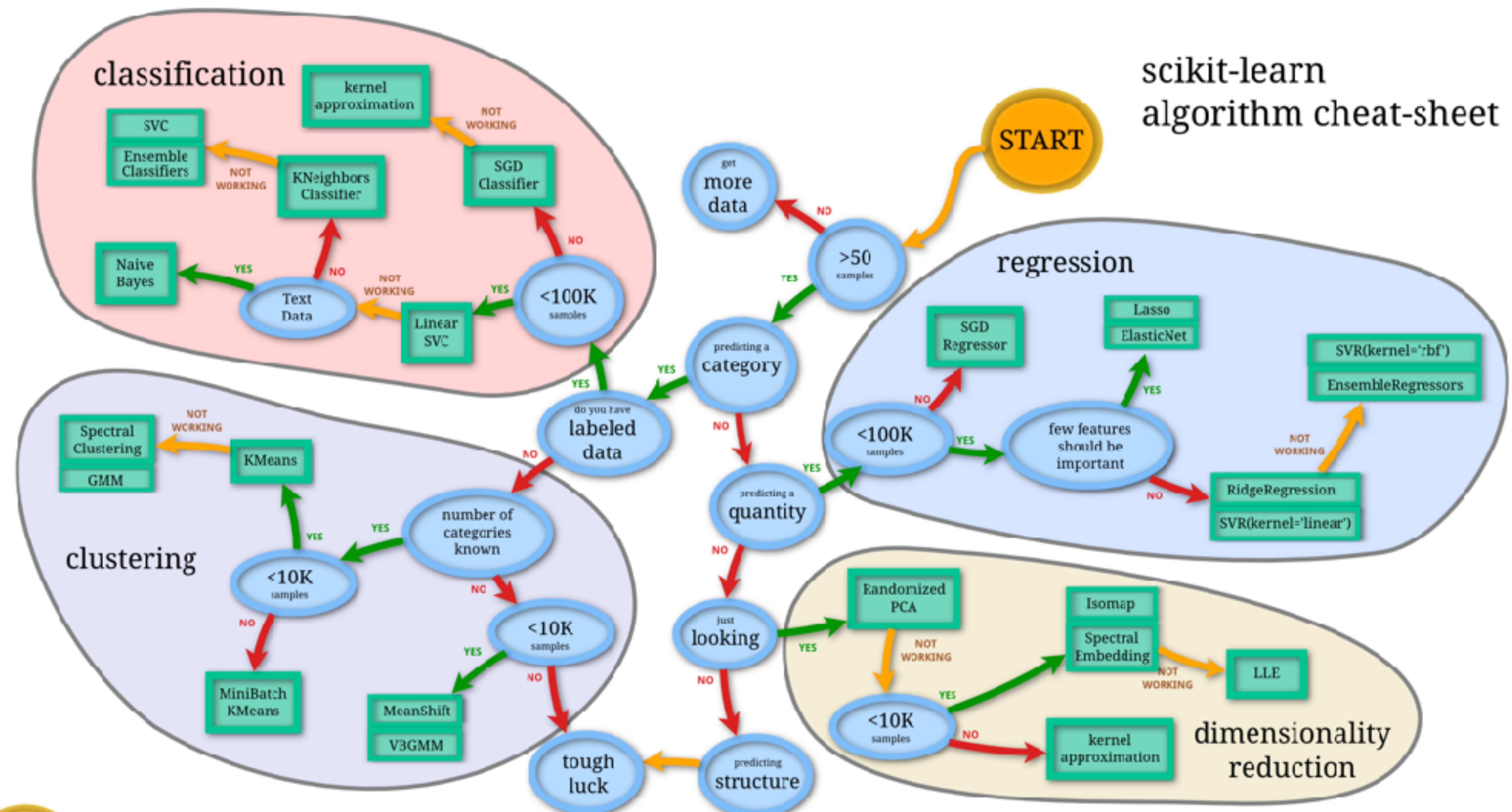
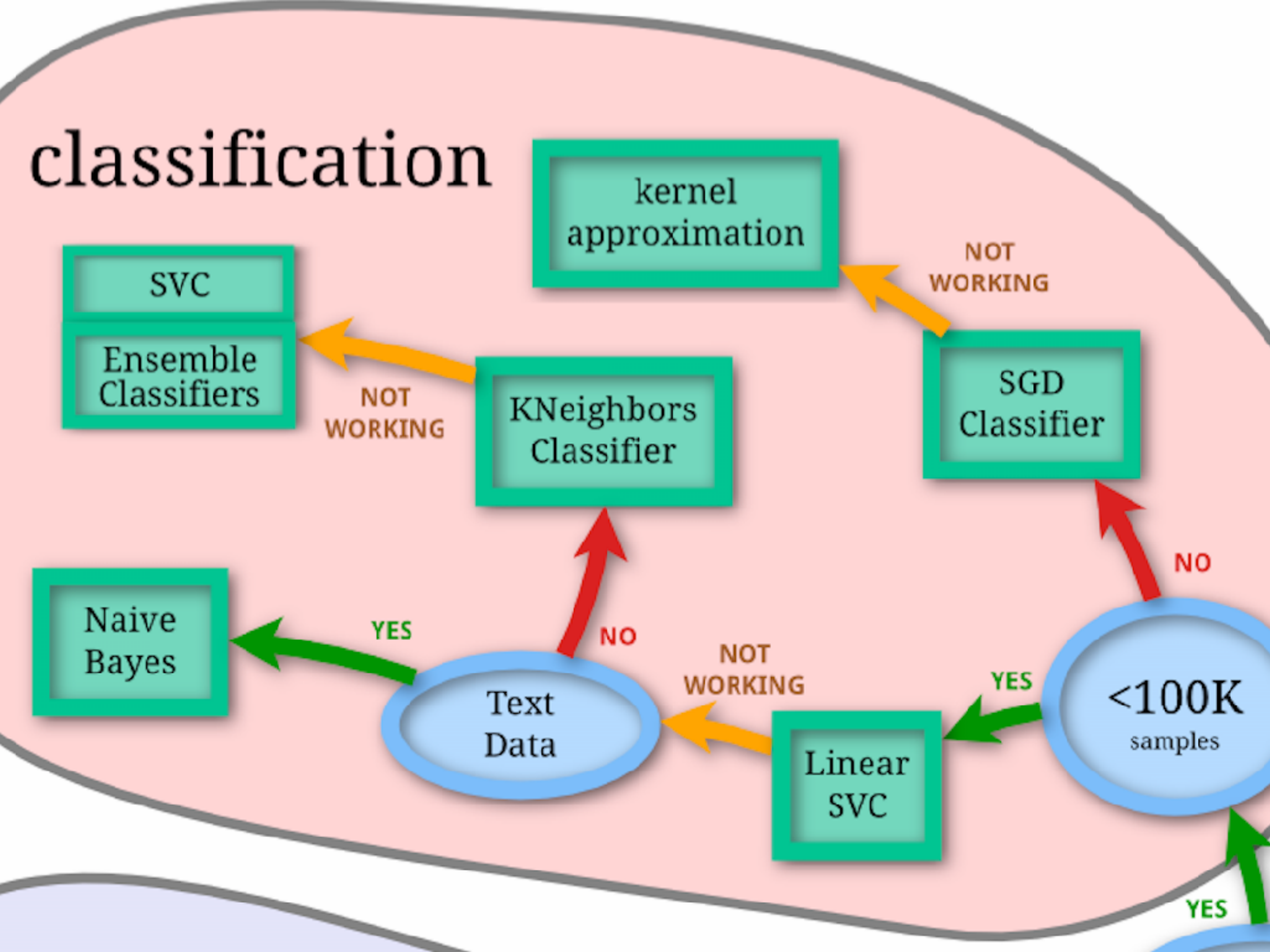
SUPPORT VECTOR MACHINES

