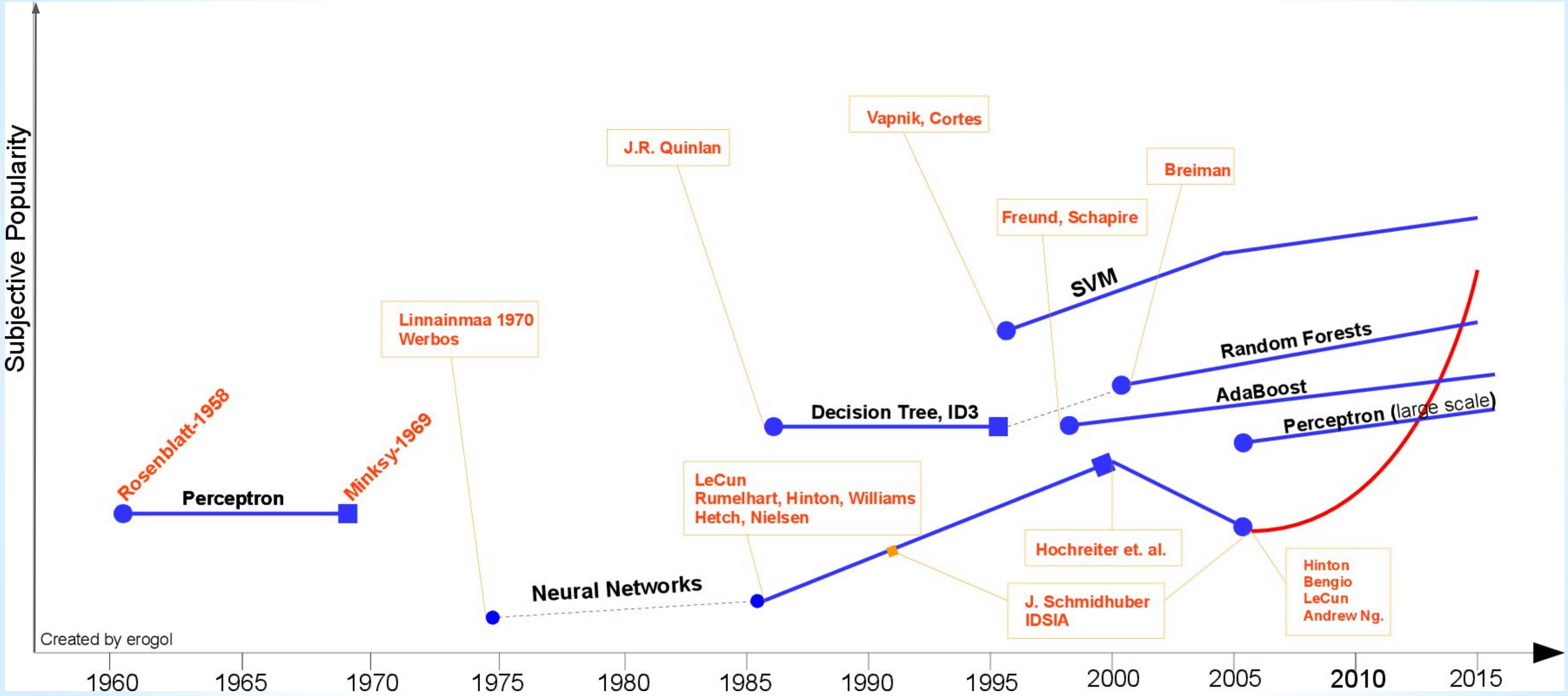
Support Vector Machines

Lior Sidi & Efrat Egozi

Y-DATA
SCHOOL OF DATA SCIENCE

Yandex

Models History

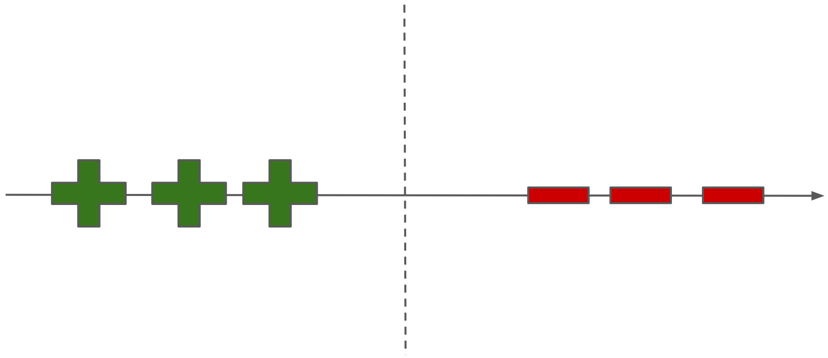


Motivation

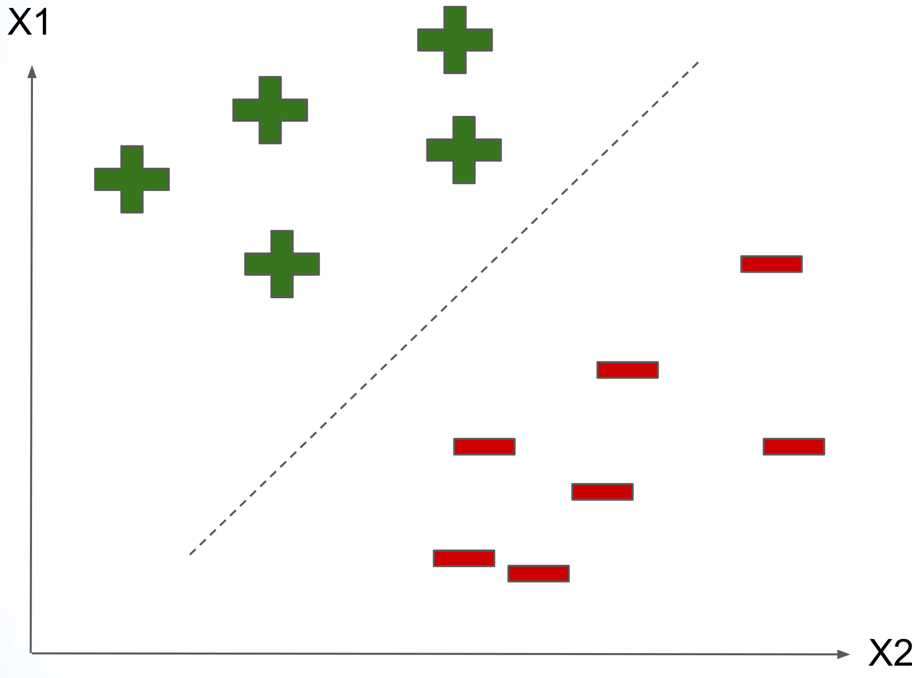
Good for difficult problems with limited data (<10K data points)

