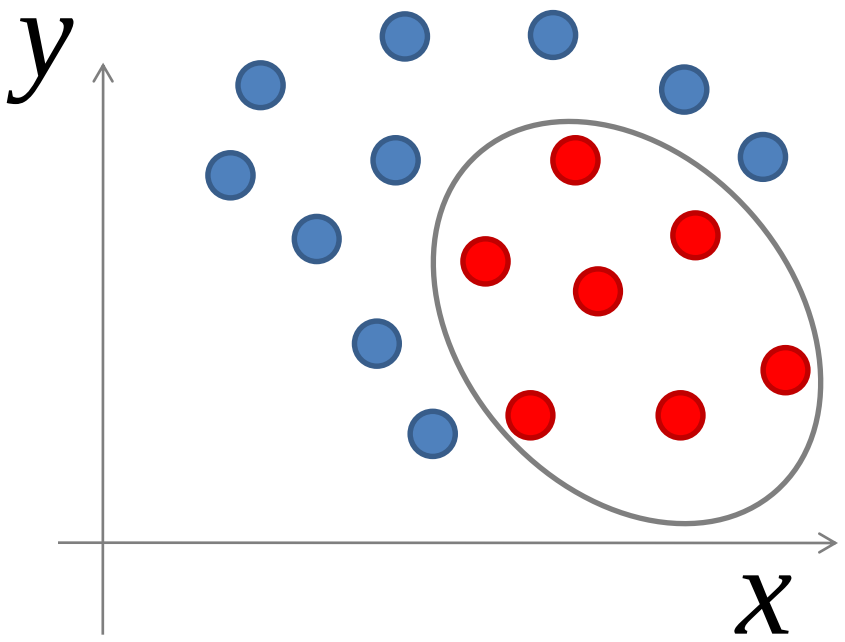
Linear discrimination

Transformation of the feature space

$$Ax^2 + Bxy + Cy^2 + Dx + Ey \gtrless F$$

hyperplane in R^5

elliptic curve in R^2



Linear discrimination

SVM: support vector machine

$$\begin{cases} \forall x \in X \quad (x, \alpha) = \sum_{i=1}^{n} x_i \cdot \alpha_i > \theta \\ \forall x' \in X' \quad (x', \alpha) = \sum_{i=1}^{n} x'_i \cdot \alpha_i < \theta \end{cases}$$

$$(\alpha, \alpha) \to \min$$

> Separate by the widest band

Linear discrimination

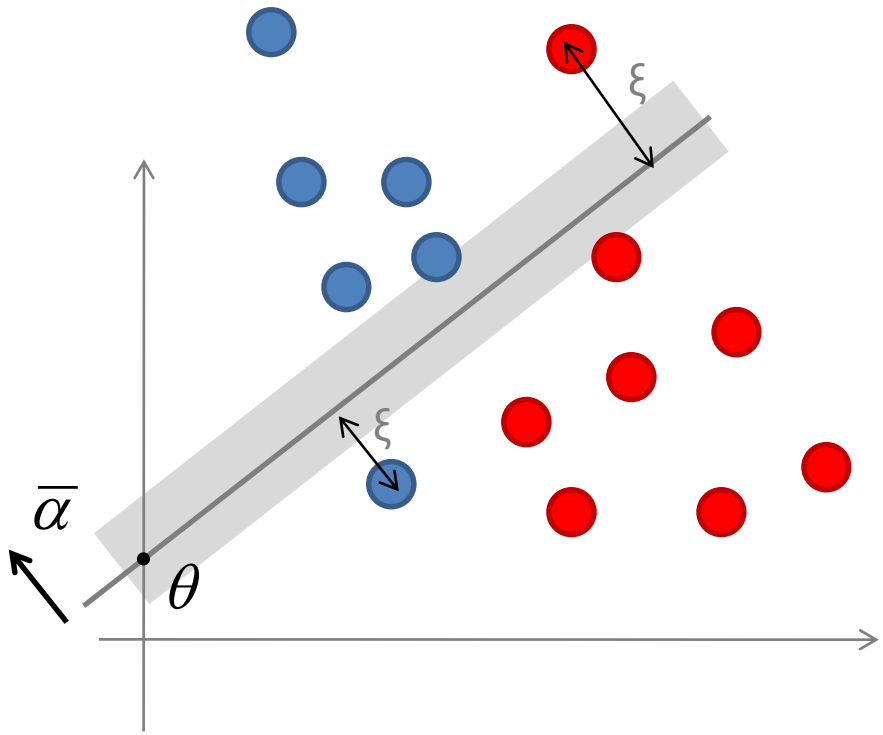
SVM: support vector machine

$$\begin{cases} \forall x \in X \quad (x,\alpha) = \sum_{i=1}^{n} x_i \cdot \alpha_i > \theta - \xi(x) \\ \forall x' \in X' \quad (x',\alpha) = \sum_{i=1}^{n} x'_i \cdot \alpha_i < \theta + \xi(x') \end{cases}$$

penalty

$$(\alpha,\alpha) + const \cdot \sum_{X,X'} \xi(x) \to \min$$

Separate by the widest band

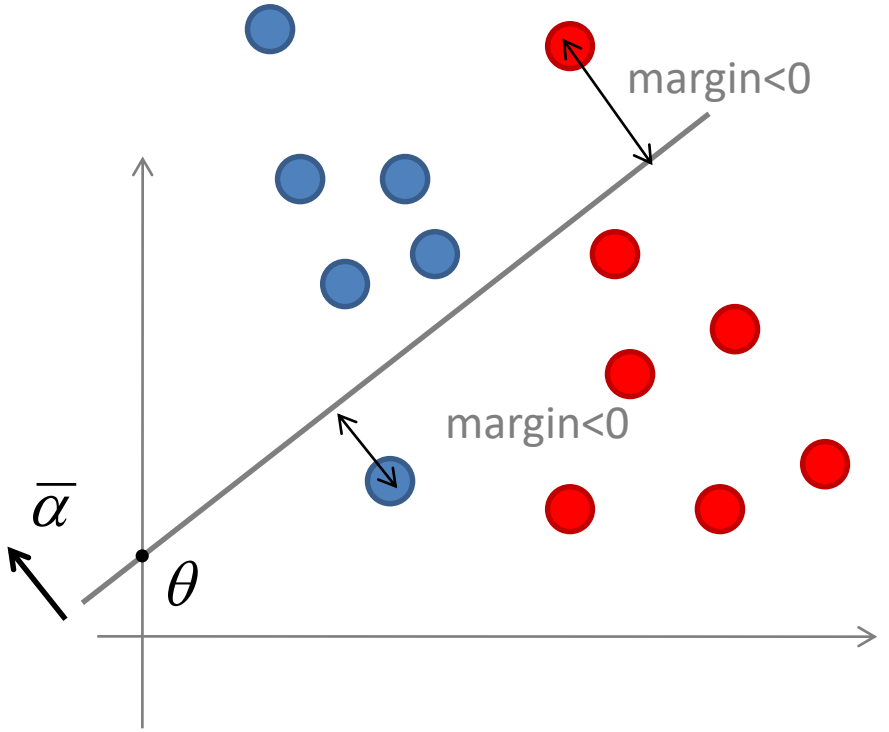


Logistic regression

Logistic regression

Target function: empirical risk = number of errors

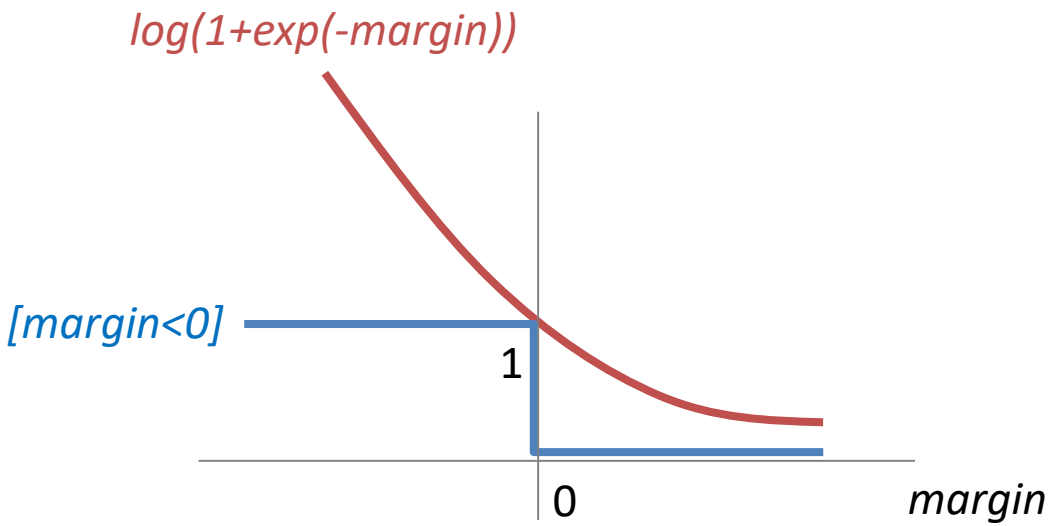
$$\#errors = \sum_{i=1}^{n} [mrgin(x_i, y_i) < 0] = \sum_{i=1}^{n} \left[y_i \cdot (a, x_i) < 0 \right] \to \min$$



Logistic regression

Target function: upper bound

$$\sum_{i=1}^{n}\left[y_i \cdot (a, x_i) < 0\right] < \sum_{i=1}^{n}\log\left(1+\exp(-y_i \cdot (a, x_i))\right) \to \min$$



Logistic regression

Relationship with P(y|x)

$$\sum_{i=1}^{n}\left[y_i \cdot \langle a, x_i \rangle < 0\right] < \sum_{i=1}^{n} \log\left(1+\exp(-y_i \cdot \langle a, x_i \rangle)\right) \to \min$$

$$\sum_{i=1}^{n} \log\left(\frac{1}{1+\exp(-y_i \cdot \langle a, x_i \rangle)}\right) \to \max$$

$$\sum_{i=1}^{n} \log\left(sigmoid\left(y_i \cdot \langle a, x_i \rangle\right)\right) \to \max$$

$$\sum_{i=1}^{n} \log\left(p\left(y_i | x_i\right)\right) \to \max$$

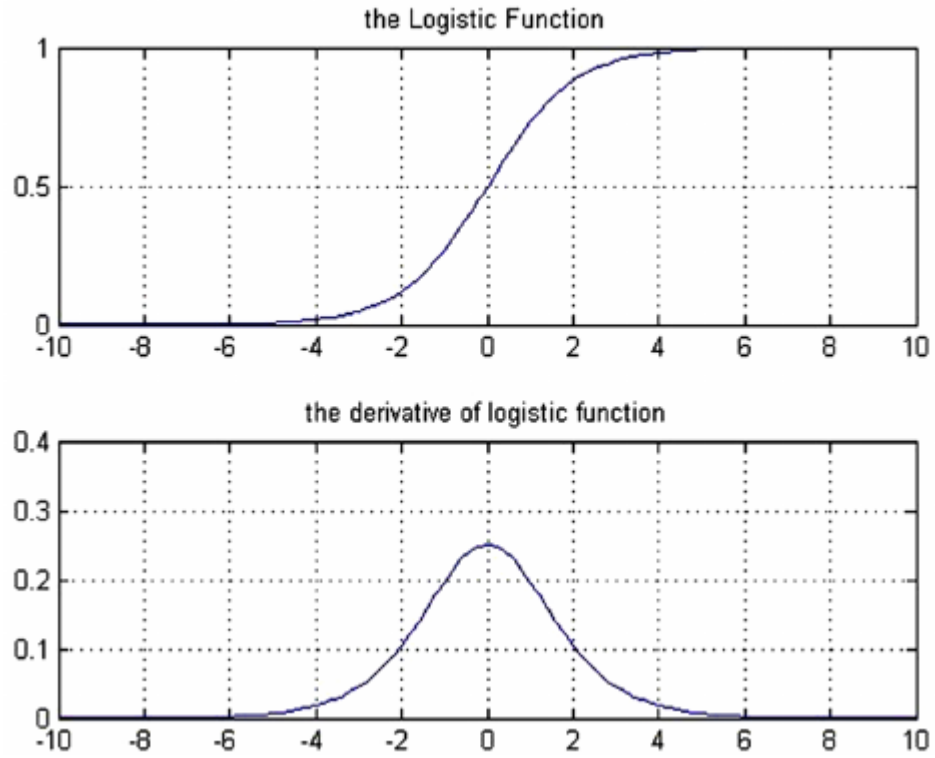
model

$$p\left(y_i | x_i\right) = sigmoid\left(y_i \cdot \langle a, x_i \rangle\right)$$

Logistic regression

Logistic function: the sigmoid

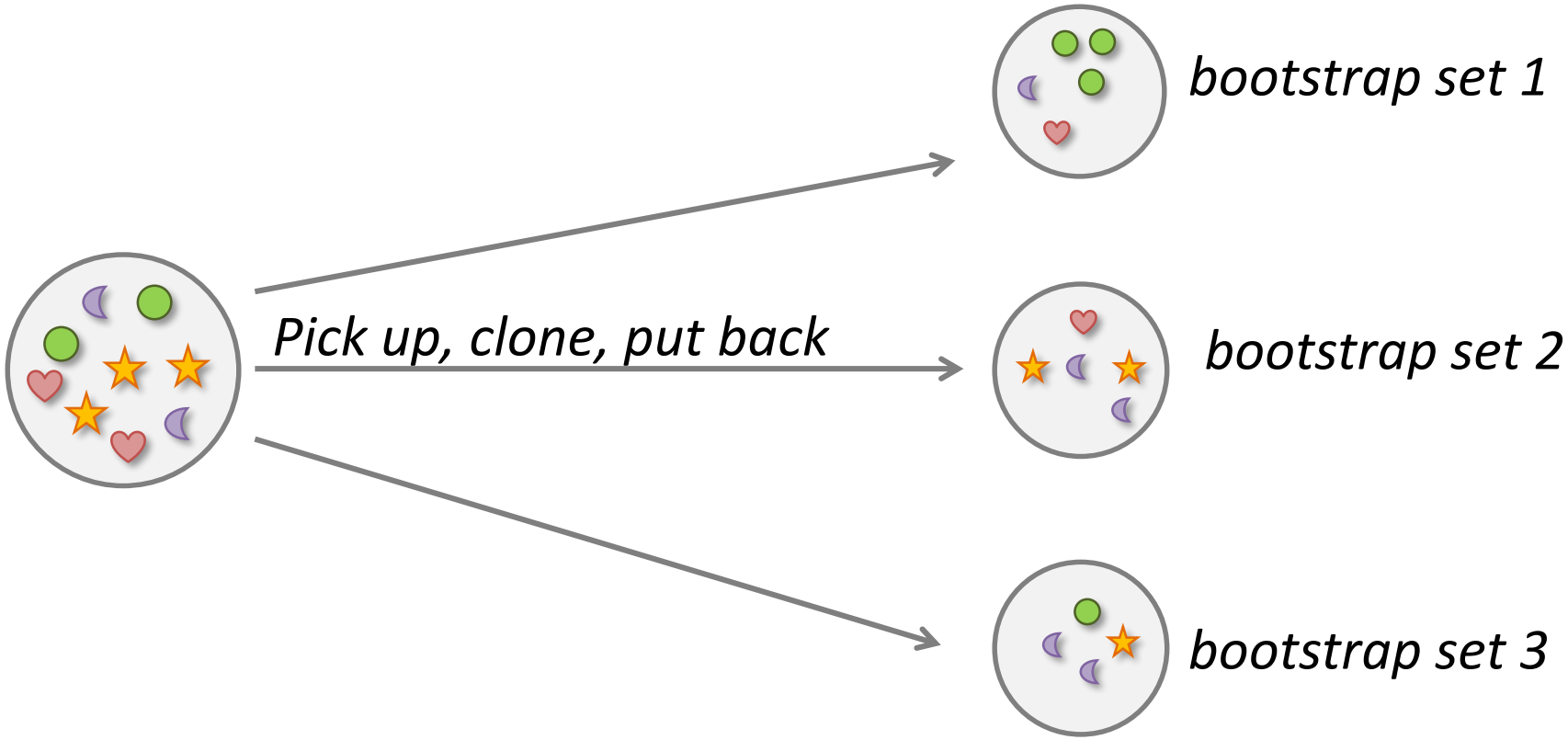
$$sigmoid(x) \qquad \frac{L}{1 + e^{-k(x-x_0)}}$$



Ensemble Learning

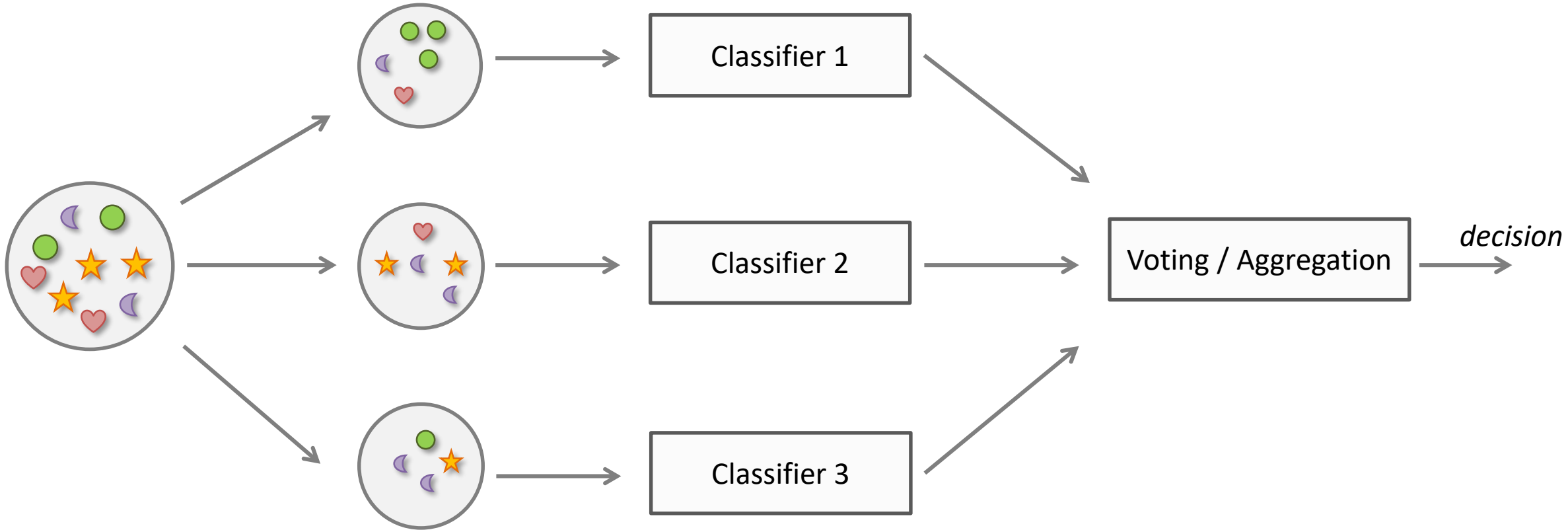
Ensemble Learning

Bootstrapping: random sampling with replacement



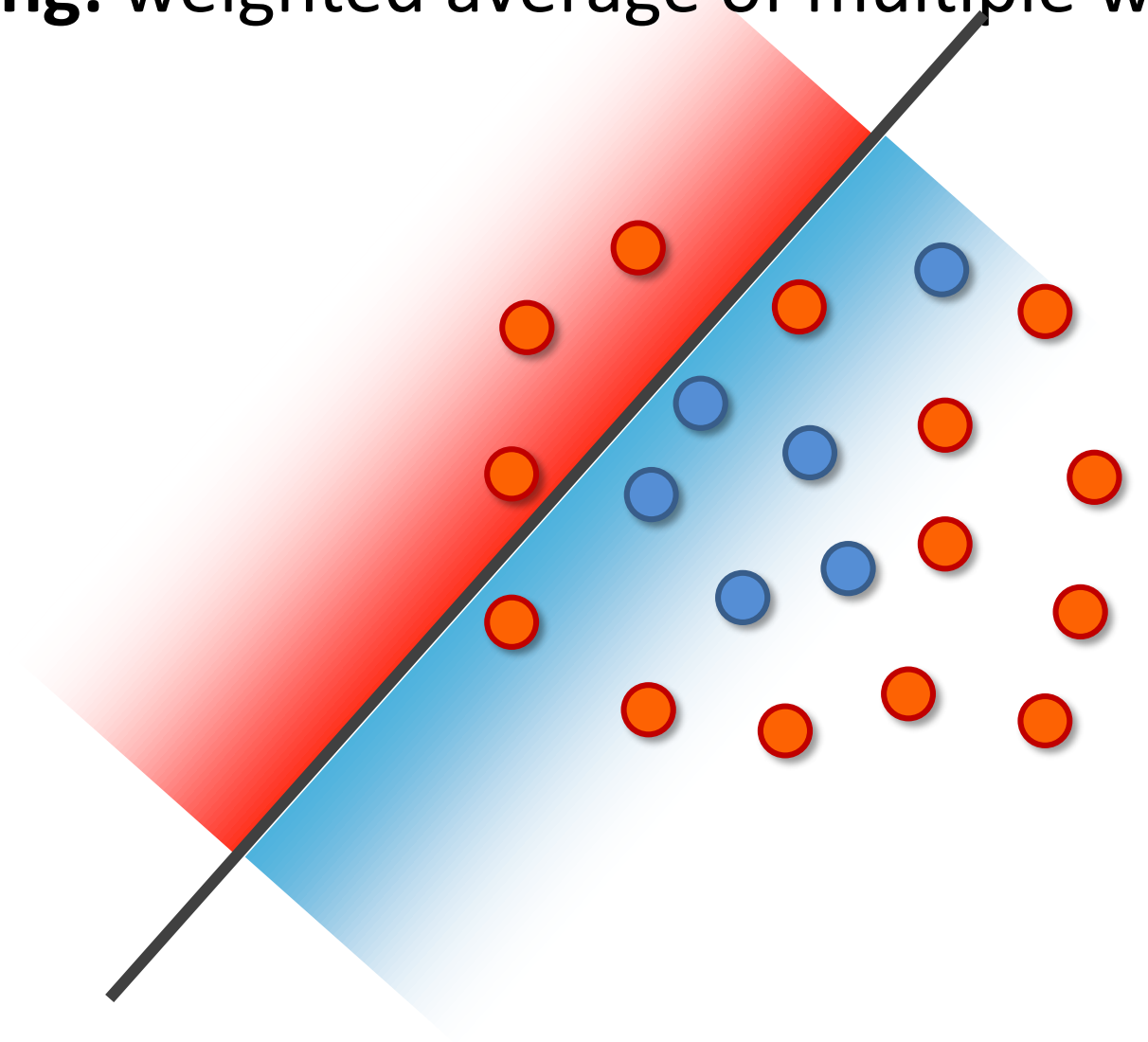
Ensemble Learning

Bagging: Bootstrap **Agg**regation



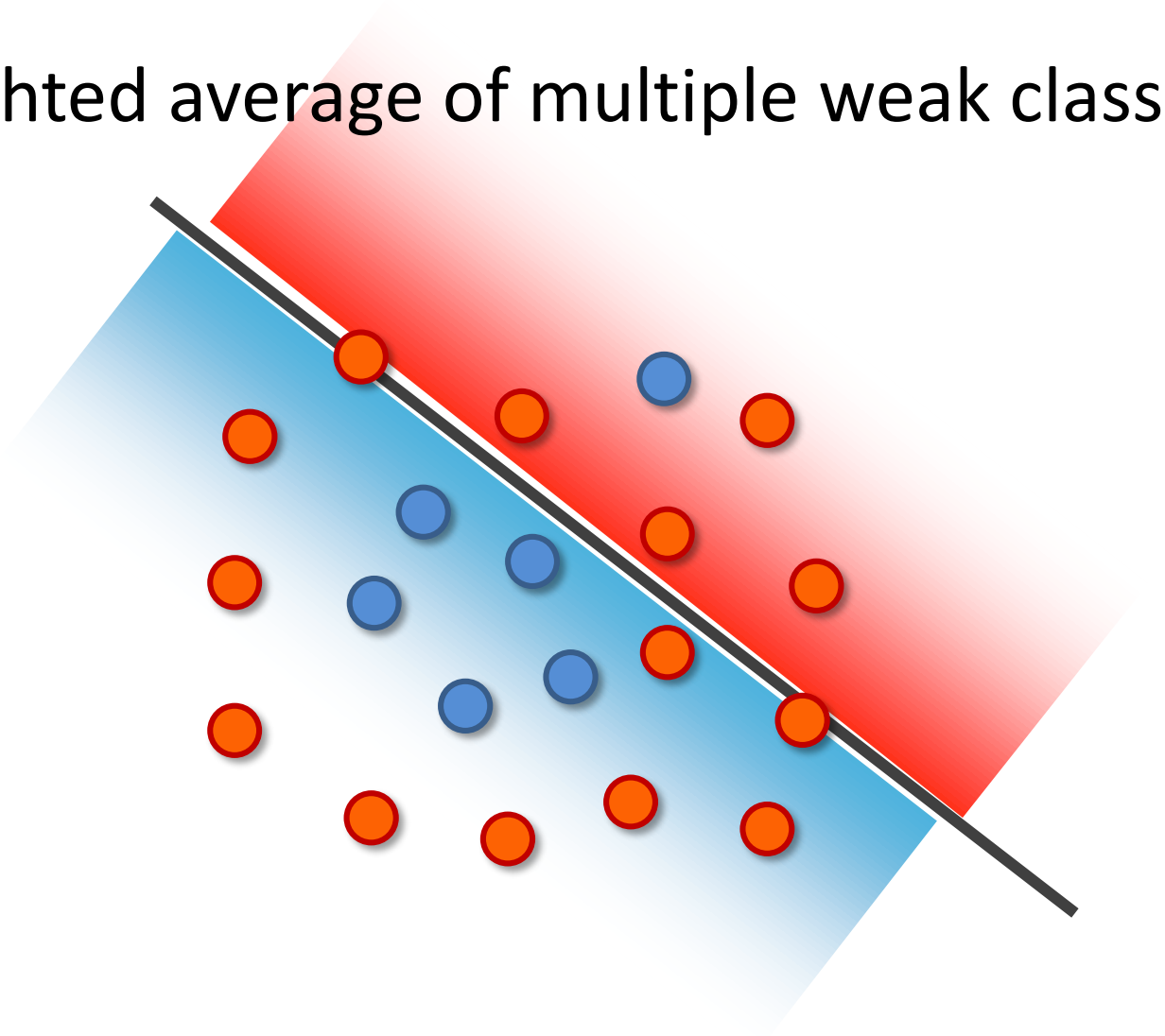
Ensemble Learning

Boosting: weighted average of multiple weak classifiers



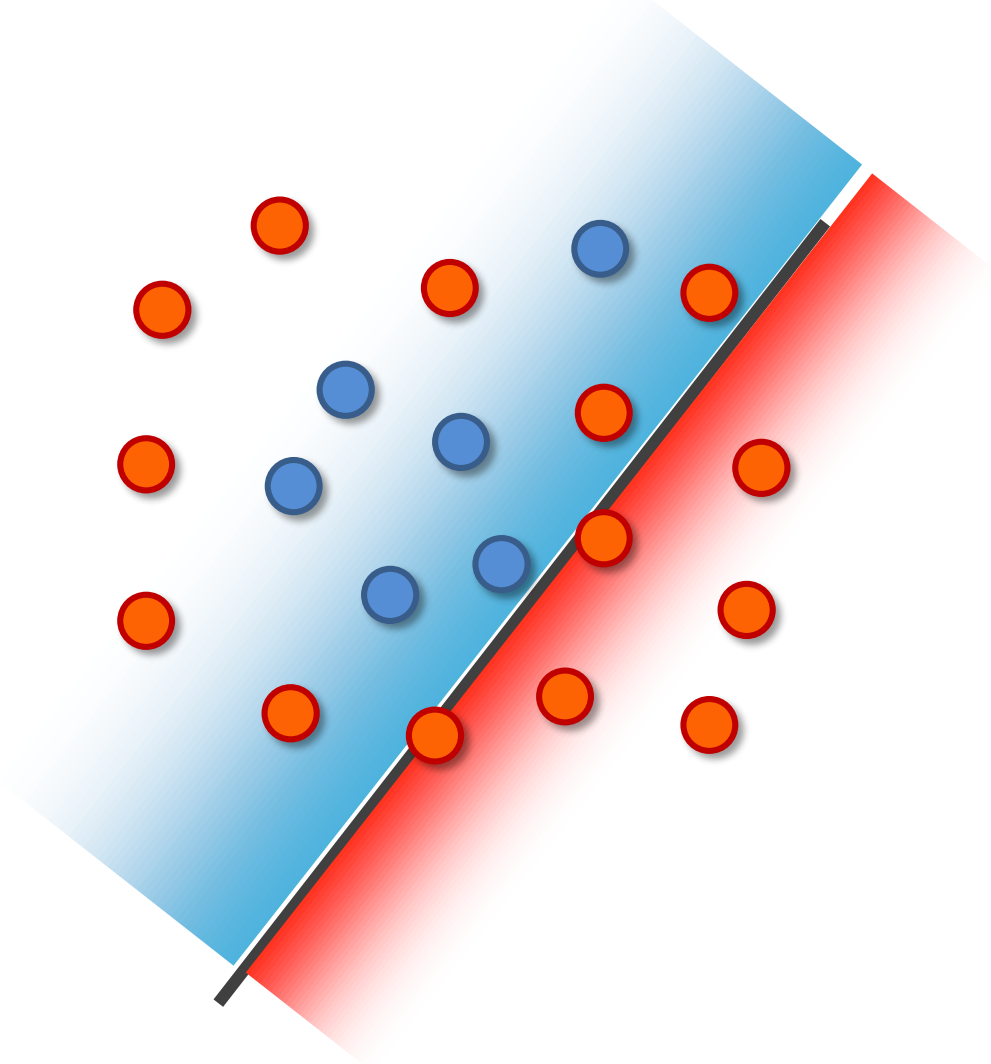
Ensemble Learning

Boosting: weighted average of multiple weak classifiers



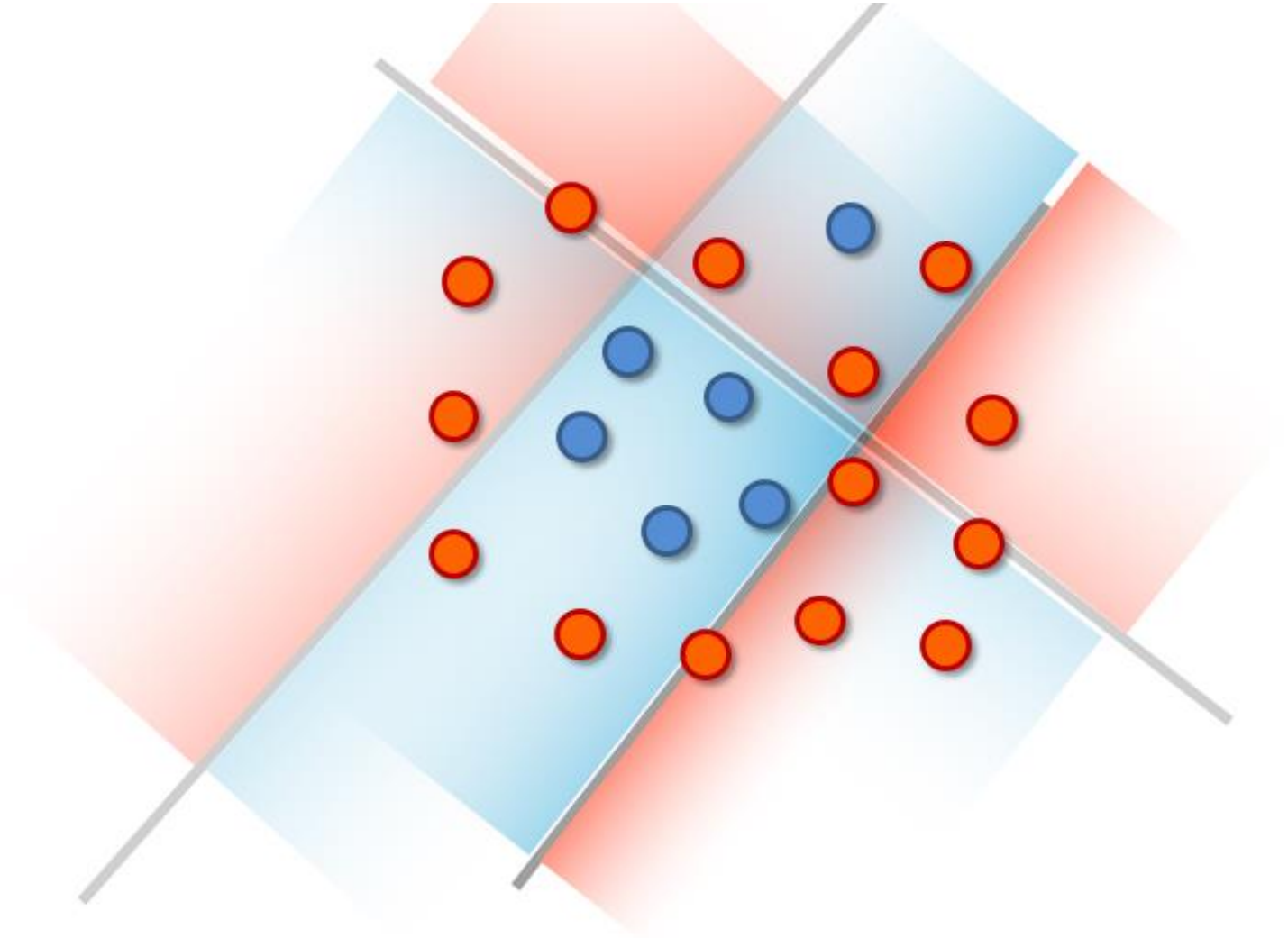
Ensemble Learning

Boosting: weighted average of multiple weak classifiers



Ensemble Learning

Boosting: weighted average of multiple weak classifiers



Ensemble Learning

Bagging

- Aims to decrease variance
- Aims to solve over-fitting problem

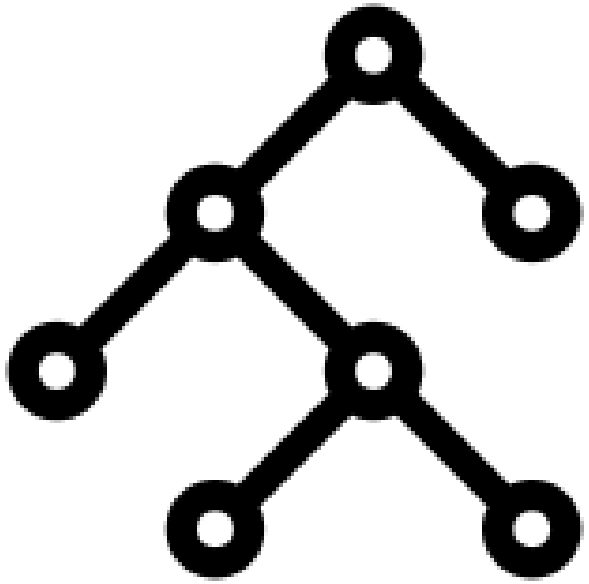
In bagging technique, a data set is divided into n samples using randomized sampling. Then, using a single learning algorithm a model is build on all samples. Later, the resultant predictions are combined using voting or averaging.
Bagging is done in parallel.

Boosting

- Aims to decrease bias

In boosting, after the first round of predictions, the algorithm weighs misclassified predictions higher, such that they can be corrected in the succeeding round. This sequential process of giving higher weights to misclassified predictions continue until a stopping criterion is reached.