Classification on the margins

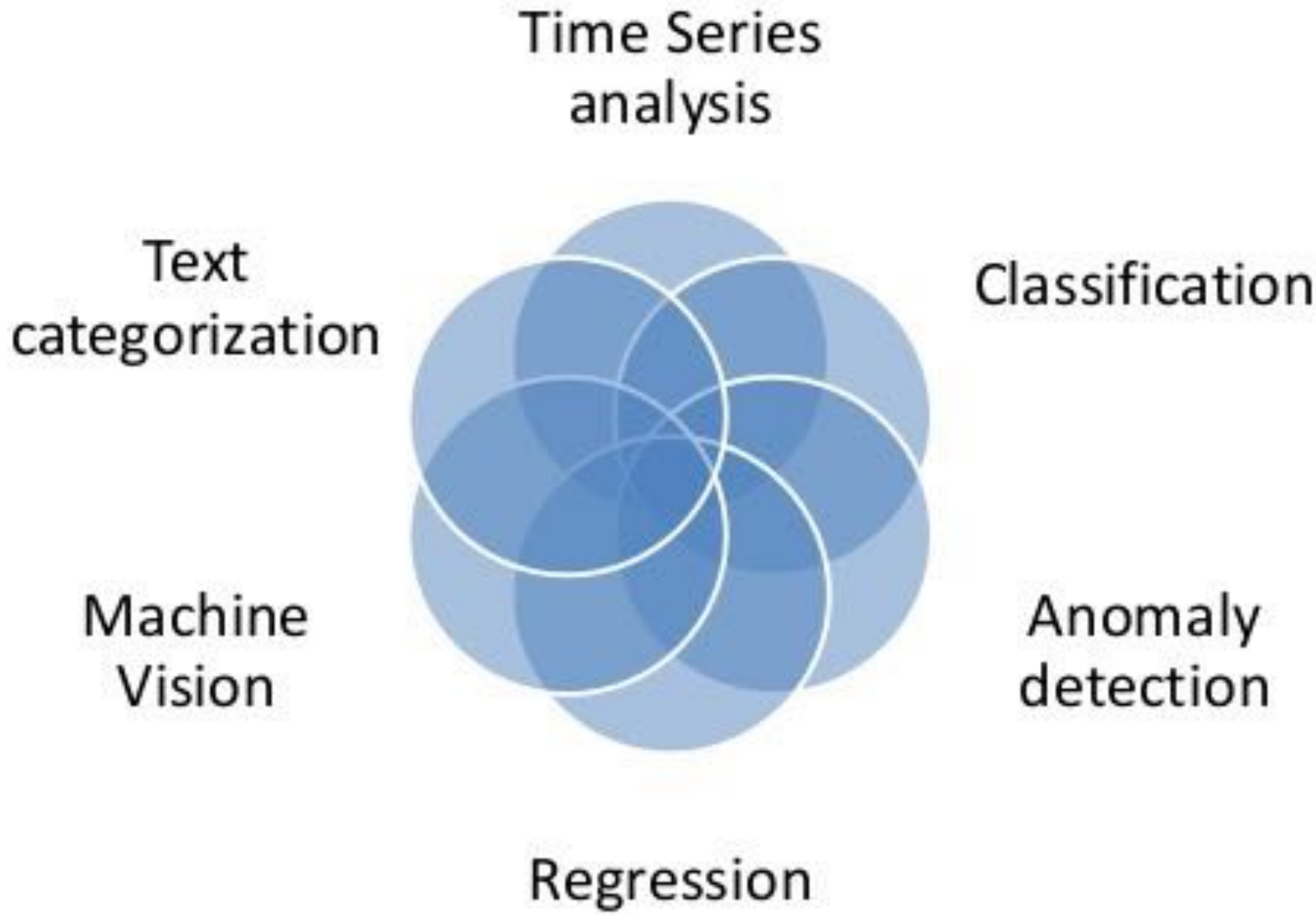


scikit-learn
algorithm cheat-sheet
START
>50 samples
get more data
predicting a category
do you have labeled data
predicting a quantity
just looking
predicting structure
tough luck
classification
<100K samples
SGD Classifier
kernel approximation
Linear SVC
Text Data
Naive Bayes
KNeighbors Classifier
SVC
Ensemble Classifiers
clustering
number of categories known
<10K samples
KMeans
Spectral Clustering
GMM
MiniBatch KMeans
MeanShift
VBGMM
regression
SGD Regressor
few features should be important
Lasso
ElasticNet
RidgeRegression
SVR(kernel='linear')
SVR(kernel='rbf')
EnsembleRegressors
dimensionality reduction
Randomized PCA
Isomap
Spectral Embedding
LLE
kernel approximation
YES
NO
NOT WORKING