- Face Detection
- Text Classification
- Protein Fold and Remote Homology Detection
- Handwriting Recognition

Linear Classification - 1 dim



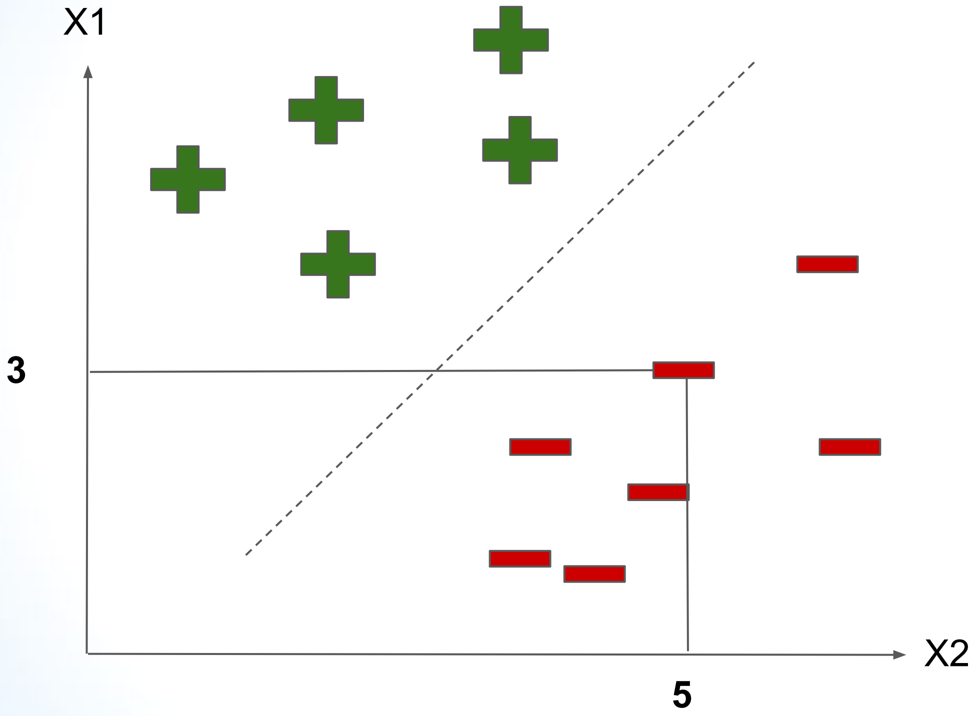
Linear Classification - 2 dim



w1*x1 = w2*x2 + b

2*x1 = 4*x2 - 3
2*x1 - 4*x2 + 3 = 0

Linear Classification - 2 dim

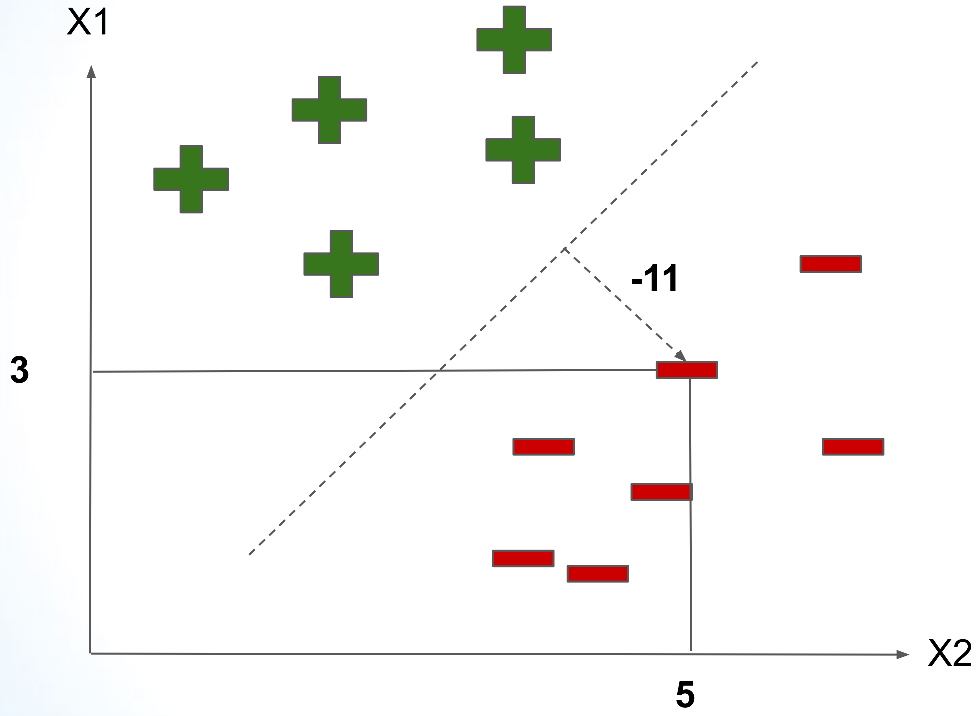


w1*x1 = w2*x2 + b

2*x1 = 4*x2 - 3
2*x1 - 4*x2 + 3 = 0

2*3 - 4*5 + 3 =

Linear Classification - 2 dim

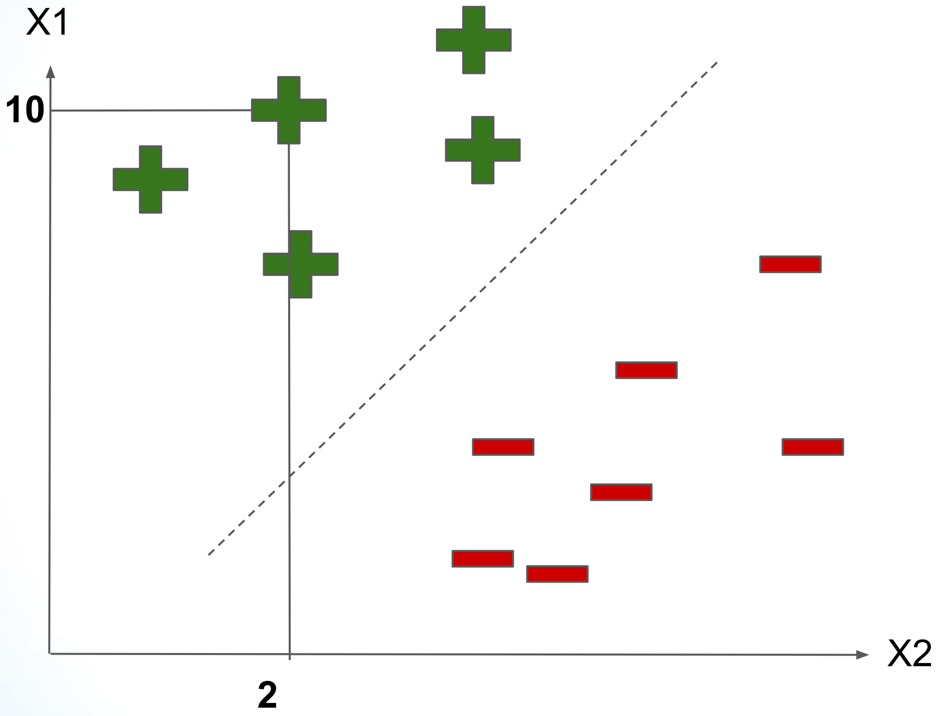