Decision Tree



Decision tree

F1	F2	F3	Target
A	A	A	0
A	A	B	0
B	B	A	1
A	B	B	1

H(0,0,1,1) = ½ log(½) + ½ log(½) = 1

Decision tree

F1	F2	F3	Target	Attempt split on F1
A	A	A	0	0
A	A	B	0	0
B	B	A	1	1
A	B	B	1	0

Information Gain = H(0011) - 3/4 H(001) - 1/4 H(1) = 0.91

Decision tree

F1	F2	F3	Target	Attempt split on F2
A	A	A	0	0
A	A	B	0	0
B	B	A	1	1
A	B	B	1	1

Information Gain = H(0011) - 1/2 H(00) - 1/2 H(11) = 1

Decision tree

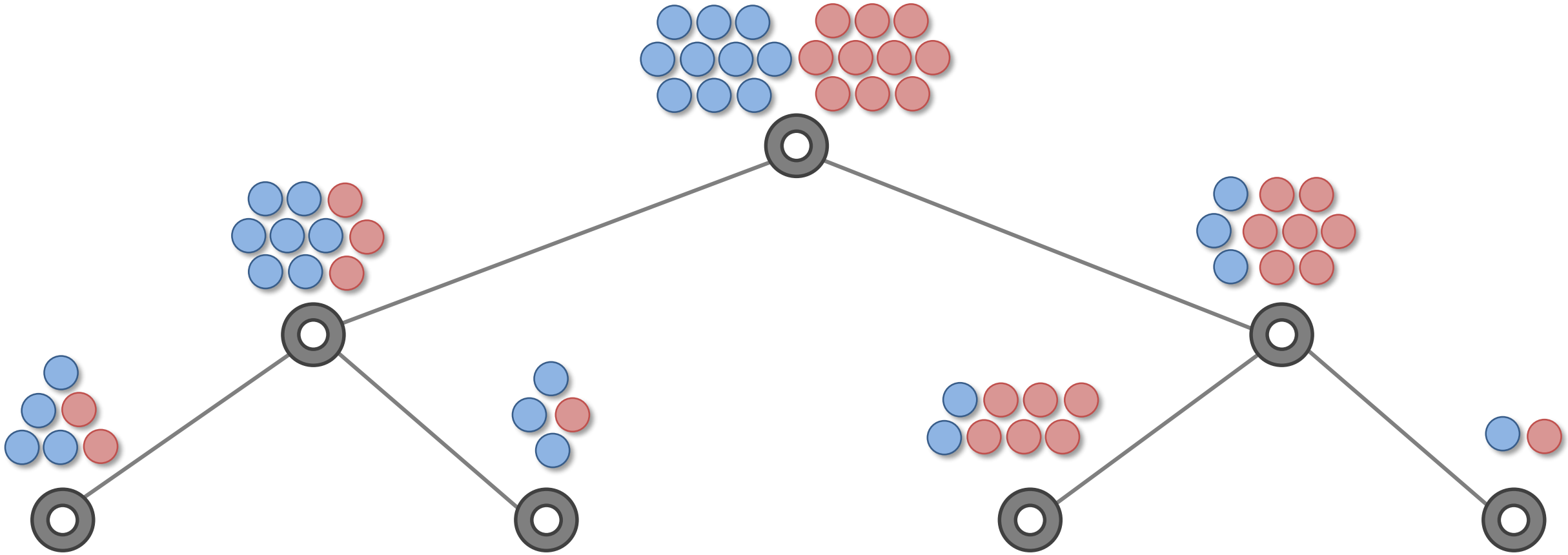
F1	F2	F3	Target	Attempt split on F3
A	A	A	0	0
A	A	B	0	0
B	B	A	1	1
A	B	B	1	1

Information Gain = H(0011) - 1/2 H(01) - 1/2 H(01) = 0

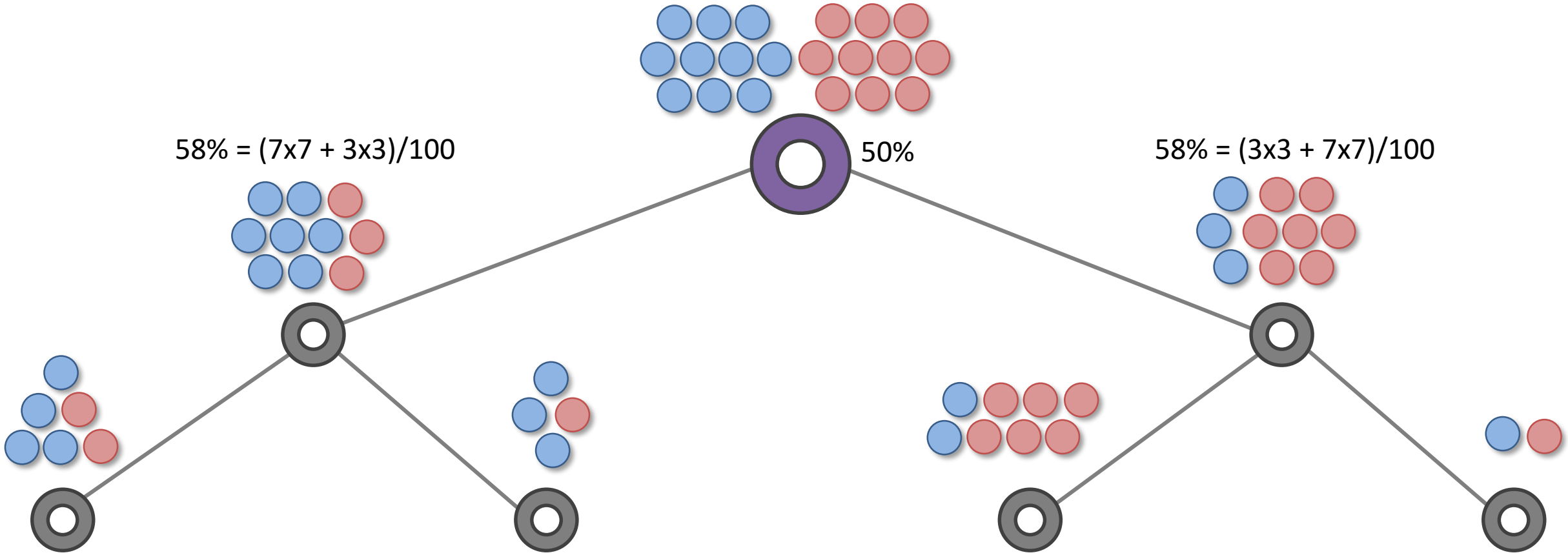
Decision tree

F1	F2	F3	Target
A	A	A	0
A	A	B	0
B	B	A	1
A	B	B	1

Decision tree: pruning



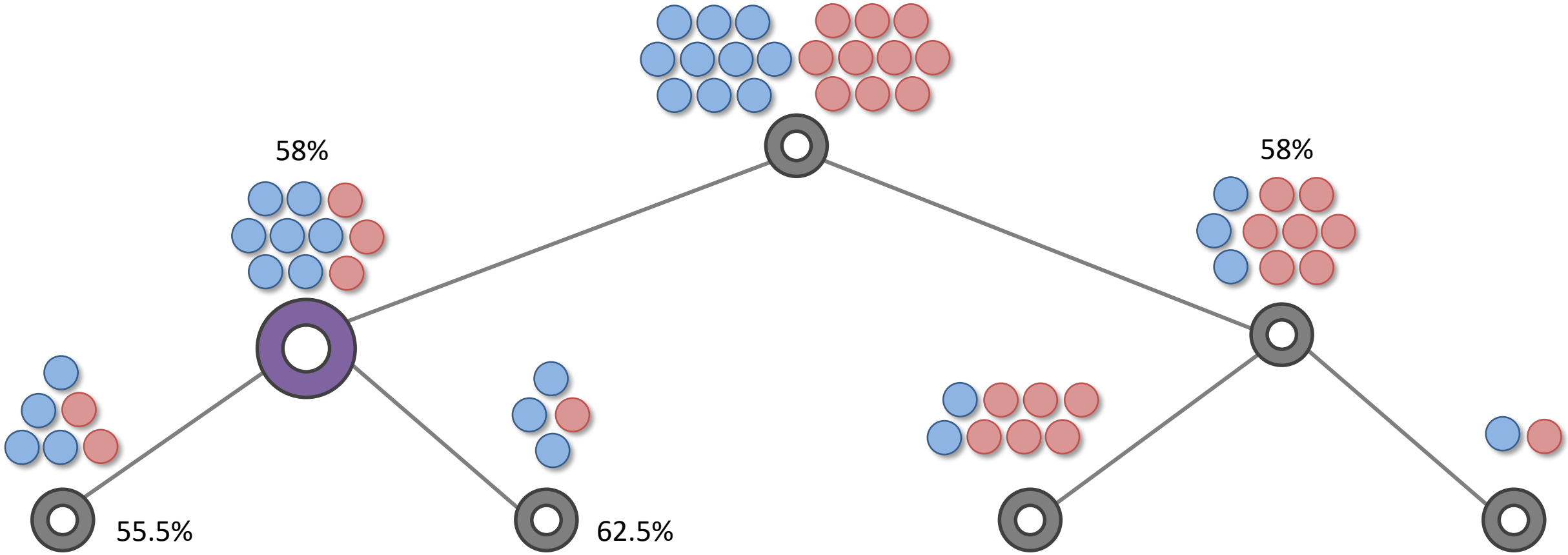
Decision tree: pruning



50% x 58% + 50% x 58% = 58%

58% > 50% OK

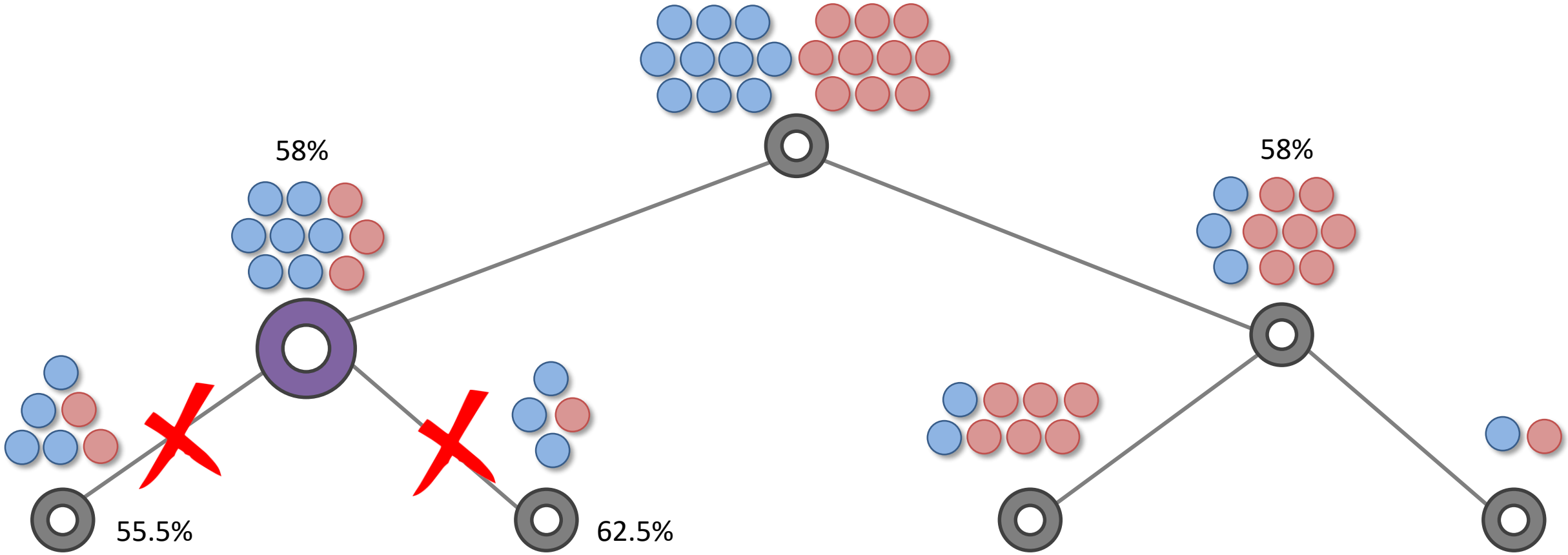
Decision tree: pruning



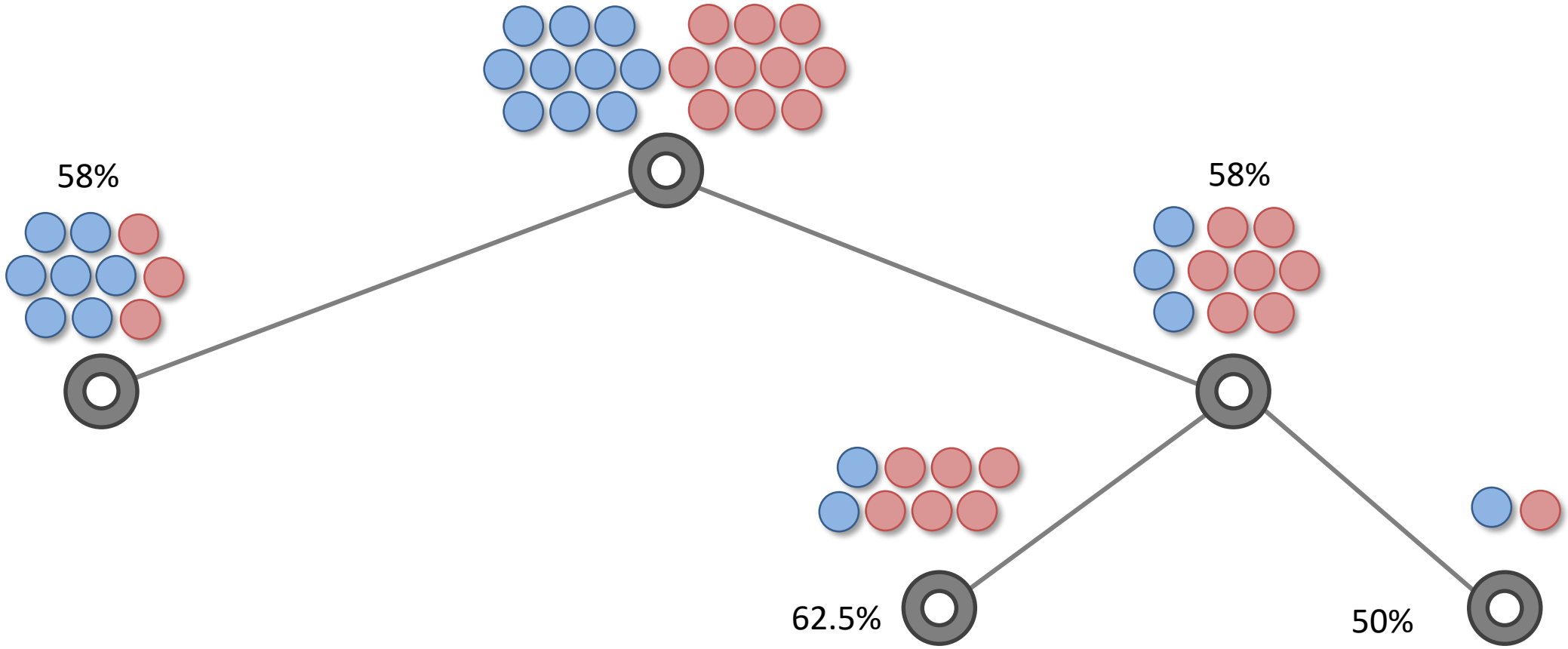
50% x 58% + 50% (60% x 55.5% + 40% x 62.5%) = 50% x 58% + 50% x 58.3 = 58.16%

58.16% > 58% *this is small improvement*

Decision tree: pruning



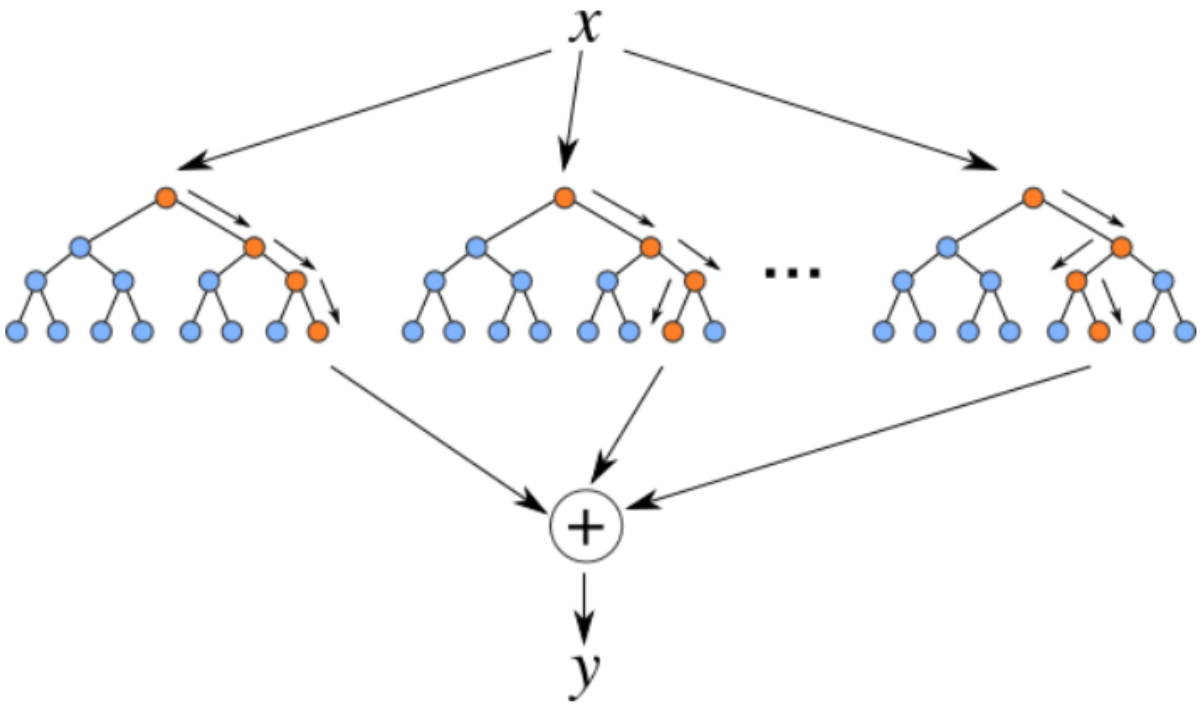
Decision tree: pruning



50% x 58% + 50% (80% x 62.5% + 20% x 50%) = 50% x 58% + 50% x 60% = 59%

59% > 58% *OK*

Random Forest

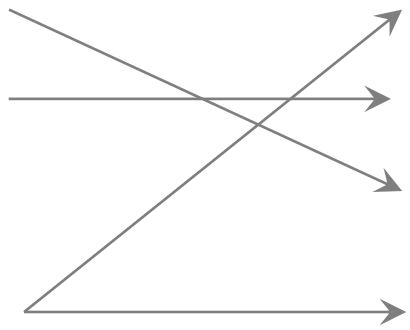


https://www.youtube.com/watch?v=J4Wdy0Wc_xQ

Random forest

Original dataset

F1	F2	F3	Target
A	A	A	0
A	A	B	0
B	B	A	1
A	B	B	1



Bootstrapped dataset

F1	F2	F3	Target
A	B	B	1
A	A	B	0
A	A	A	0
A	B	B	1

Random forest

Bootstrapped dataset

F1	F2	F3	Target
A	B	B	1
A	A	B	0
A	A	A	0
A	B	B	1

candidate candidate

Determine the best root node out of randomly selected sub-set of candidates

Random forest

Bootstrapped dataset

F1	F2	F3	Target
A	B	B	1
A	A	B	0
A	A	A	0
A	B	B	1

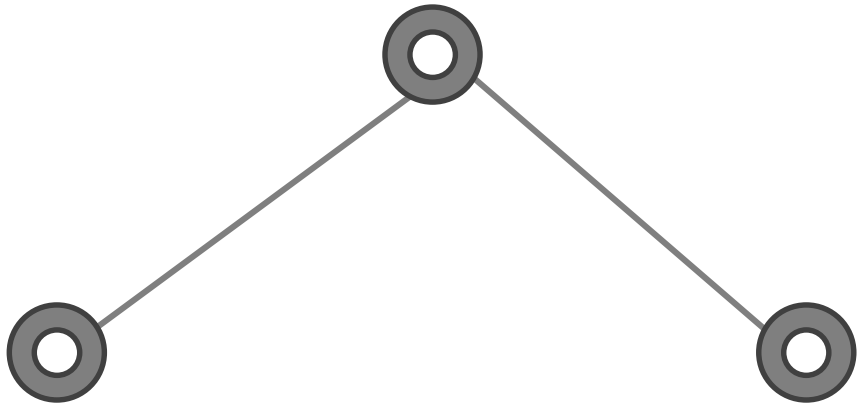
root

Random forest

Bootstrapped dataset

F1	F2	F3	Target
A	B	B	1
A	A	B	0
A	A	A	0
A	B	B	1

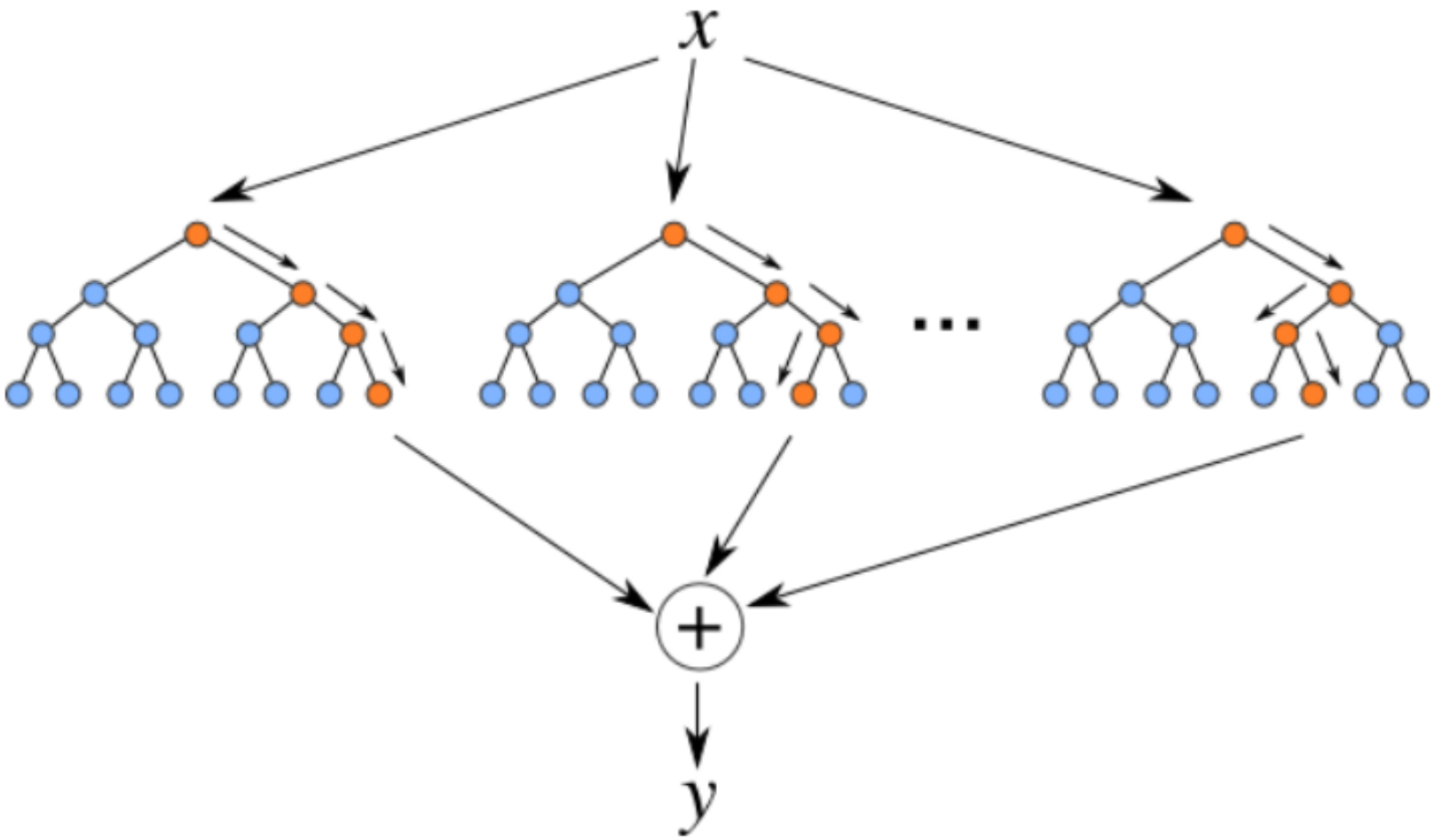
candidate candidate



Build tree as usual, but considering a random subset of features at each step

Random forest

Bagging: do aggregate decisions against the bootstrapped data



Random forest

Validation: run aggregated decisions against out-of-bag dataset

Original dataset

F1	F2	F3	Target
A	A	A	0
A	A	B	0
B	B	A	1
A	B	B	1

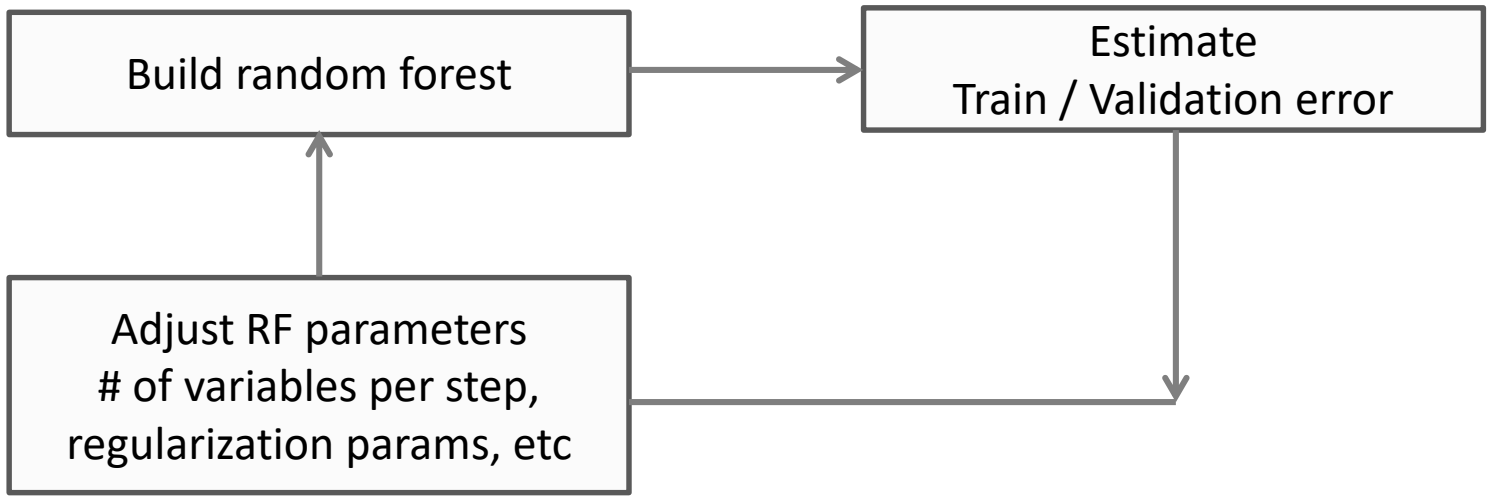
out-of-bag

Bootstrapped dataset

F1	F2	F3	Target
A	B	B	1
A	A	B	0
A	A	A	0
A	B	B	1

Random forest

The process



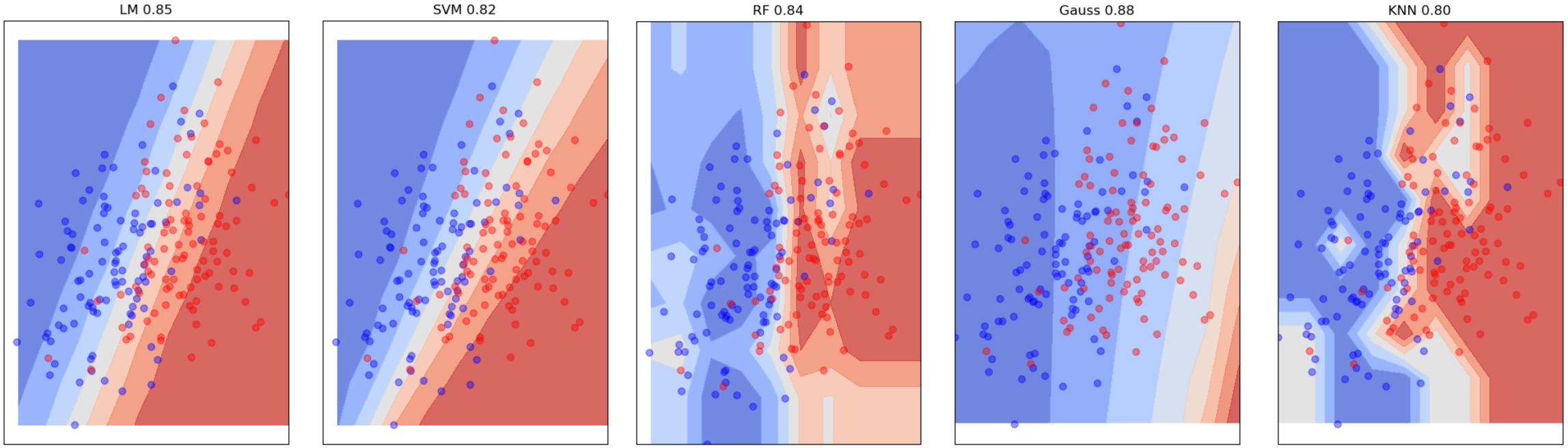
RF vs GBM

RF uses bagging technique to make predictions.
GBM uses boosting techniques to make predictions.

Random forest improves model accuracy by reducing variance (mainly). The trees grown are uncorrelated to maximize the decrease in variance.

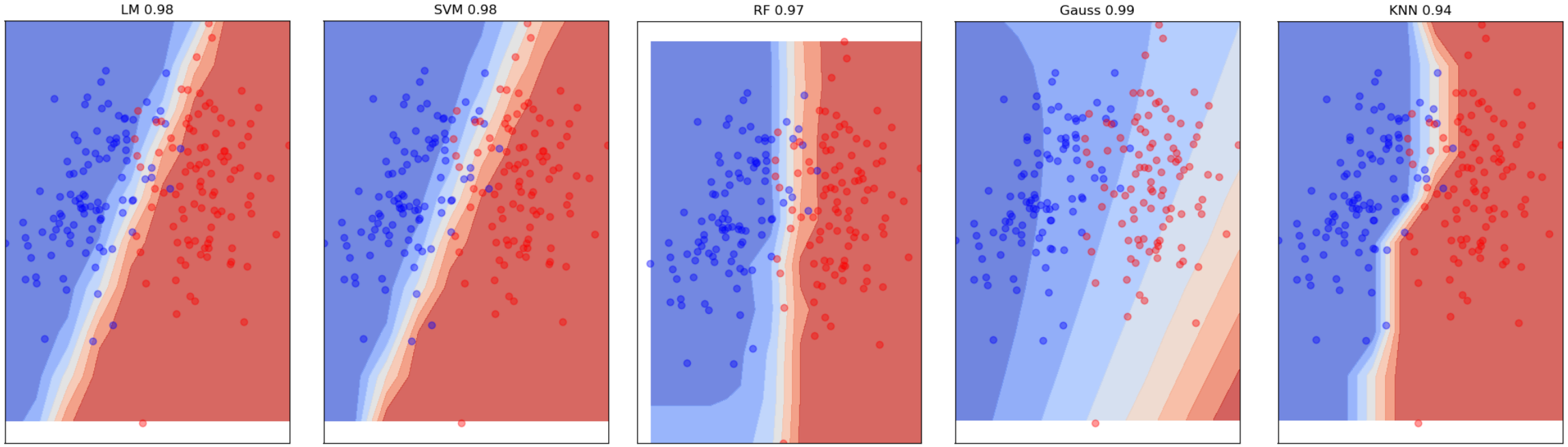
On the other hand, GBM improves accuracy by reducing both bias and variance in a model.

Classification examples: 2D data, 2 classes



https://github.com/dryabokon/ML/blob/master/ex_02_01_train_test_2_classes.py

Classification examples: 2D data, 2 classes



https://github.com/dryabokon/ML/blob/master/ex_02_01_train_test_2_classes.py

Adaboost


Robert Schapire


Yoav Freund

Adaboost

Definitions

$x_1, x_2, x_i, \dots, x_N$ Training features

$k_1, k_2, k_i, \dots, k_N$ = {-1 ,1} Training targets

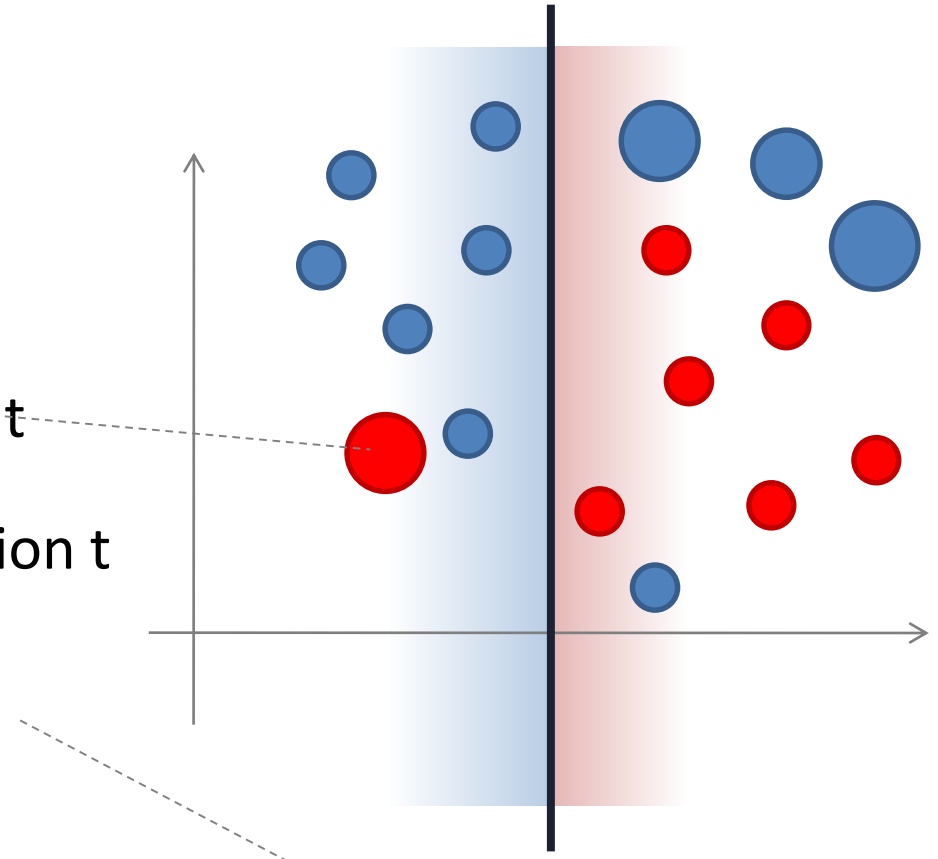
$t = 1,2,3\dots$ Iteration number

$W_t(x_i)$ Weight of the element on iteration t

α_t Weight if a weak classifier on iteration t

$h_t(x)$ Weak classifier on iteration t

$$k(x) \cong \sum_t \alpha_t \cdot h_t(x)$$



$$h(x) = \{x > \theta \,?\, 0 : 1\}$$

Adaboost

Algorithm

Weak classifier to minimize the weighted error

$$h_t(x) = \arg\min_h \sum_i W_t(i) \cdot [k_i \neq h(x_i)]$$

Update weights

$$W_{t+1}(i) = W_t(i) \cdot \exp\{-\alpha_t \cdot k_i \cdot h_t(x_i)\} = W_t(i) \times \begin{cases} \sqrt{\frac{1-r_t}{1+r_t}} & if\ h_t(x) = k \\ \sqrt{\frac{1+r_t}{1-r_t}} & if\ h_t(x) \neq k \end{cases}$$

Calculate the weight od classifier

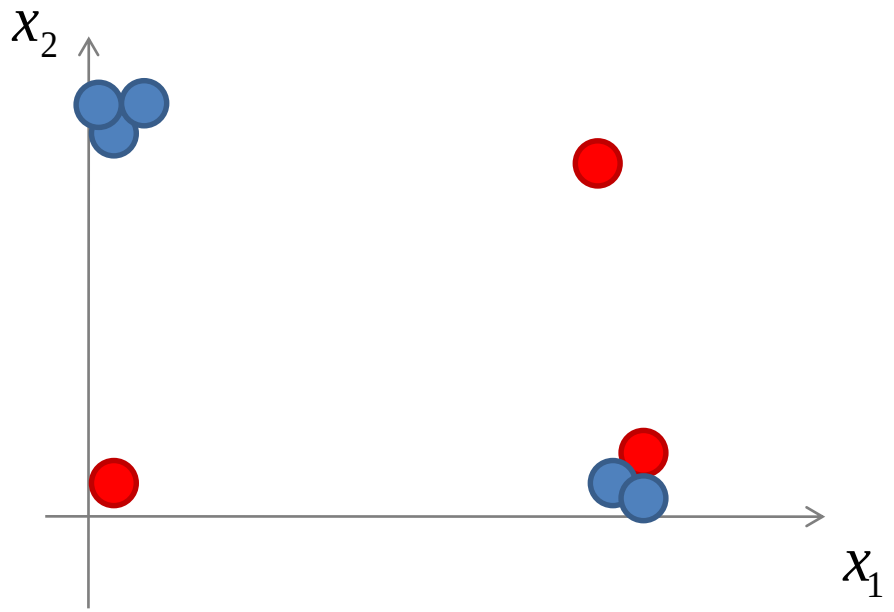
$$\alpha_t = \frac{1}{2}\log\left(\frac{1+r_t}{1-r_t}\right)$$