Back

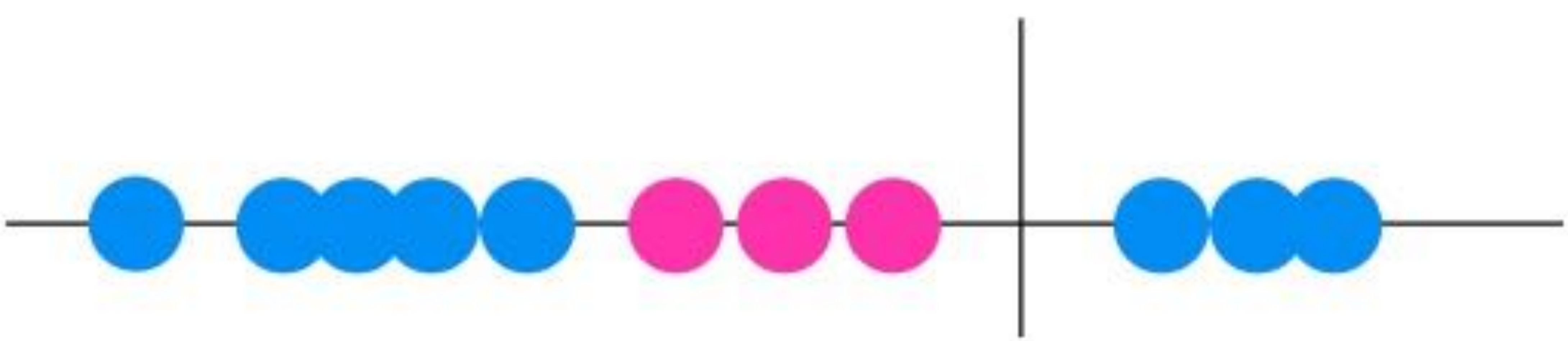

classification
kernel approximation
NOT WORKING
SVC
Ensemble Classifiers
NOT WORKING
KNeighbors Classifier
SGD Classifier
NO
Naive Bayes
YES
NO
NOT WORKING
YES
Text Data
<100K samples
Linear SVC
YES

APPLICATIONS OF SUPPORT VECTOR MACHINES

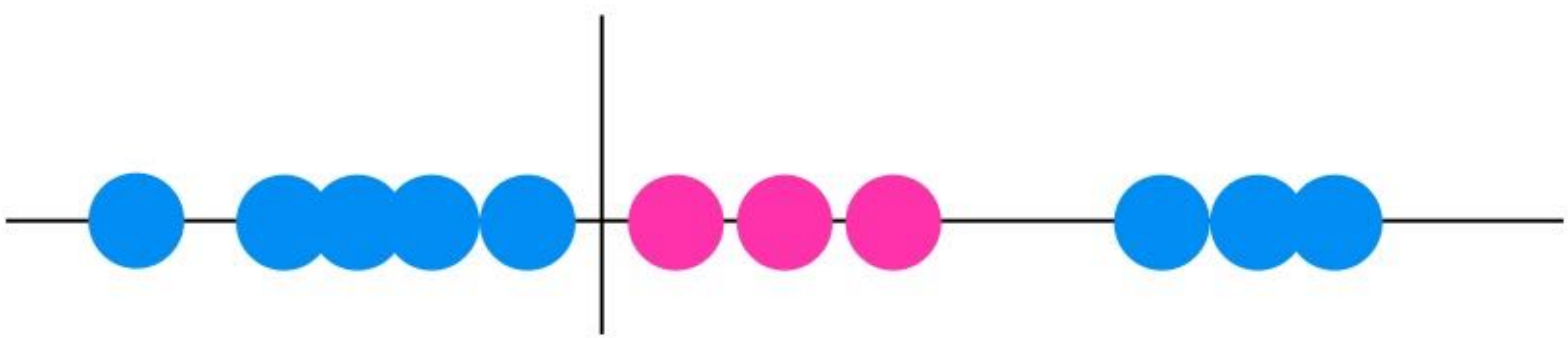


HOW TO SPLIT THIS DATA?