w1*x1 = w2*x2 + b

2*x1 = 4*x2 - 3
2*x1 - 4*x2 + 3 = 0

2*3 - 4*5 + 3 = 6 - 20 + 3 = -11

Linear Classification - 2 dim



w1*x1 = w2*x2 + b

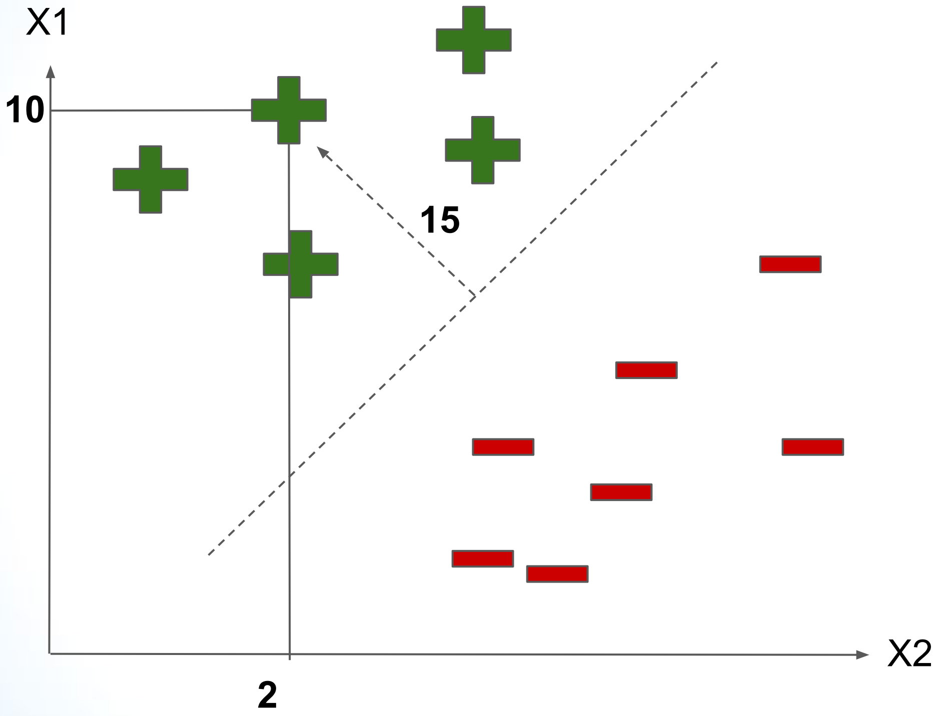
2*x1 = 4*x2 - 3
2*x1 - 4*x2 + 3 = 0

2*3 - 4*5 + 3 = 6 - 20 + 3 = -11
2*10 - 4*2 + 3 = 20 - 8 + 3 = 15

Y-DATA
SCHOOL OF DATA SCIENCE

Yandex

Linear Classification - 2 dim

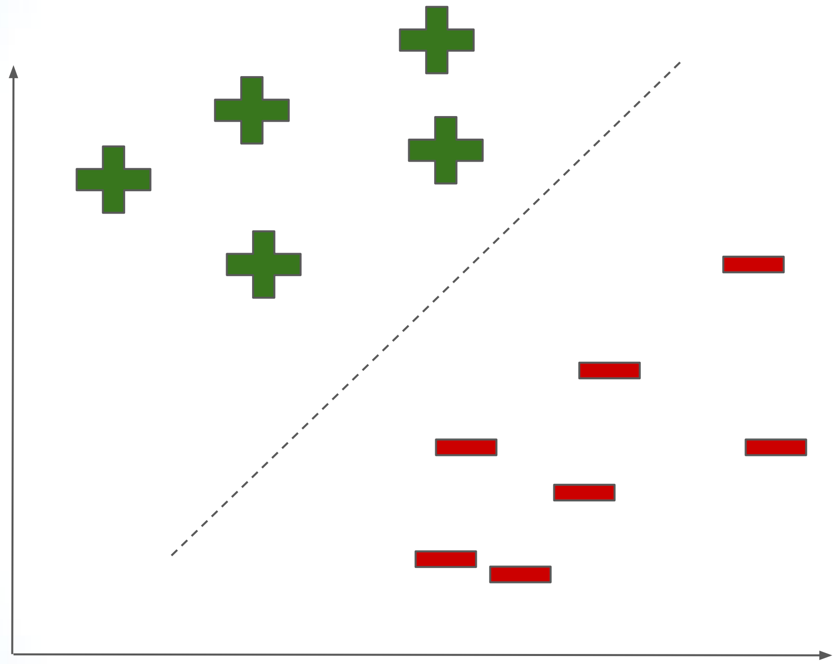


w1*x1 = w2*x2 + b

2*x1 = 4*x2 - 3
2*x1 - 4*x2 + 3 = 0