$$r_t = \sum_i W_t(i) \cdot h(x_i) \cdot k_i \Big/ \sum_i W_t(i)$$

Adaboost

Example: step 1

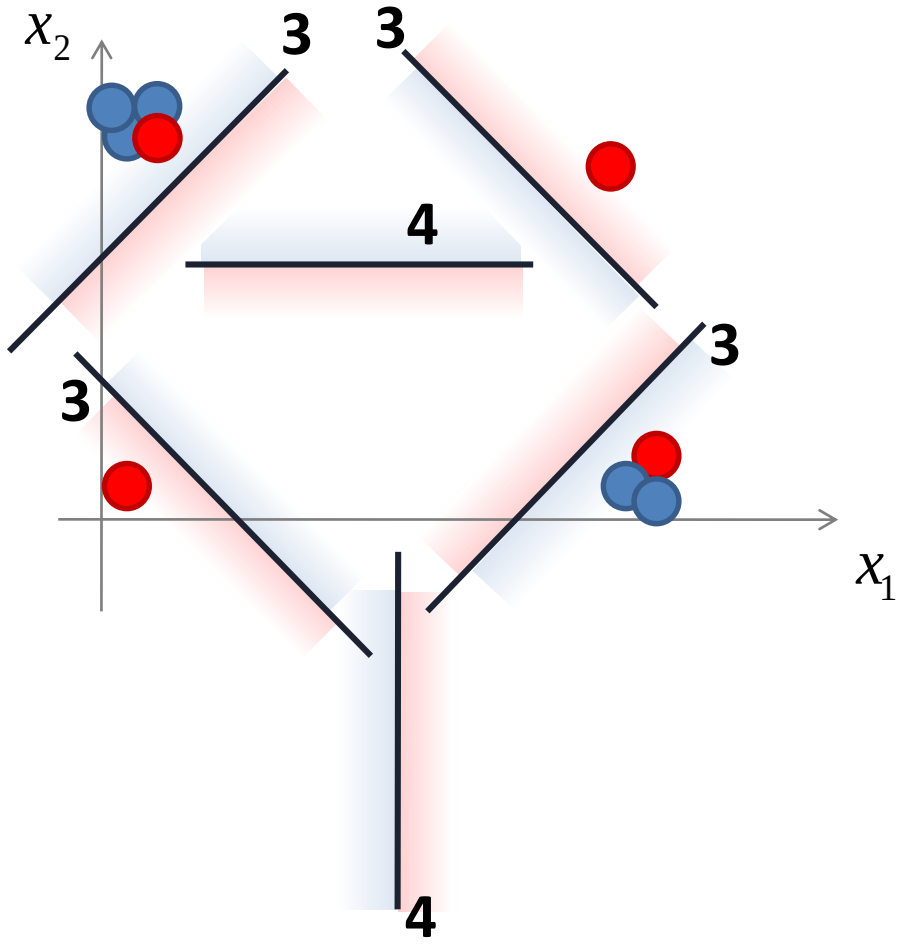
$$h_{t=1}(\bar{x}) =$$



Adaboost

Example: step 1

$$h_{t=1}(\bar{x}) =$$

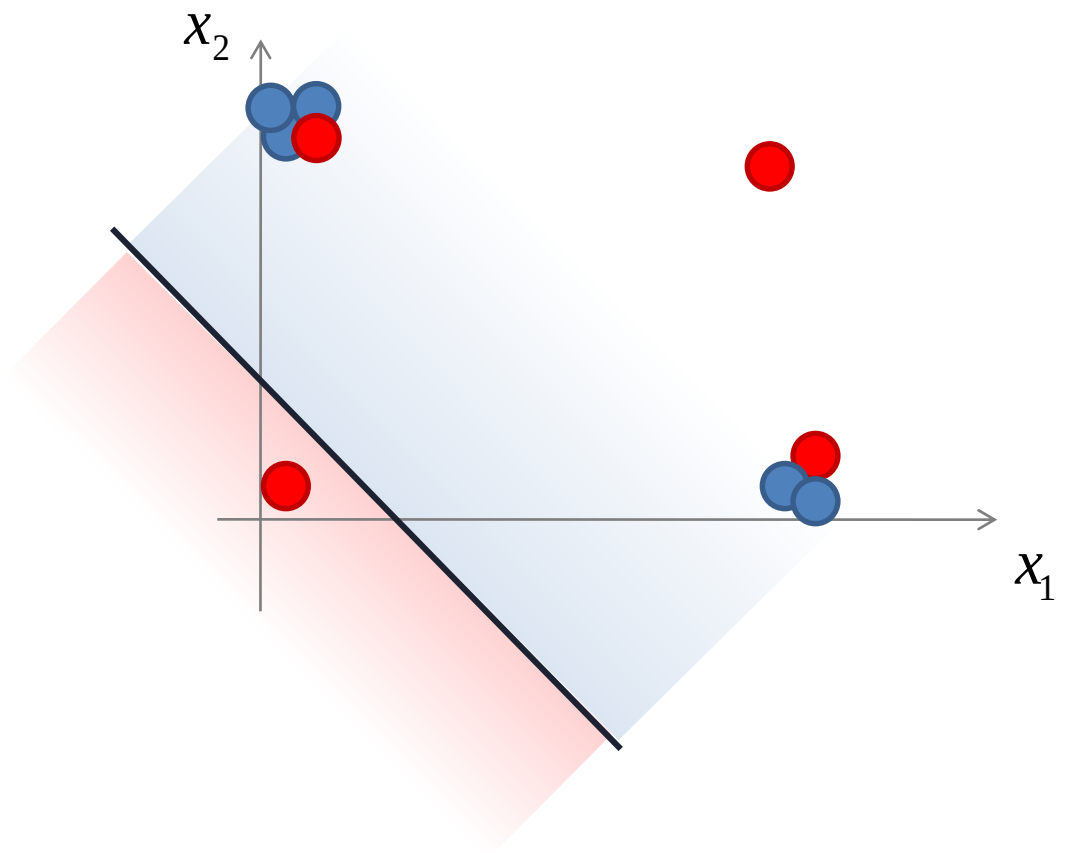


Adaboost

Example: step 1

$$h_{t=1}(\bar{x}) = \{x_1 + x_2 < 0.5? \text{ ● } : \text{ ● }\}$$

$$r_{t=1} = 0.33 \quad \alpha_{t=1} = 0.34$$



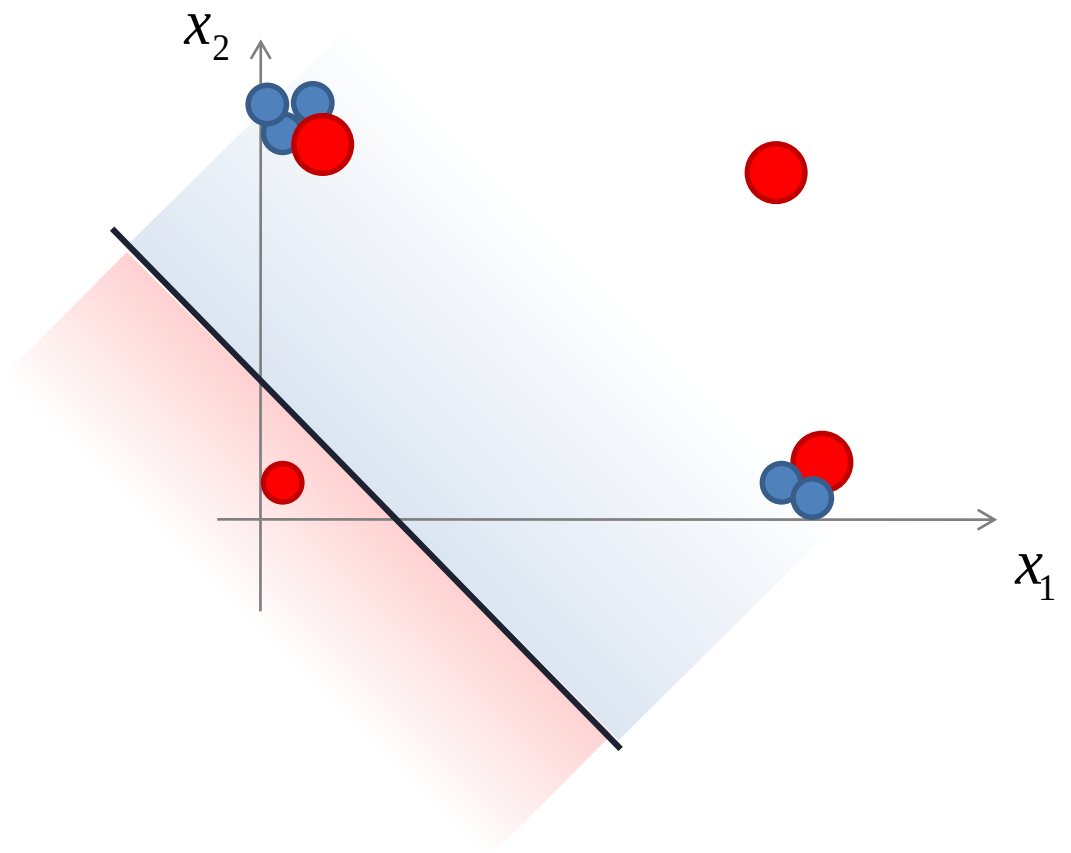
Adaboost

Example: step 1

$$h_{t=1}(\bar{x}) = \{x_1 + x_2 < 0.5? \text{ ● } : \text{ ●} \}$$

$$r_{t=1} = 0.33 \quad \alpha_{t=1} = 0.34$$

$$W_{t=2}(i) = W_1(i) \cdot \exp\{-\alpha_1 \cdot k_i \cdot h_1(x_i)\}$$



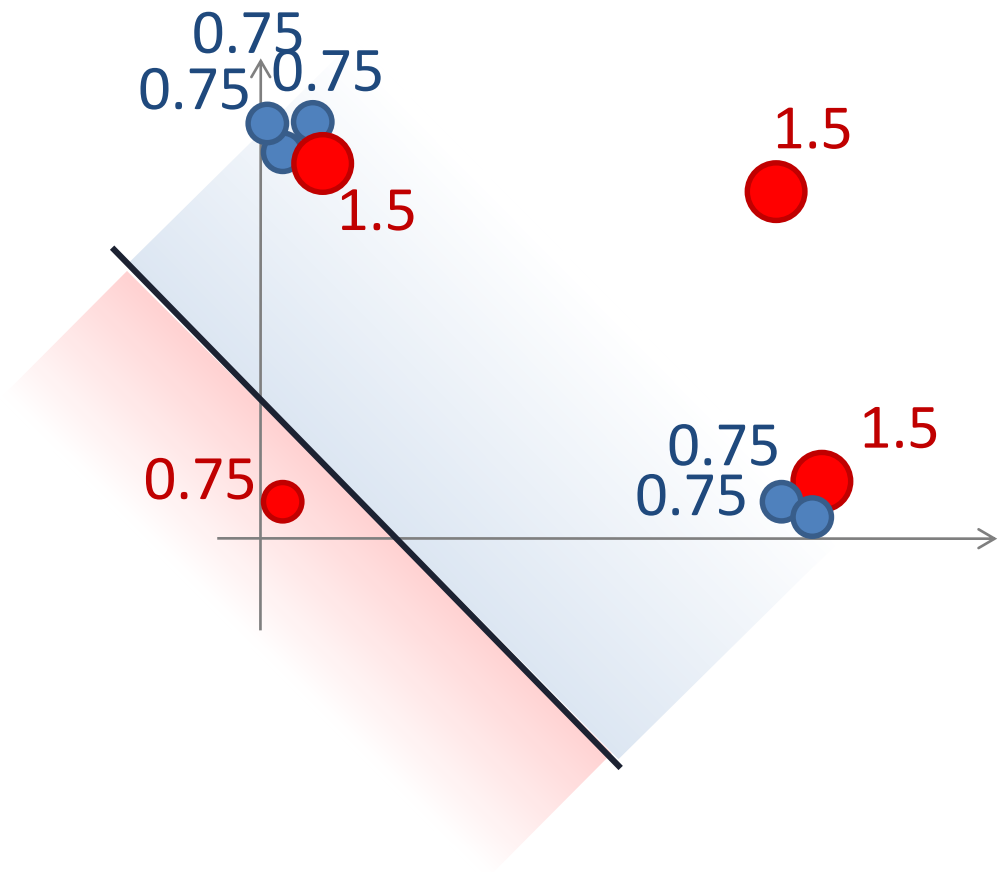
Adaboost

Example: step 1

$$h_{t=1}(\bar{x}) = \{x_1 + x_2 < 0.5? \text{ ●} : \text{ ●}\}$$

$$r_{t=1} = 0.33 \quad \alpha_{t=1} = 0.34$$

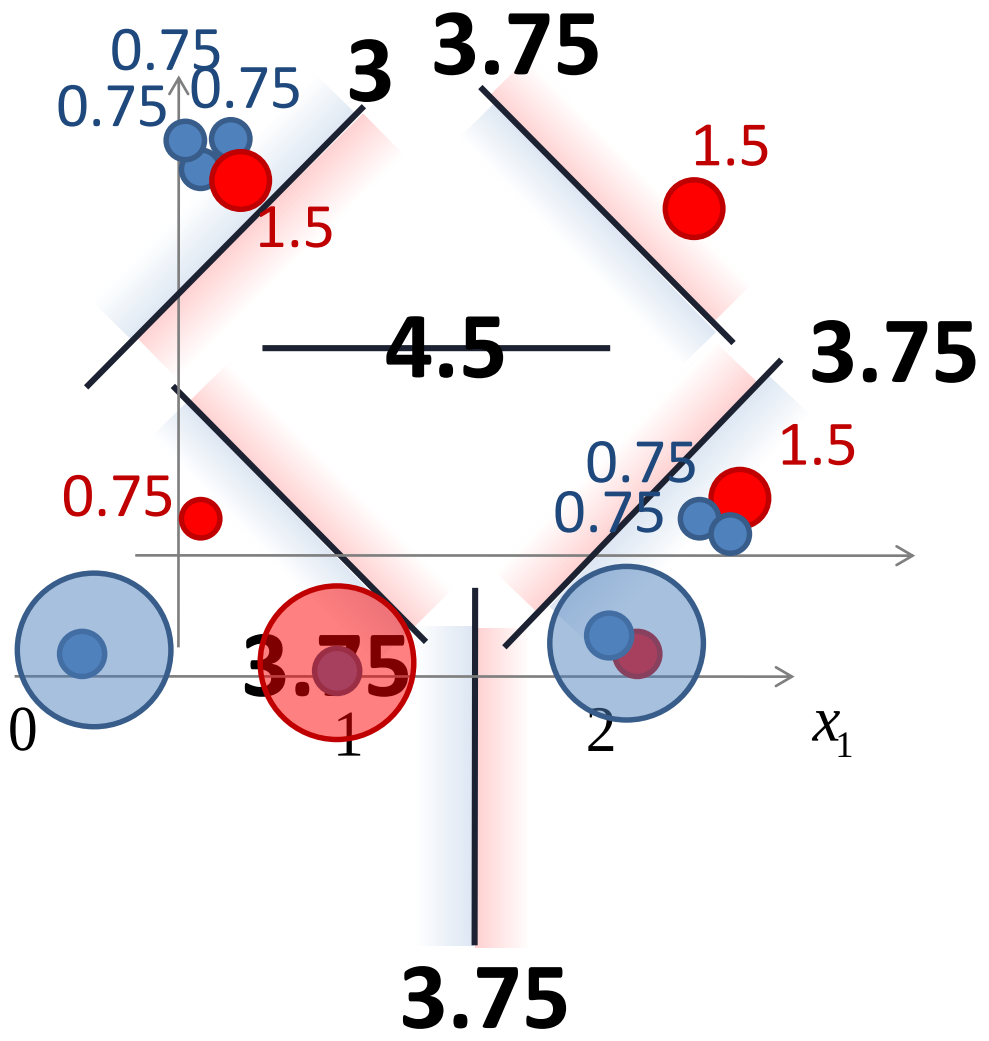
$$W_{t=2}(i) = W_1(i) \cdot \exp\{-\alpha_1 \cdot k_i \cdot h_1(x_i)\}$$



Adaboost

Example: step 2

$$h_{t=2}(\bar{x}) =$$

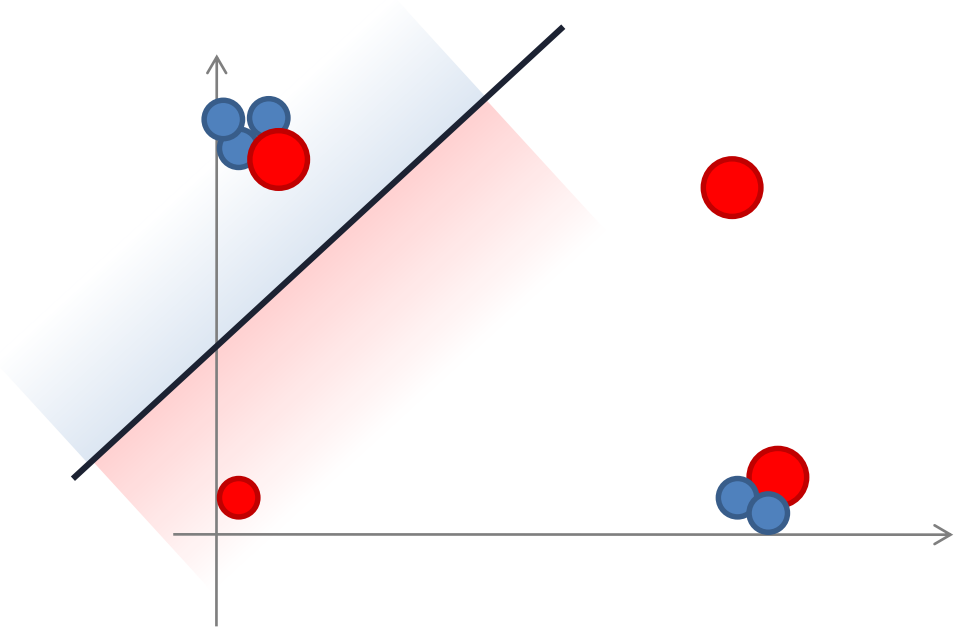


Adaboost

Example: step 2

$$h_{t=2}(\bar{x}) = \{x_1 - x_2 < -0.5? \text{ ● } : \text{ ● }\}$$

$$r_{t=2} = 0.33 \quad \alpha_{t=2} = 0.34$$



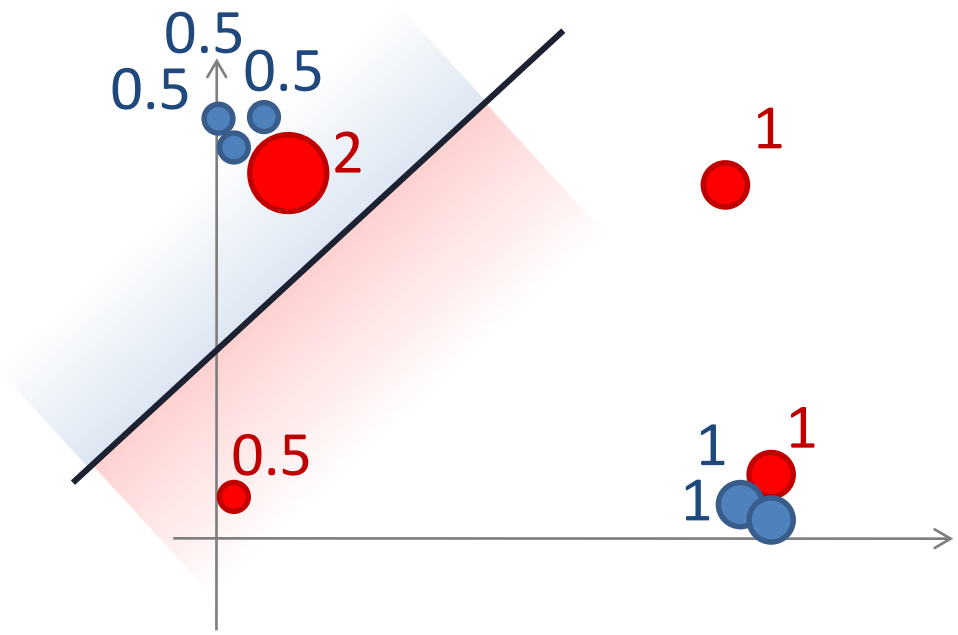
Adaboost

Example: step 2

$$h_{t=2}(\bar{x}) = \{x_1 - x_2 < -0.5? \text{ ● } : \text{ ●} \}$$

$$r_{t=2} = 0.33 \quad \alpha_{t=2} = 0.34$$

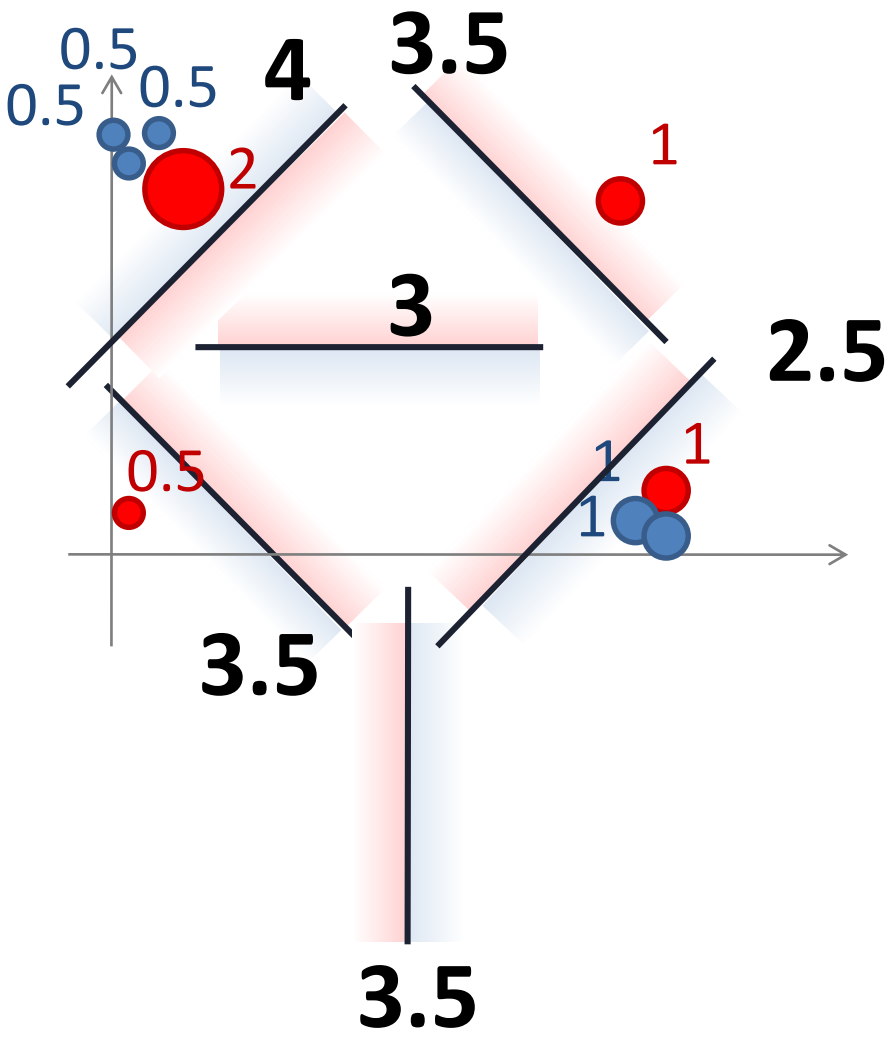
$$W_{t=2}(i) = W_1(i) \cdot \exp\{-\alpha_1 \cdot k_i \cdot h_1(x_i)\}$$



Adaboost

Example: step 3

$$h_{t=3}(\bar{x}) =$$

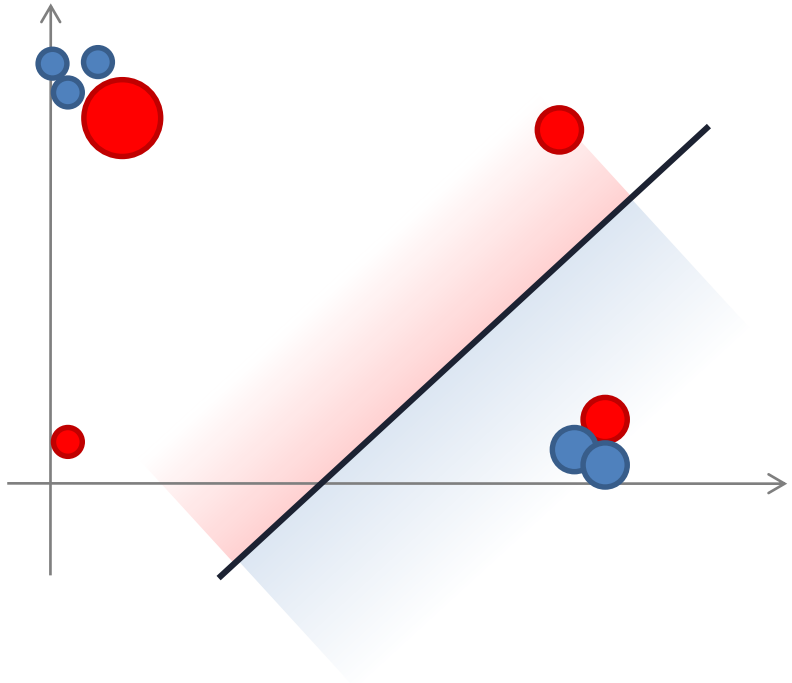


Adaboost

Example: step 3

$$h_{t=3}(\bar{x}) = \{x_1 - x_2 > 0.5? \ \bullet : \bullet\}$$

$$r_{t=3} = 0.38 \quad \alpha_{t=3} = 0.39$$

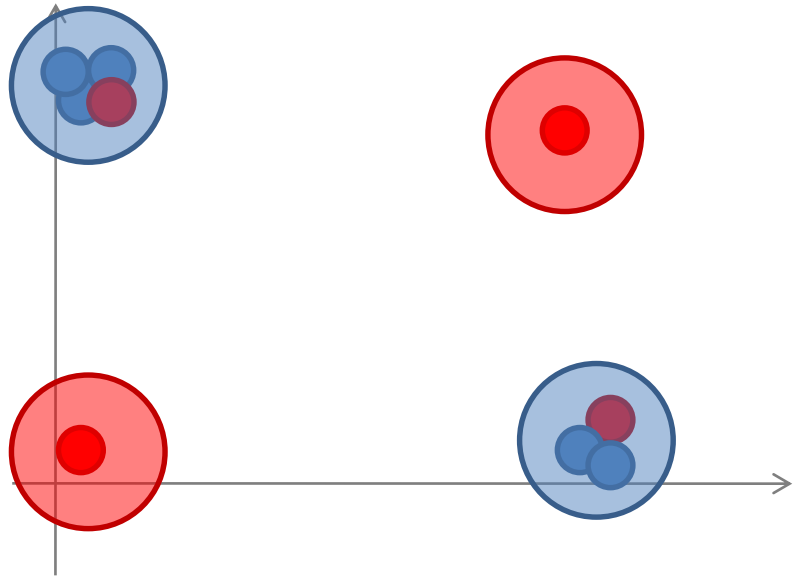


Adaboost

Example: result

$$k(x) \cong \sum_t \alpha_t \cdot h_t(x)$$

$h_{t=3}$ $h_{t=2}$

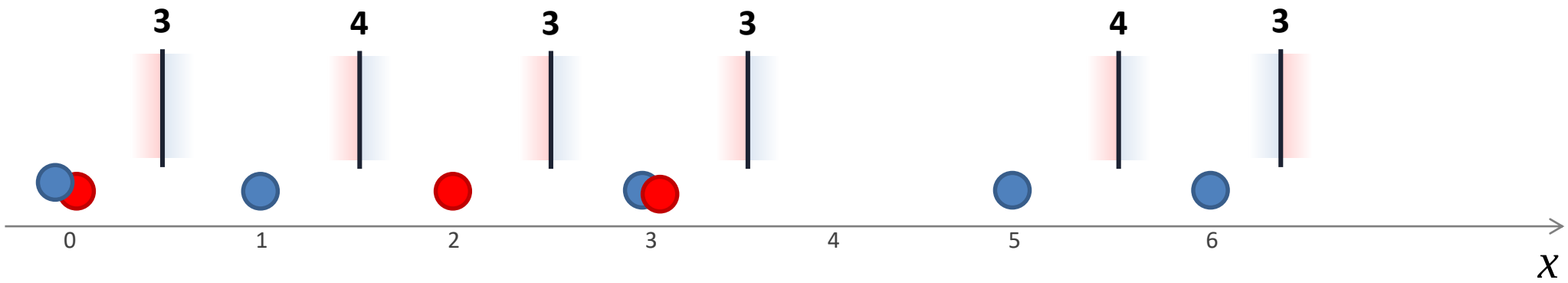


	x	(0,0)	(1,0)	(1,0)	(1,0)	(0,1)	(0,1)	(0,1)	(0,1)	(0,1)	(1,1)
$\alpha_1 = 0.35$	h_1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1
$\alpha_2 = 0.35$	h_2	1	1	1	1	-1	-1	-1	-1	-1	1
$\alpha_3 = 0.39$	h_3	1	-1	-1	-1	1	1	1	1	1	1
$k(x) \cong \sum_t \alpha_t \cdot h_t(x)$		1	-1	-1	-1	-1	-1	-1	-1	-1	1

Adaboost

Another example

Adaboost



Adaboost

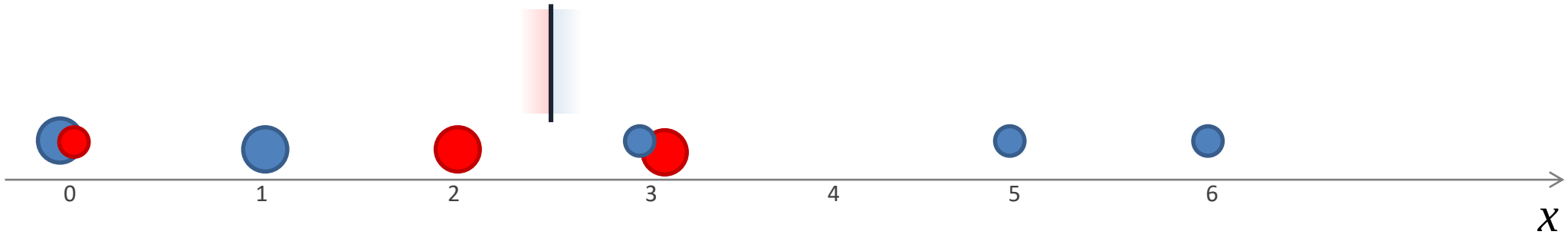
$$h_{t=1}(\bar{x}) = \{x < 2.5?\ \ +1\ \ :\ \ -1\}$$

$$r_{t=1} = \frac{1}{\sum_i W_1(i)} \sum_i W_1(i) \cdot h_1(x_i) \cdot k_i = 2/8$$

$$\alpha_t = \frac{1}{2} \log\left(\frac{1+r_t}{1-r_t}\right)$$

$$W_{t=2}(i) = W_1(i) \cdot \exp\{-\alpha_1 \cdot k_i \cdot h_1(x_i)\} = W_1(i) \times \begin{cases} \sqrt{\frac{1-r_t}{1+r_t}} & if\ h_1(x) = k \\ \sqrt{\frac{1+r_t}{1-r_t}} & if\ h_1(x) \neq k \end{cases}$$

Adaboost

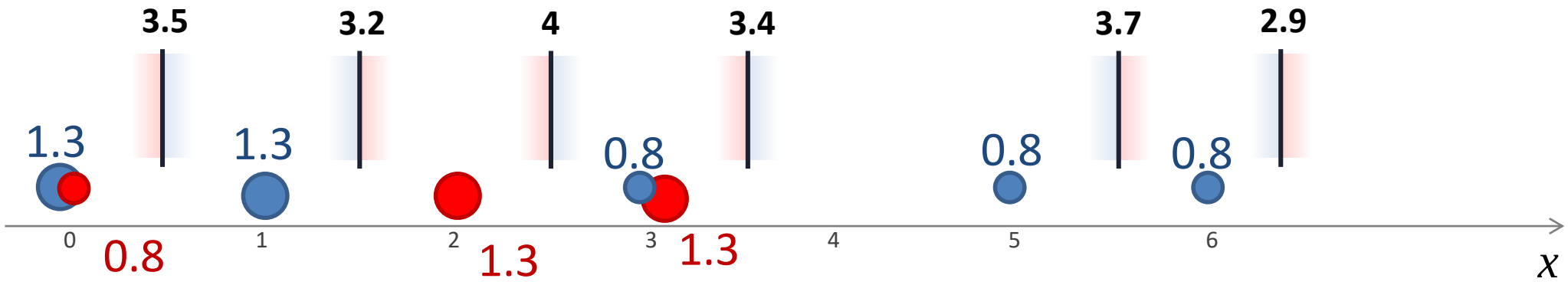


$$r_{t=1} = \frac{1}{\sum_i W_1(i)} \sum_i W_1(i) \cdot h_1(x_i) \cdot k_i = 2/8$$

$$\alpha_t = \frac{1}{2} \log\left(\frac{1+r_t}{1-r_t}\right)$$

$$W_{t=2}(i) = W_1(i) \cdot \exp\{-\alpha_1 \cdot k_i \cdot h_1(x_i)\} = W_1(i) \times \begin{cases} \sqrt{\frac{3}{5}} & if\ h_1(x) = k \\ \sqrt{\frac{5}{3}} & if\ h_1(x) \neq k \end{cases}$$

Adaboost



Adaboost

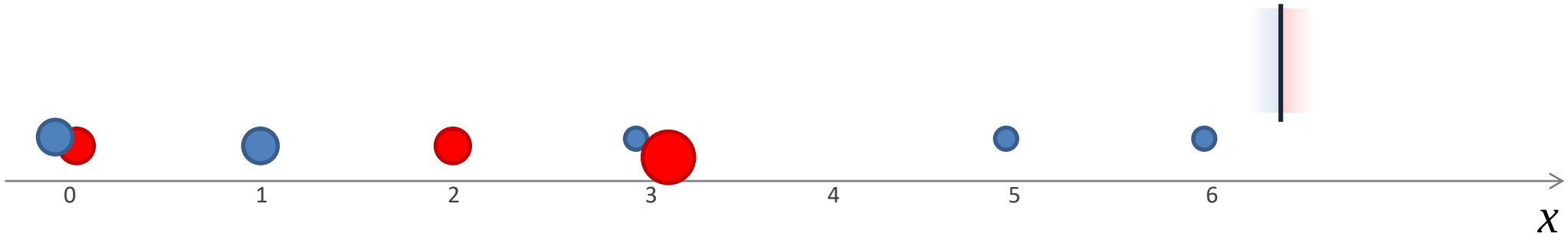
$$h_{t=2}(\bar{x}) = \{-1\}$$

$$W_{t=3}(i) = W_2(i) \times \begin{cases} \sqrt{\frac{1-r_t}{1+r_t}} & if\ h_2(x) = k \\ \sqrt{\frac{1+r_t}{1-r_t}} & if\ h_2(x) \neq k \end{cases}$$

$$r_{t=2} = \frac{1}{\sum_i W_2(i)} \sum_i W_2(i) \cdot h_2(x_i) \cdot k_i = 2.9$$

$$\alpha_t = \frac{1}{2} \log\left(\frac{1+r_t}{1-r_t}\right)$$

Adaboost

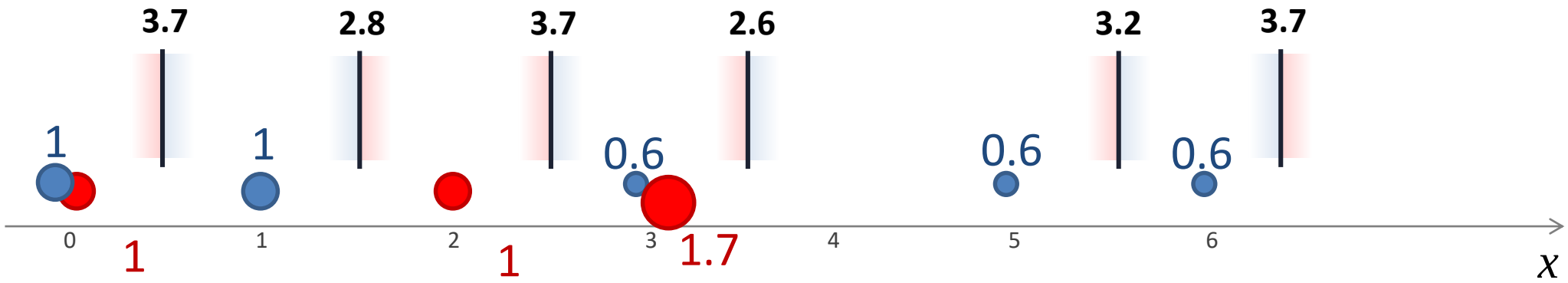


$$W_{t=3}(i) = W_2(i) \times \begin{cases} \sqrt{\frac{1-r_t}{1+r_t}} & if\ h_2(x) = k \\ \sqrt{\frac{1+r_t}{1-r_t}} & if\ h_2(x) \neq k \end{cases}$$

$$r_{t=2} = \frac{1}{\sum_i W_2(i)} \sum_i W_2(i) \cdot h_2(x_i) \cdot k_i = 2.9$$

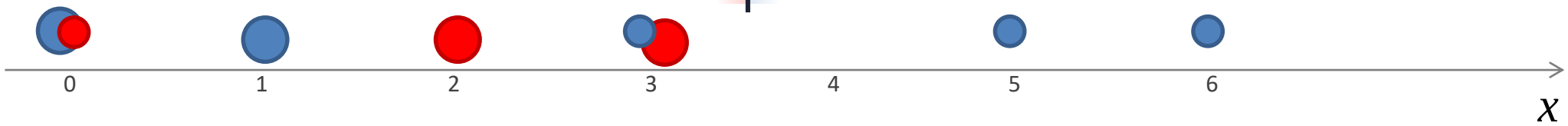
$$\alpha_t = \frac{1}{2} \log\left(\frac{1+r_t}{1-r_t}\right)$$

Adaboost



Adaboost

$$h_{t=3}(\bar{x}) = \{x < 3.5?\ \ +1\ \ :\ \ -1\}$$



$\alpha_1 = 0.25$	1	1	1	-1	-1	-1
$\alpha_2 = 0.27$	-1	-1	-1	-1	-1	-1
$\alpha_3 = 0.33$	1	1	1	1	-1	-1
	1	1	1	-1	-1	-1

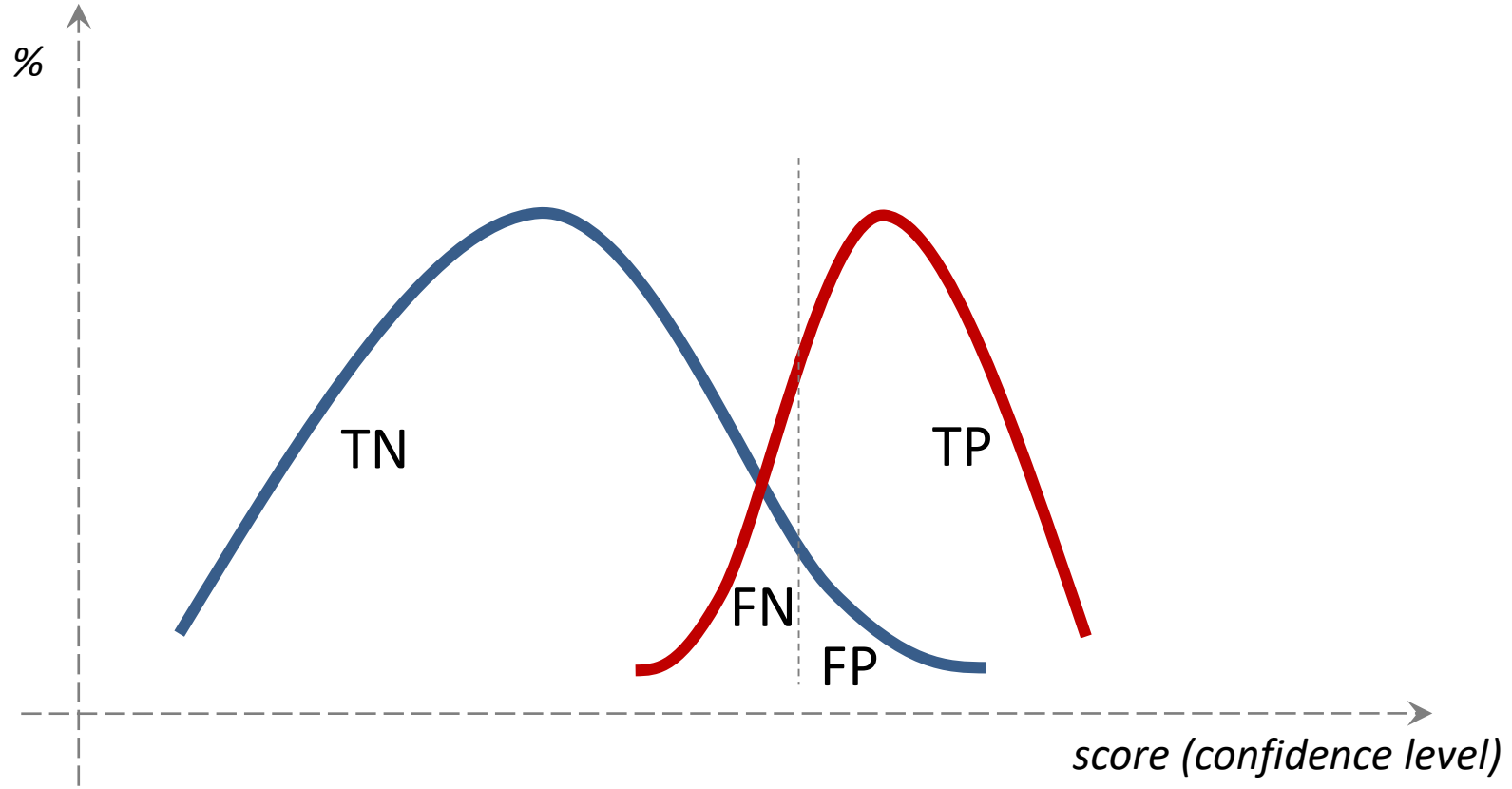
Part 3: Accuracy and Benchmarking

ROC-Curve
Precision, Recall, Accuracy, F1 score
Benchmarking the Classifiers

ROC curve

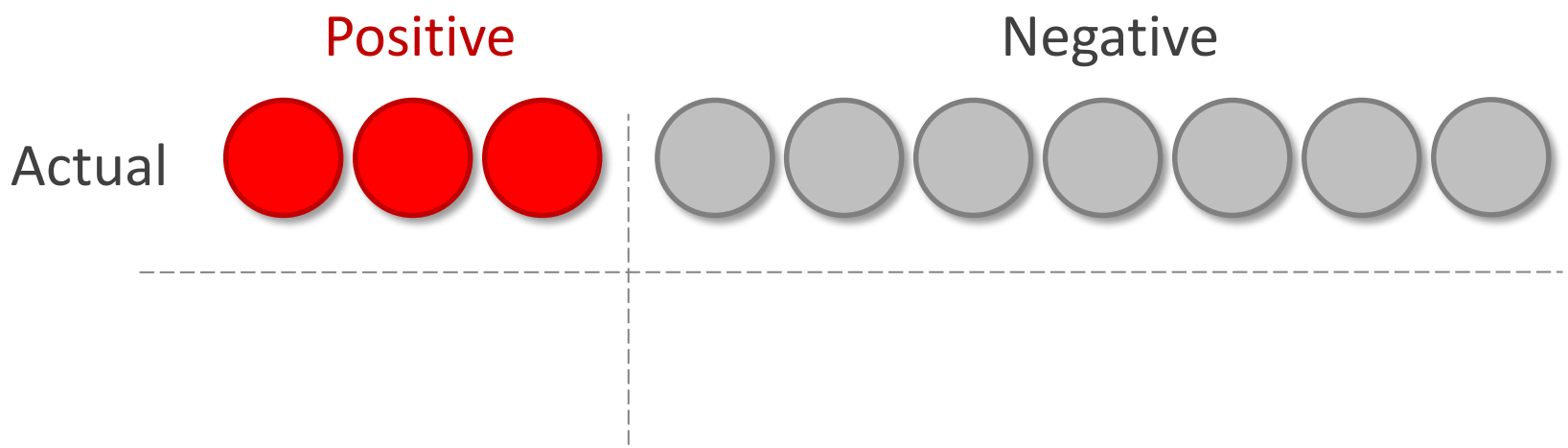
https://github.com/dryabokon/ML/blob/master/ex_03_01_ROC.py