NOT SO GOOD…

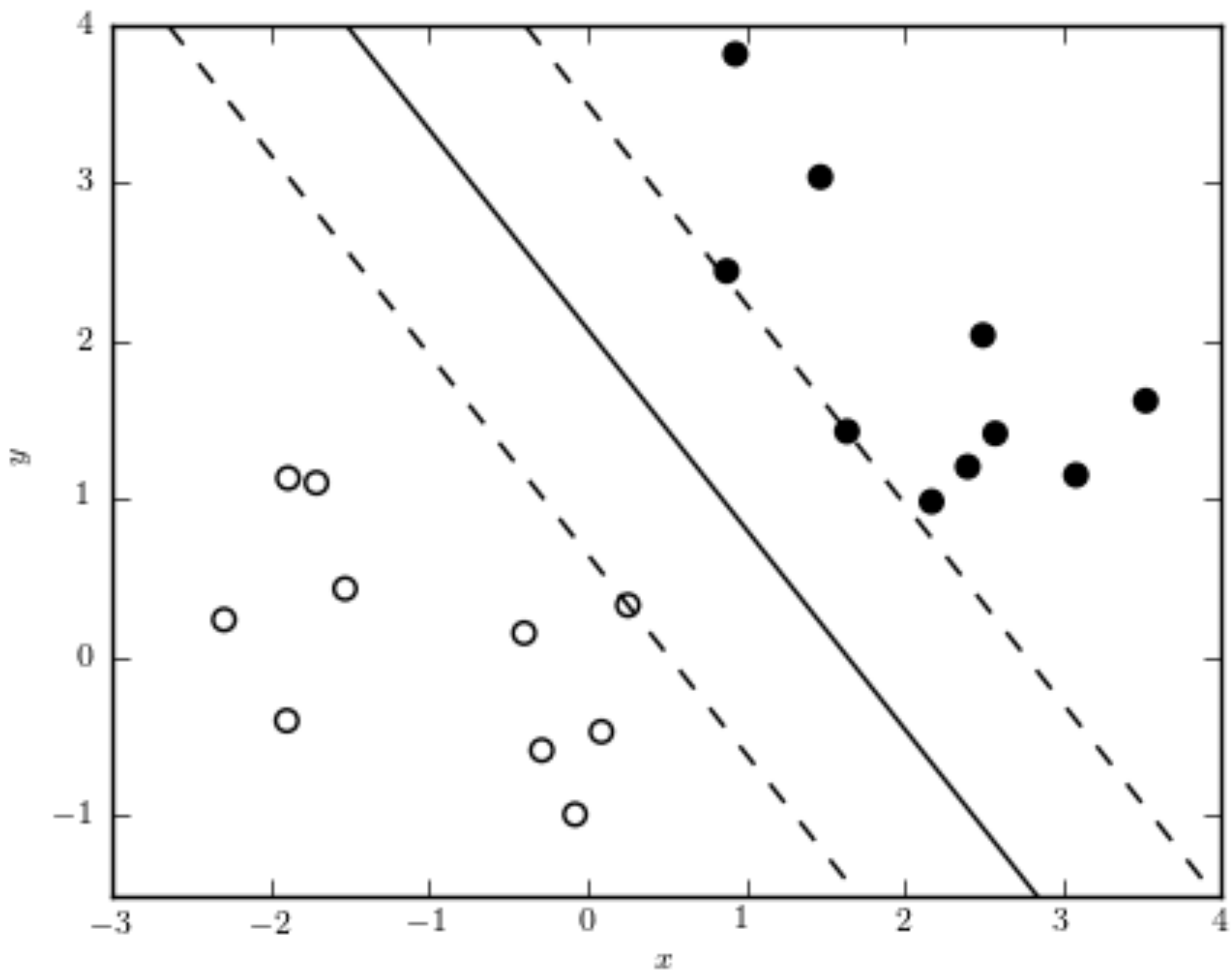


NOT SO GOOD EITHER…



BASIC CONCEPTS OF LINEAR SVM

$$\max_{\beta_0,\beta}(m) = \quad \text{subject to} \quad \frac{1}{\|\beta\|} y_i(\beta_0 + \beta^T \mathbf{x}_i) \geq m, \quad \forall\ i$$



BASIC CONCEPTS OF LINEAR SVM

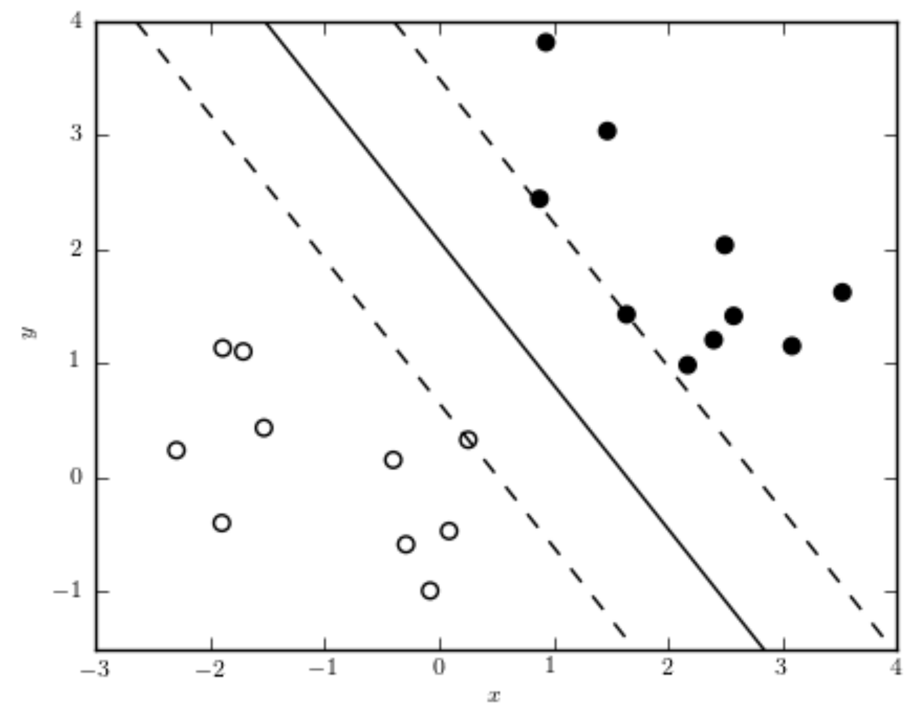
$$0 = \beta^T \mathbf{x} + \beta_0$$

$$d = \hat{\beta}^T |\mathbf{x}_i - \mathbf{x}|$$

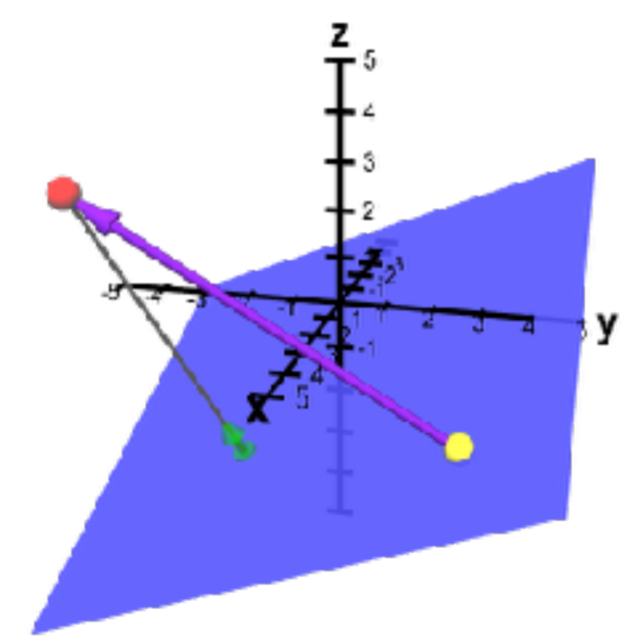
$$d = \frac{1}{||\beta||} |\beta^T \mathbf{x_i} - \beta^T \mathbf{x}|$$