2*3 - 4*5 + 3 = 6 - 20 + 3 = -11
2*10 - 4*2 + 3 = 20 - 8 + 3 = 15

Linear Classification - 2 dim



w1*x1 = w2*x2 + b

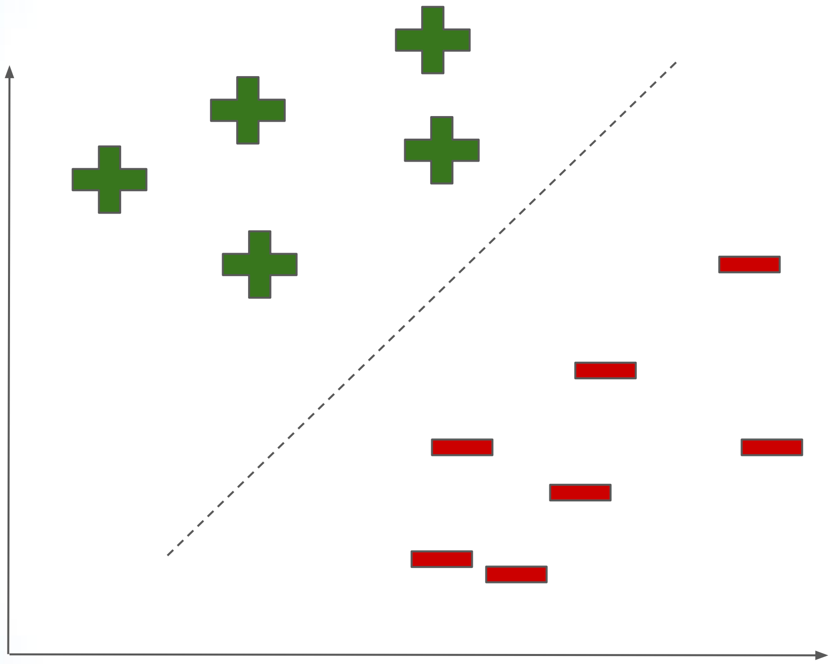
f(X,W) = w1*x1 + w2*x2 =

For simplicity We are going to eliminate the bias term Which can be added as a vectors of ones

Y-DATA SCHOOL OF DATA SCIENCE

Yandex

Linear Classification - 2 dim



$w1*x1 = w2*x2 + b$

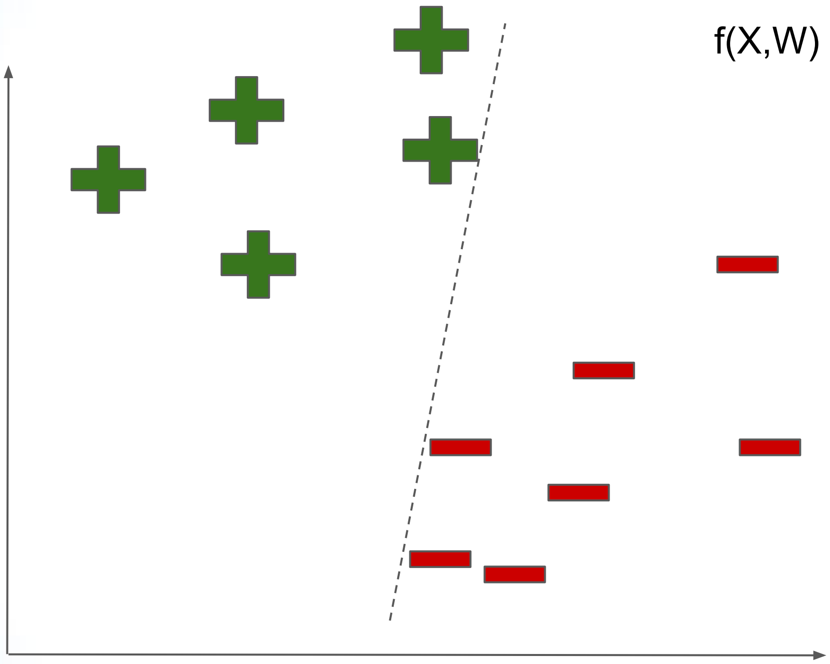
$f(X,W) = w1*x1 + w2*x2 =$
$= \Sigma WX = W^tX = 0$
$\Rightarrow sign(WX)$