ROC curve



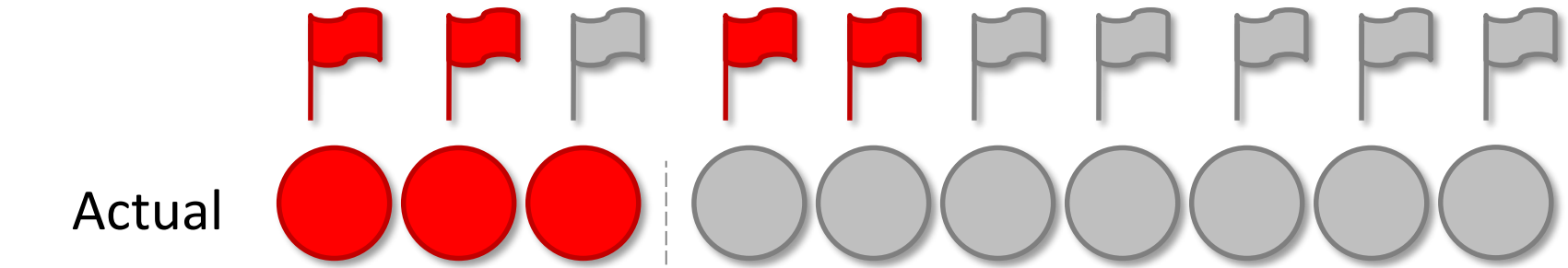
ROC curve

Error types



ROC curve

Error types

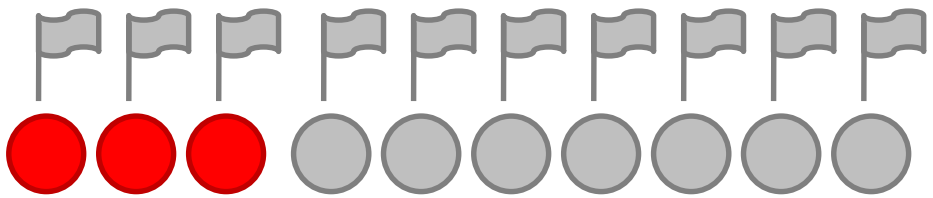


true pos **TP: 2/3**

false neg **FN: 1/3**

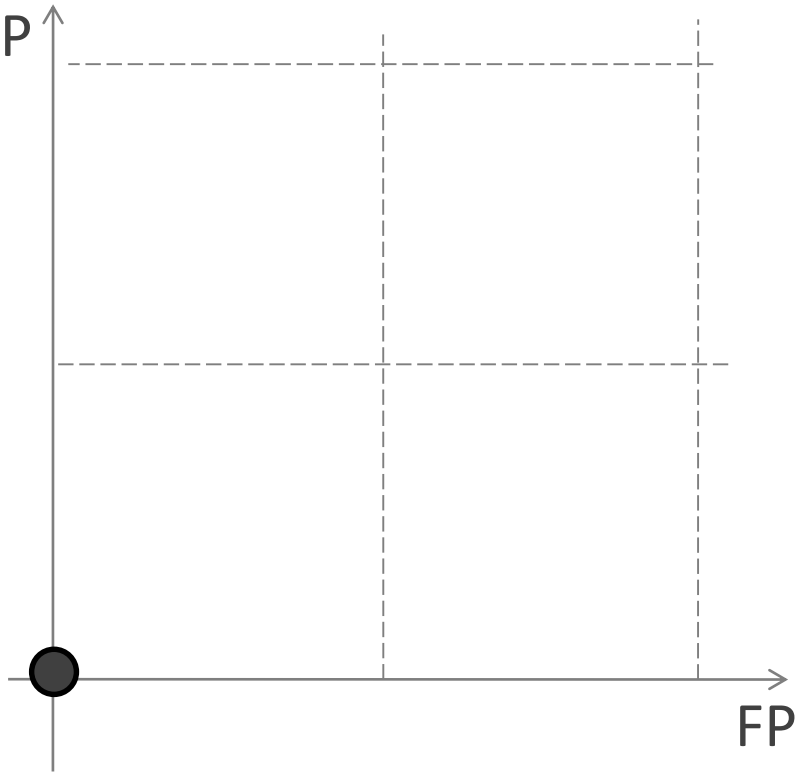
FP: 2/7 false pos

ROC curve

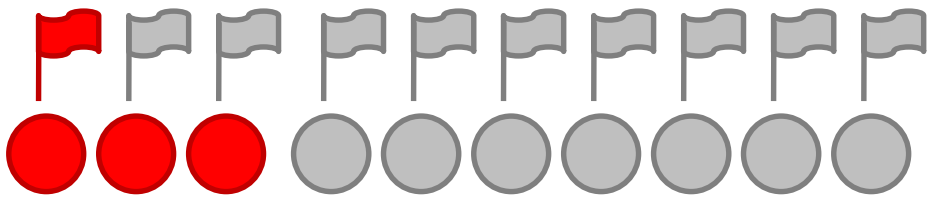


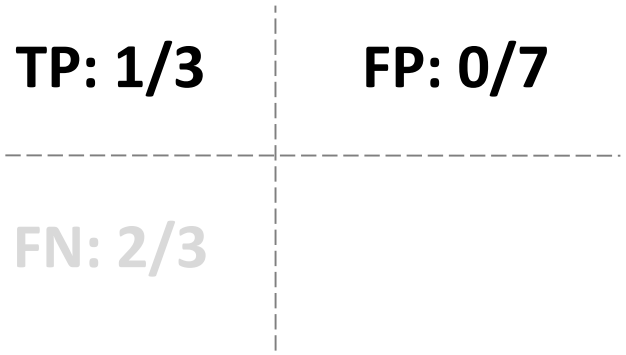
TP: 0/3 FP: 0/7

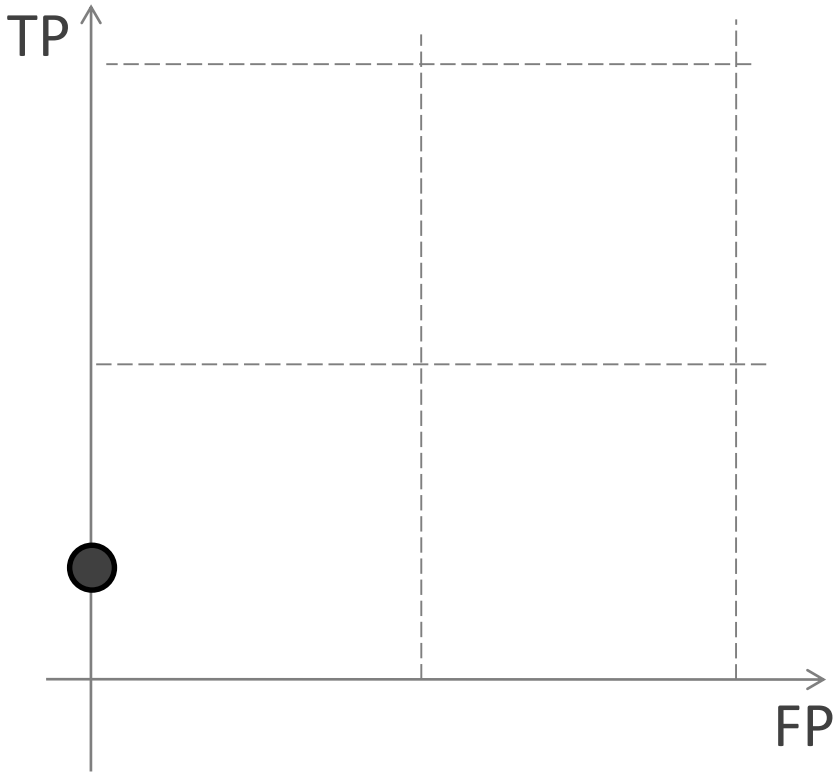
FN: 3/3



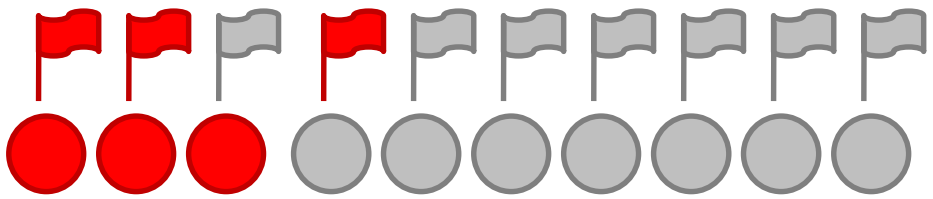
ROC curve

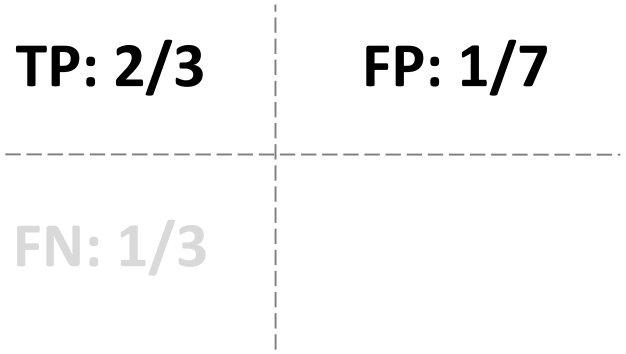


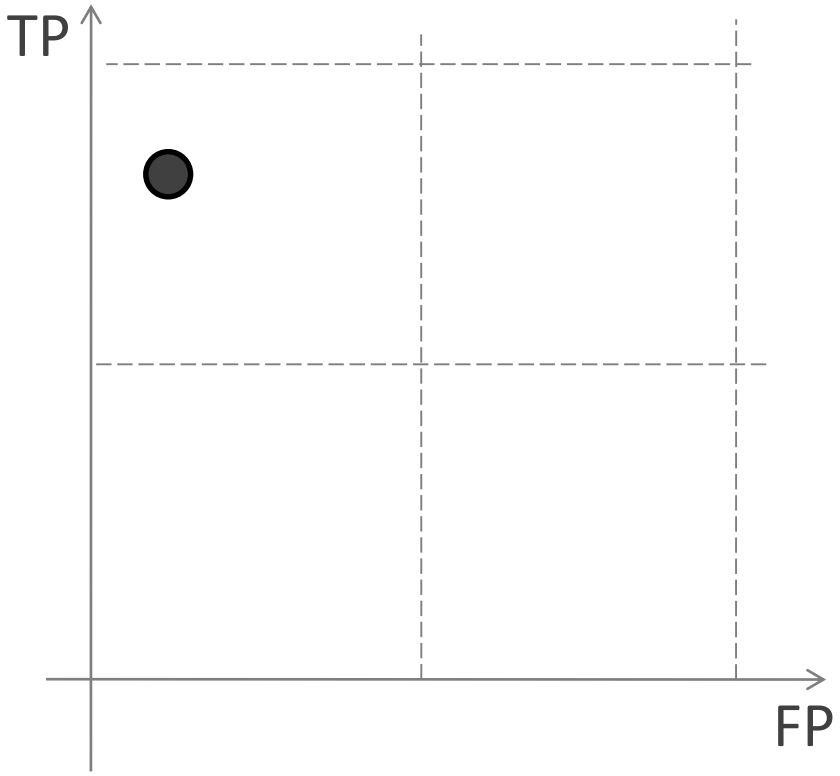




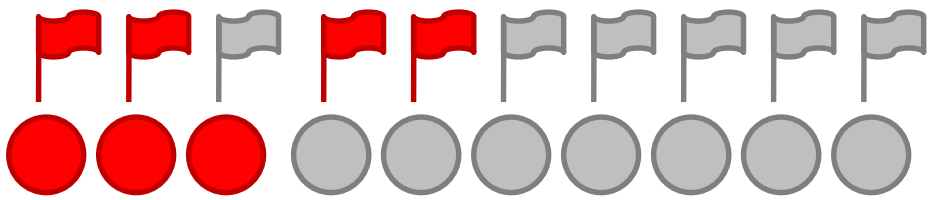
ROC curve

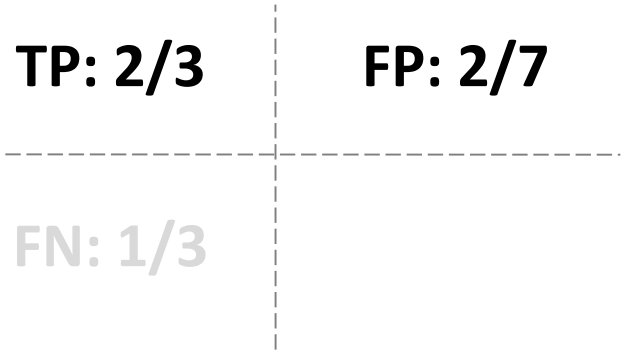


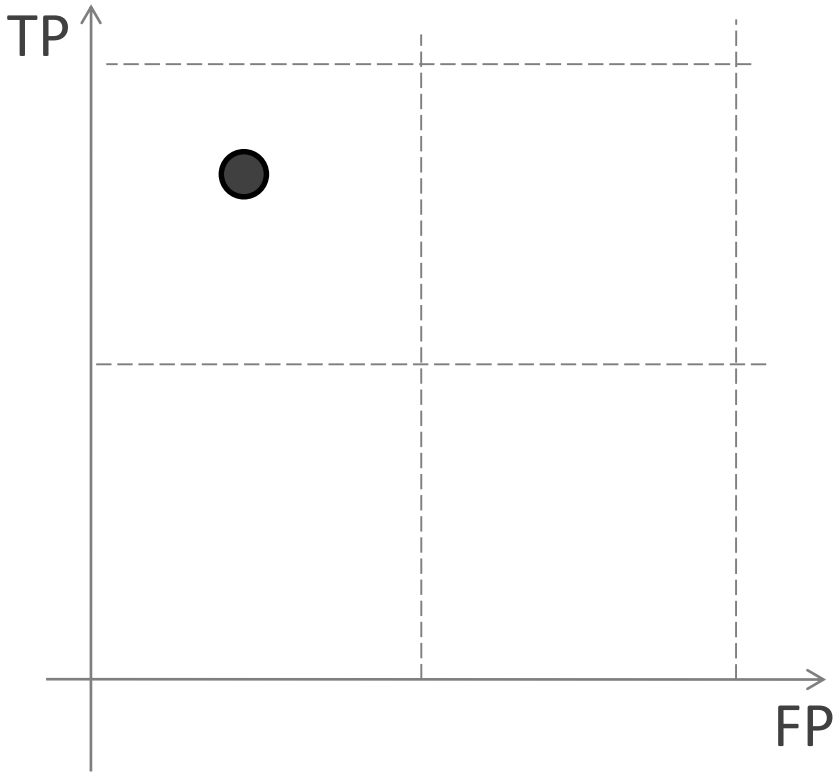




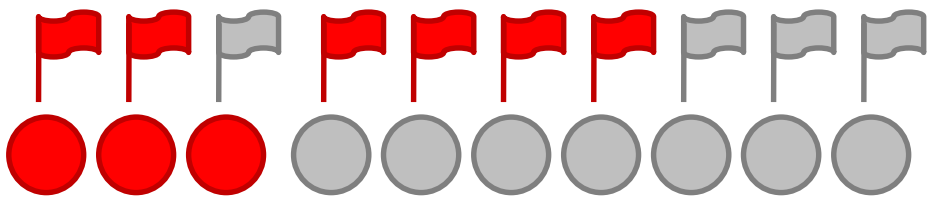
ROC curve

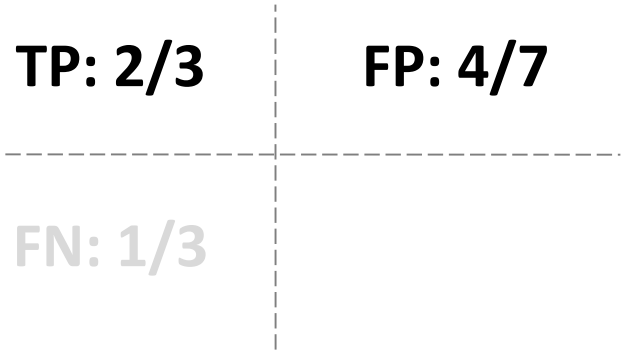


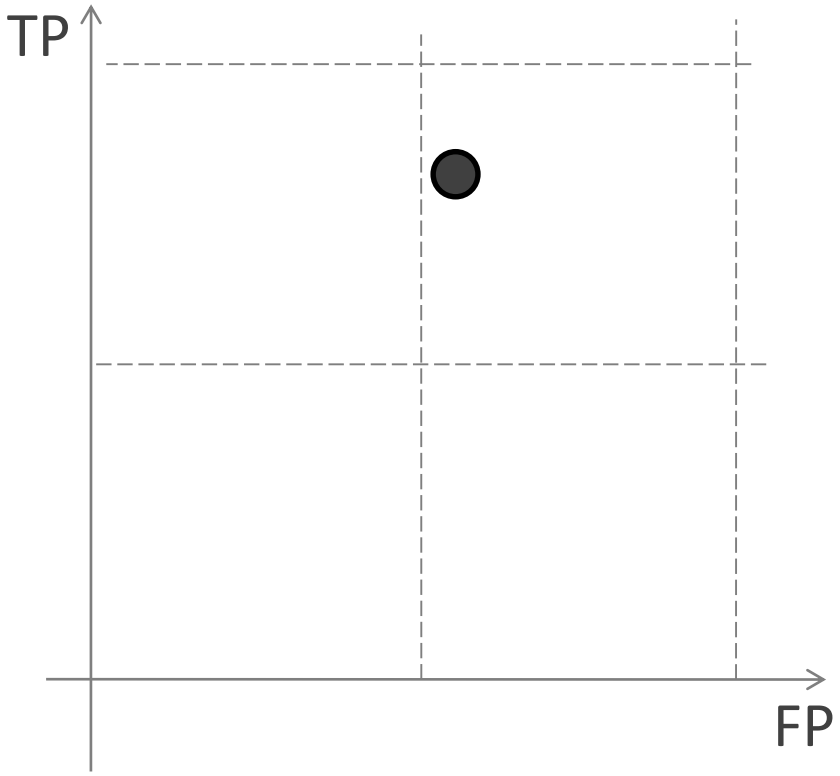




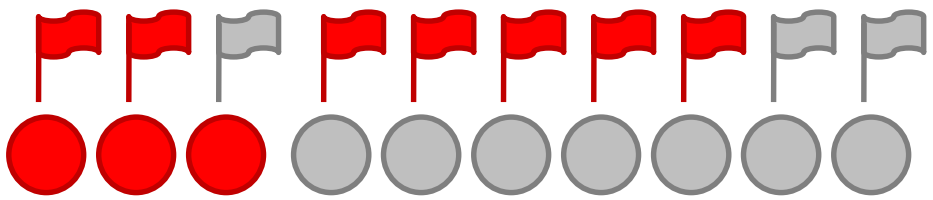
ROC curve





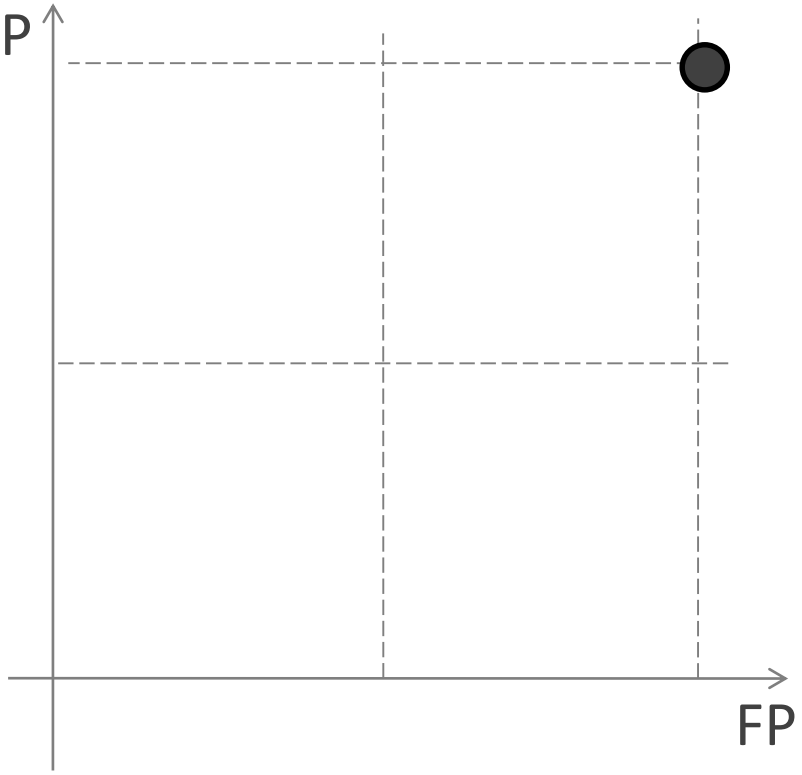


ROC curve



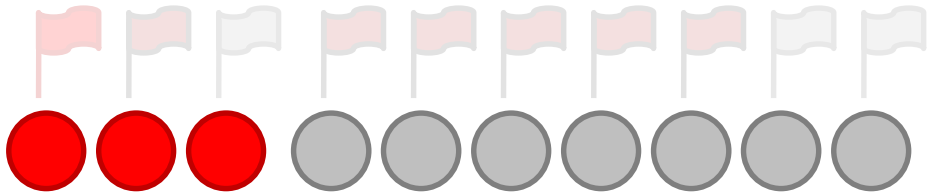
TP: 3/3 FP: 7/7

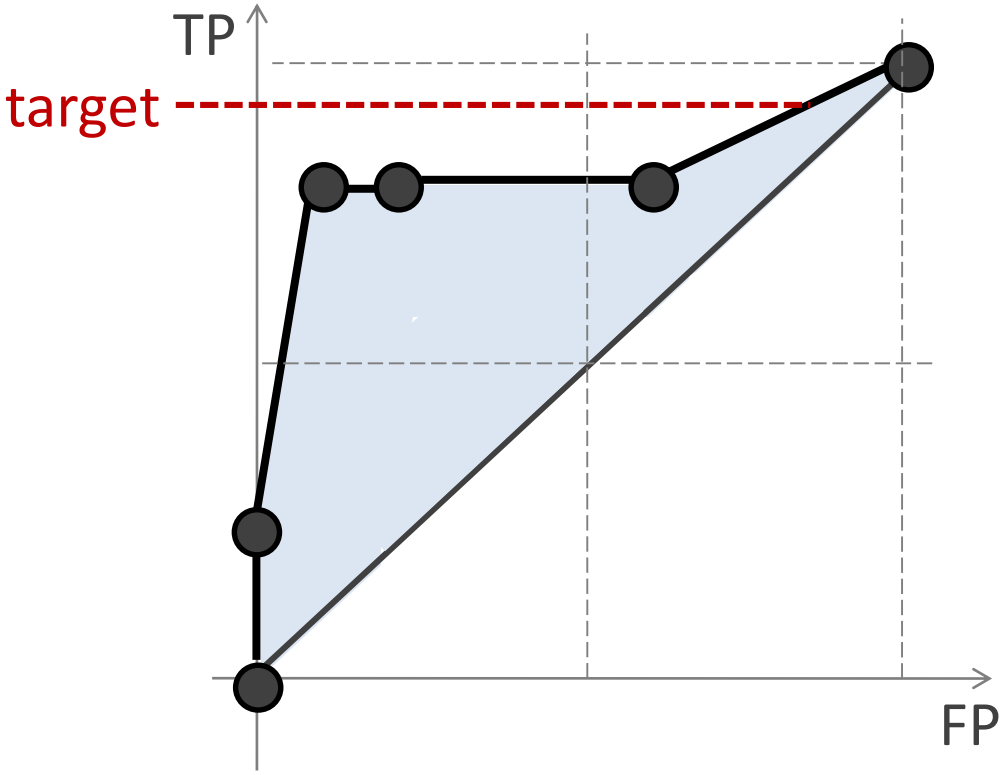
FN: 0/3



ROC curve

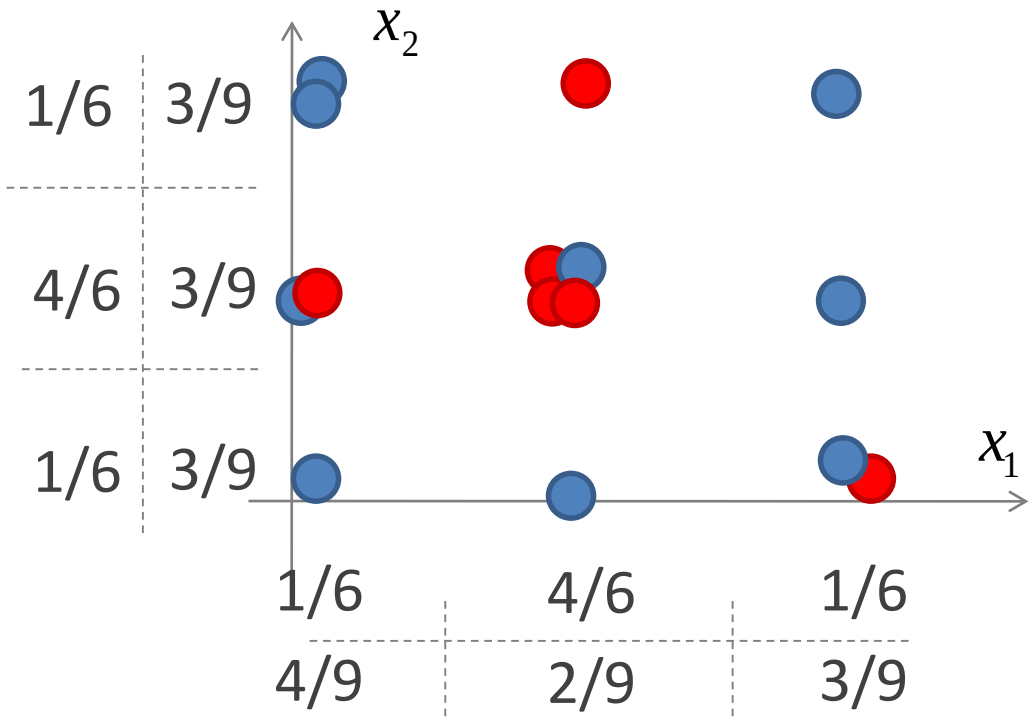
AUC: area under ROC curve





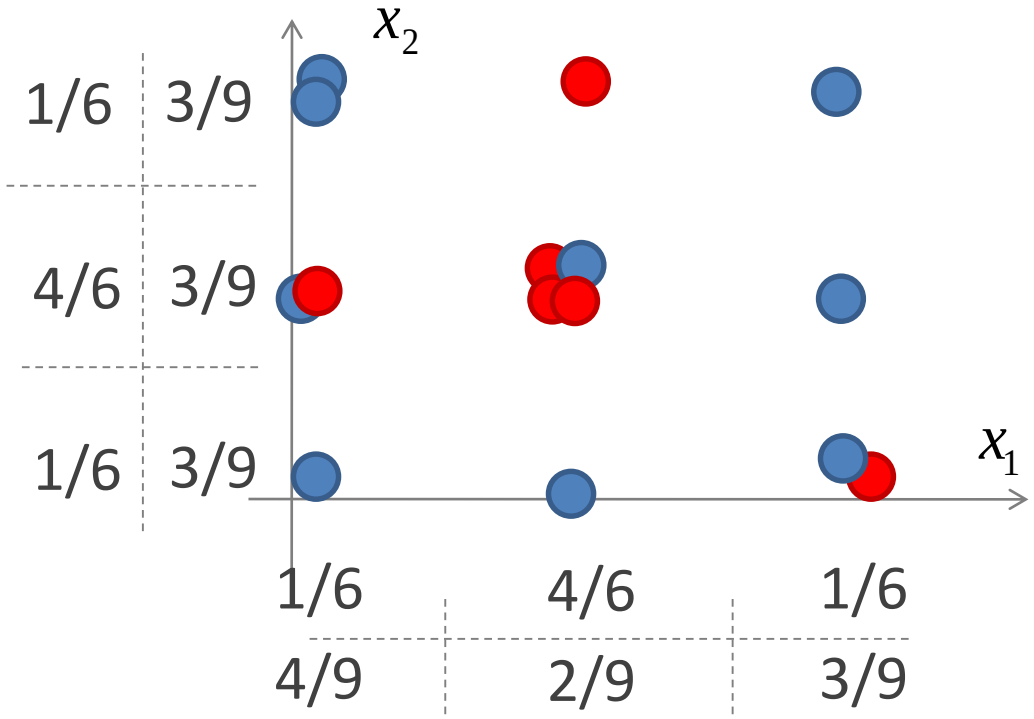
ROC curve

Example: Naive Bayesian classifier



ROC curve

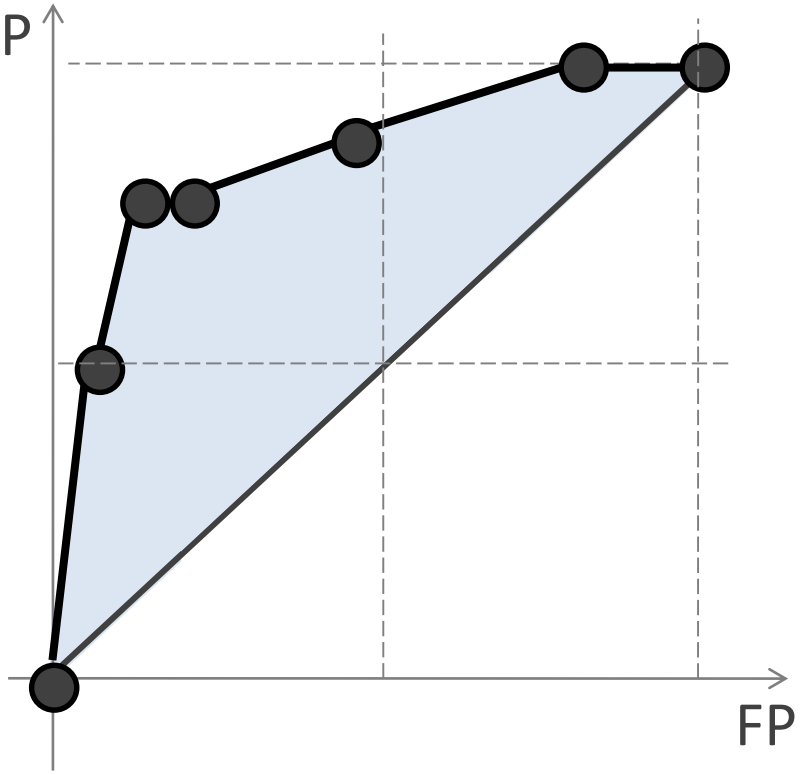
Example: Naive Bayesian classifier



x	(0,0)	(1,0)	(2,0)	(2,0)	(0,1)	(0,1)	(1,1)	(1,1)	(1,1)	(1,1)	(2,1),	(0,2)	(0,2)	(1,2)	(2,2)
k	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
$\frac{p(1\|x)}{p(2\|x)}$	0.19	1.50	0.25	0.25	0.75	0.75	6.00	6.00	6.00	6.00	1.00	0.19	0.19	1.50	0.25

ROC curve

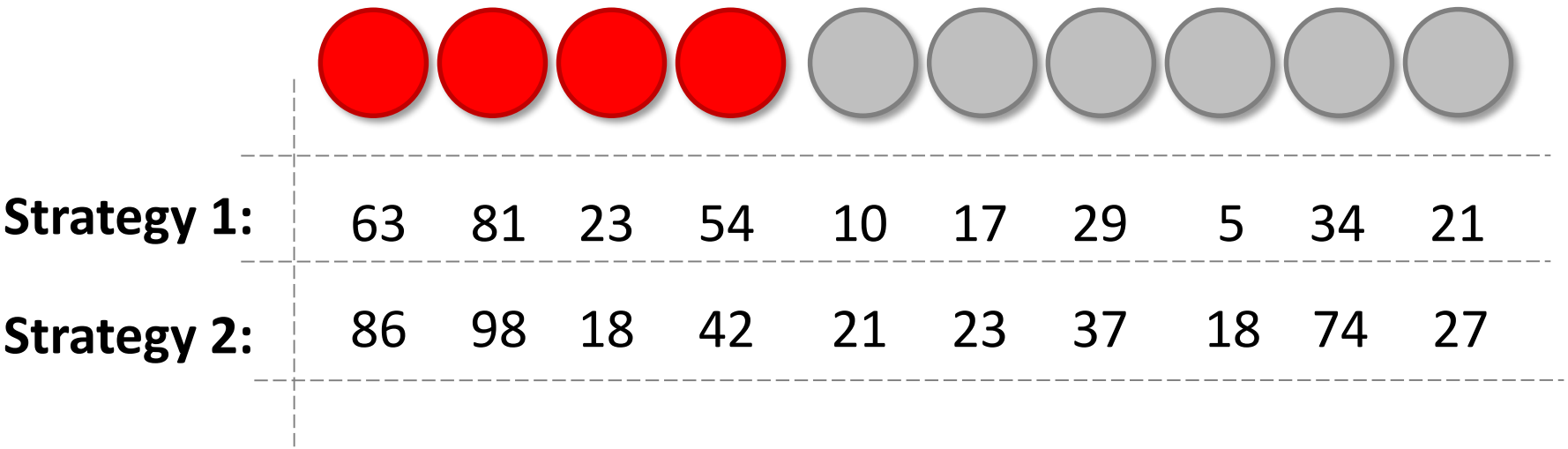
Example: Naive Bayesian classifier



k															
$\frac{p(1\|x)}{p(2\|x)}$	6.00	6.00	6.00	6.00	1.50	1.50	1.00	0.75	0.75	0.25	0.25	0.25	0.19	0.19	0.19

ROC curve

The task: compare accuracy of two strategies



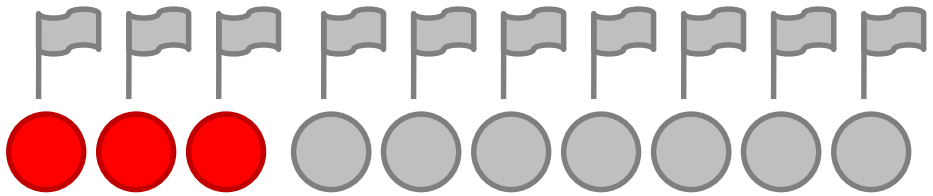
Precision-Recall

Precision - Recall

Precision: TP / (TP+FP)
Recall: TP / (TP+FN)

Accuracy: (TP+TN) /Total
F1-score: 2 x Precision x Recall / (Precision+ Recall)