$$d = \frac{1}{||\beta||} |\beta^T \mathbf{x_i} + \beta_0 - \beta^T \mathbf{x} - \beta_0|$$

$$d = \frac{1}{||\beta||} = m$$



$$y_i(\beta_0 + \beta^T \mathbf{x}_i) \geq 1$$



FLIPPING THE PROBLEM

$$\max \frac{1}{\|\beta\|} \to \min \|\beta\| = \beta^T \beta \qquad y_i(\beta_0 + \beta^T \mathbf{x}_i) \geq 1$$

- KKT Lagrangian Formulation - Physics!

$$\mathcal{L} = \frac{1}{2}\beta^T \beta - \sum_{i=1}^{N} \alpha_i (y_i(\beta_0 + \beta^T \mathbf{x}_i) - 1)$$

Lagrange multiplier

LAGRANGE FORMULATION

$$\mathcal{L} = \frac{1}{2}\beta^T\beta - \sum_{i=1}^{N}\alpha_i(y_i(\beta_0 + \beta^T\mathbf{x}_i) - 1)$$

Equations of Motion

$$\boldsymbol{\nabla}_\beta\mathcal{L} = \beta - \sum_{i=1}^{N}\alpha_i y_i \mathbf{x_i} = \mathbf{0}$$

$$\frac{\partial\mathcal{L}}{\partial\beta_0} = -\sum_{i=1}^{N}\alpha_i y_i = 0$$

Solutions

$$\beta = \sum_{i=1}^{N}\alpha_i y_i \mathbf{x_i}$$

$$\sum_{i=1}^{N}\alpha_i y_i = 0$$

LAGRANGE FORMULATION

$$\mathcal{L}(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j y_i y_j \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j$$

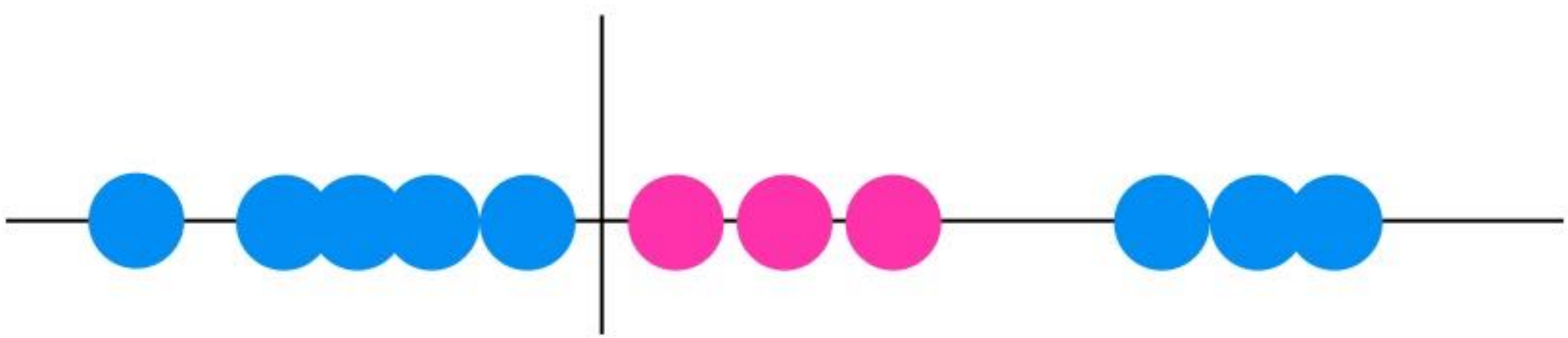
$$\downarrow$$

$$\min_\alpha \quad \frac{1}{2} \alpha^T \begin{pmatrix} y_1 y_1 x_1 x_1 & y_1 y_2 x_1 x_2 & \cdots & y_1 y_N x_1 x_N \\ y_2 y_1 x_2 x_1 & y_2 y_2 x_2 x_2 & \cdots & y_2 y_N x_2 x_N \\ \vdots & \vdots & \vdots & \vdots \\ y_N y_1 x_N x_1 & y_N y_2 x_N x_2 & \cdots & y_N y_N x_N x_N \end{pmatrix} \alpha + (-\mathbf{1}^T)\alpha$$