For simplicity We write
W^tX as WX

Y-DATA SCHOOL OF DATA SCIENCE

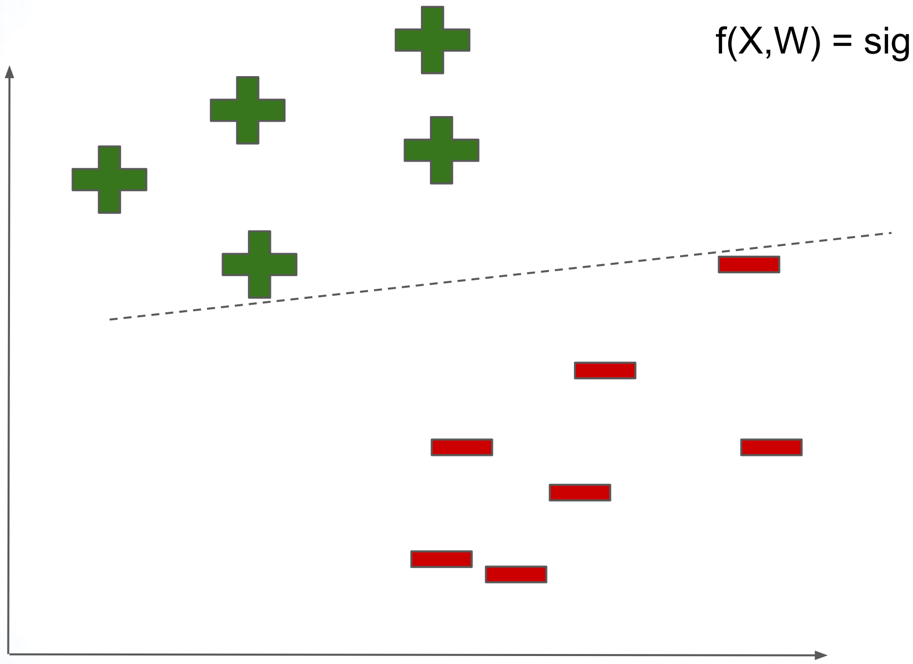
Yandex

Linear Classification



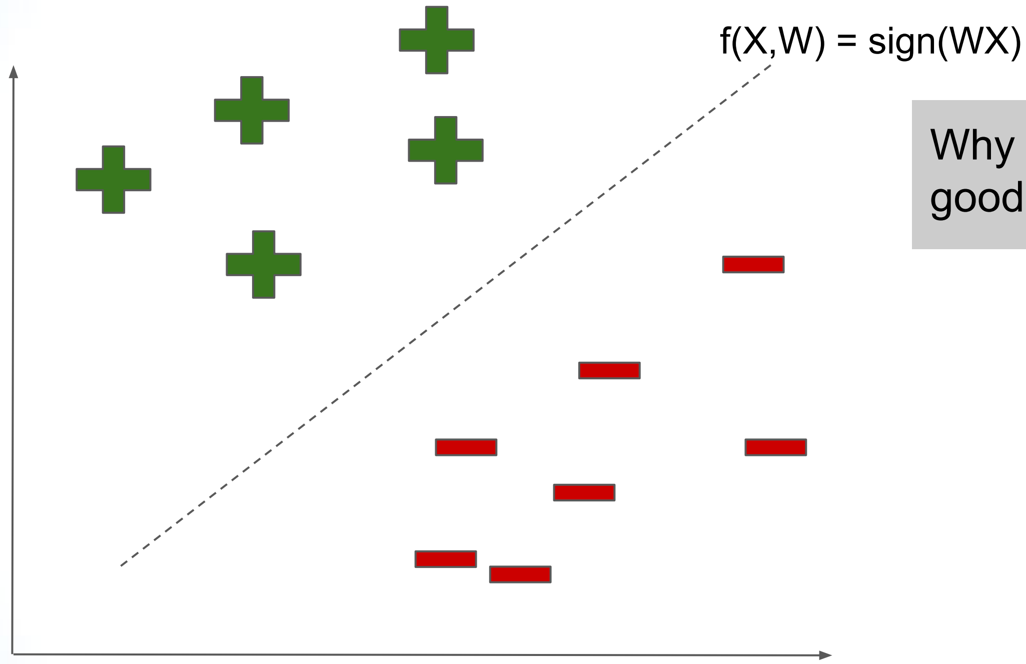
$f(X,W) = \text{sign}(WX)$

Linear Classification



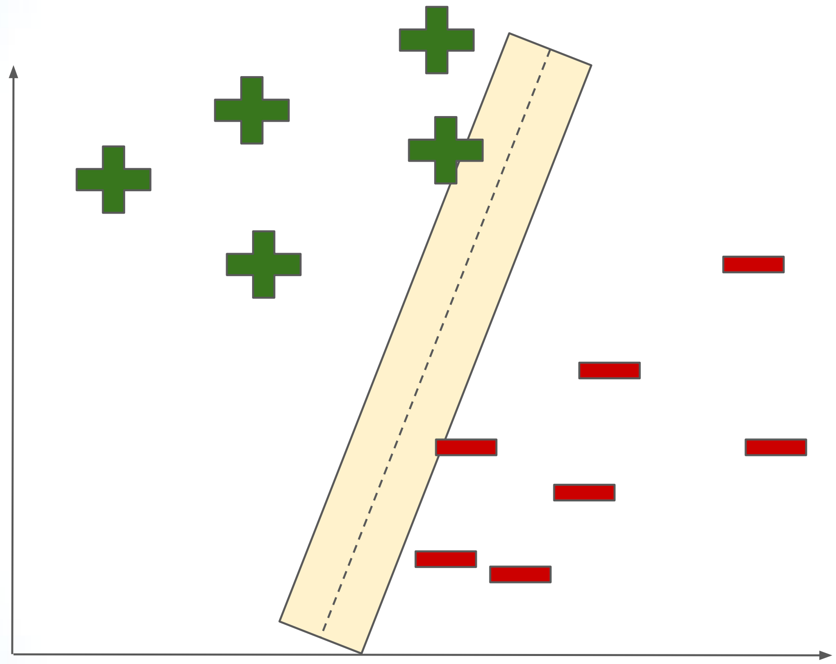
$f(X,W) = \text{sign}(WX)$

Linear Classification



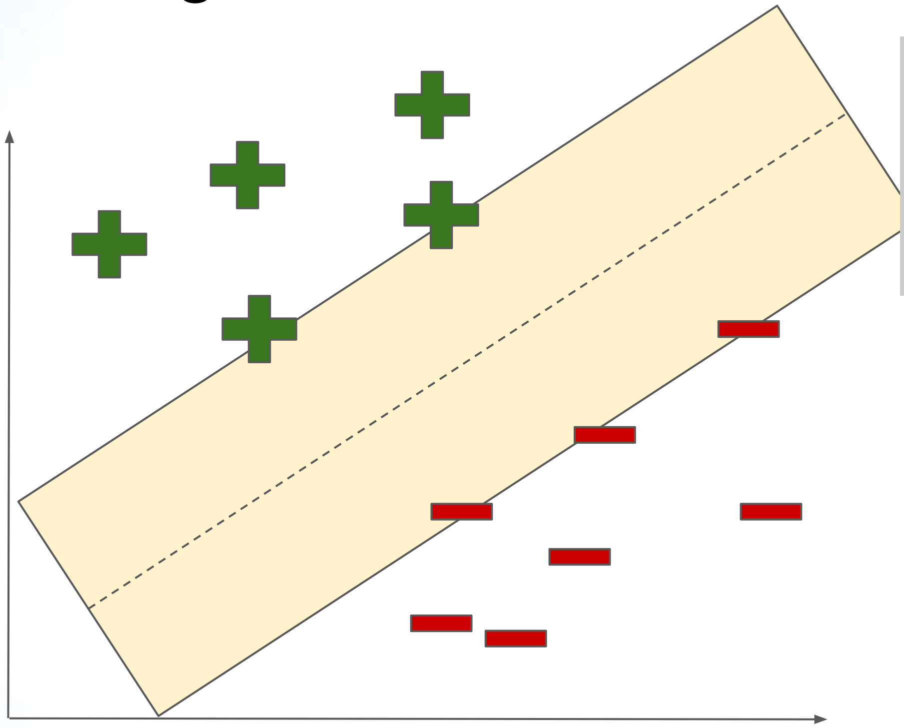
Why is this seems as a good separator?