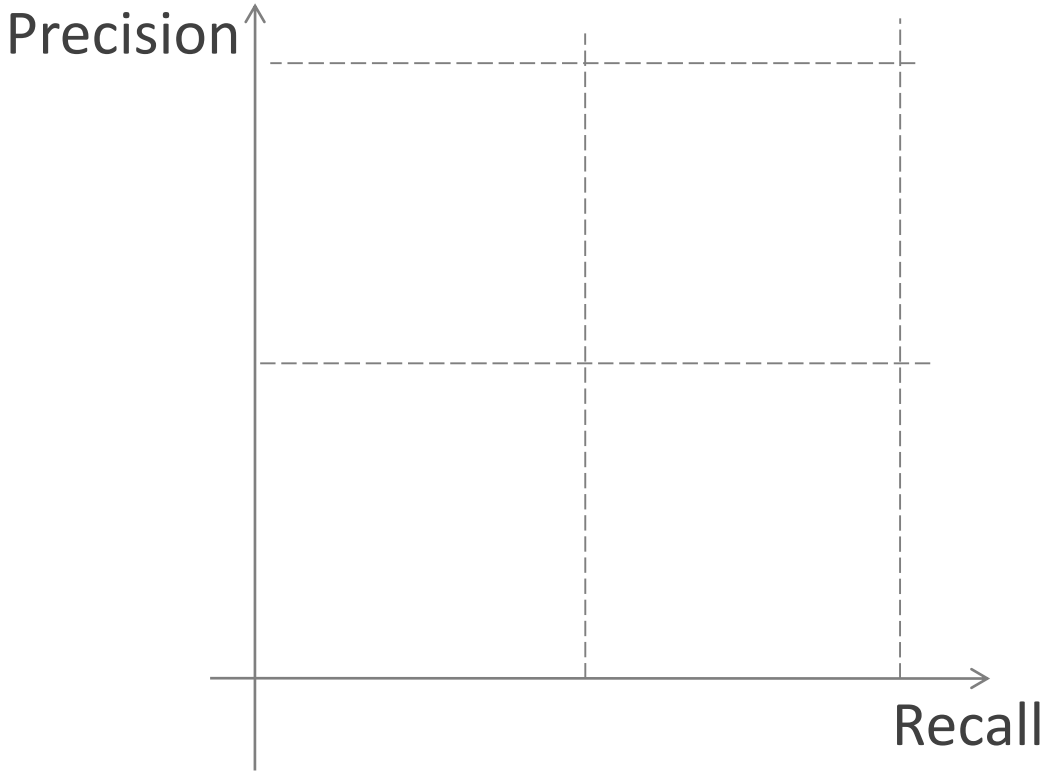
Precision - Recall



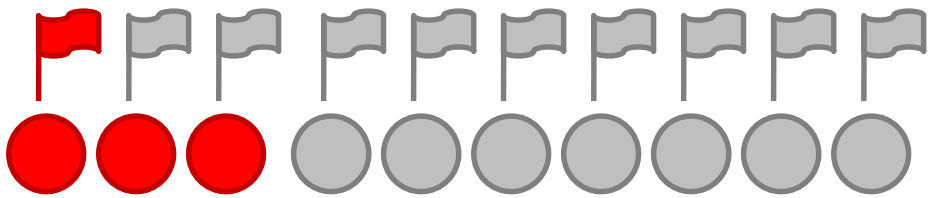
TP: 0	FP: 0
FN: 3	

Precision: 0/0

Recall: 0/3



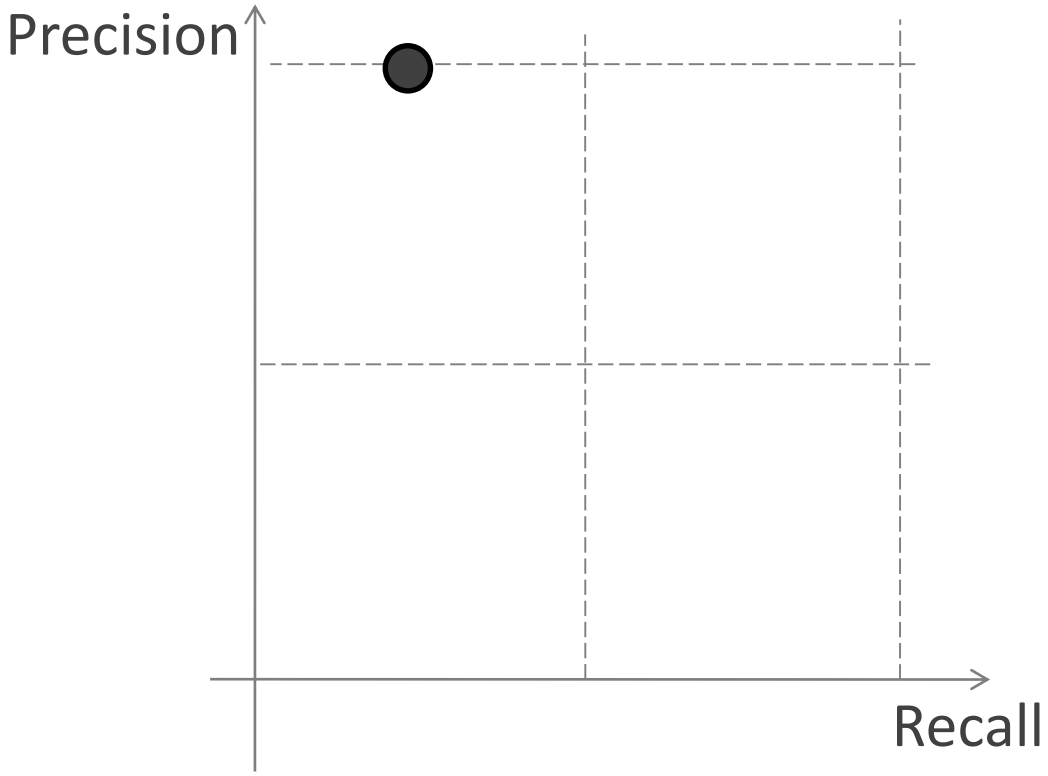
Precision - Recall



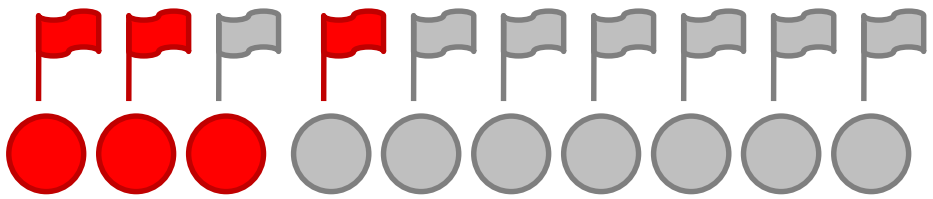
TP: 1	FP: 0
FN: 2	

Precision: 1

Recall: 1/3



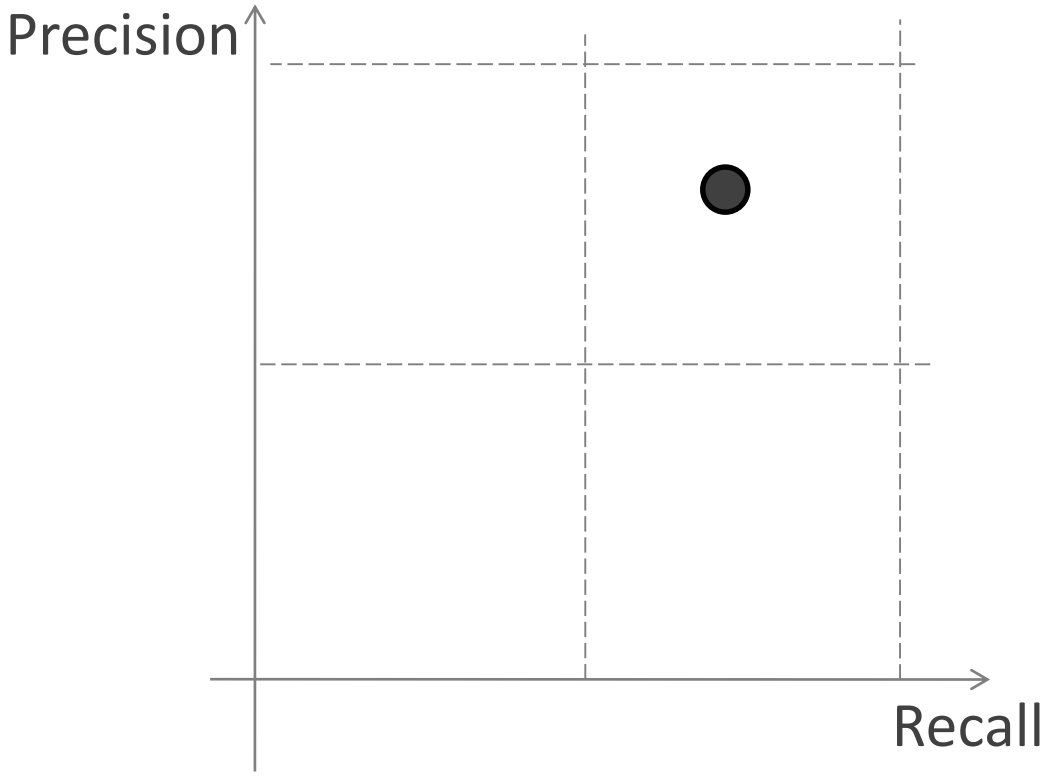
Precision - Recall



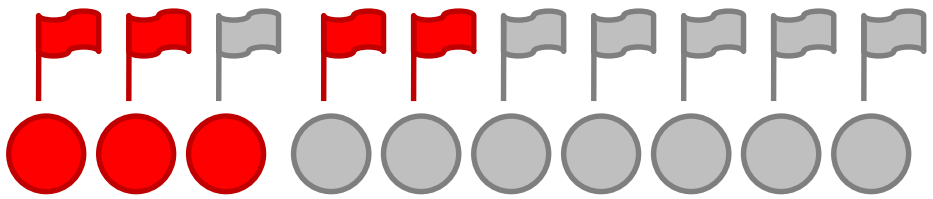
TP: 2	FP: 1
FN: 1	

Precision: 2/3

Recall: 2/3



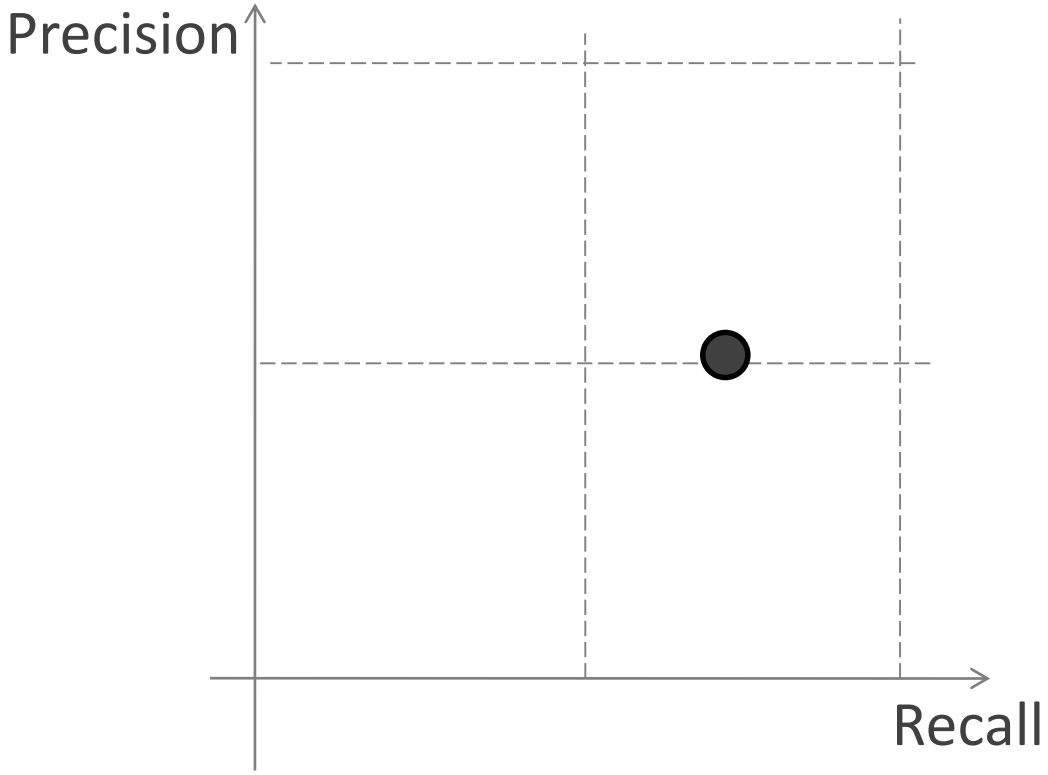
Precision - Recall



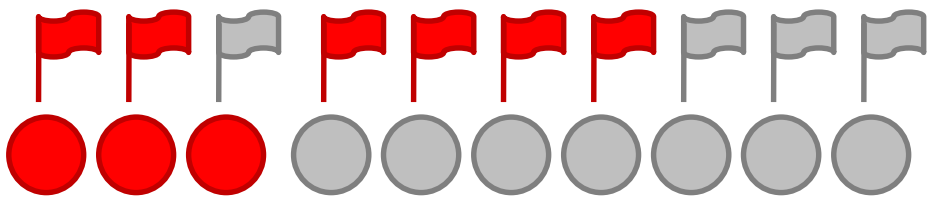
TP: 2	FP: 2
FN: 1	

Precision: 1/2

Recall: 2/3



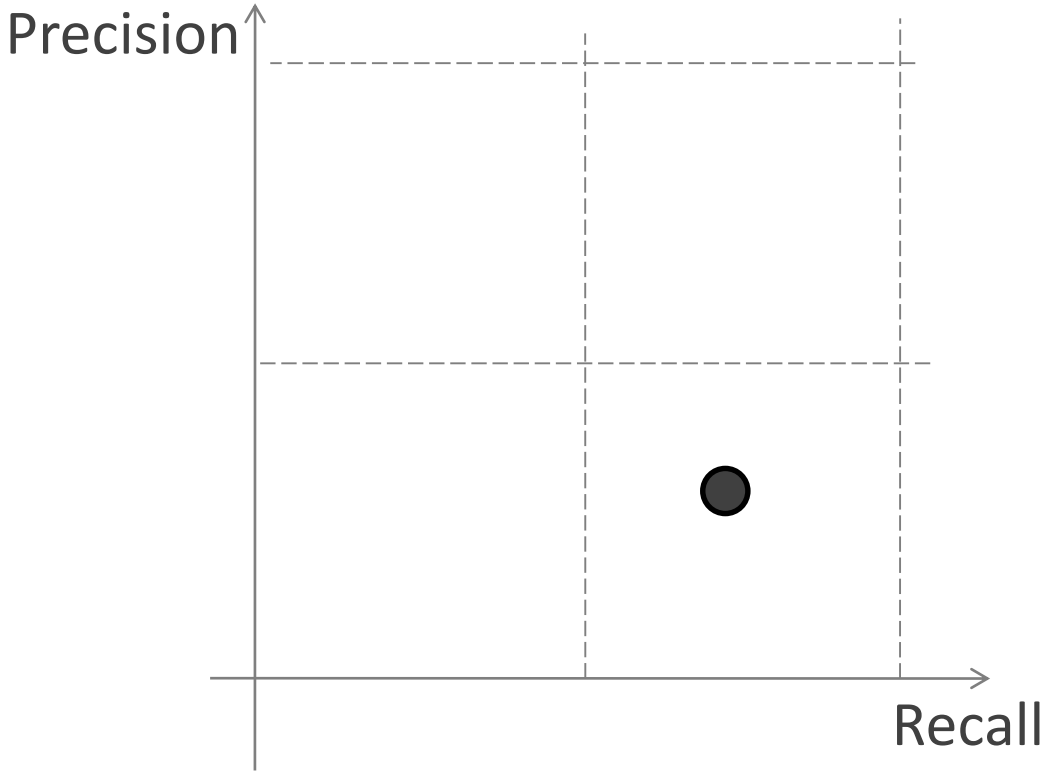
Precision - Recall



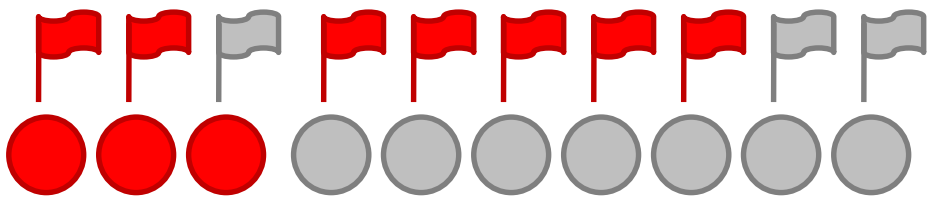
TP: 2	**FP: 4**
FN: 1	

Precision: 2/6

Recall: 2/3



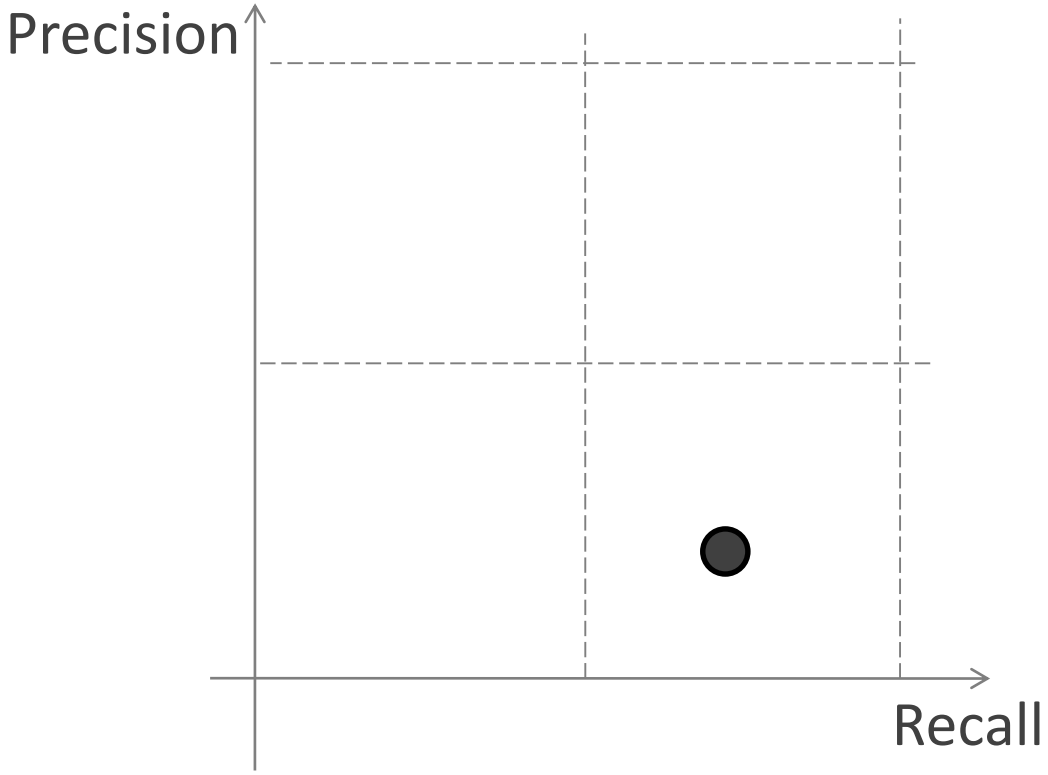
Precision - Recall



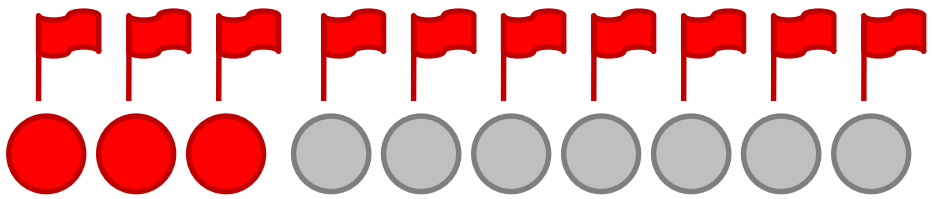
TP: 2	FP: 5
FN: 1	

Precision: 2/7

Recall: 2/3



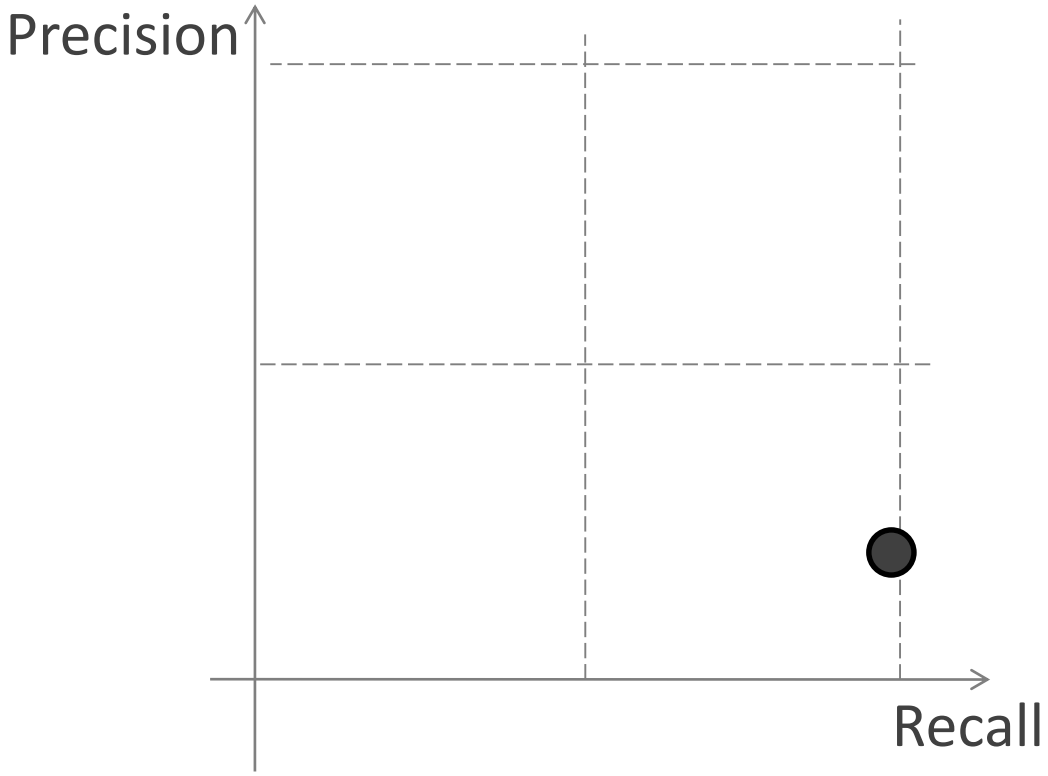
Precision - Recall



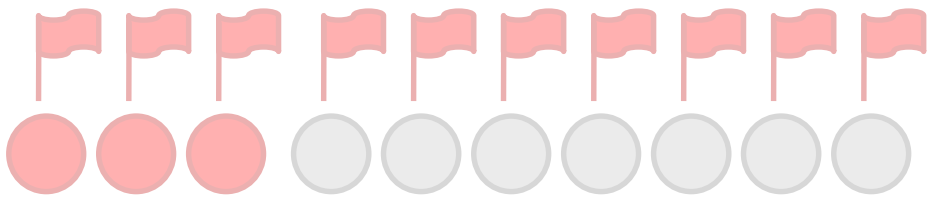
TP: 3 **FP: 7** **Precision: 3/7**

FN: 0

Recall: 1

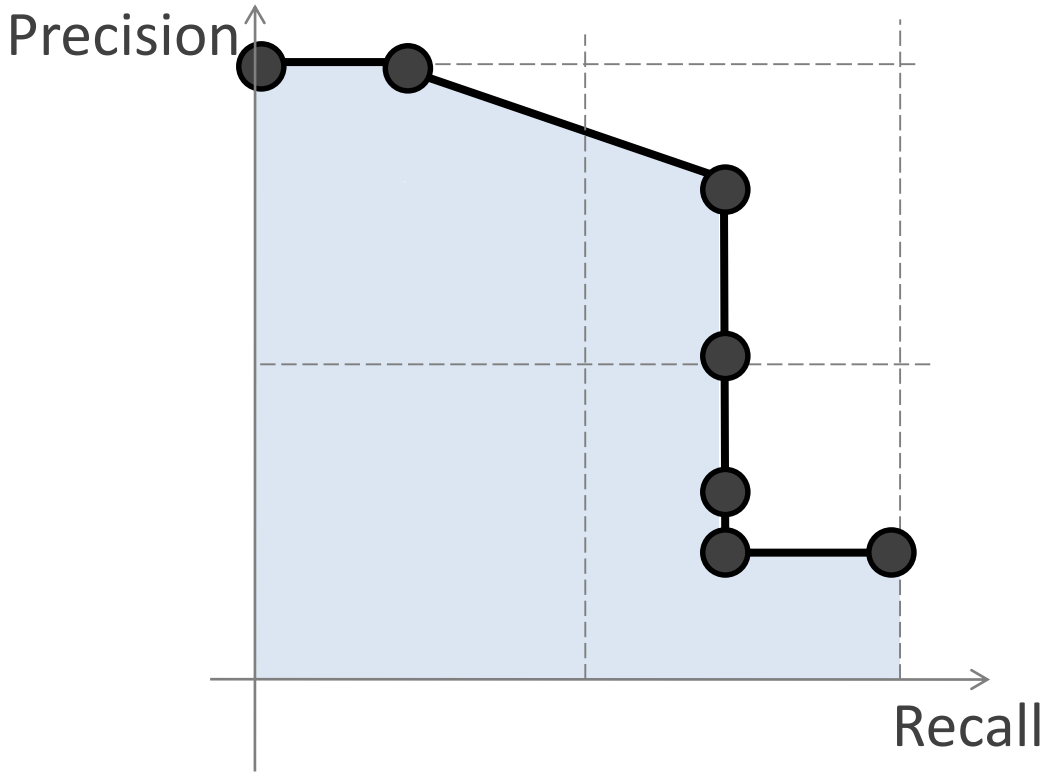


Precision - Recall



TP: 3	FP: 7
FN: 0	

Recall: 1



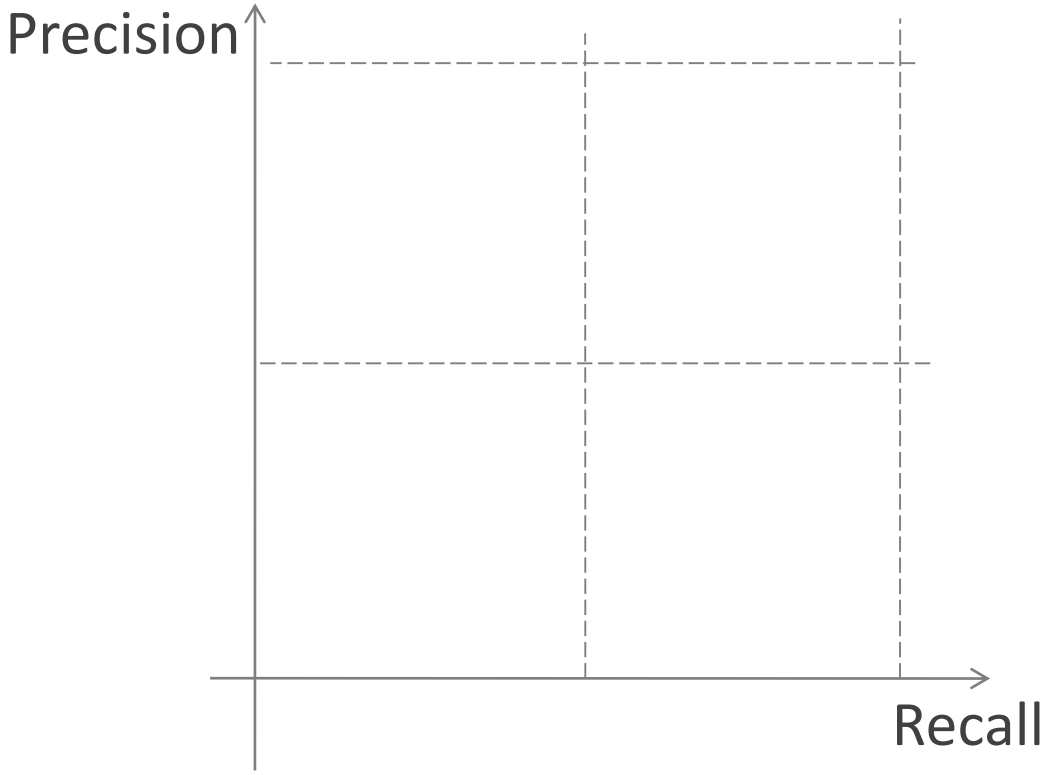
Precision - Recall

TP: 0 **FP: 0**

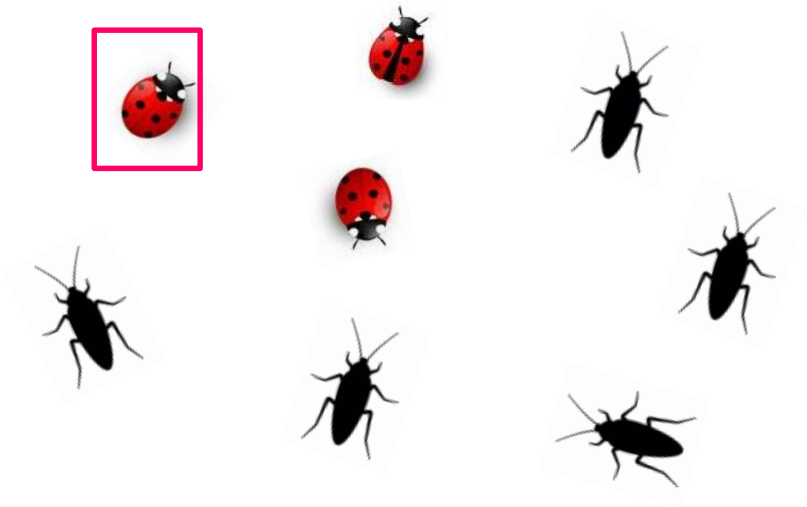
FN: 3

Precision: 0

Recall: 0



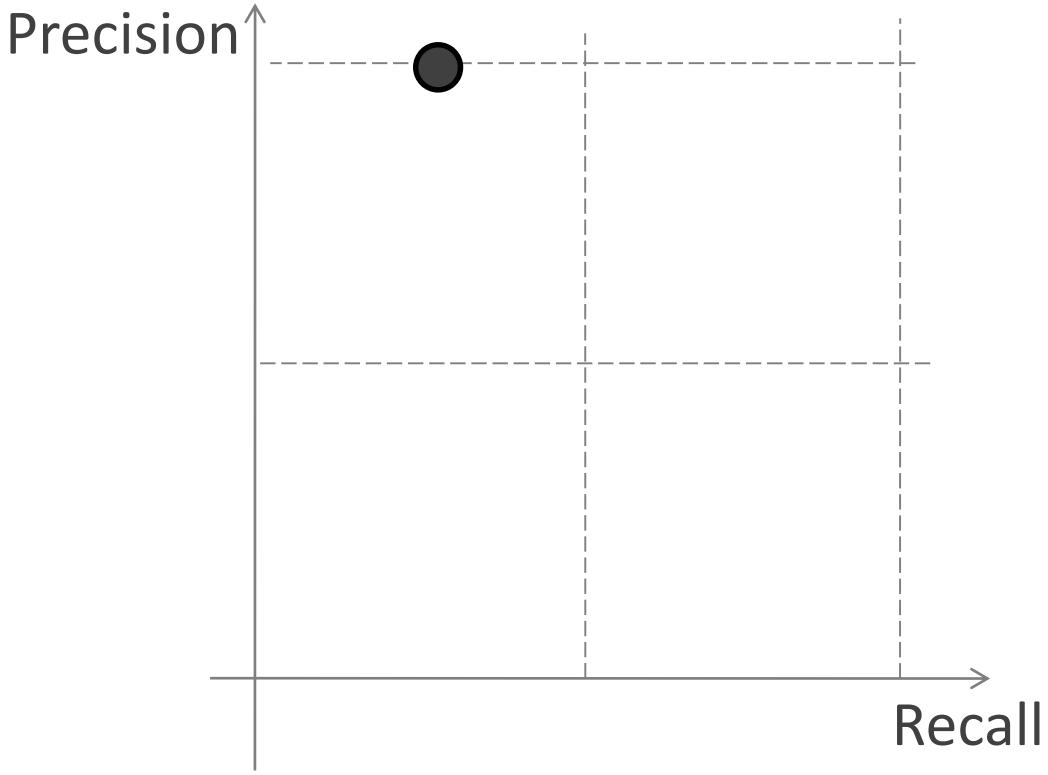
Precision - Recall



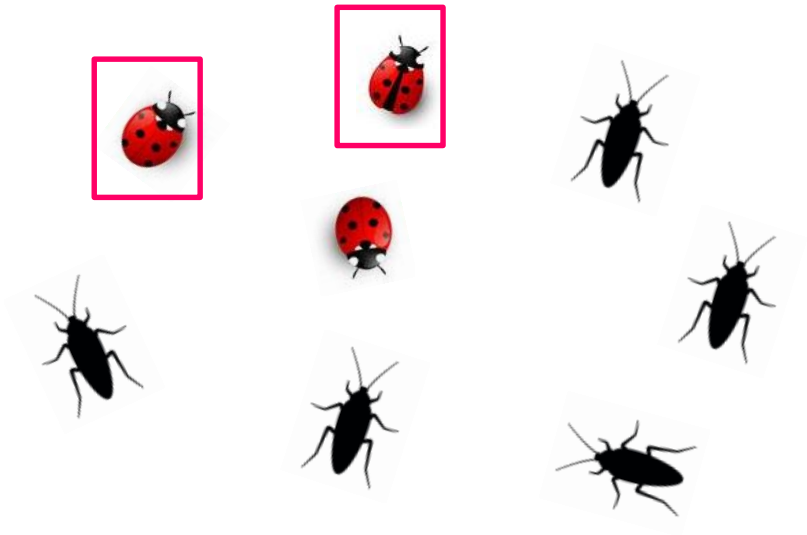
TP: 1	FP: 0
FN: 2	

Precision: 1

Recall: 1/3



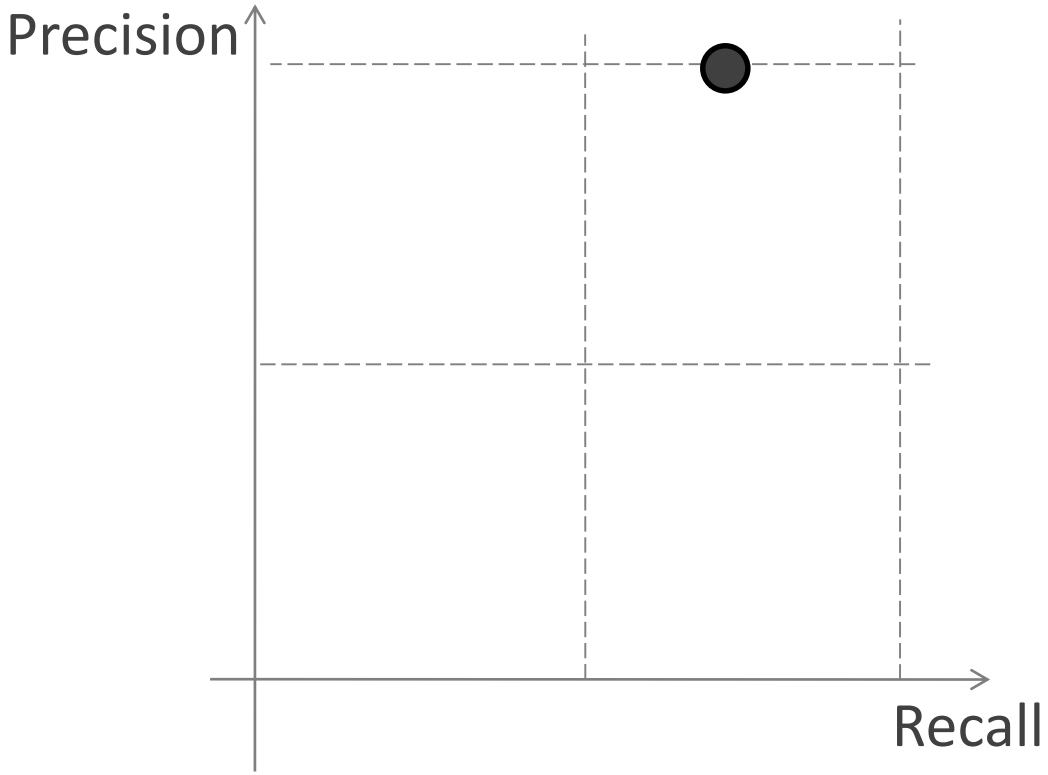
Precision - Recall



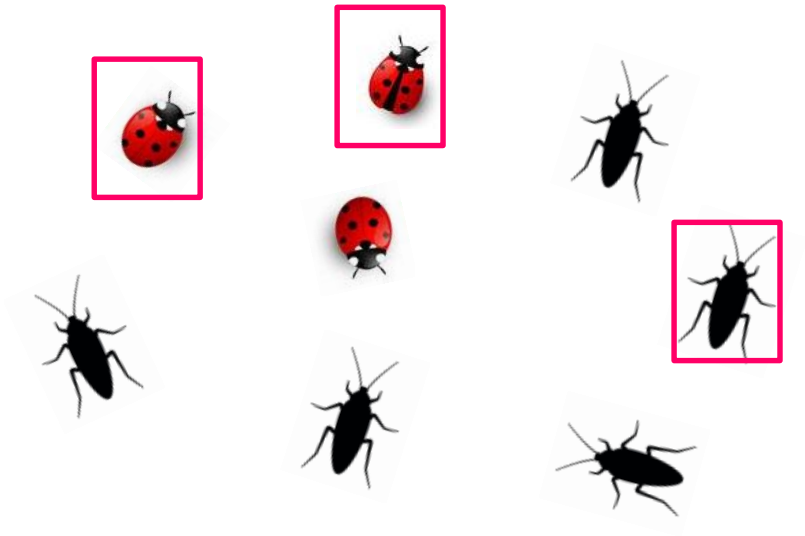
TP: 2	FP: 0
FN: 1	

Precision: 1

Recall: 2/3



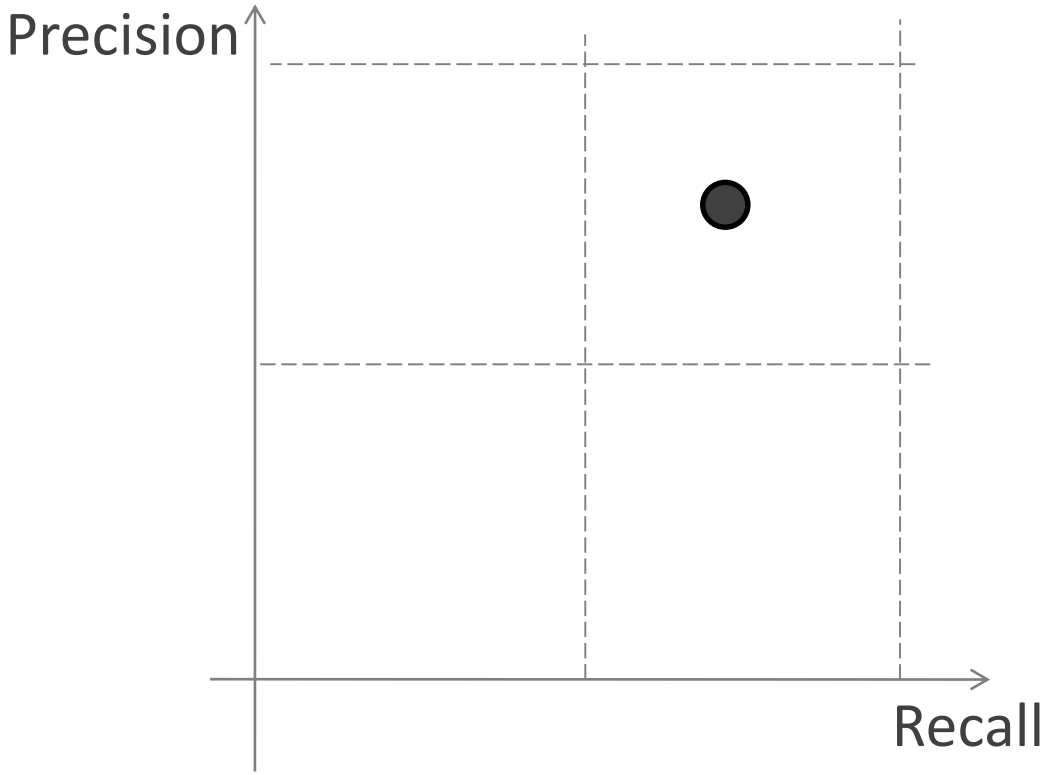
Precision - Recall



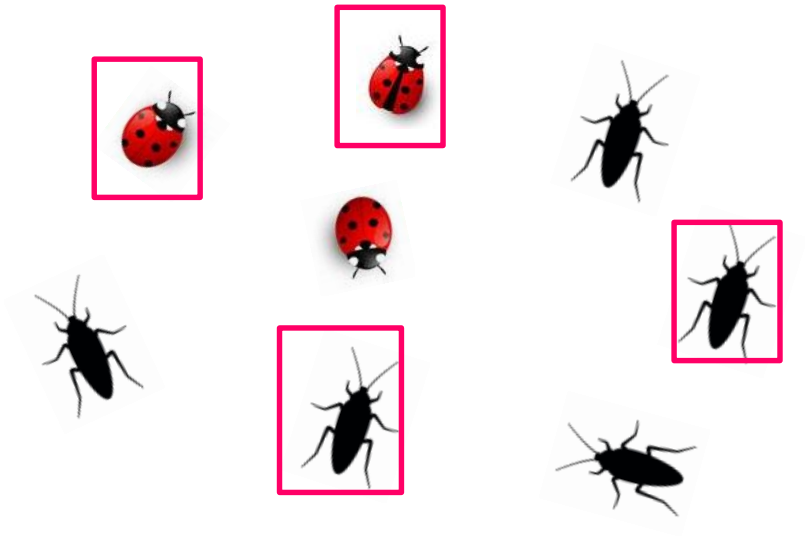
TP: 2	**FP: 1**
FN: 1	

Precision: 2/3

Recall: 2/3



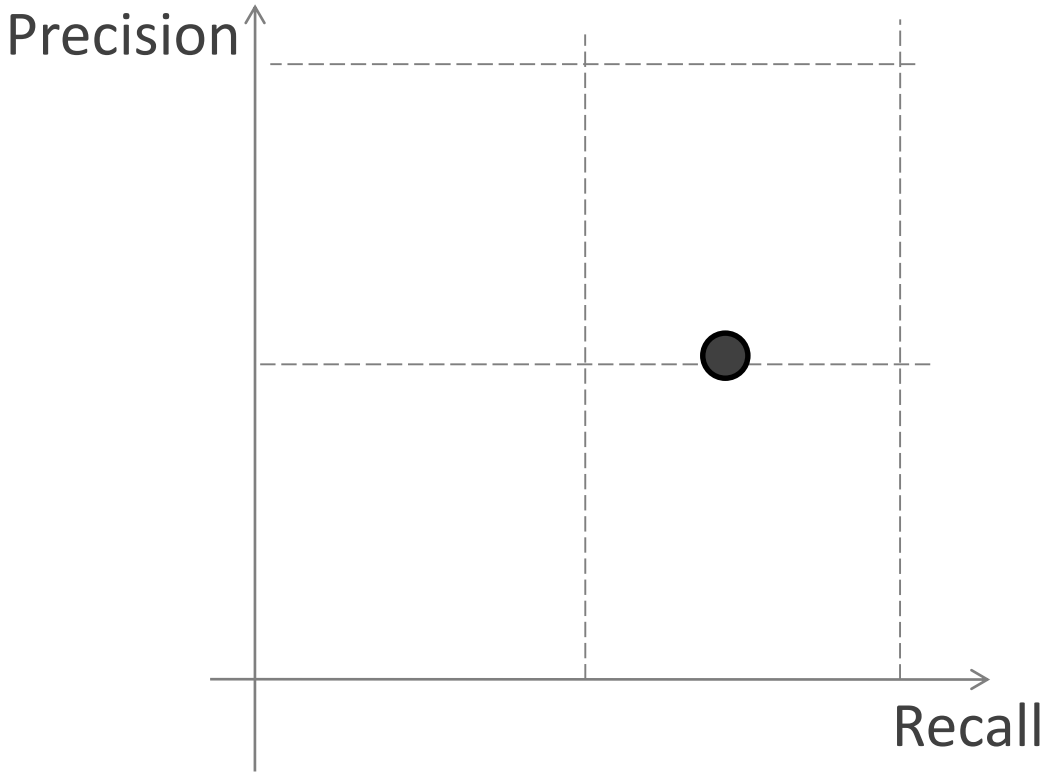
Precision - Recall



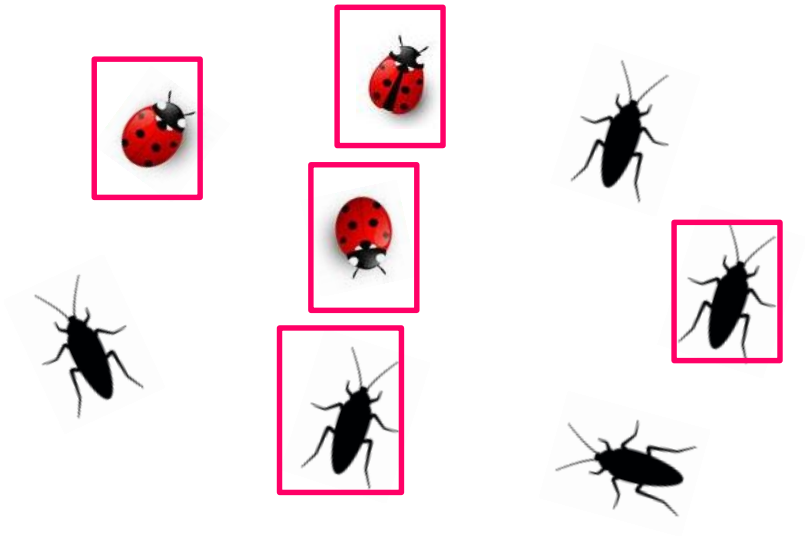
TP: 2	FP: 2
FN: 1	

Precision: 2/4

Recall: 2/3



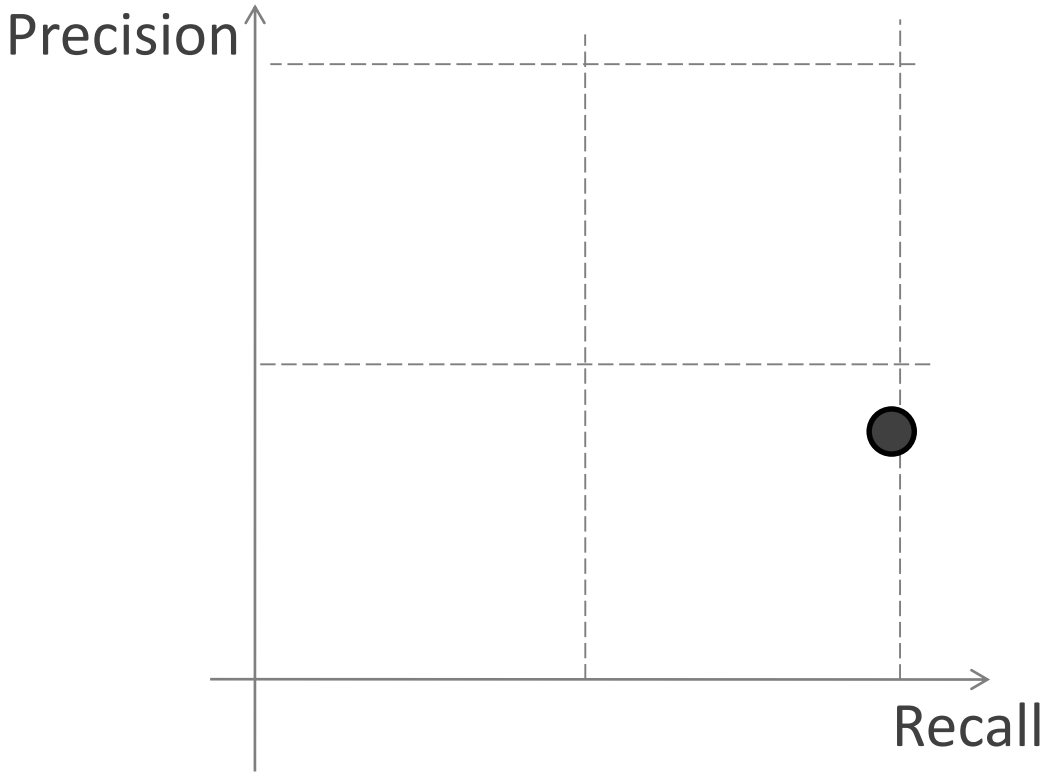
Precision - Recall



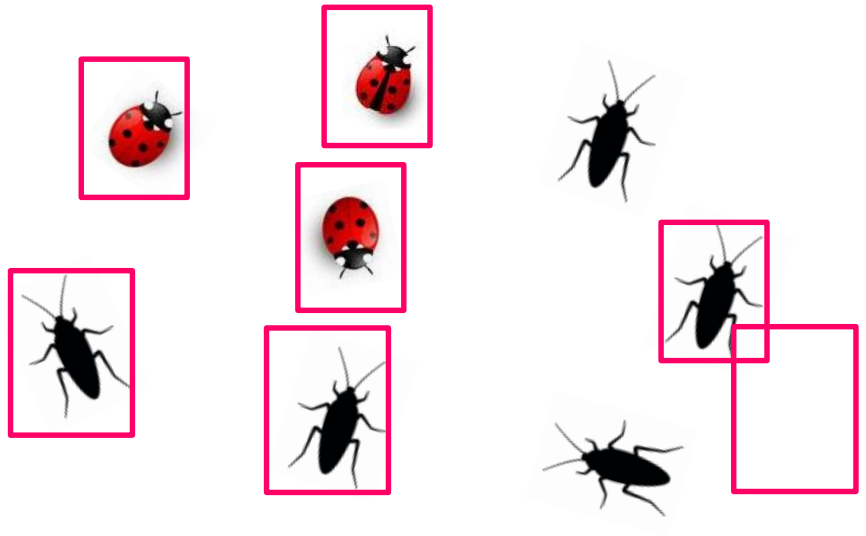
TP: 3 **FP: 2** **Precision: 2/5**

FN: 0

Recall: 1



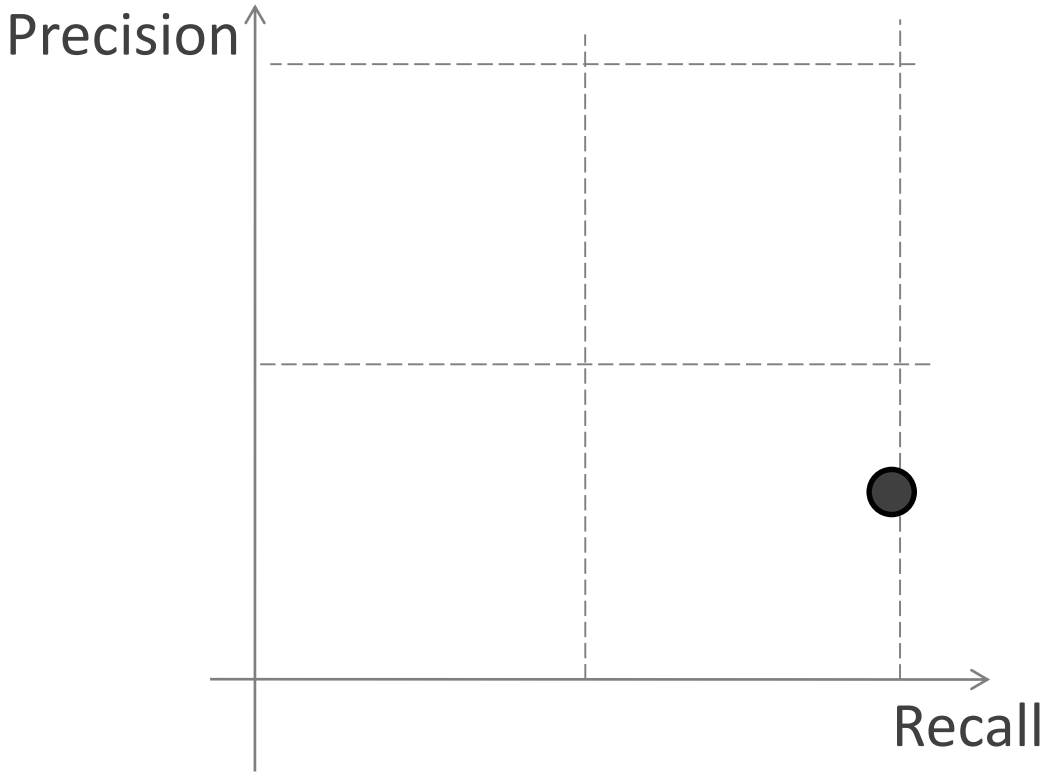
Precision - Recall



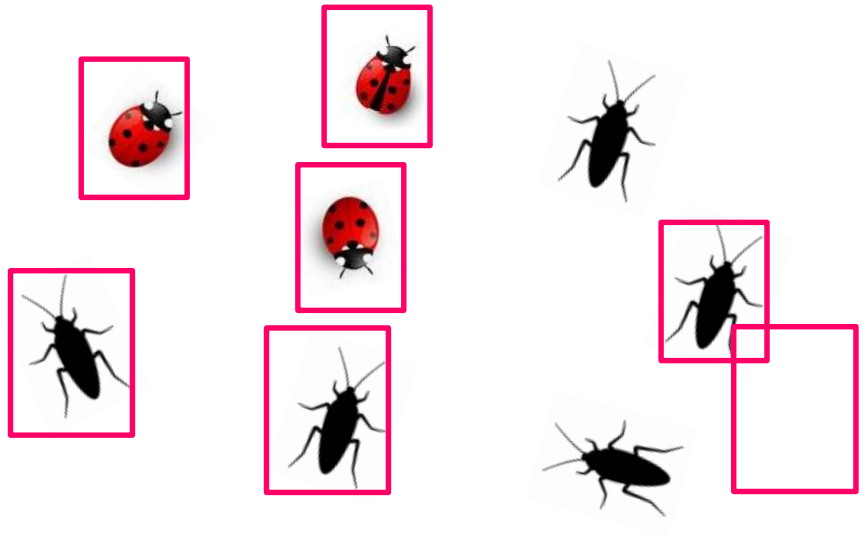
TP: 3 **FP: 4** **Precision: 3/7**

FN: 0

Recall: 1



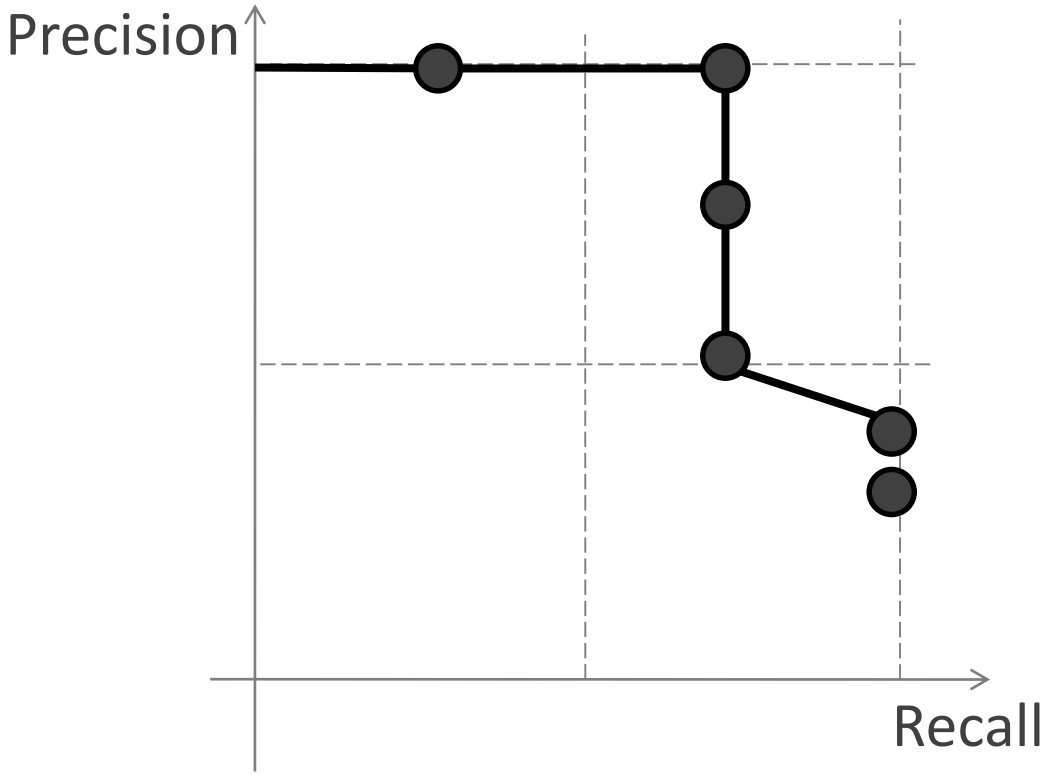
Precision - Recall



TP: 3 **FP: 4** **Precision: 3/7**

FN: 0

Recall: 1



Feature importance

https://github.com/dryabokon/ML/blob/master/ex_03_02_feature_importance.py

Feature importance

```
def example_01():
    dataset = load_wine()
    X,Y = dataset.data, dataset.target
    model = XGBClassifier()
    model.fit(X, Y)

    feature_importances = model.get_booster().get_score()

    print(feature_importances)
    plot_importance(model)
    plt.show()

    return
```