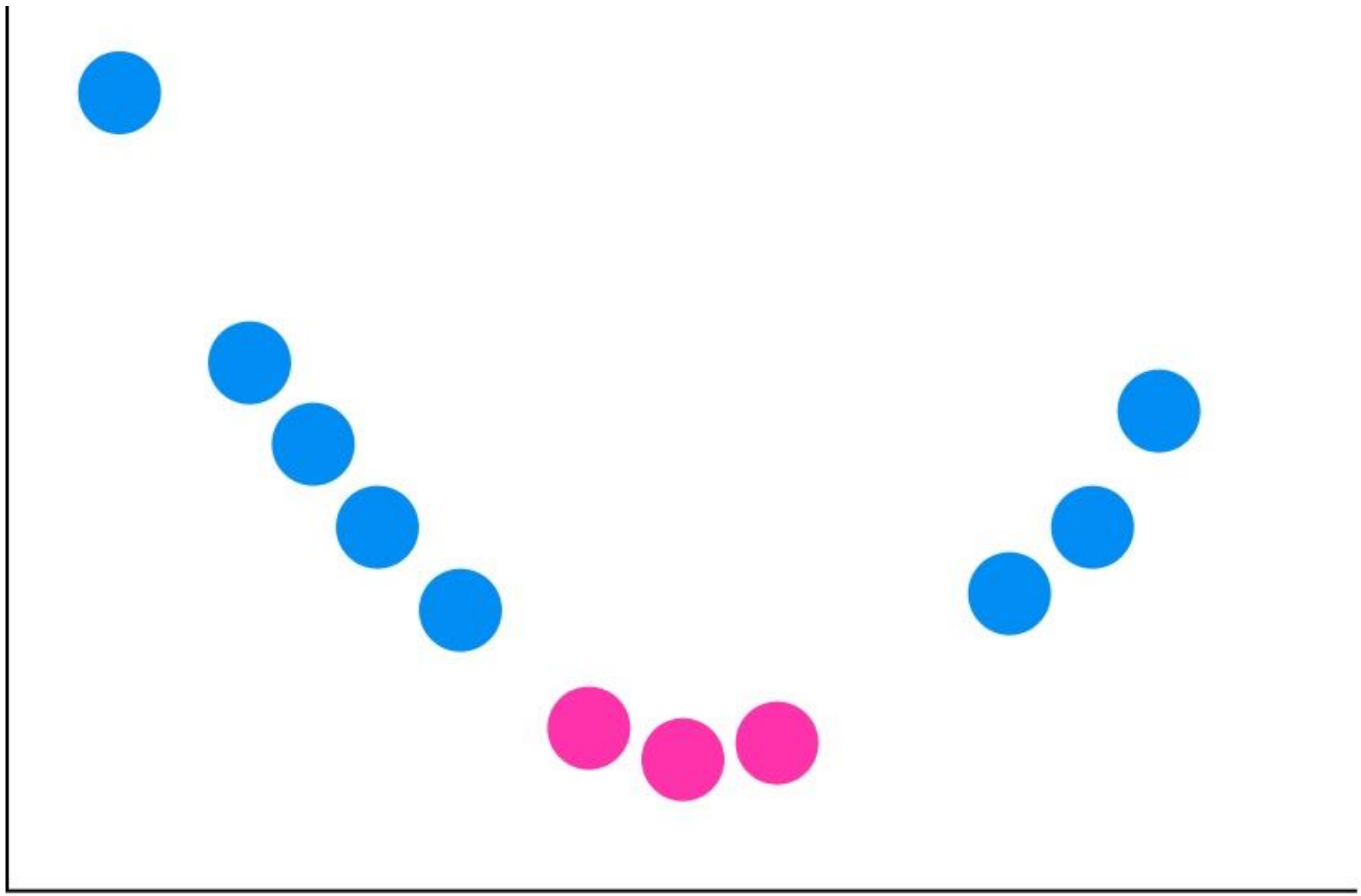
$$\beta = \sum_{\alpha_i > 0} \alpha_i y_i \mathbf{x}_i \qquad\qquad y_m(\beta^T \mathbf{x}_m + \beta_0) = 1$$

BUT THIS WASN'T REALLY SO GOOD...

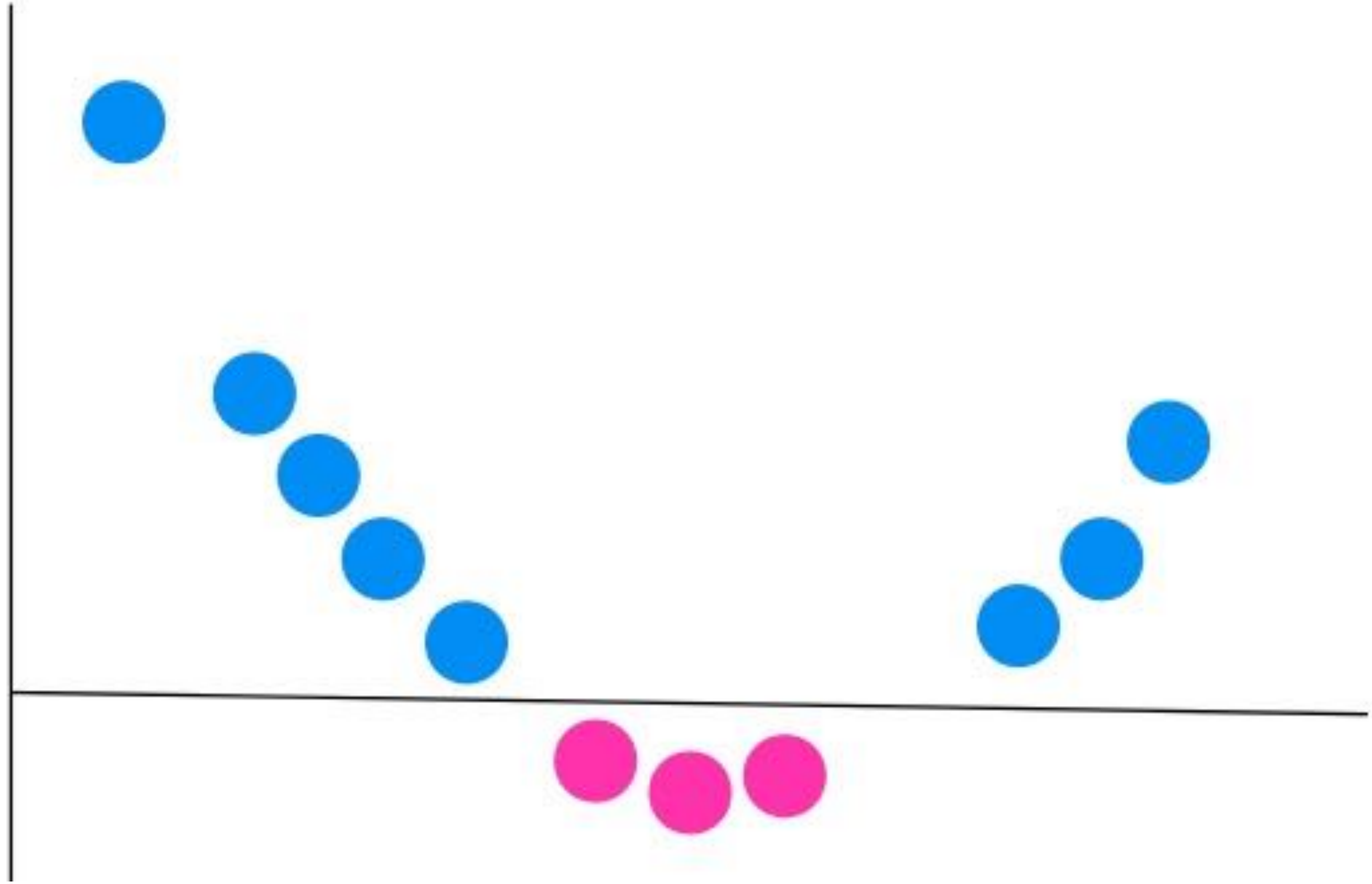
$$\mathcal{L}(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j y_i y_j \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j$$