Classifier Margin



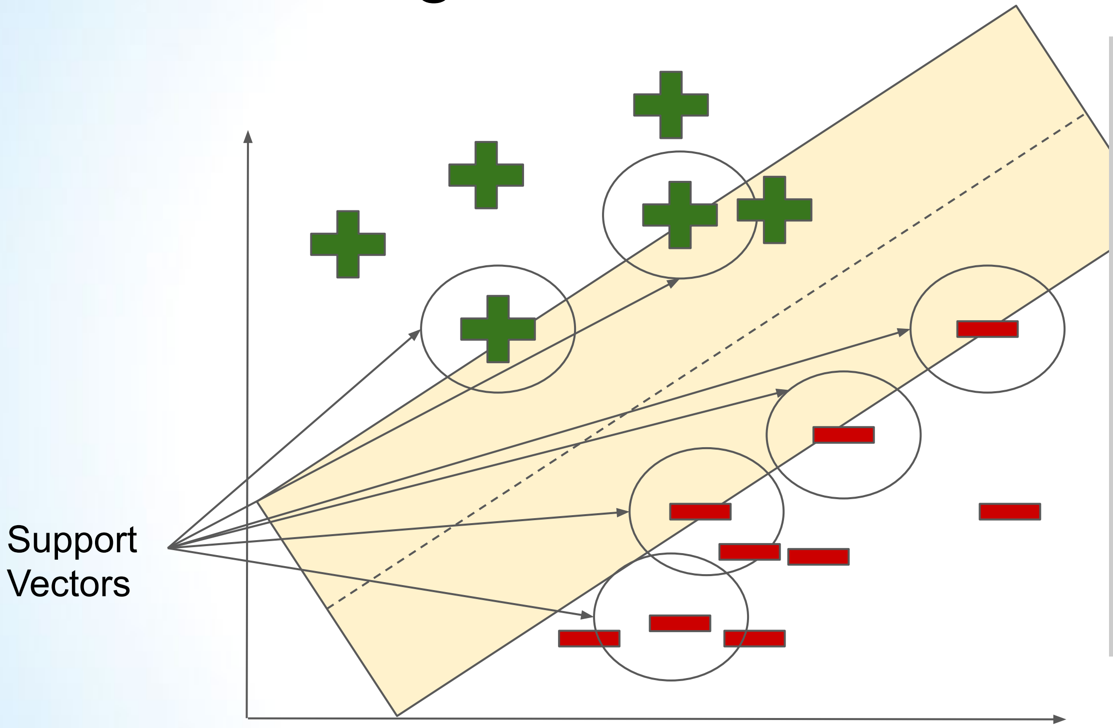
A margin in linear classifiers is the boundary width the touches the datapoint

Maximum margin



A maximum margin in linear classifiers is the Max boundary width the touches the datapoint

Maximum Margin



A maximum margin in linear classifiers is the Max boundary width the touches the datapoint

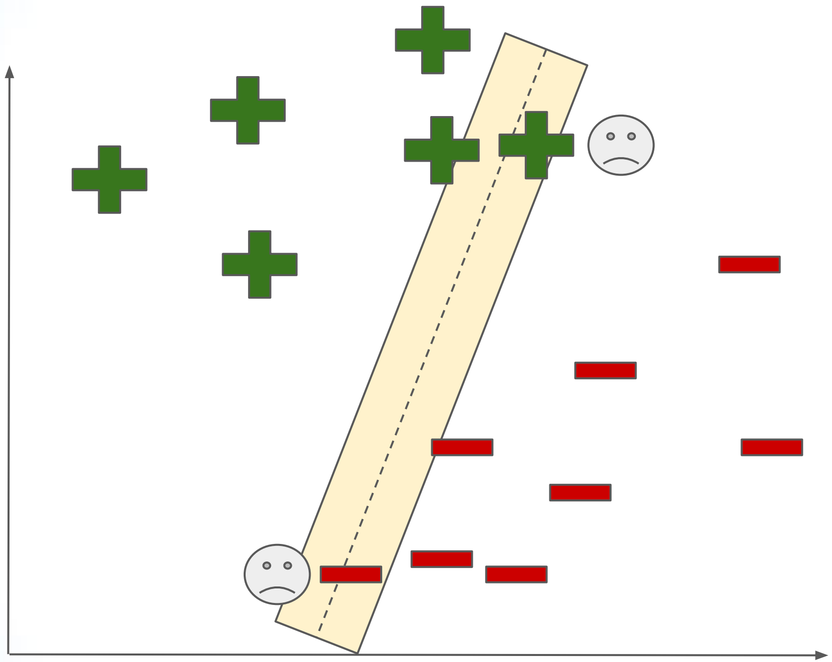
The points on the margins are called Support Vectors

VC dimension **can show that the maximum margin is a good approach to linearly separable problems.**

Y-DATA SCHOOL OF DATA SCIENCE

Yandex

Classifier margine



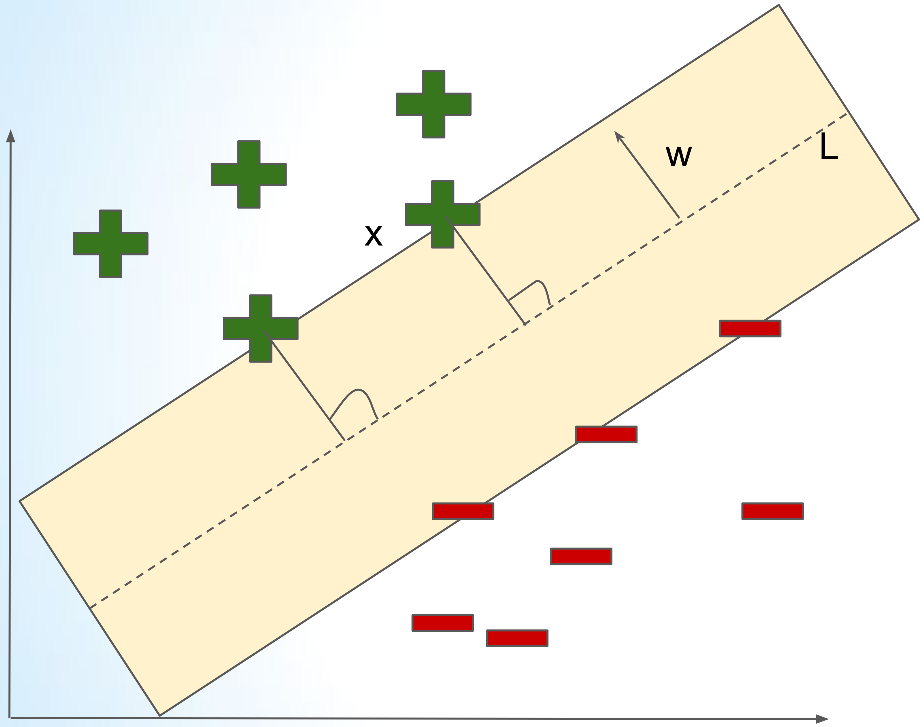
A maximum margin in linear classifiers is the Max boundary width the touches the datapoint

The points on the margins are called Support Vectors

VC dimension can show that the maximum margin is a good approach to linearly separable problems.