Benchmarking the classifiers

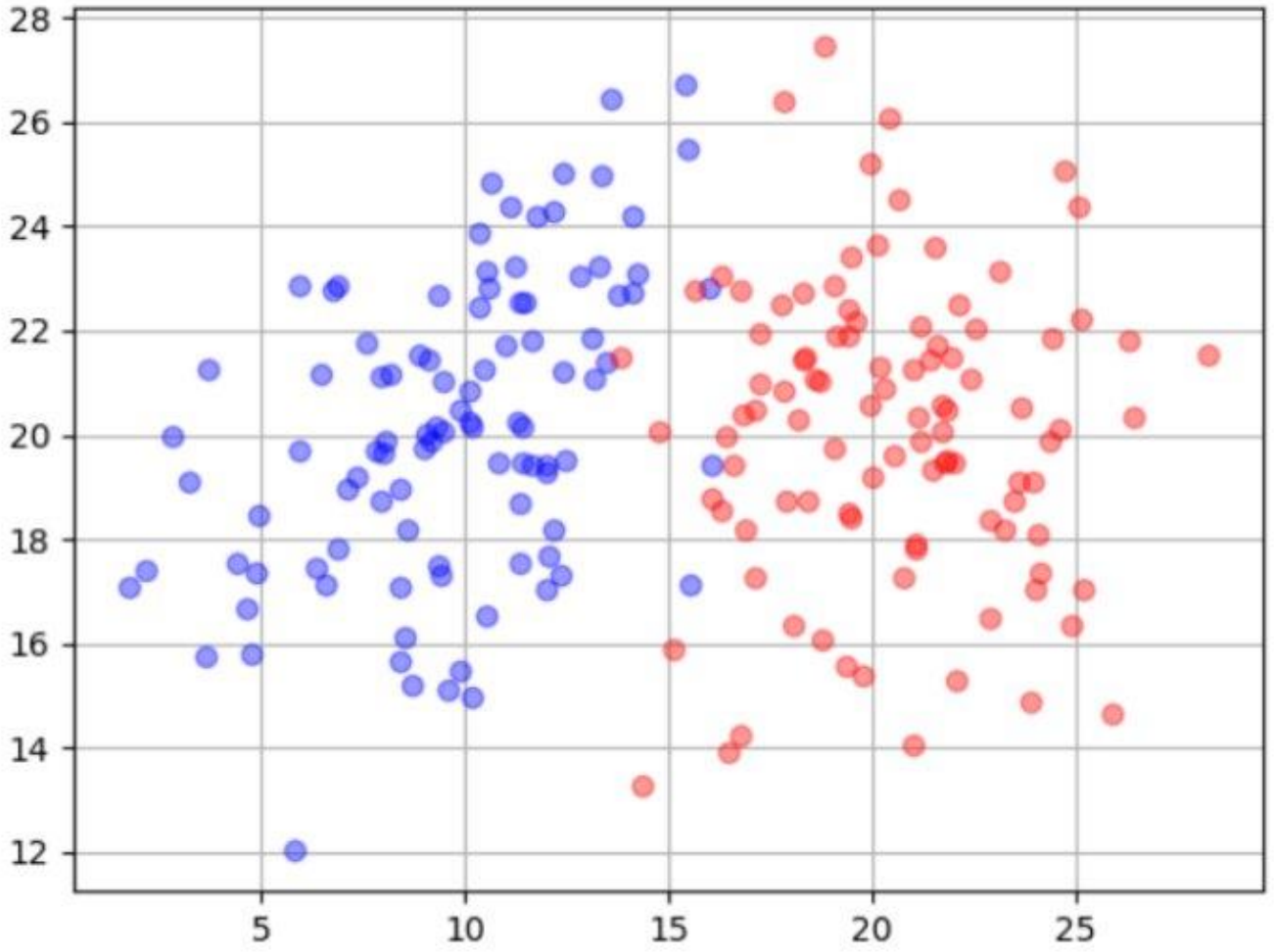
github.com/dryabokon/ML/blob/master/ex_03_03_benchmark_classifiers_k2.py
github.com/dryabokon/ML/blob/master/ex_03_03_benchmark_classifiers_multi_class.py

Benchmarking the Classifiers

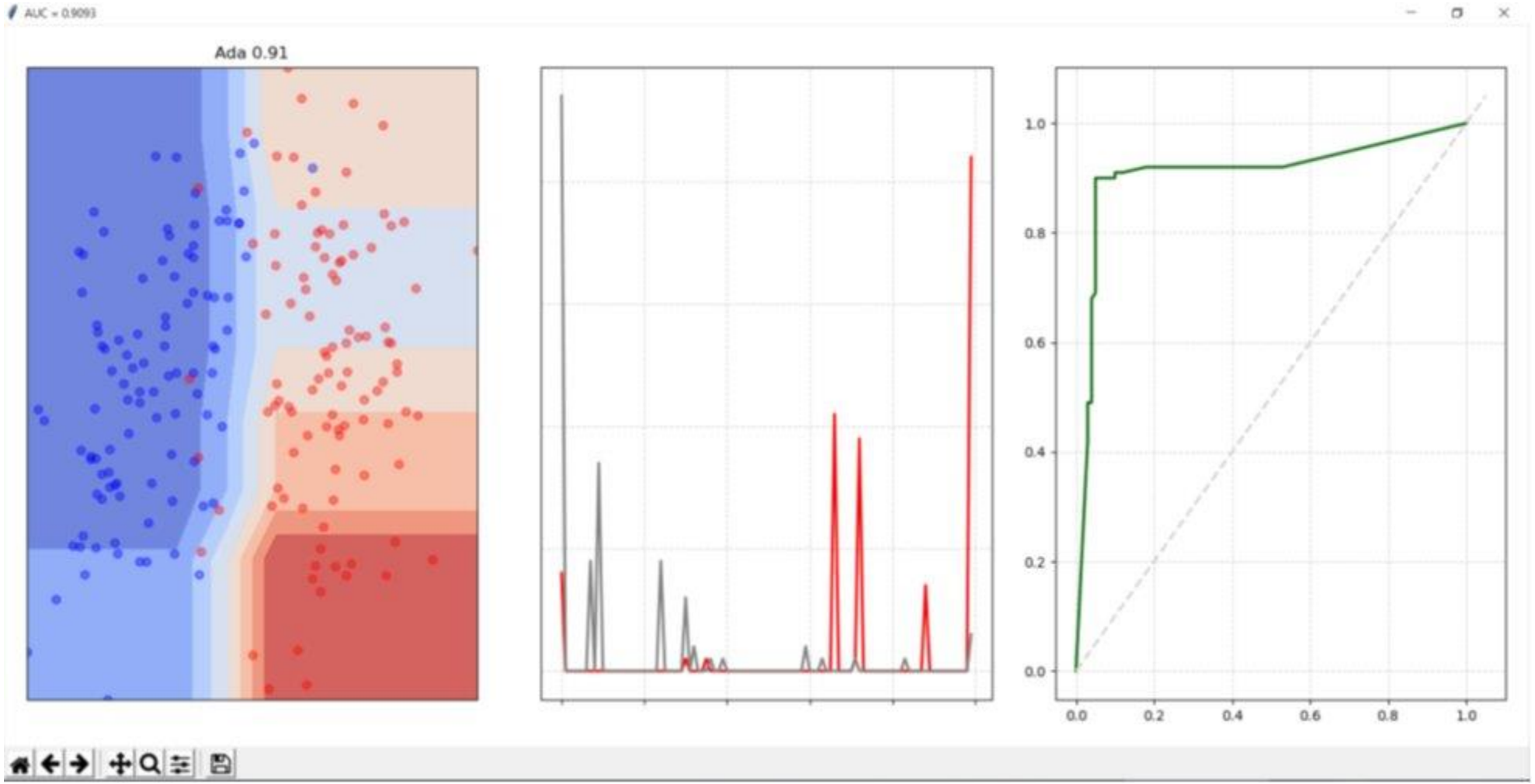
- Regression
- Naive Bayes
- Gaussian
- SVM
- Nearest Neighbors
- Decision Tree
- Random Forest
- AdaBoost
- xgboost
- Neural Net