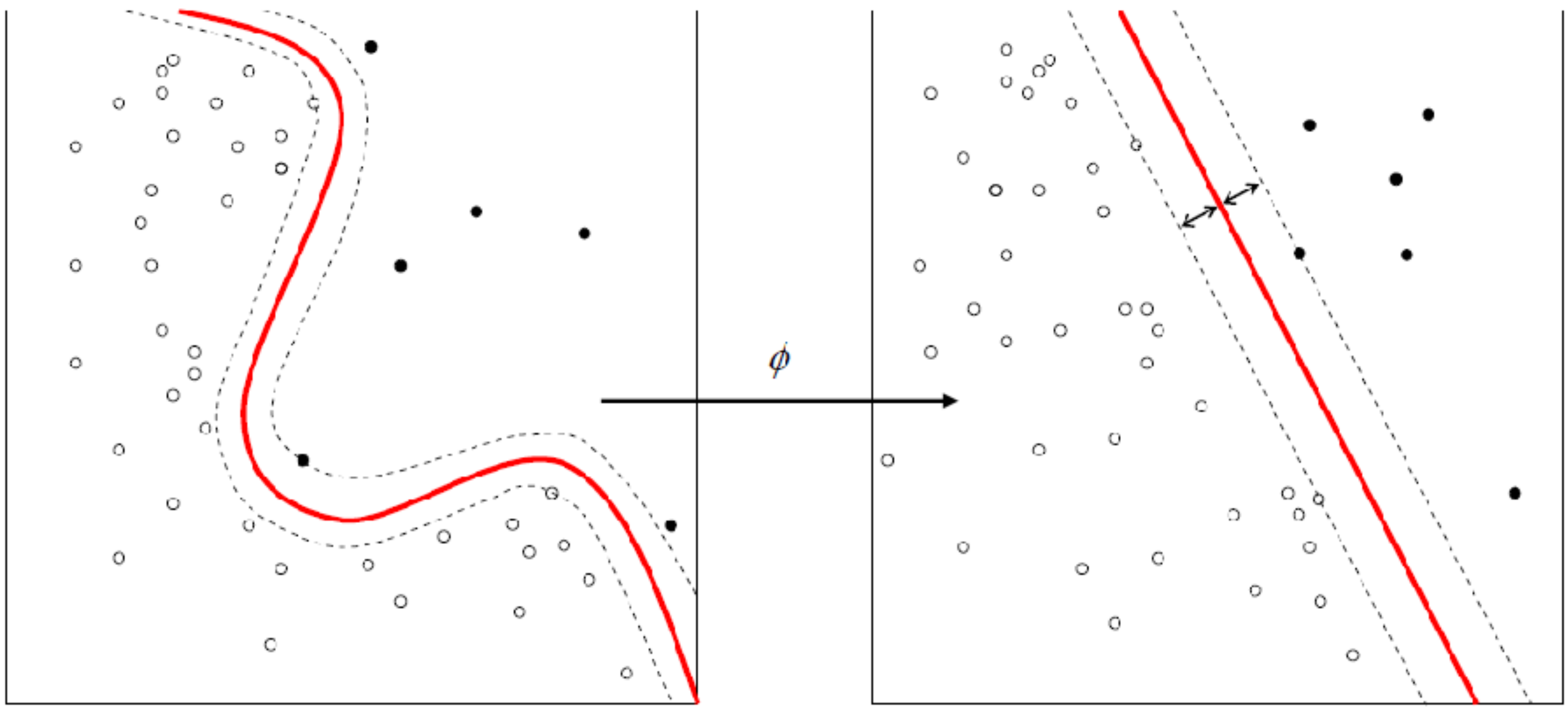
MAP THE DATA INTO A HIGHER DIMENSION OF SPACE



NOW THE DATA IS LINEARLY SEPARABLE



$$\mathcal{X} \rightarrow \mathcal{Z}$$



$$\mathcal{L}(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j y_i y_j \alpha_i \alpha_j \mathbf{z}_i^T \mathbf{z}_j$$

COMMON TYPES OF SUPPORT VECTOR MACHINES

- linear

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$$

- polynomial

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma \mathbf{x}_i^T \mathbf{x}_j + r)^d, \gamma > 0$$

- radial basis function (RBF) (Gaussian)

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\Big(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\Big), \gamma > 0$$