Allows a more flexibility around the decision boundary

Hard SVM



A distance of datapoint i can be defined by

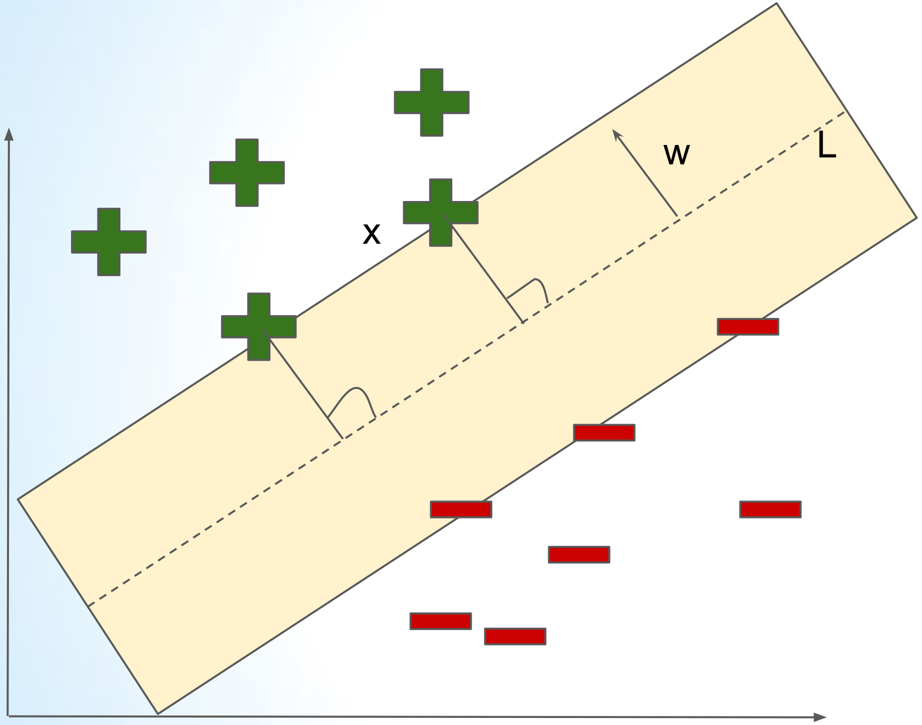
$$\rho_i = y_i w x_i$$

y is the point label which can be 1 or -1

Y-DATA SCHOOL OF DATA SCIENCE

Yandex

Hard SVM



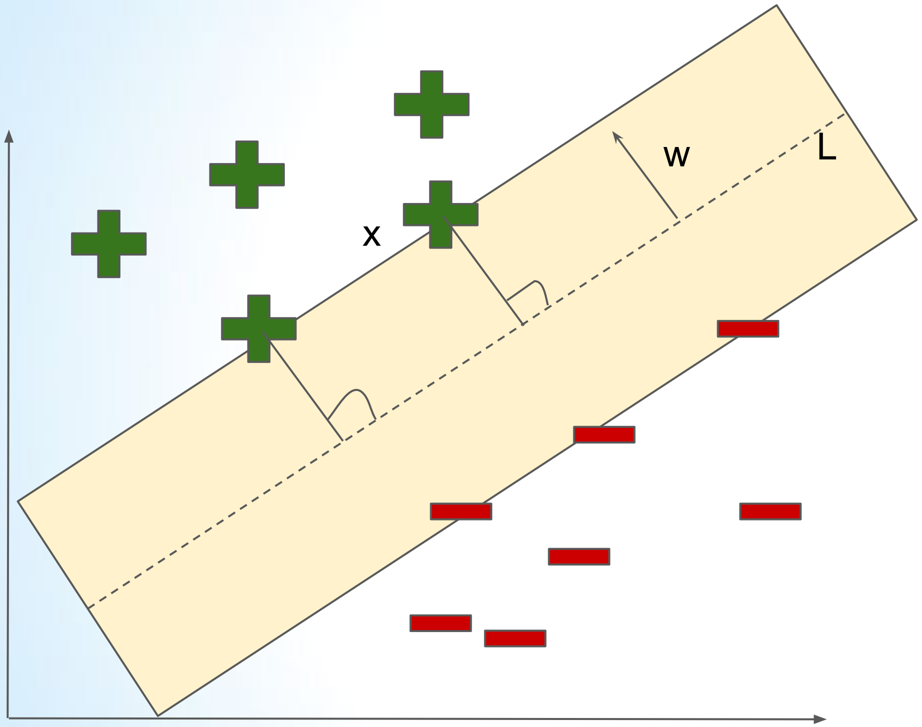
A distance of datapoint i can be defined by

$$\rho_i = y_i w x_i$$

The support vectors has the minimum margin

$$\rho = min\ y_i w x_i$$

Hard SVM



A distance of datapoint i can be defined by

$$\rho_i = y_i w x_i$$

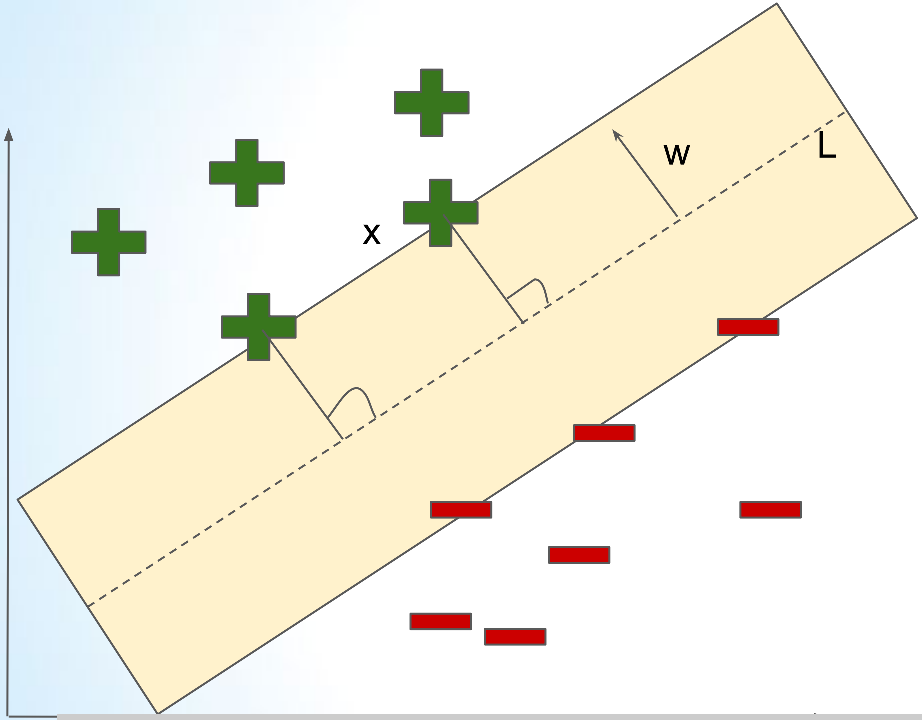
The support vectors has the minimum margin

$$\rho = min\ y_i w x_i$$

We want to find the w that maximize the margin

$$argmax_w = \rho$$

Hard SVM



Add ||w|| to constraint the w scaling
Define a margin that not depends on the scale of w