Benchmarking the Classifiers

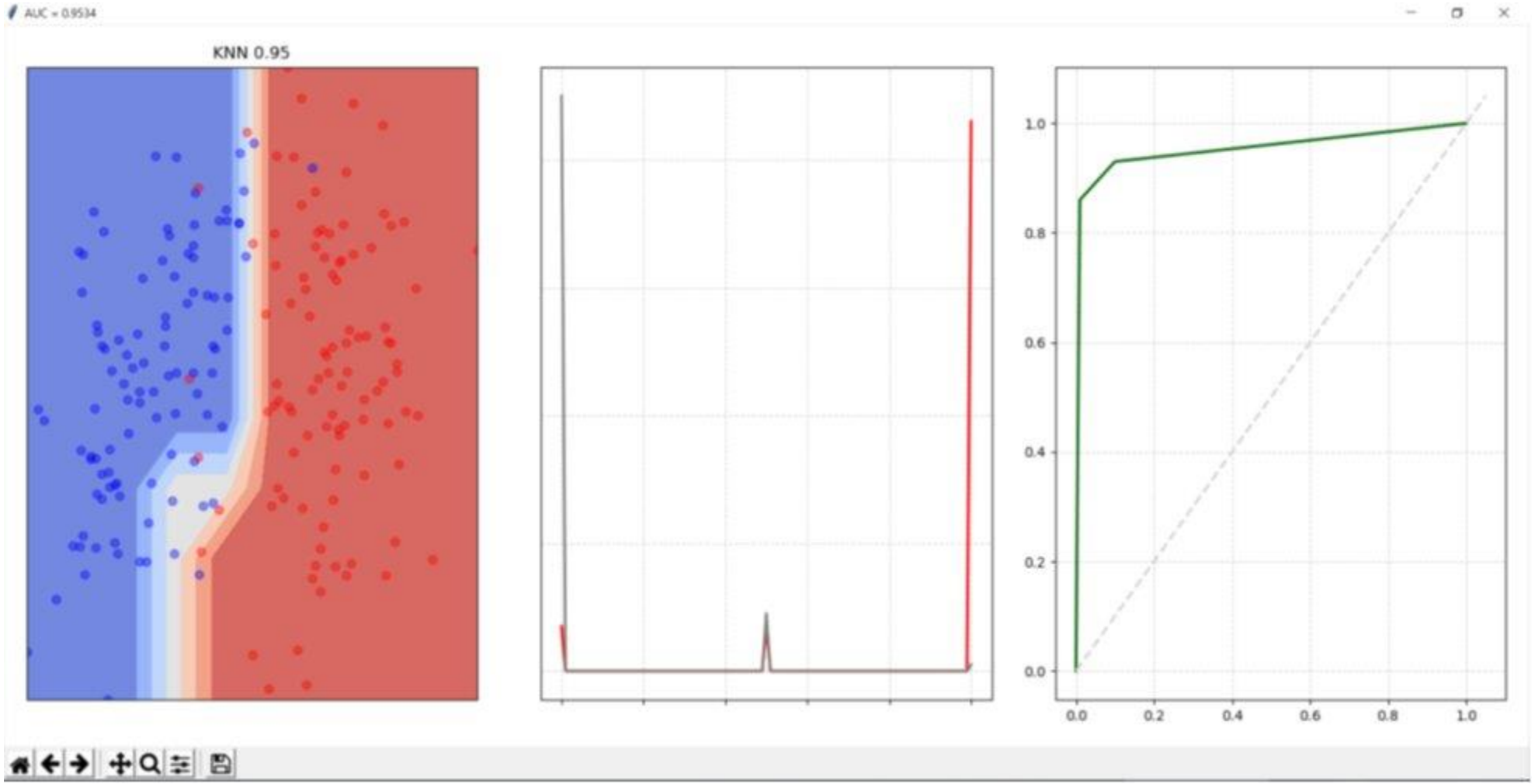
Gaussian distribution



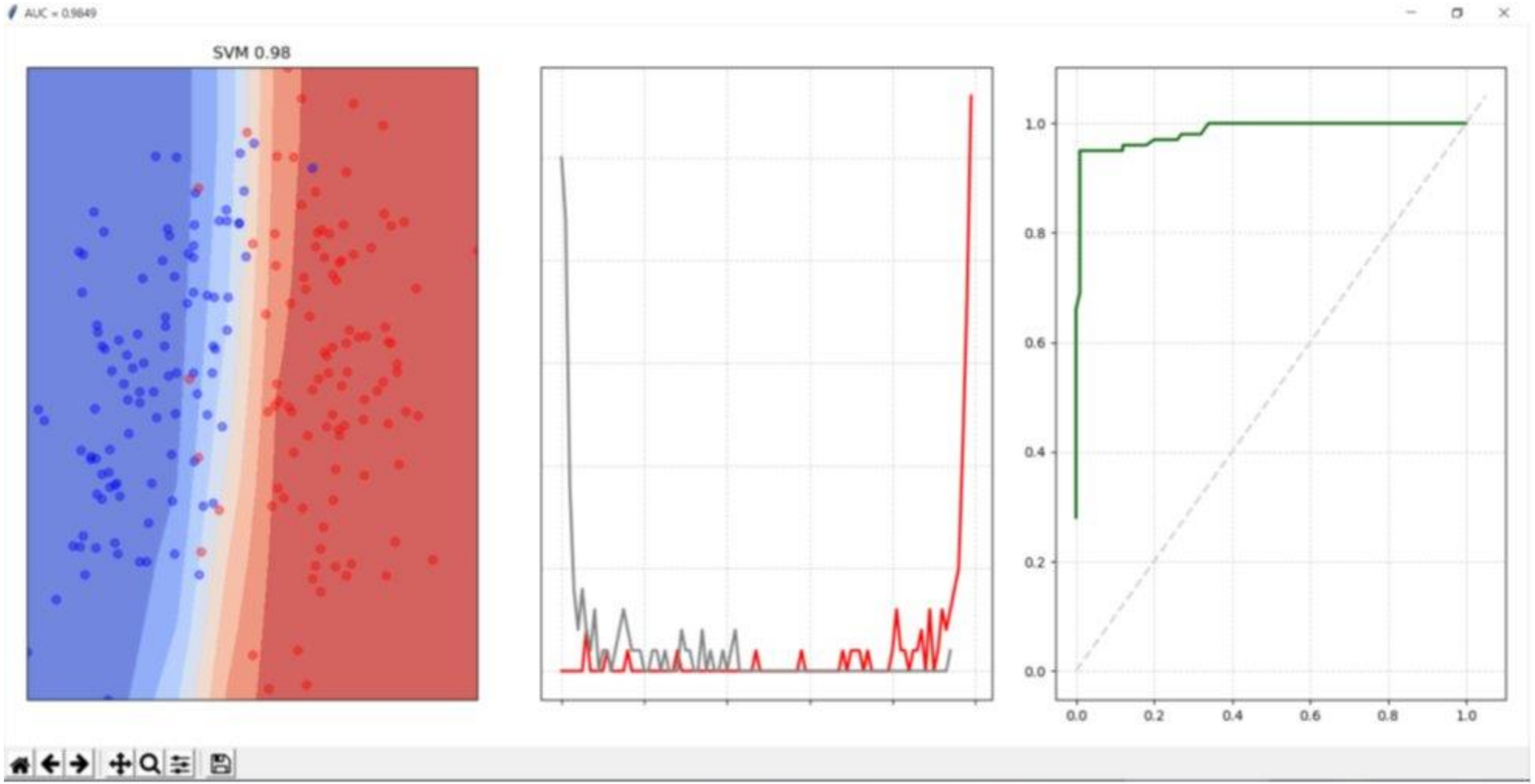
Benchmarking the Classifiers



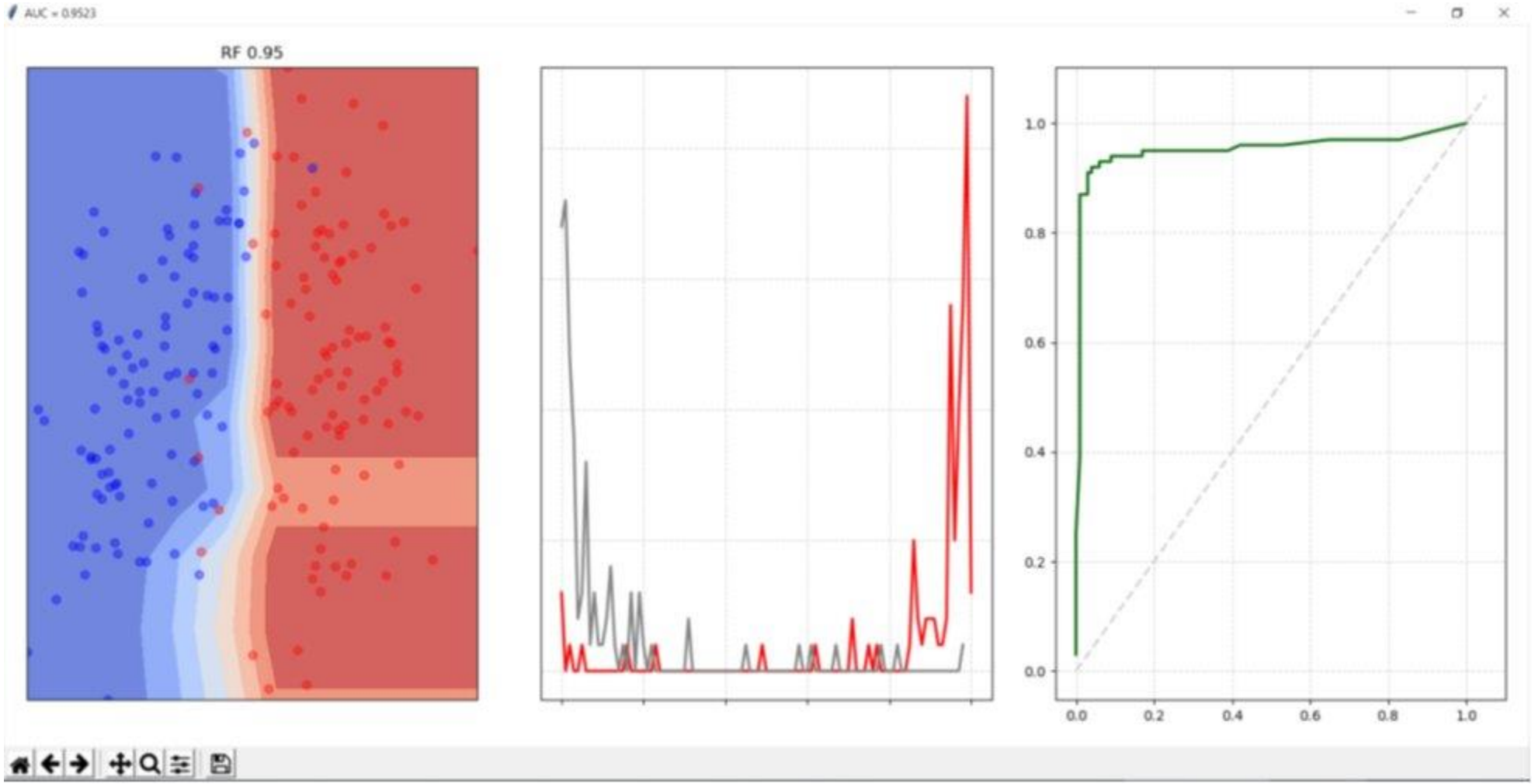
Benchmarking the Classifiers



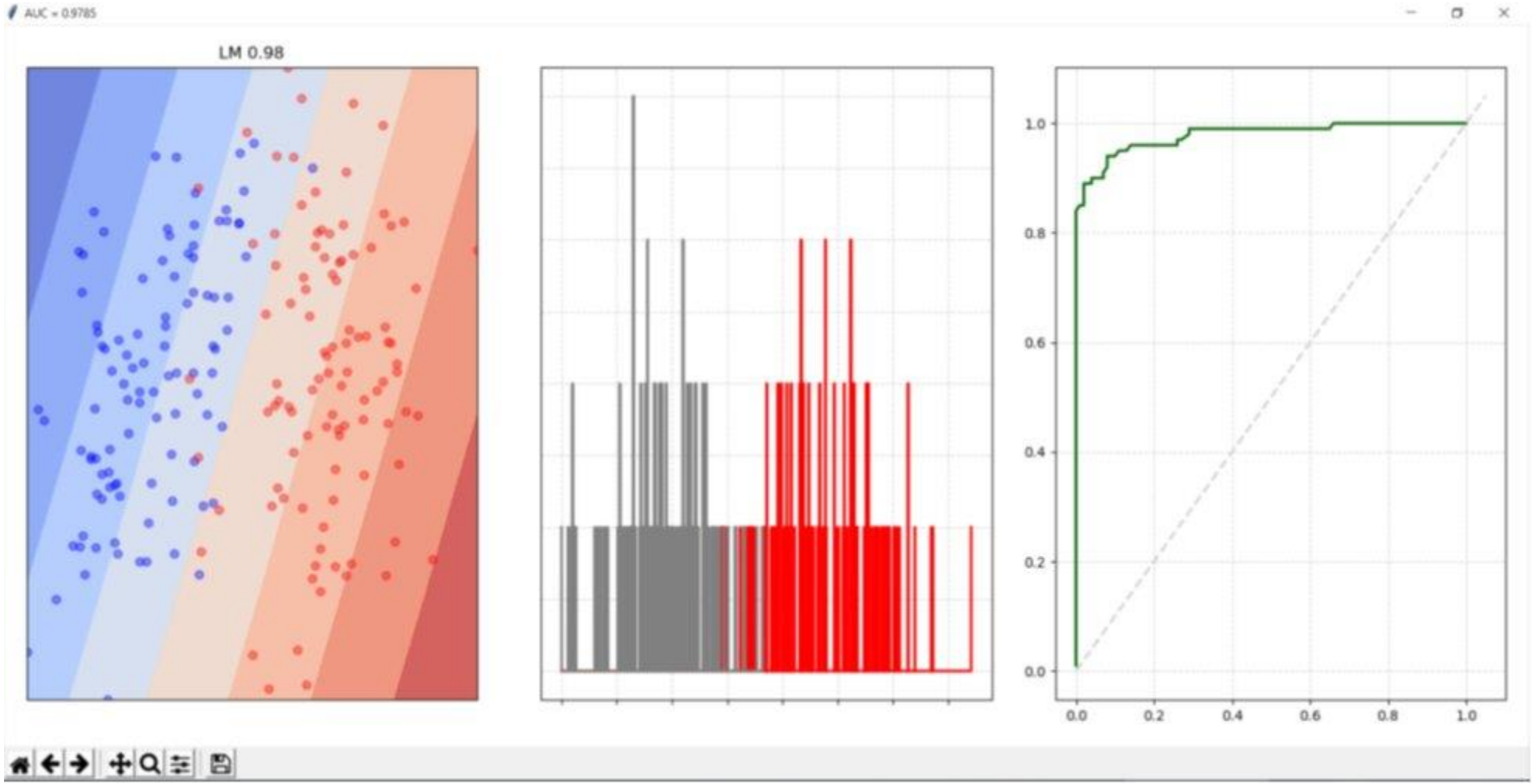
Benchmarking the Classifiers



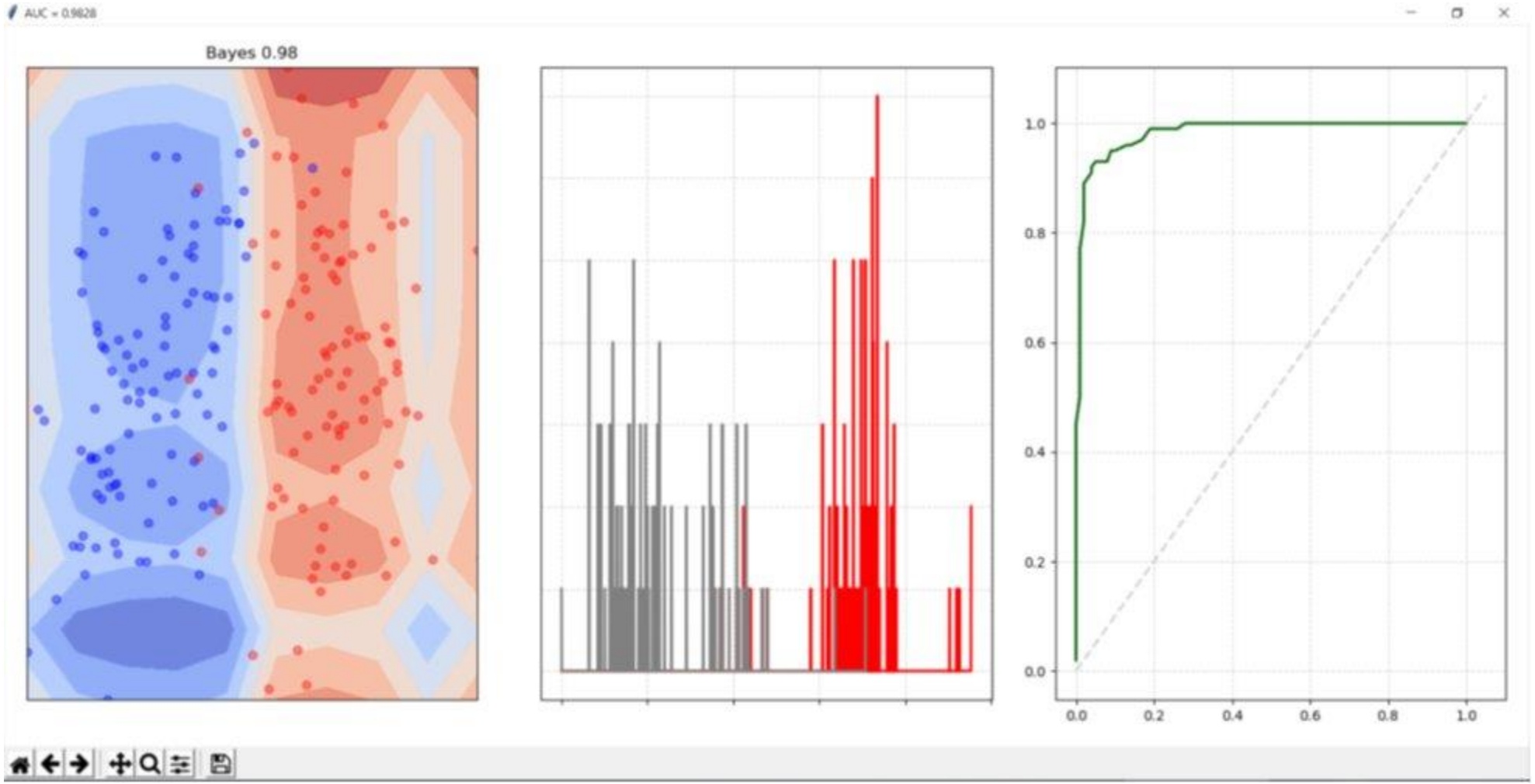
Benchmarking the Classifiers



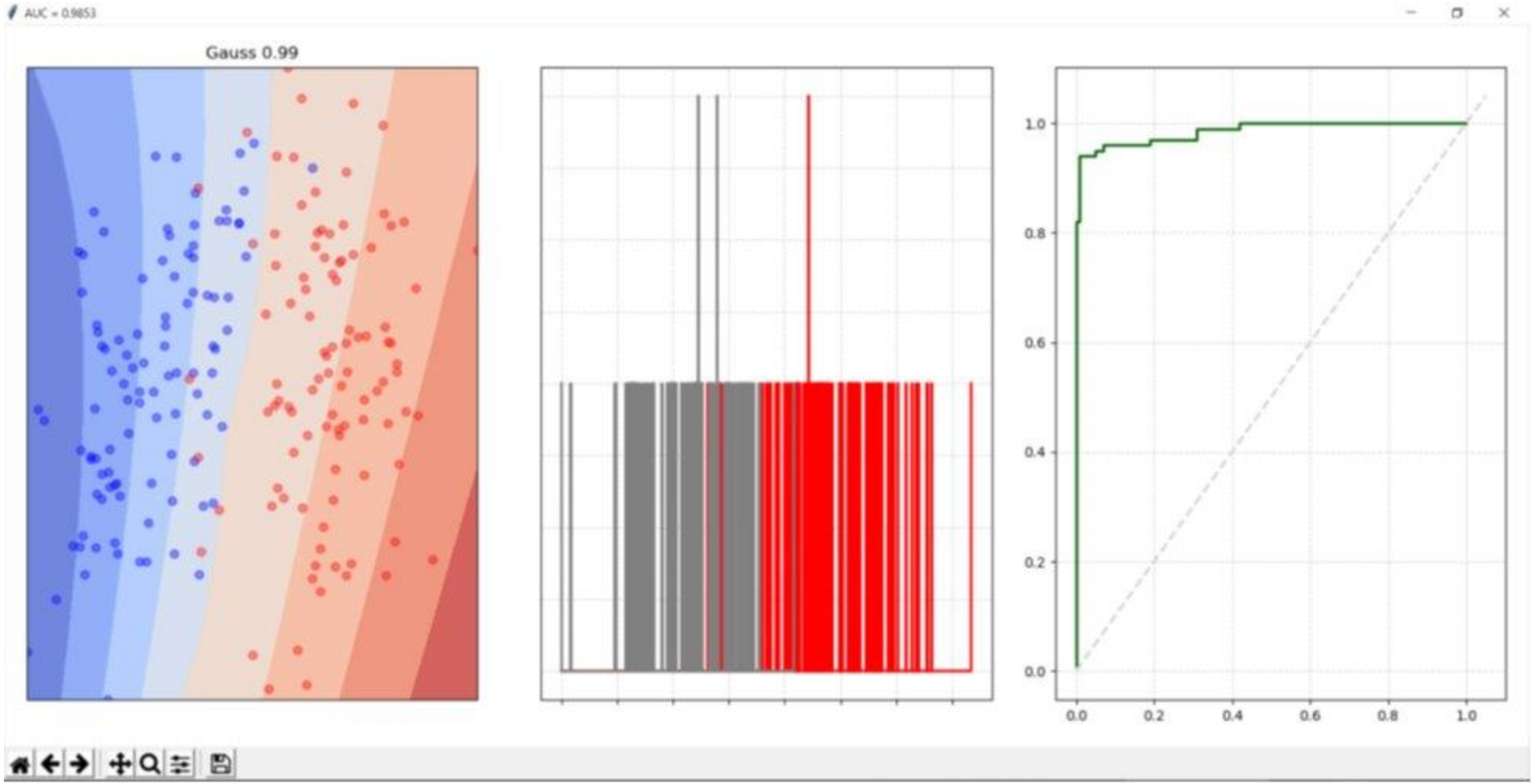
Benchmarking the Classifiers



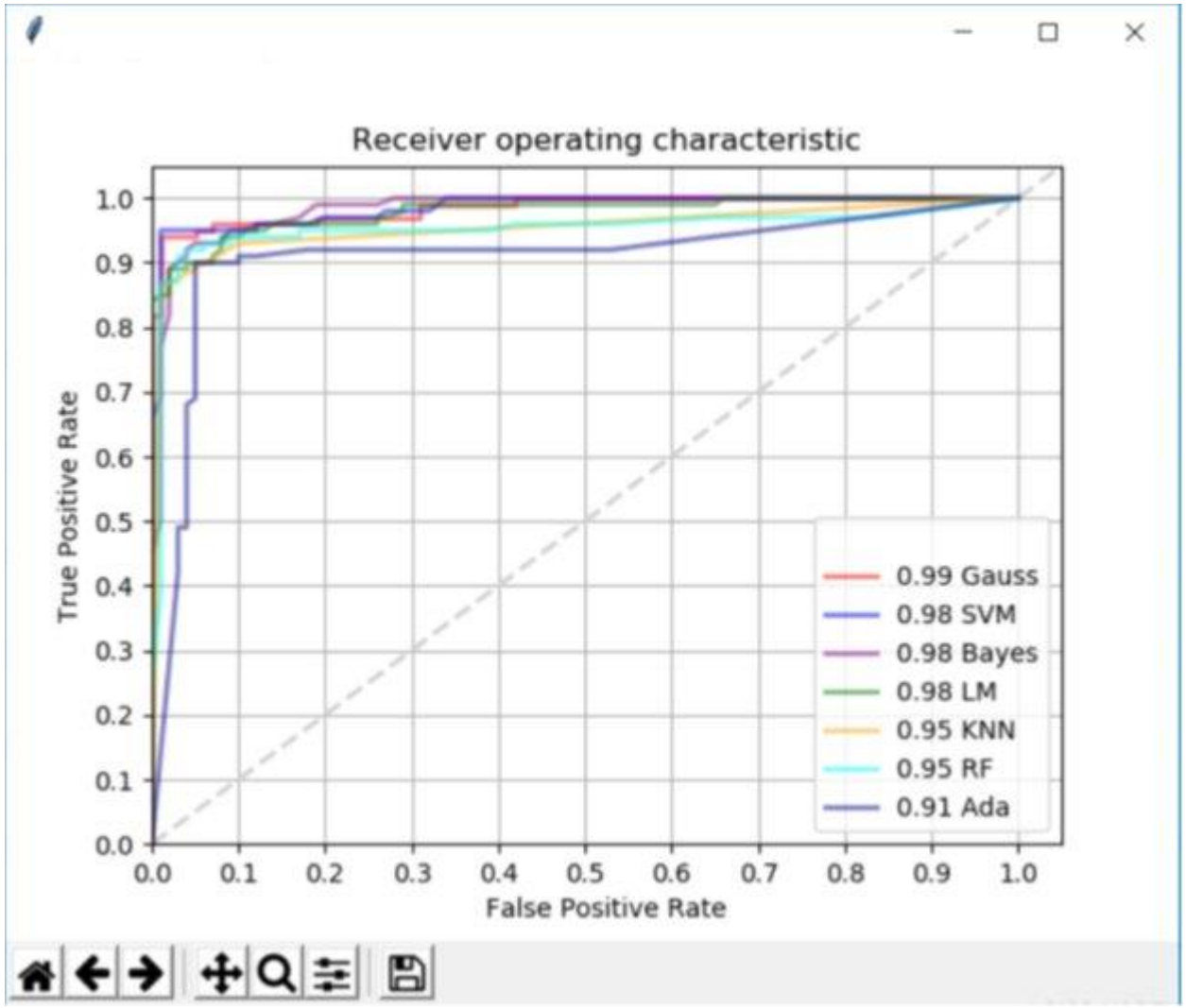
Benchmarking the Classifiers



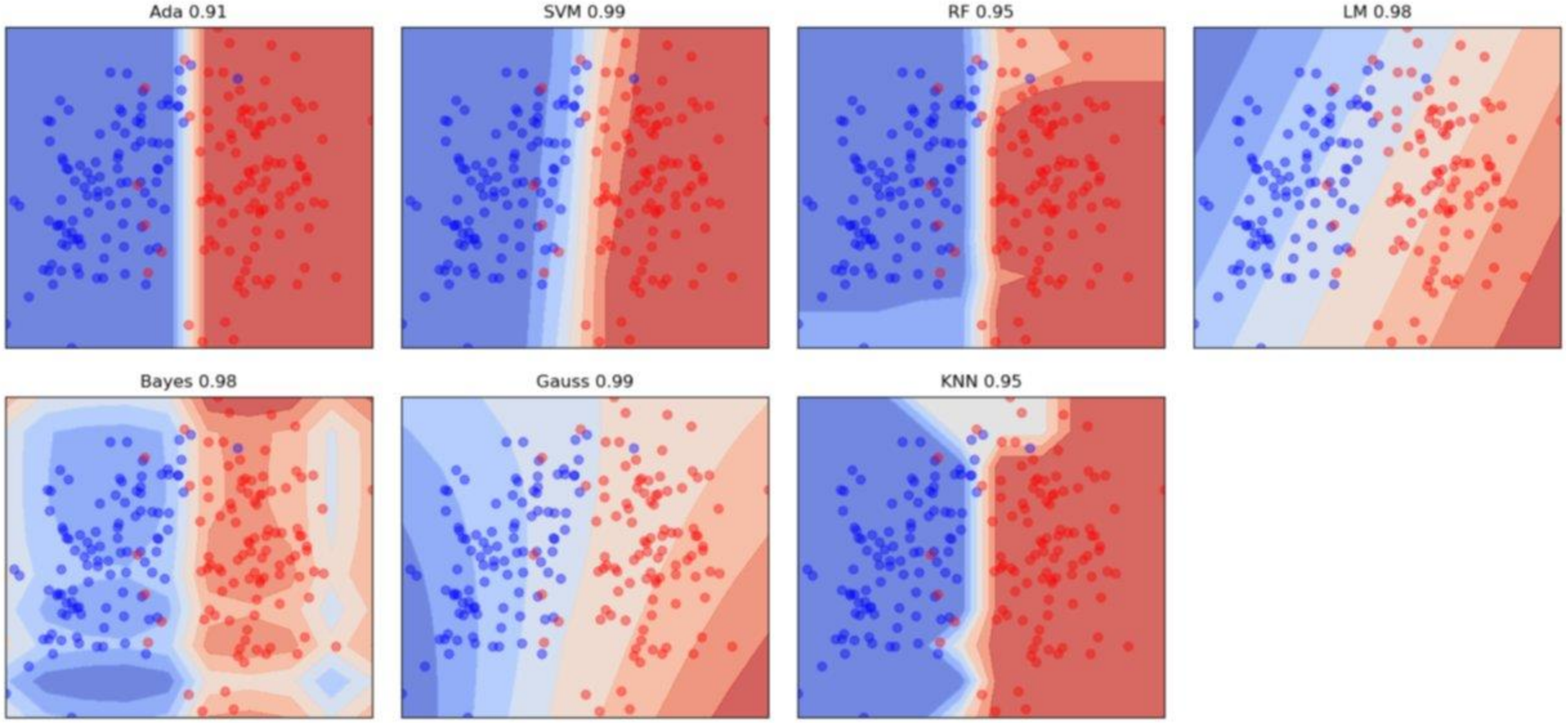
Benchmarking the Classifiers



Benchmarking the Classifiers

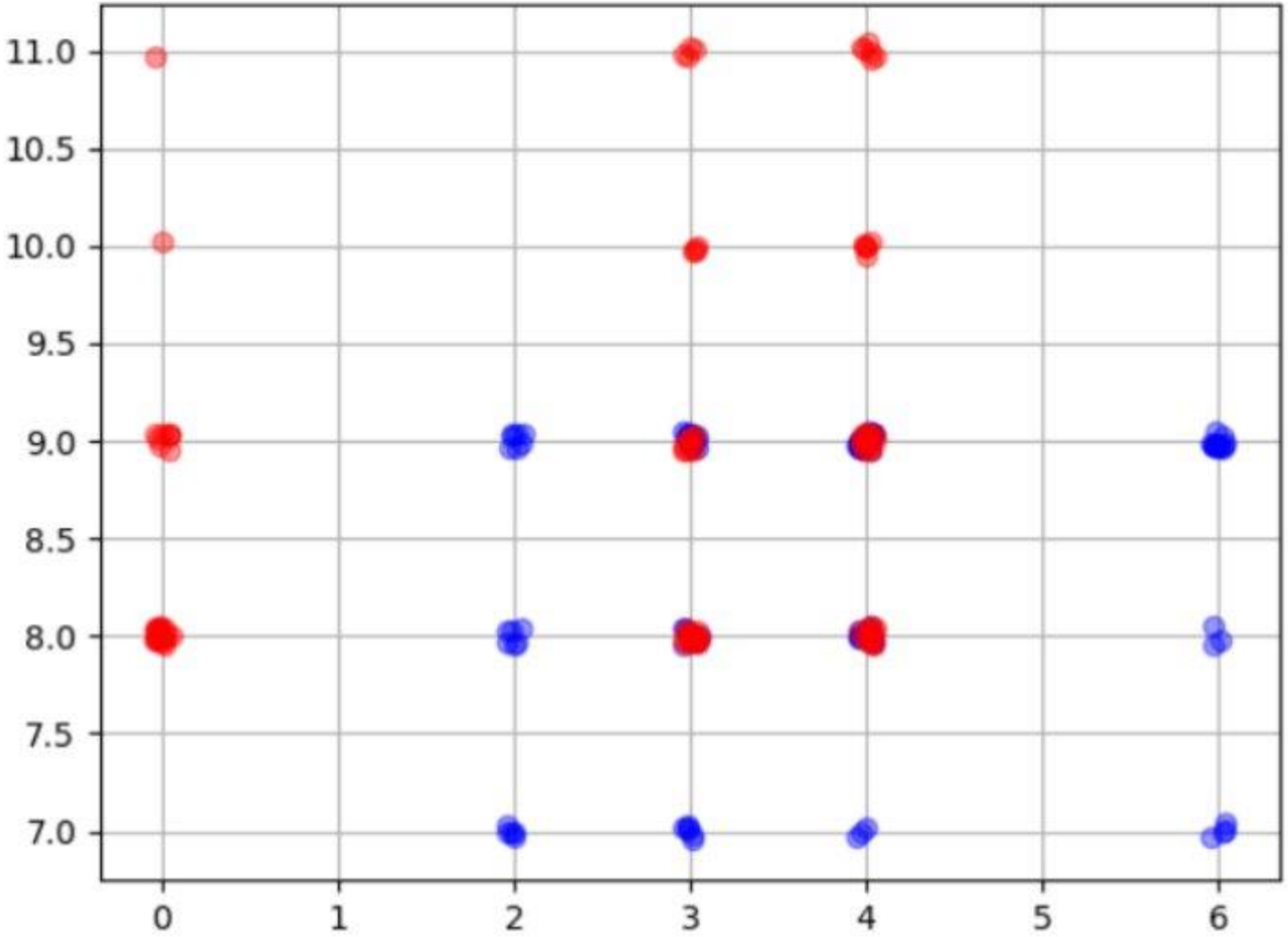


Benchmarking the Classifiers

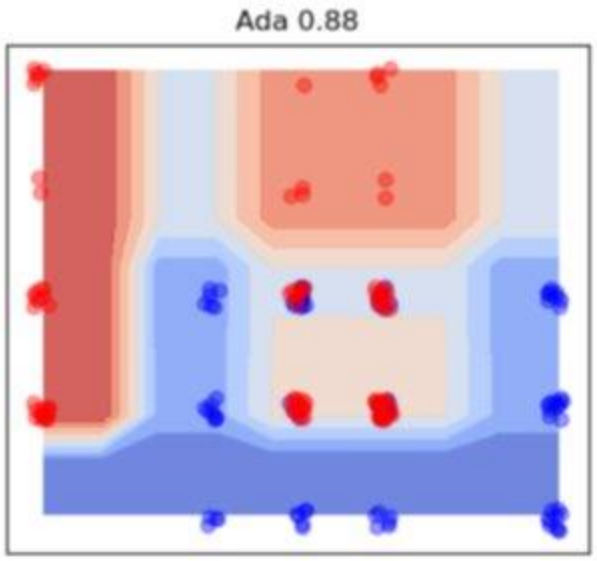


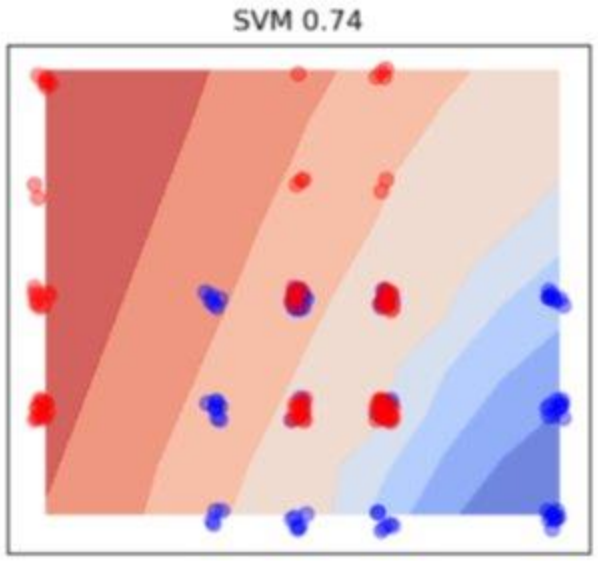
Benchmarking the Classifiers

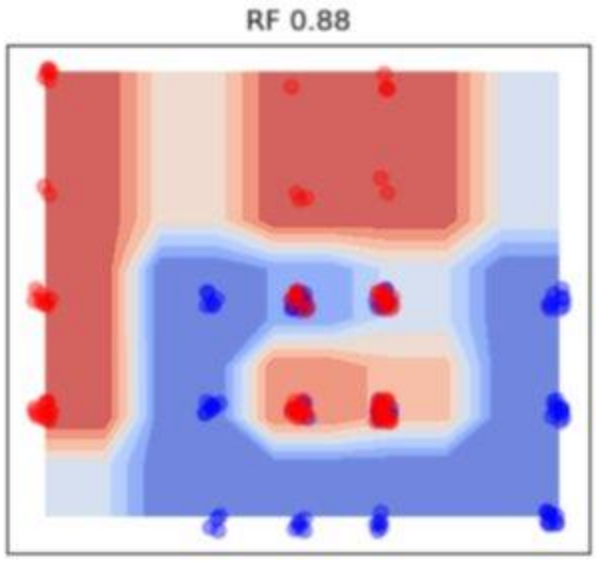
Non-gaussian distribution

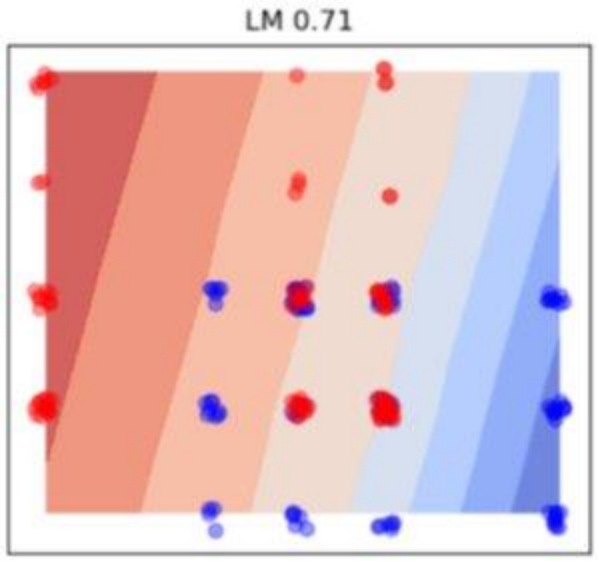


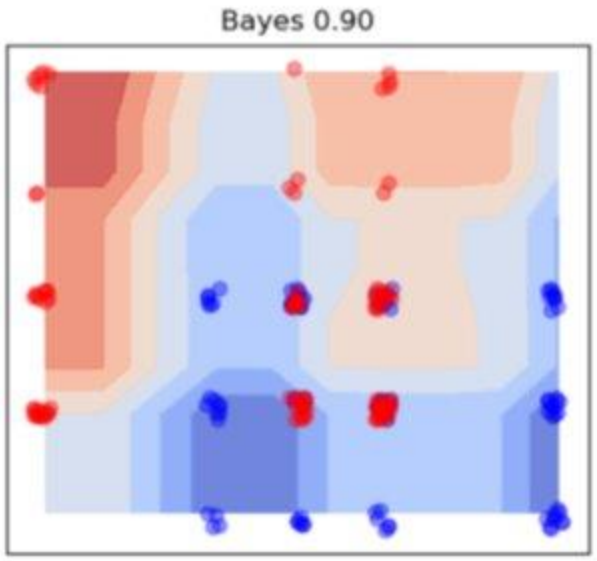
Benchmarking the Classifiers

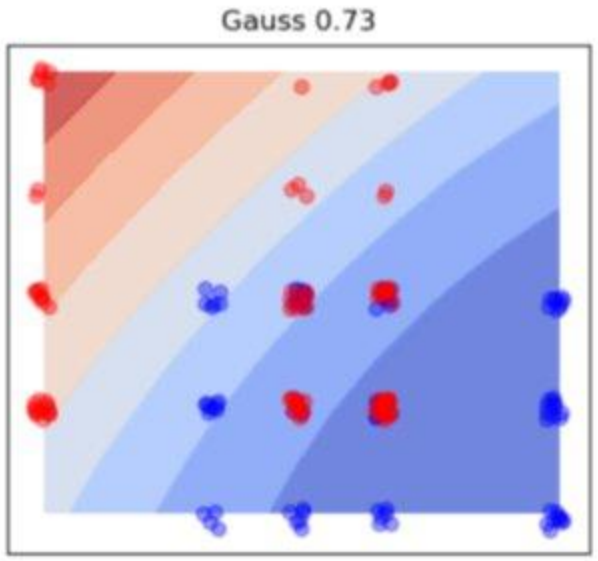


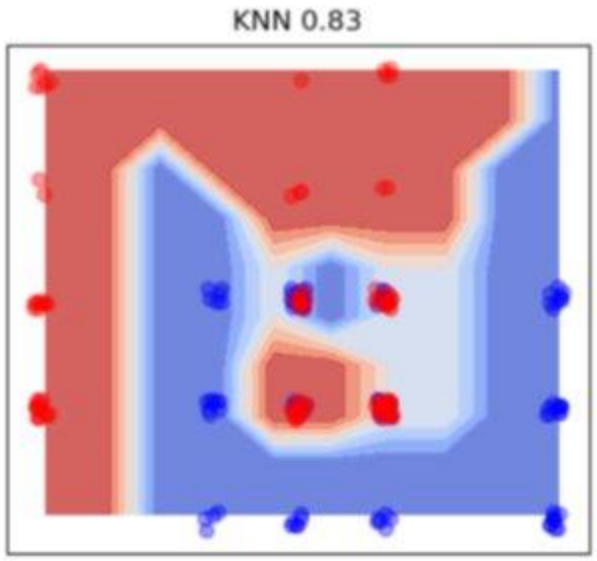




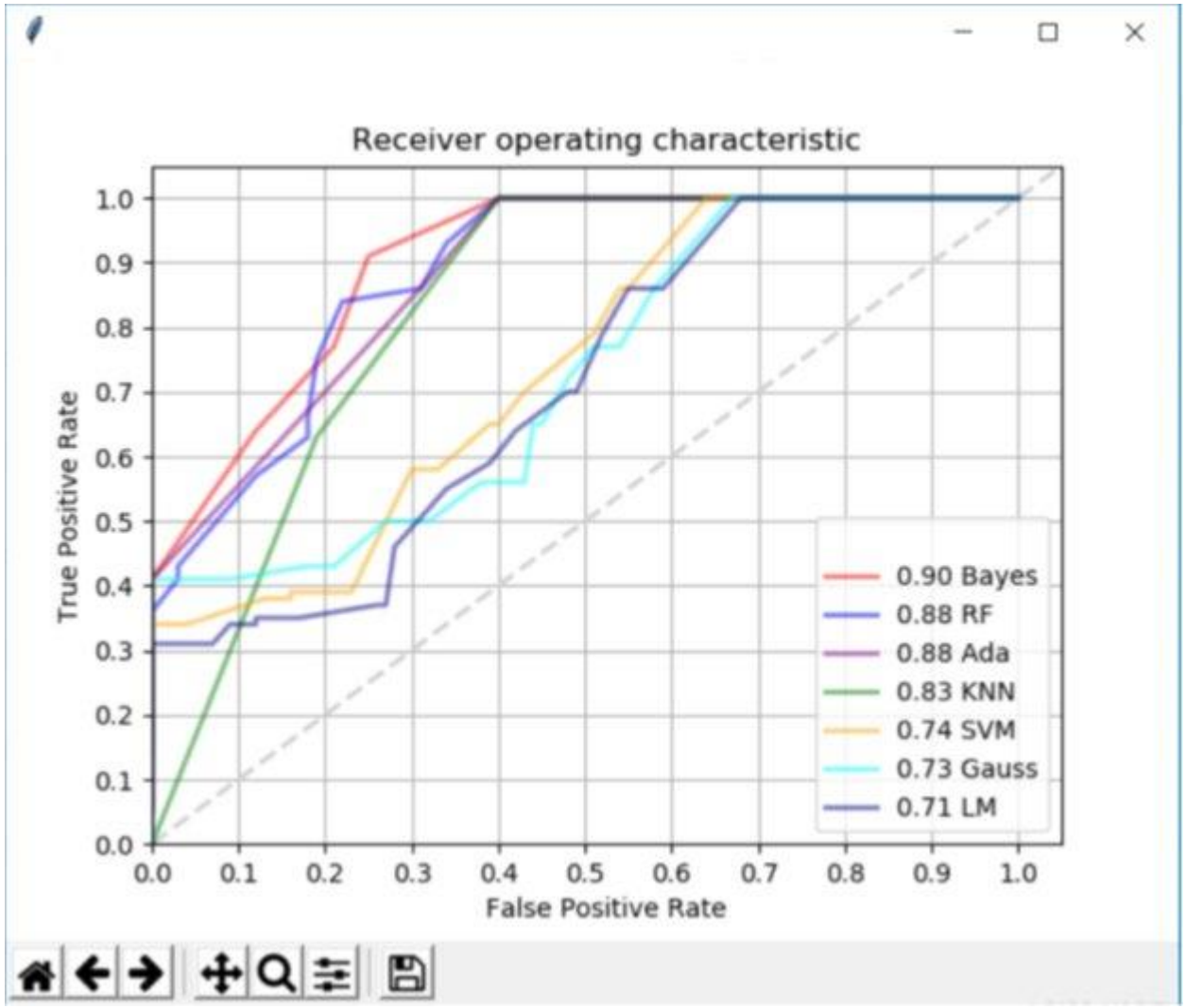








Benchmarking the Classifiers



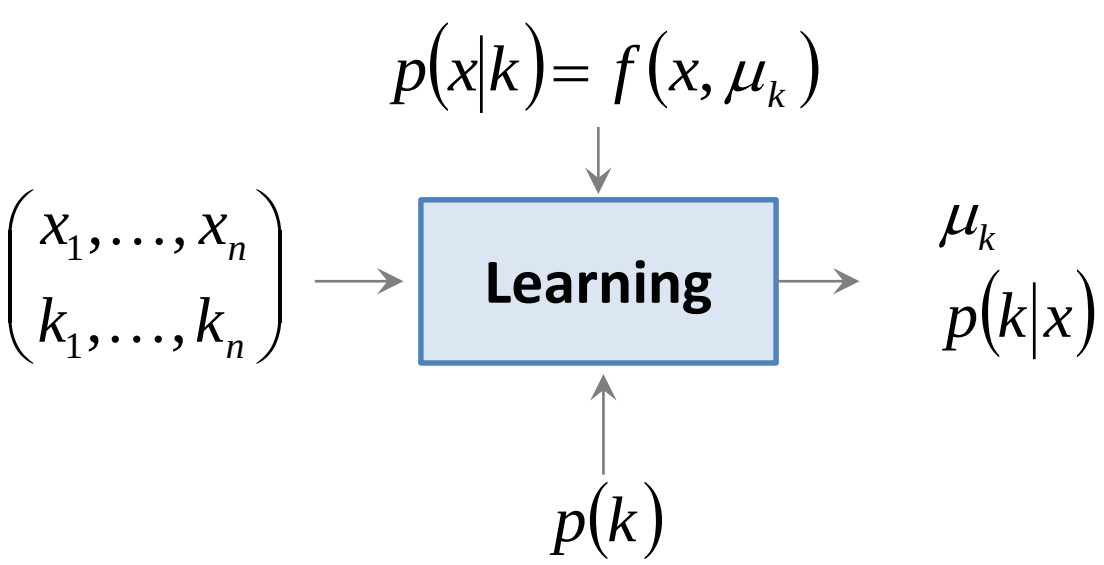
Part 4: Other Approaches in ML

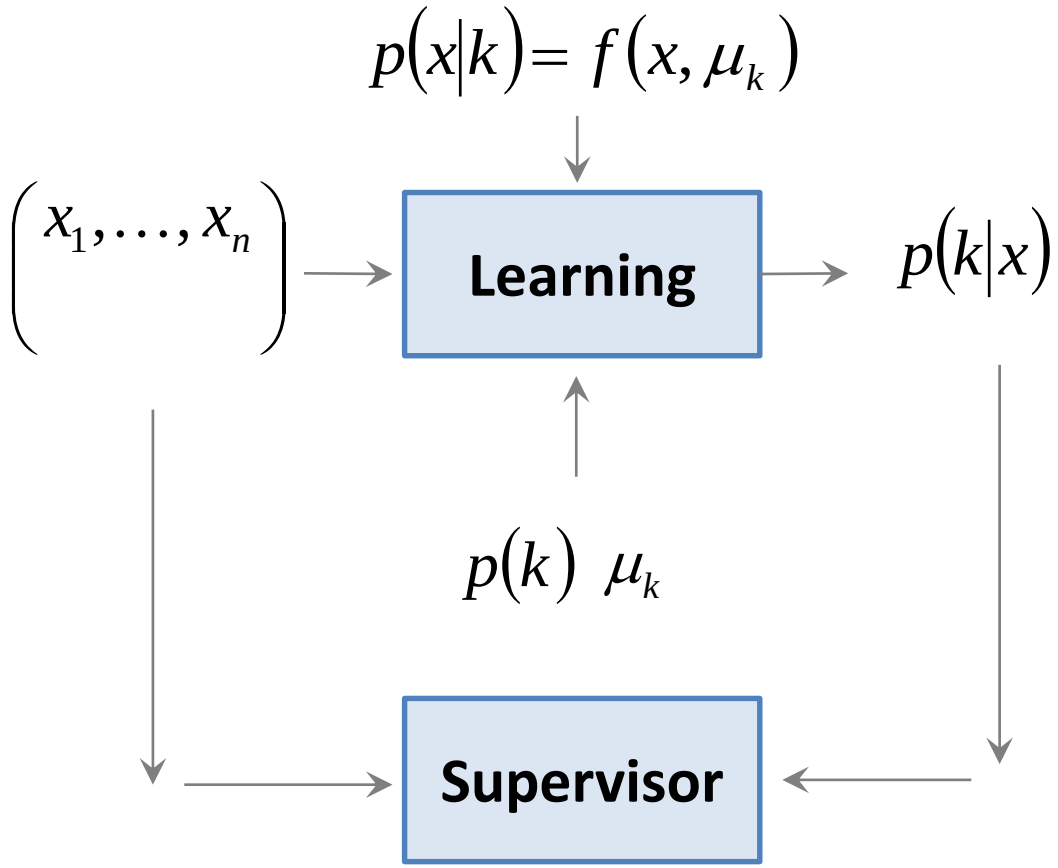
Unsupervised learning
Non-Bayesian approach
Time Series

Unsupervised learning

Unsupervised learning

EM algorithm





EM algorithm

EM algorithm

Unsupervised learning

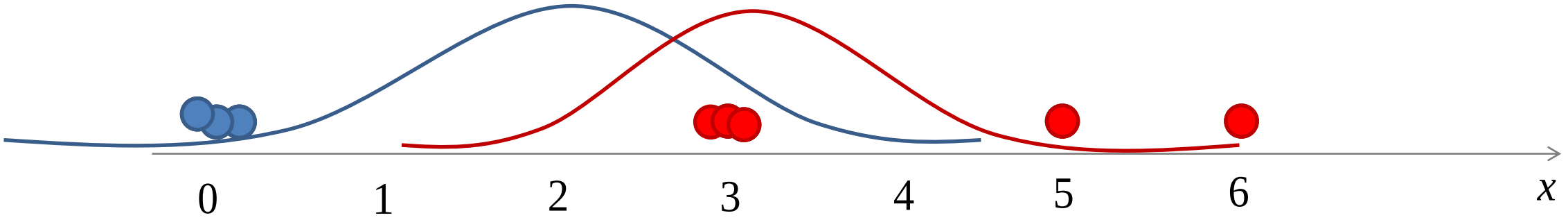
EM algorithm

$$\mu_{\bullet} = \tfrac{1}{4}(0+0+3+5) = 2 \qquad \mu_{\bullet} = \tfrac{1}{4}(0+3+3+6) = 3$$

$$\sigma_{\bullet} = \tfrac{1}{4}\left(2^2+2^2+1^2+3^2\right) = 4.5 \qquad \sigma_{\bullet} = \tfrac{1}{4}\left(3^2+0^2+0^2+3^2\right) = 4.5$$

Unsupervised learning

EM algorithm

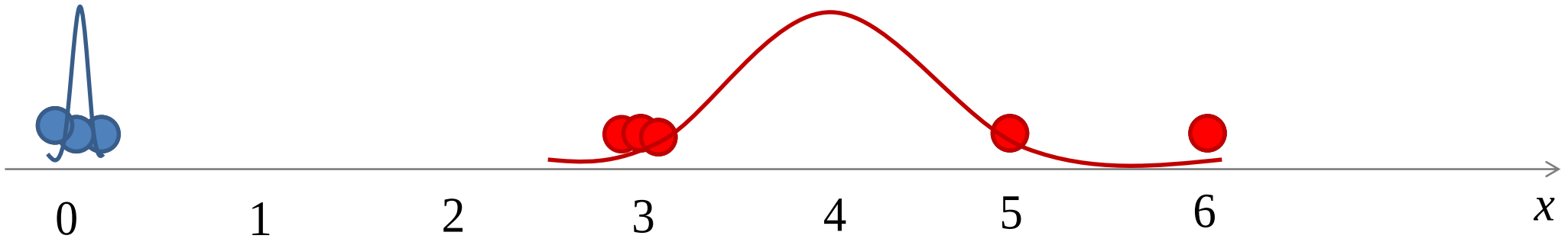


$$\mu_{\bullet} = \tfrac{1}{4}(0+0+3+5) = 2 \qquad \mu_{\bullet} = \tfrac{1}{4}(0+3+3+6) = 3$$

$$\sigma_{\bullet} = \tfrac{1}{4}\left(2^2+2^2+1^2+3^2\right) = 4.5 \qquad \sigma_{\bullet} = \tfrac{1}{4}\left(3^2+0^2+0^2+3^2\right) = 4.5$$

Unsupervised learning

EM algorithm



$$\mu_{\bullet} = \tfrac{1}{3}(0+0+0) = 0 \qquad \mu_{\bullet} = \tfrac{1}{2}(3+3+3+5+6) = 4$$

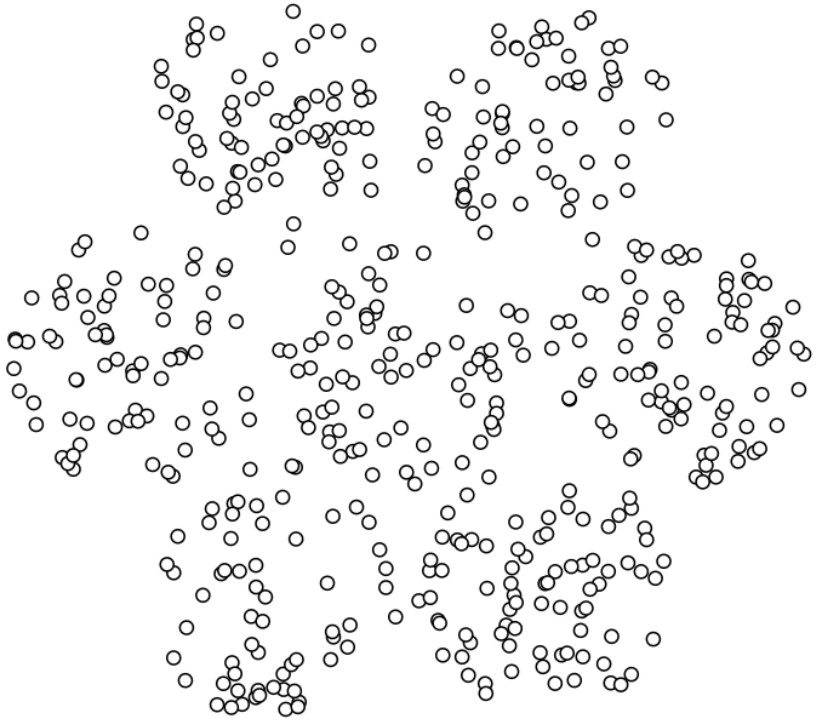
$$\sigma_{\bullet} = \tfrac{1}{3}\left(0^2+0^2+0^2\right) = 0 \qquad \sigma_{\bullet} = \tfrac{1}{5}\left(1^2+1^2+1^2+1^2+2^2\right) = 1.6$$

Unsupervised learning

K-means

K-means is an unsupervised clustering algorithm

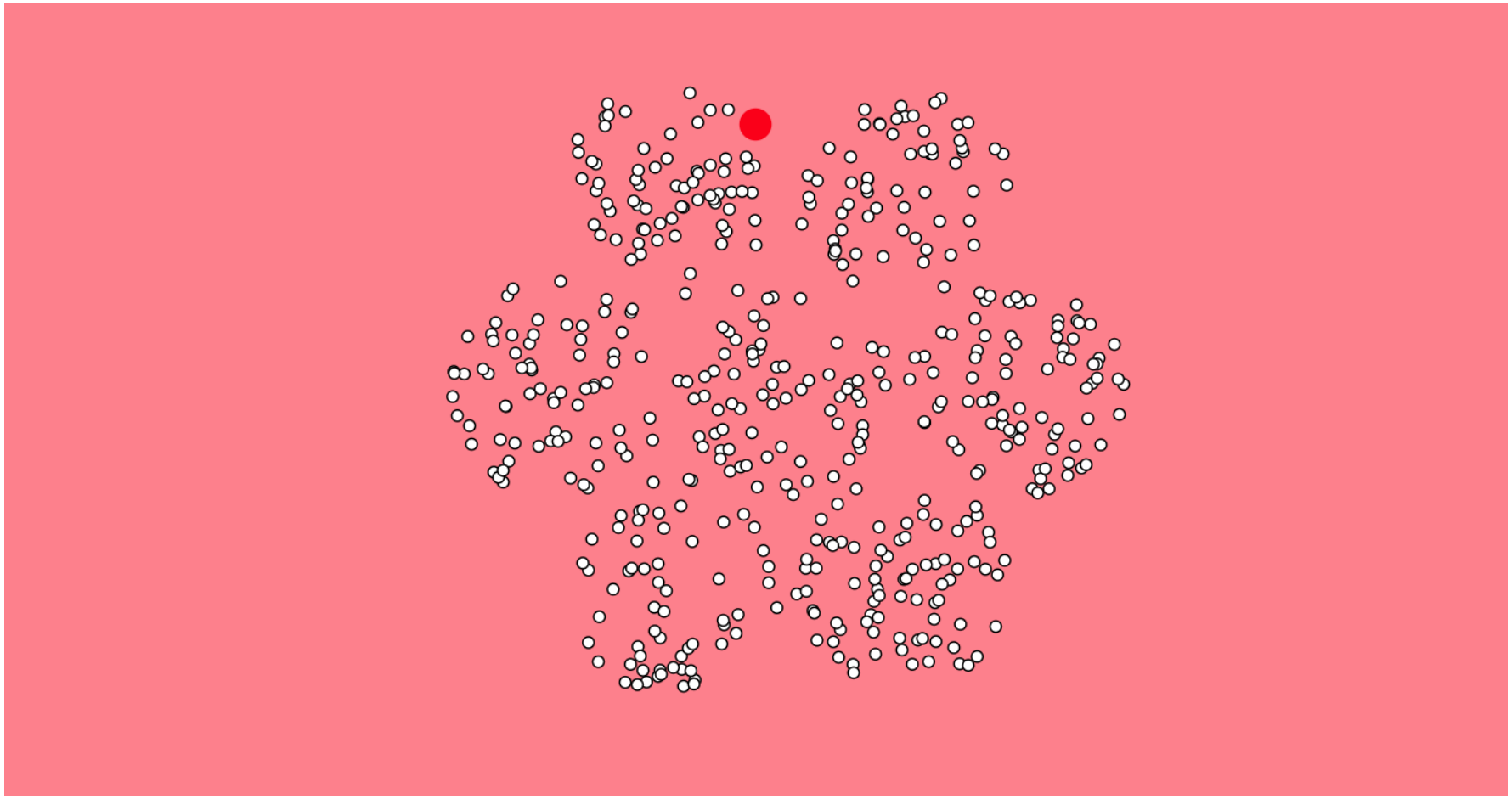
Do not confuse with KNN **classification** (or regression) algorithm which classifies an unlabeled observation based on its k surrounding neighbors.



https://www.naftaliharris.com/blog/visualizing-k-means-clustering/
http://stanford.edu/class/ee103/visualizations/kmeans/kmeans.html