- sigmoid

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh\big(\gamma \mathbf{x}_i^T \mathbf{x}_j + r\big)$$

$$\mathcal{L}(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j y_i y_j \alpha_i \alpha_j K(x_i, x_j)$$

$$\downarrow$$

$$\frac{1}{2}\alpha^T \begin{pmatrix} y_1 y_1 K(x_1, x_1) & y_1 y_2 K(x_1, x_2) & \cdots & y_1 y_N K(x_1, x_N) \\ y_2 y_1 x_2 x_1 & y_2 y_2 K(x_2, x_2) & \cdots & y_2 y_N K(x_2, x_N) \\ \vdots & \vdots & \vdots & \vdots \\ y_N y_1 K(x_N, x_1) & y_N y_2 K(x_N, x_2) & \cdots & y_N y_N K(x_N, x_N) \end{pmatrix} \alpha + (-\mathbf{1}^T)\alpha$$

THE SAME ANSWER!

$$\beta = \sum_{\alpha_i > 0} \alpha_i y_i \mathbf{z}_i \qquad\qquad y_m(\beta^T \mathbf{z}_m + \beta_0) = 1$$

EVALUATION OF THE MODEL

$$g(\mathbf{x}) = \mathrm{sign}(\beta^T z + \beta_0)$$

$$\mathbb{E}[E_{\mathrm{out}}] \leq \frac{\mathbb{E}[N_{\mathrm{SV}}]}{N-1}$$

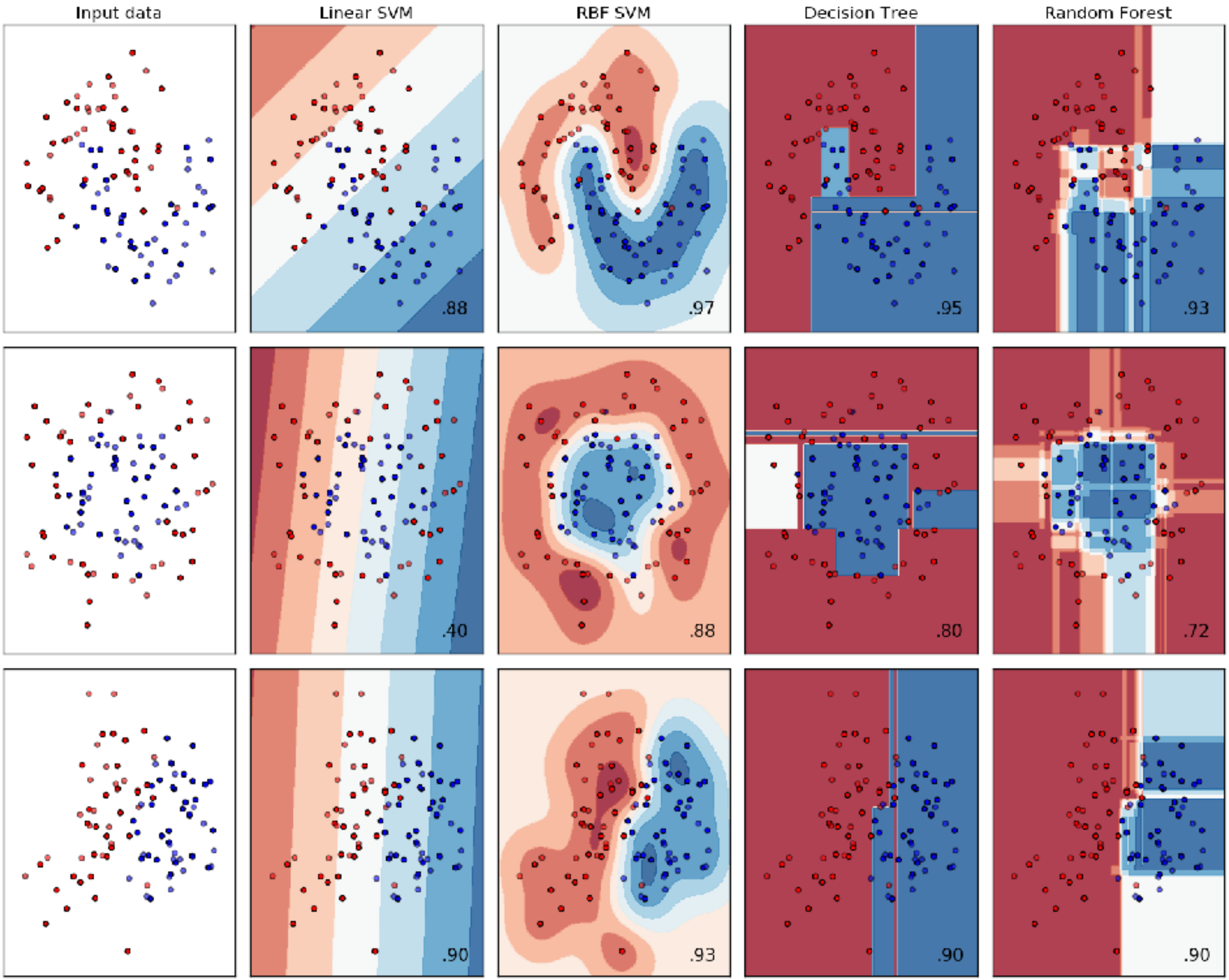
SCIKIT LEARN

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier

names = ["Linear SVM", "RBF SVM",
         "Decision Tree", "Random Forest"]


classifiers = [
    SVC(kernel="linear", C=0.025),
    SVC(gamma=2, C=1),
    DecisionTreeClassifier(max_depth=5),
    RandomForestClassifier(max_depth=5, n_estimators=10, max_features=1),
    ]

datasets = [make_moons(noise=0.3, random_state=0),
            make_circles(noise=0.2, factor=0.5, random_state=1),
            linearly_separable
            ]
```


Input data
Linear SVM
RBF SVM
Decision Tree
Random Forest
.88
.97
.95
.93
.40
.88
.80
.72
.90
.93
.90
.90

REFERENCES

1. Ivezic et al. Chapter 9
2. Brain Lange http://www.slideshare.net/brianjlange/machine-learning-in-5-minutes-classification/22
3. Classifier comparison http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html#sphx-glr-auto-examples-classification-plot-classifier-comparison-py
4. https://en.wikipedia.org/wiki/Karush–Kuhn–Tucker_conditions