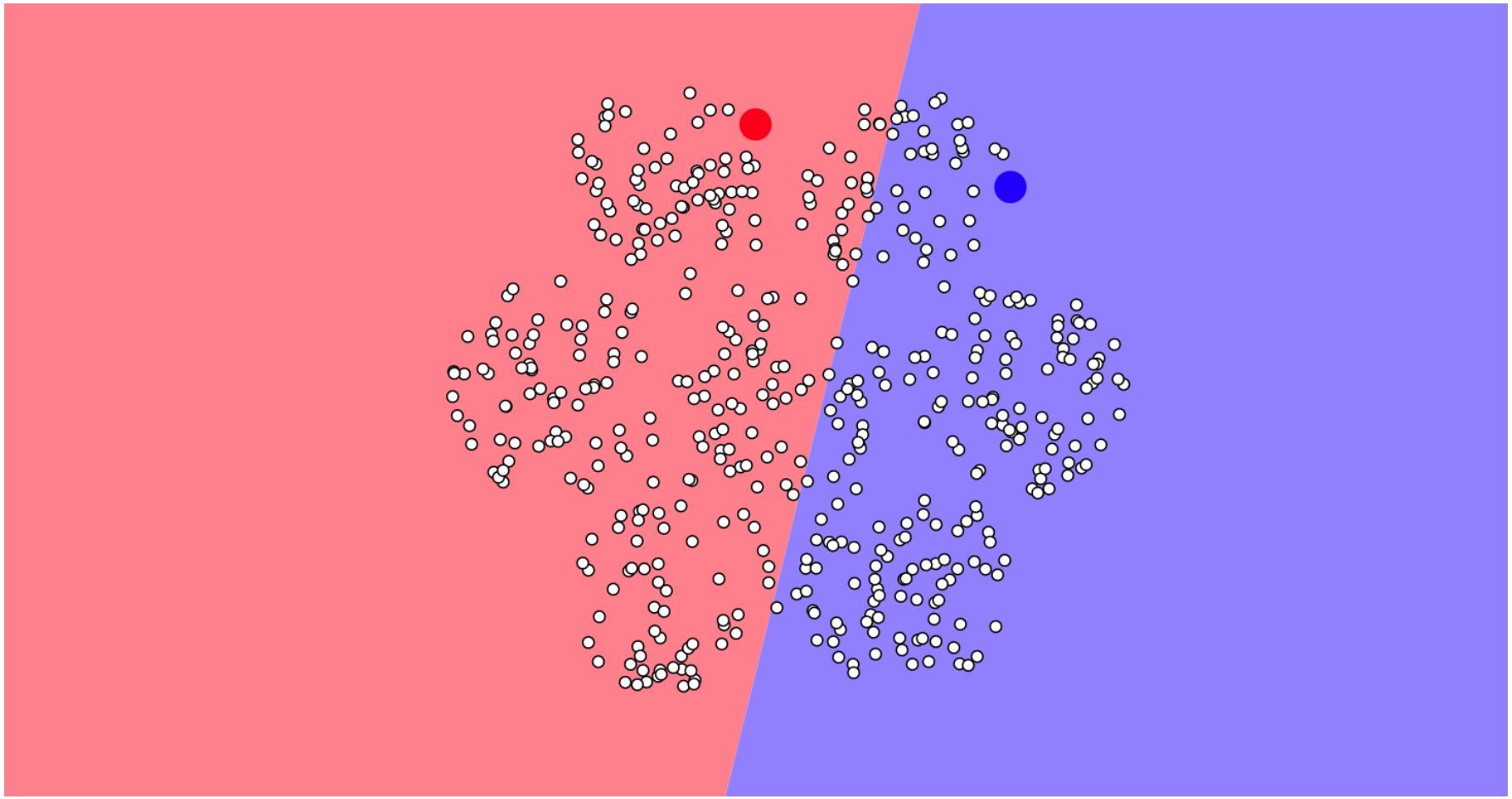
Unsupervised learning

K-means



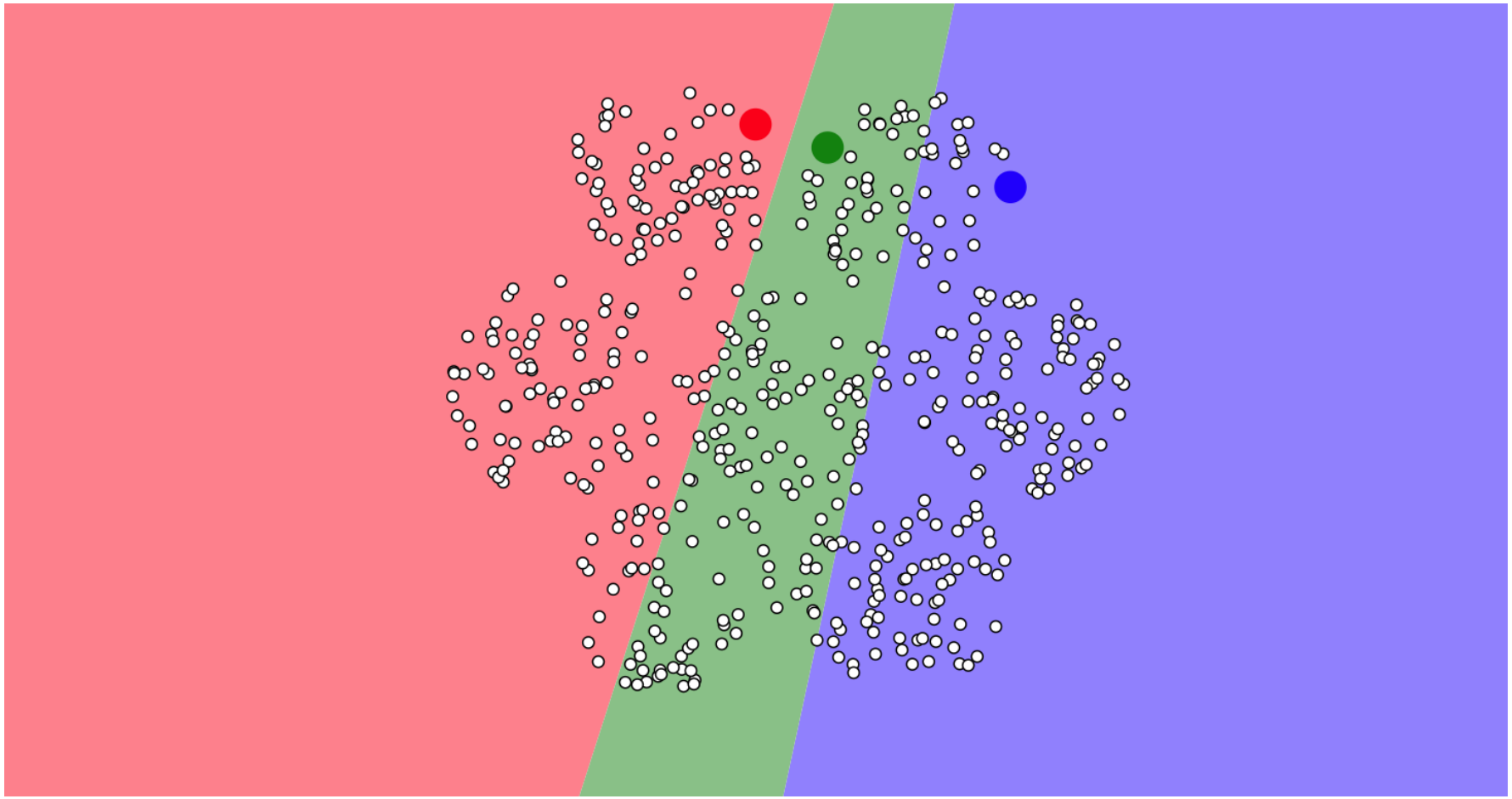
Unsupervised learning

K-means



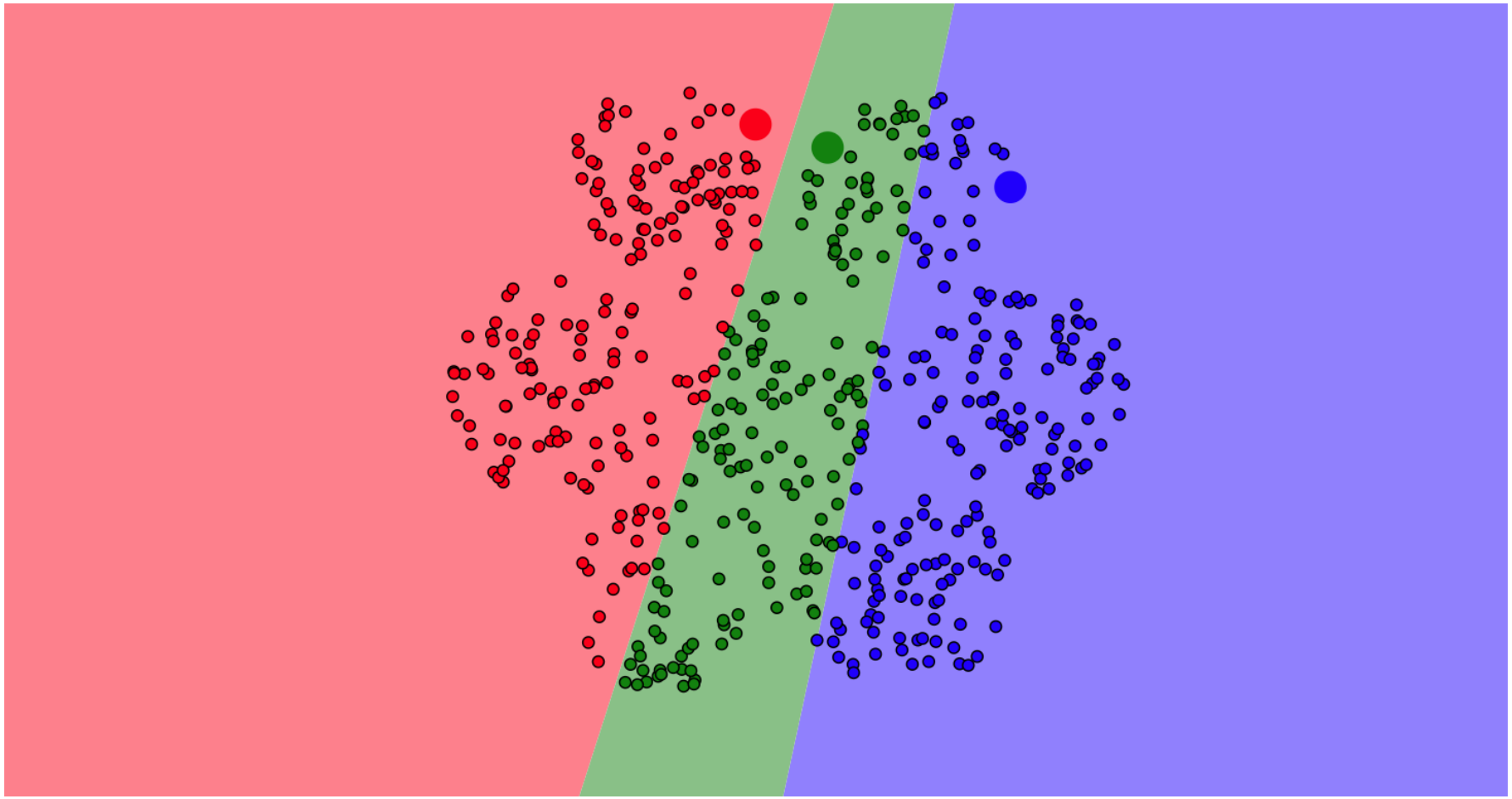
Unsupervised learning

K-means



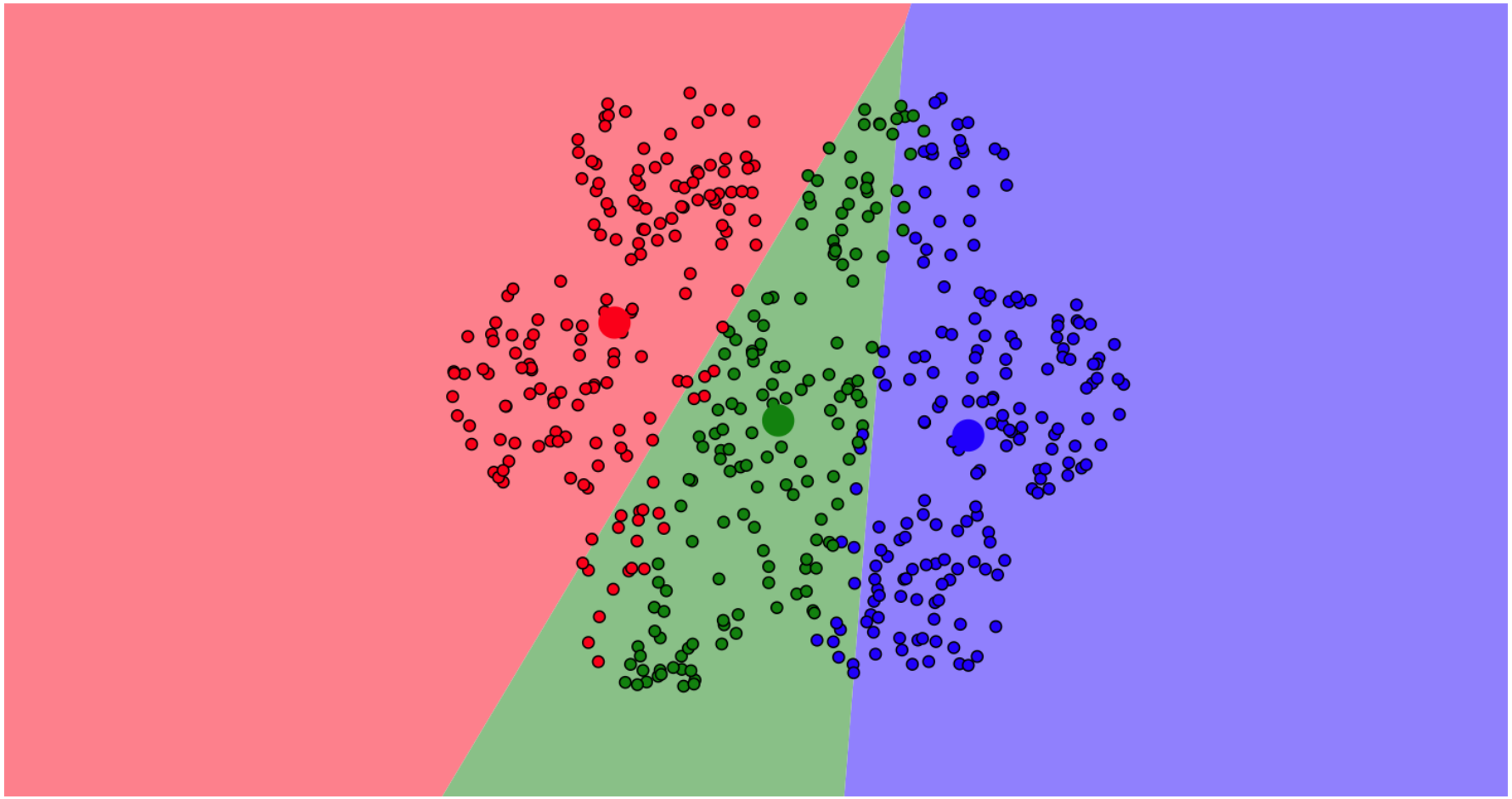
Unsupervised learning

K-means



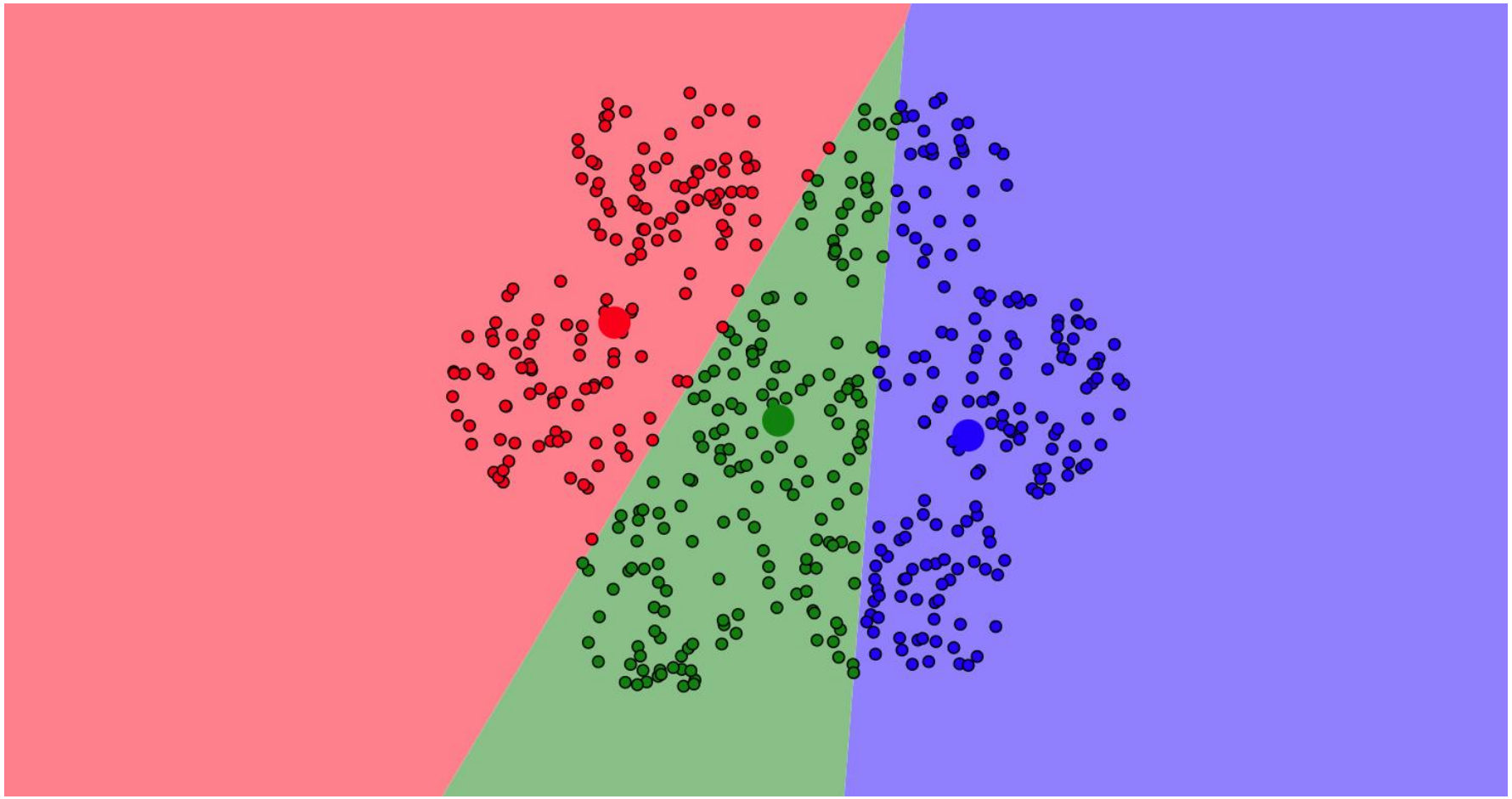
Unsupervised learning

K-means



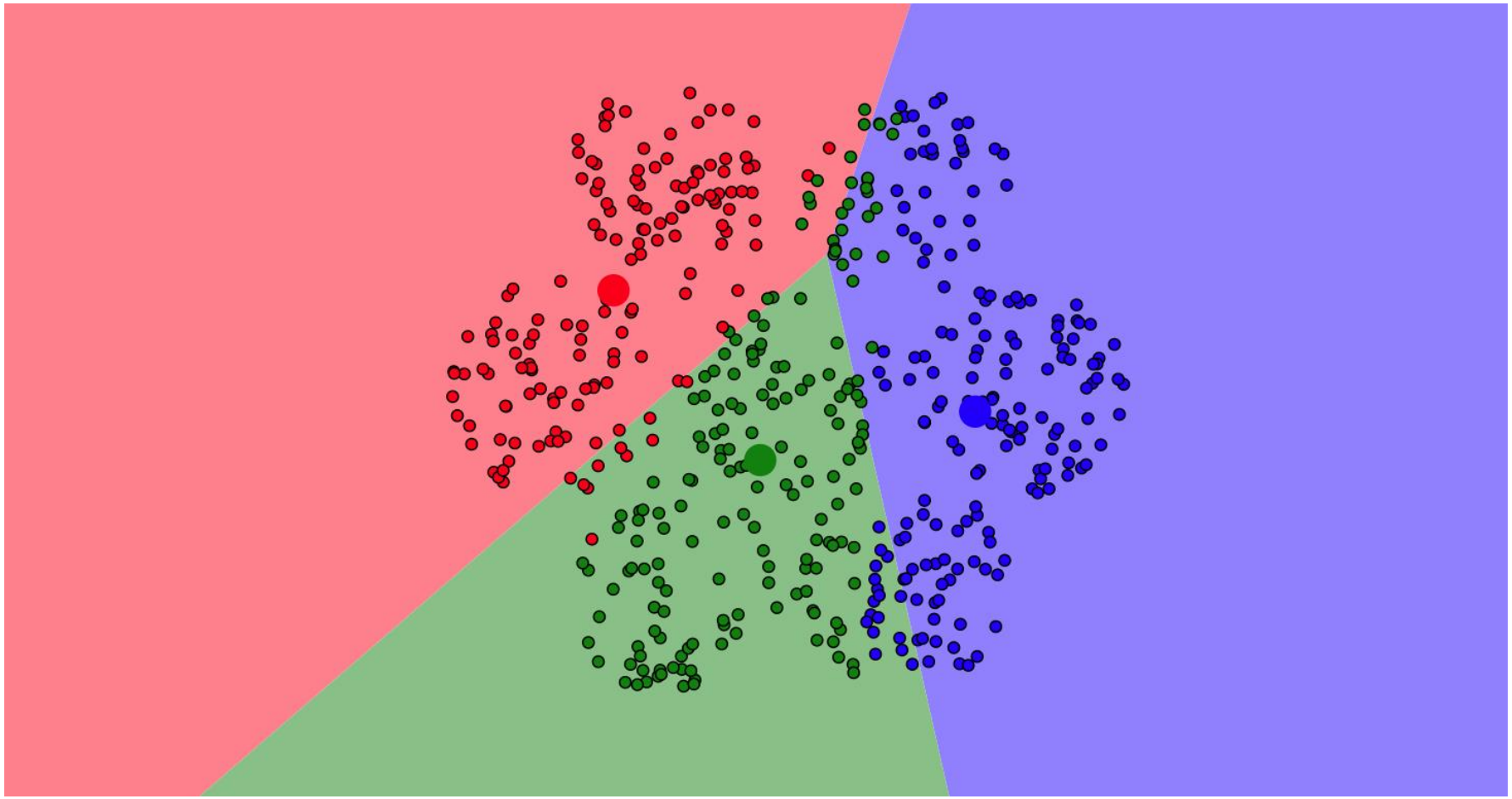
Unsupervised learning

K-means



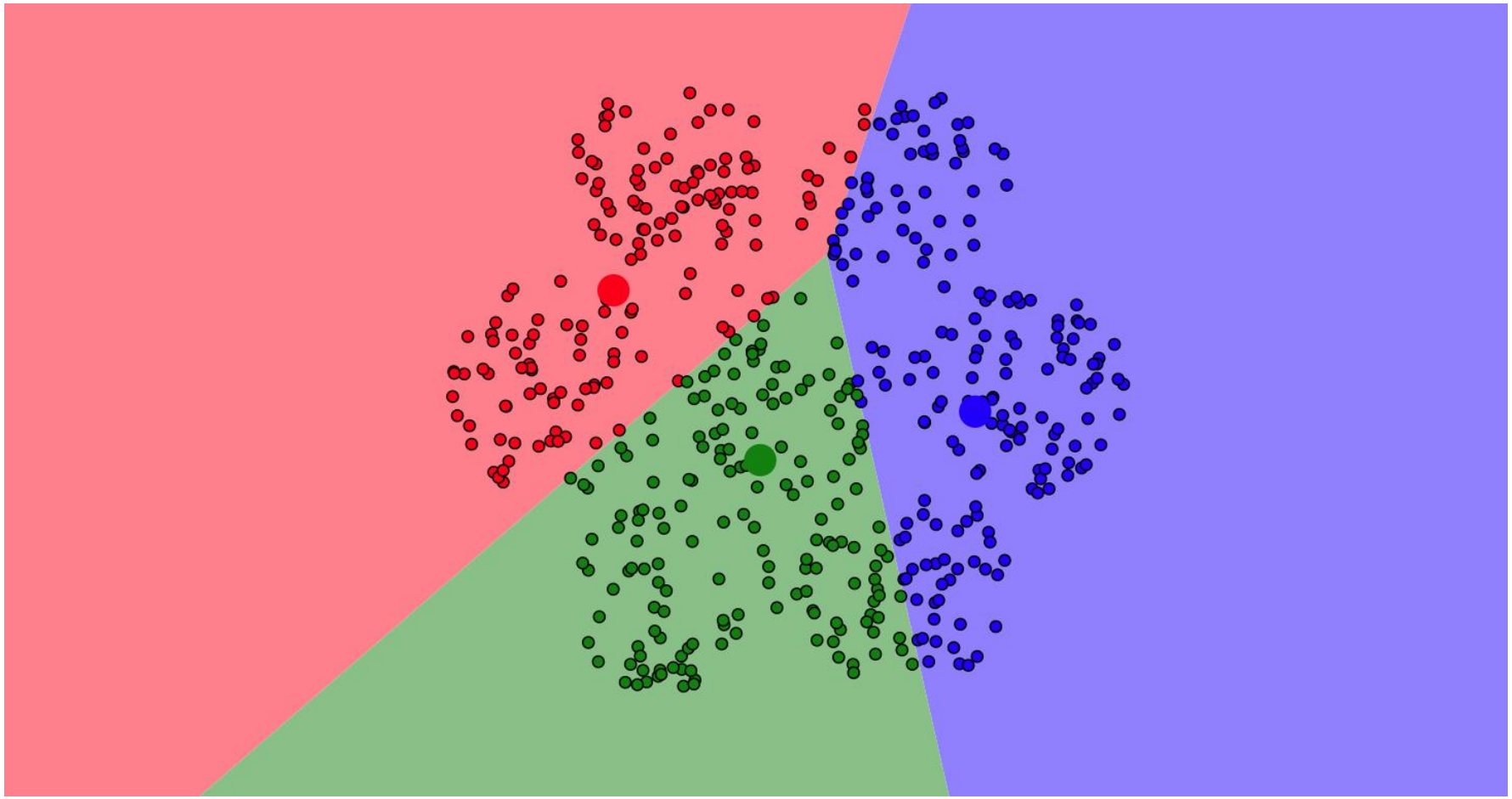
Unsupervised learning

K-means



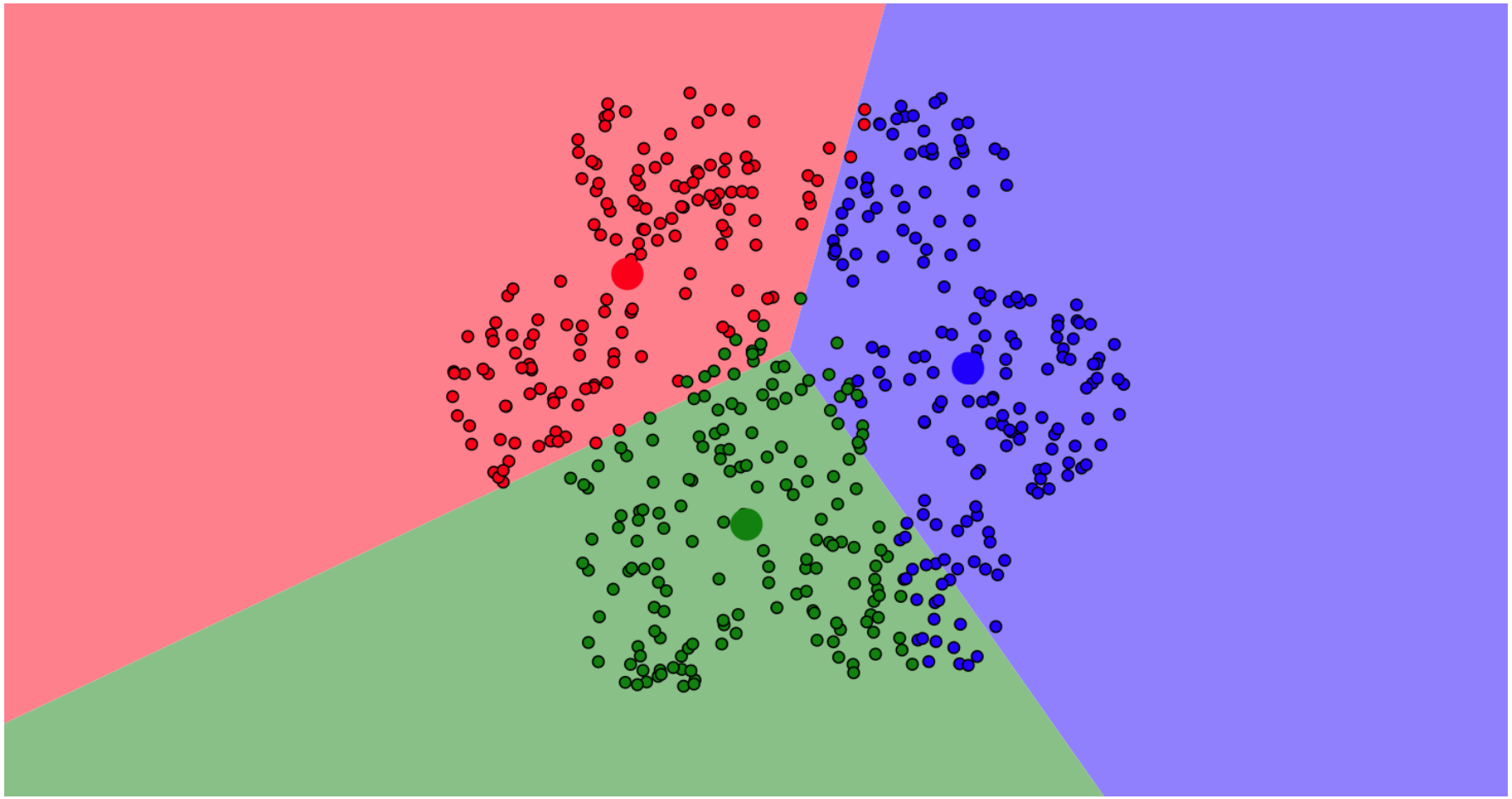
Unsupervised learning

K-means



Unsupervised learning

K-means



Non-Bayesian approach

Neyman

Neyman-Pearson approach

Limitations of the Bayesian approach

$x \in X$ feature

$k \in \mathrm{K}$ hidden state

$p(x|k)$ conditional probability to expose *x by* state *k*

$p(k)$ priory probability of the state

$W : \mathrm{K} \times \mathrm{K} \to R$ penalty function

Neyman-Pearson approach

Limitations of the Bayesian approach

$x \in X$ feature

$k \in \mathrm{K}$ hidden state

$p(x|k)$ conditional probability to expose *x by* state *k*

~~$p(k)$ priory probability of the state~~

~~$W : \mathrm{K} \times \mathrm{K} \to R$ penalty function~~

Neyman-Pearson approach

Limitations of the Bayesian approach: example

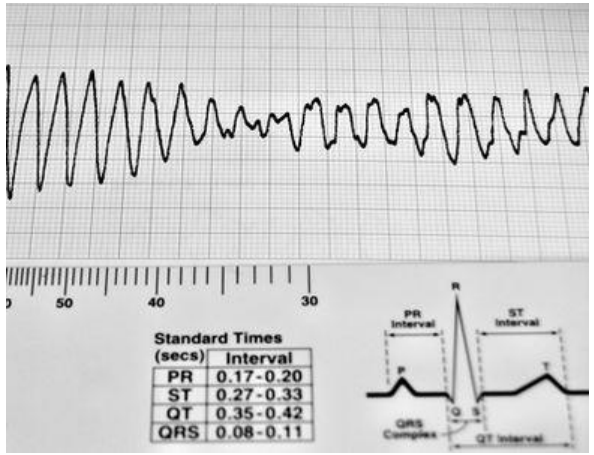
$x \in X$ heartbeat

$k \in \mathrm{K}$ state: normal, sick

$p(x|k)$ distribution of the heartbeat for normal and sick

$p(k)$???

W(normal, sick)
W(sick, normal) ???



Neyman-Pearson approach

The problem

$x \in X \quad k \in K = \{nornal, risky\}$

$p(x|k = norm) \quad X_{norm} \cup X_{risk} = X$

$p(x|k = risk) \quad X_{norm} \cap X_{risk} = 0$

input

$$error\ type\ 1 = P(false\ alarm) = \sum_{x \in X_{risk}} p(x|norm) \rightarrow min$$

$$error\ type\ 2 = P(miss\ the\ risk) = \sum_{x \in X_{norm}} p(x|risk) \leq \varepsilon$$

$X_{norm}, \ X_{risk} = ?$

output

Neyman-Pearson approach

The problem

$$x \in X \qquad k \in K = \{nornal, risky\}$$

$$\alpha_{norm}(x) = \begin{cases} 1\ when\ x \in X_{norm} \\ 0\ when\ x \notin X_{norm} \end{cases}$$

$$\alpha_{risk}(x) = \begin{cases} 1\ when\ x \in X_{risk} \\ 0\ when\ x \notin X_{risk} \end{cases}$$

Neyman-Pearson approach

The problem

$$\sum_{x \in X_{risk}} p(x|norm) \to min$$

$$\sum_{x \in X_{norm}} p(x|risk) \leq \varepsilon$$

$$\begin{cases} \sum_{x \in X} \alpha_{risk}(x) \cdot p(x|norm) \to min \\ -\sum_{x \in X} \alpha_{norm}(x) \cdot p(x|norm) \geq -\varepsilon \\ \alpha_{norm}(x) + \alpha_{risk}(x) = 1 \quad \forall\, x \in X \\ \alpha_{norm}(x) \geq 0 \quad \forall\, x \in X \\ \alpha_{risk}(x) \geq 0 \quad \forall\, x \in X \end{cases}$$

Neyman-Pearson approach

$$p_1 \cdot x_1 + \ \dots \ + p_n \cdot x_n \to max$$

$$a_{11} \cdot x_1 + \ \dots \ + a_{1n} \cdot x_n \le b_1$$

$$a_{k1} \cdot x_1 + \ \dots \ + a_{kn} \cdot x_n \le b_k$$

$$a_{k+1,1} \cdot x_1 + \ \dots \ + a_{k+1,n} \cdot x_n = b_{k+1}$$

$$a_{m1} \cdot x_1 + \ \dots \ + a_{mn} \cdot x_n = b_m$$

$$x_1 \ge 0$$

$$x_s \ge 0$$

$$x_{s+1} - св. \ \ перем.$$

$$x_n - св. \ \ перем.$$

$$b_1 \cdot y_1 + \ \dots \ + b_m \cdot y_m \to min$$

$$y_1 \ge 0$$

$$y_k \ge 0$$

$$y_{k+1} - св. \ \ перем.$$

$$y_m - св. \ \ перем.$$

$$a_{11} \cdot y_1 + \ \dots \ + a_{m1} \cdot y_m \ge p_1$$

$$a_{1s} \cdot y_1 + \ \dots \ + a_{ms} \cdot y_m \ge p_s$$

$$a_{1,s+1} \cdot y_1 + \ \dots \ + a_{m,s+1} \cdot y_m = p_{s+1}$$

$$a_{1n} \cdot y_1 + \ \dots \ + a_{mn} \cdot y_m = p_n$$

Neyman-Pearson approach

$$
\begin{cases}
p_1 \cdot x_1 + \ \dots \ + p_n \cdot x_n \to max \\
a_{11} \cdot x_1 + \ \dots \ + a_{1n} \cdot x_n \le b_1 \\
a_{k1} \cdot x_1 + \ \dots \ + a_{kn} \cdot x_n \le b_k \\
a_{k+1,1} \cdot x_1 + \ \dots \ + a_{k+1,n} \cdot x_n = b_{k+1} \\
a_{m1} \cdot x_1 + \ \dots \ + a_{mn} \cdot x_n = b_m \\
x_1 \ge 0 \\
x_s \ge 0 \\
x_{s+1} - св. \quad перем. \\
x_n - св. \quad перем.
\end{cases}
\qquad
\begin{cases}
b_1 \cdot y_1 + \ \dots \ + b_m \cdot y_m \to min \\
y_1 \ge 0 \\
y_k \ge 0 \\
y_{k+1} - св. \quad перем. \\
y_m - св. \quad перем. \\
a_{11} \cdot y_1 + \ \dots \ + a_{m1} \cdot y_m \ge p_1 \\
a_{1s} \cdot y_1 + \ \dots \ + a_{ms} \cdot y_m \ge p_s \\
a_{1,s+1} \cdot y_1 + \ \dots \ + a_{m,s+1} \cdot y_m = p_{s+1} \\
a_{1n} \cdot y_1 + \ \dots \ + a_{mn} \cdot y_m = p_n
\end{cases}
$$

$$
\left(a_{11} \cdot x_1 + \ \dots \ + a_{1n} \cdot x_n - b_1\right) \cdot y_1 = 0
$$

$$
\left(a_{11} \cdot y_1 + \ \dots \ + a_{m1} \cdot y_m - p_1\right) \cdot x_1 = 0
$$

Neyman-Pearson approach

Solution

$$\begin{cases} \sum_{x \in X} \alpha_{risk}(x) \cdot p(x|norm) \to min \\ -\sum_{x \in X} \alpha_{norm}(x) \cdot p(x|norm) \geq -\varepsilon \\ \alpha_{norm}(x) + \alpha_{risk}(x) = 1 \quad \forall\, x \in X \\ \alpha_{norm}(x) \geq 0 \quad \forall\, x \in X \\ \alpha_{risk}(x) \geq 0 \quad \forall\, x \in X \end{cases}$$

$$\begin{cases} -\varepsilon \cdot \tau + 1 \cdot t(x_1) + \cdots + 1 \cdot t(x_i) \to max \\ \tau \geq 0 \\ t(x)\ free\ var \\ -p(x|risk) \cdot \tau + 1 \cdot t(x) \leq 0 \\ 1 \cdot t(x) \leq p(x|norm) \end{cases}$$

Neyman-Pearson approach

Solution

$$x \in X_{norm} \Rightarrow \alpha_{norm}(x) = 1, \alpha_{risk}(x) = 0 \quad \Rightarrow \quad \begin{cases} -p(x|risk) \cdot \tau + 1 \cdot t(x) = 0 \\ 1 \cdot t(x) < p(x|norm) \end{cases} \quad \Rightarrow$$

$$\Rightarrow \quad p(x|risk) \cdot \tau \leq p(x|norm)$$

$$x \in X_{risk} \Rightarrow \alpha_{norm}(x) = 0, \ \alpha_{risk}(x) = 1 \quad \Rightarrow \quad \begin{cases} -p(x|risk) \cdot \tau + 1 \cdot t(x) \leq 0 \\ 1 \cdot t(x) = p(x|norm) \end{cases} \quad \Rightarrow$$

$$\Rightarrow \quad p(x|risk) \cdot \tau \geq p(x|norm)$$

Neyman-Pearson approach

Solution

$$\frac{p(x|norm)}{p(x|risk)} \underset{\text{risk}}{\overset{\text{norm}}{\gtrless}} \tau$$

Neyman-Pearson approach

Some Variations: two risky states

$$\sum_{x \in X_{risk}} p(x|norm) \rightarrow min$$

$$\sum_{x \in X_{norm}} p(x|risk1) \leq \varepsilon$$

$$\sum_{x \in X_{norm}} p(x|risk2) \leq \varepsilon$$

Neyman-Pearson approach

Some Variations: two normal states

$$\sum_{x \in X_{risk}} p(x|norm1) + \sum_{x \in X_{risk}} p(x|norm2) \rightarrow min$$

$$\sum_{x \in X_{norm}} p(x|risk) \leq \varepsilon$$

Neyman-Pearson approach

Minimax problem

$$max\left\{\sum_{x\notin X_1} p(x|1)\ ,\sum_{x\notin X_2} p(x|2)\ ..,\sum_{x\notin X_K} p(x|K)\ \right\}\rightarrow min$$

Time Series

https://github.com/dryabokon/ML/blob/master/ex_04_03_Time_Series.py

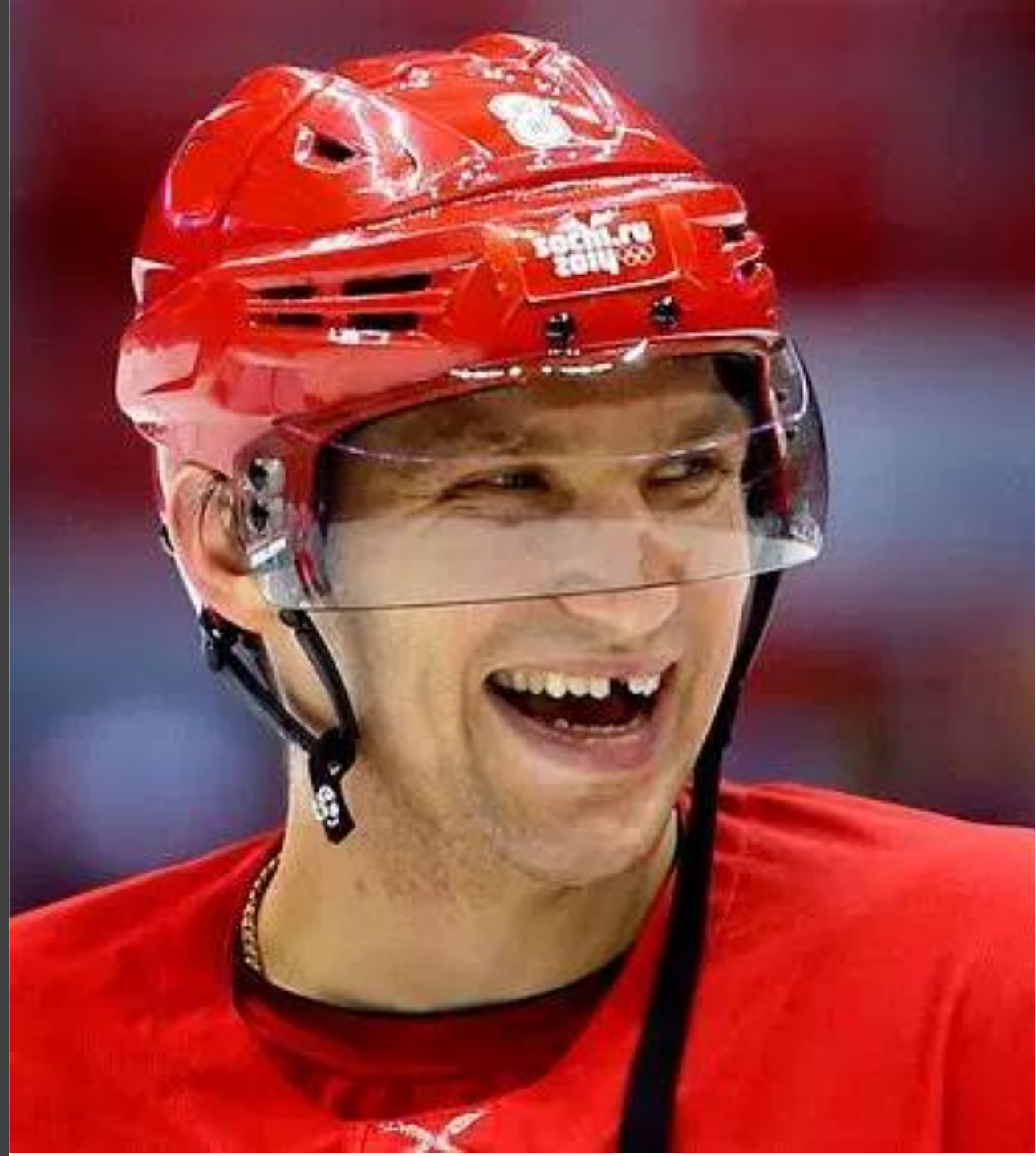
Time Series

- Naive
- LinearRegression
- AutoRegression
- Holt
- VAR
- ARIMA
- LSTM

Part 5: Optimization

Missing values: the approach
Bias vs Variance
Regularization
Variable importance
Multi-collinearity
Reducing data dimensions
Back propagation

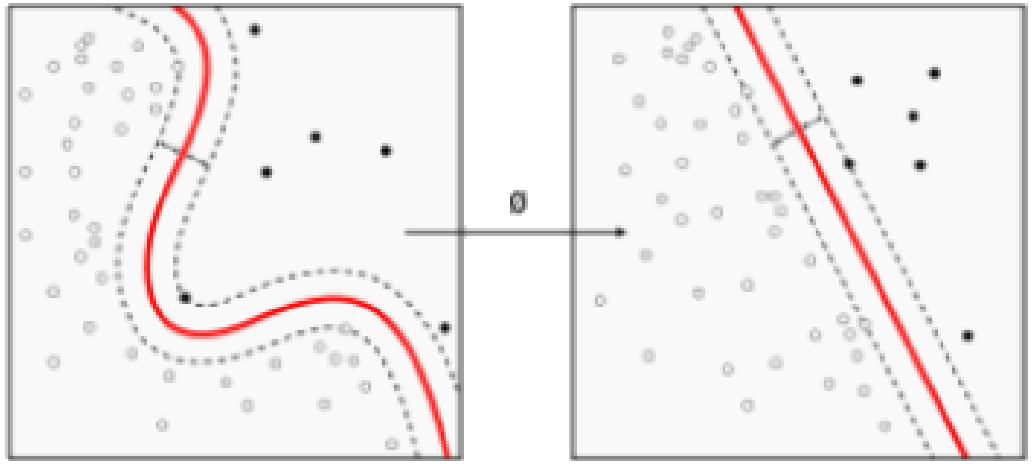
Missing values



Missing values

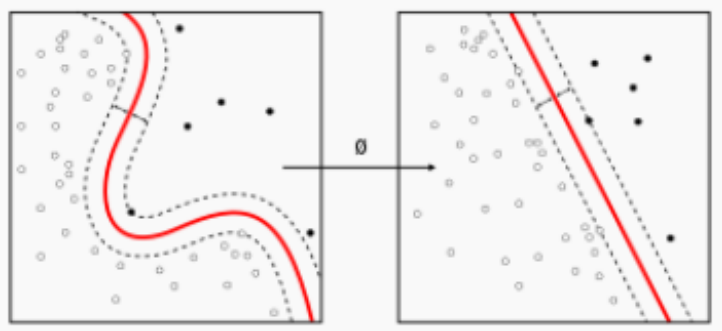
- Delete missing rows
- Replace with mean/median
- Encode as another level of a categorical variable
- Create a new engineered feature
- Hot deck

Bias vs Variance



Bias vs Variance

The **bias** is an error from wrong assumptions in the learning algorithm
High bias can cause **underfit**: the algorithm can miss relations between features and target



The **variance** is an error from sensitivity to small fluctuations in the training set.
High variance can cause **overfit**: the algorithm can model the random noise in the training data, rather than the intended outputs

Bias vs Variance

How to avoid overfit (high variance)

- Keep the model simpler
- Do cross-validation (e.g. k-folds)
- Use regularization techniques (e.g. LASSO, Ridge)
- Use top n features from variable importance chart

Regularization

Regularization: ridge (L2) and lasso (L1)

Regularization becomes necessary when the model begins to ovefit / underfit.
This technique introduces a cost term for bringing in more features with the objective function.

Hence, it tries to push the coefficients for many variables to zero and hence reduce cost term. This helps to reduce model complexity so that the model can become better at predicting (generalizing).

The **key difference** between these techniques is that Lasso shrinks the less important feature's coefficient to zero thus, removing some feature altogether. So, this works well for **feature selection** in case we have a huge number of features.
Traditional methods like cross-validation, stepwise regression to handle overfitting and perform feature selection work well with a small set of features but these techniques are a great alternative when we are dealing with a large set of features.

In presence of many variables with small / medium sized effect, use ridge regression.
Conceptually, we can say, lasso regression (L1) does both variable selection and parameter shrinkage, whereas Ridge regression only does parameter shrinkage and end up including all the coefficients in the model. In presence of correlated variables, ridge regression might be the preferred choice. Also, ridge regression works best in situations where the least square estimates have higher variance.

Regularization: when # of variables > # of observation

In such high dimensional data sets, we can't use classical regression techniques, since their assumptions tend to fail. When p > n, we can no longer calculate a unique least square coefficient estimate, the variances become infinite, so OLS (Ordinary Least Squares) cannot be used at all.

To combat this situation, we can use penalized regression methods like lasso, LARS, ridge which can shrink the coefficients to reduce variance. Precisely, ridge regression works best in situations where the least square estimates have higher variance.

Variable importance

Variable importance

Following are the methods of variable selection you can use:

- Remove the correlated variables prior to selecting important variables
- Use linear regression and select variables based on p values
- Use Forward Selection, Backward Selection, Stepwise Selection
- Use Random Forest, Xgboost and plot variable importance chart
- Use Lasso Regression
- Measure information gain for the available set of features and select top n features accordingly.

Multicollinearity

Multicollinearity

Multicollinearity is problem that you can run into when you're fitting a regression model, or other linear model. It refers to predictors that are correlated with other predictors in the model. Unfortunately, the effects of multicollinearity can feel murky and intangible, which makes it unclear whether it's important to fix.

Moderate multicollinearity may not be problematic. However, severe multicollinearity is a problem because it can increase the variance of the coefficient estimates and make the estimates very sensitive to minor changes in the model. The result is that the coefficient estimates are unstable and difficult to interpret. Multicollinearity saps the statistical power of the analysis, can cause the coefficients to switch signs, and makes it more difficult to specify the correct model

Upon correlation matrix - **remove high correlated variables** (eg above 75%).

In addition, calculate **VIF (variance inflation factor) to check the presence of multicollinearity**. VIF value <= 4 suggests no multicollinearity whereas a value of >= 10 implies serious multicollinearity.