A distance of datapoint i can be defined by

$$\rho_i = \frac{y_i w x_i}{\|w\|}$$

The support vectors has the minimum margin

$$\rho = min \frac{y_i w x_i}{\|w\|}$$

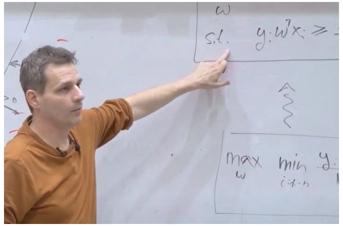
We want to find the w that maximize the margin

$$argmax_w = \rho$$

$$argmax_w \ \ min\frac{y_i w x_i}{\|w\|}$$

$$argmax_w \ min \frac{y_i w x_i}{\|w\|}$$

equivalent to

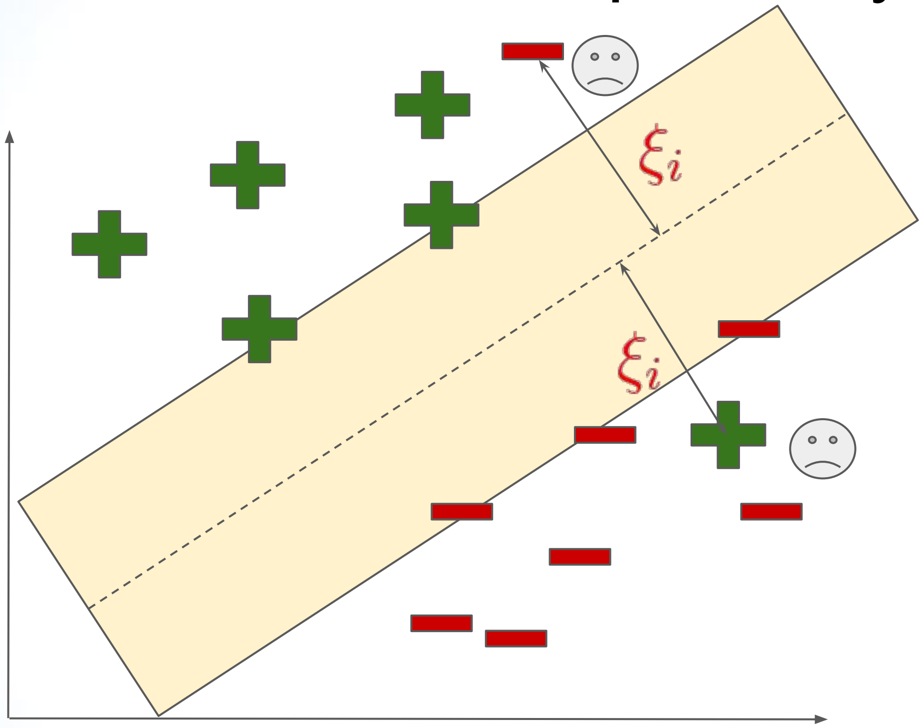


https://youtu.be/LceLJvKMbBk?t=3613

$$min_w \|w\|^2 \ \ s.t \ \ \forall i, y_i w x_i \geq 1$$

*Assumes full separability (1.1 in HW)

What if there is no linear separability?

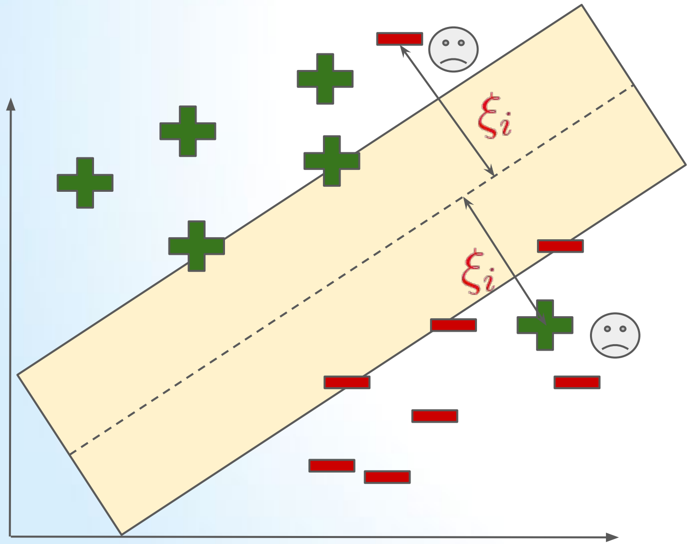


Y-DATA SCHOOL OF DATA SCIENCE

Yandex

Soft SVM - Supporting error data points

$$argmin_{w,\xi}\left(\lambda\left\|w\right\|^2 + C\frac{1}{m}\sum_{i=1}^{m}\xi_i\right)$$



$$s.t\ \forall y_i(wx_i) >= 1-\xi_i$$

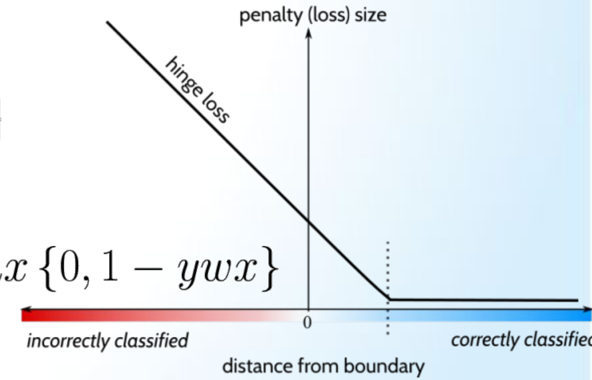
Soft SVM - Supporting error data points

$$argmin_{w,\xi}\left(\lambda\left\|w\right\|^2 + C\frac{1}{m}\sum_{i=1}^{m}\xi_i\right)$$

$$s.t\ \forall y_i(wx_i) >= 1 - \xi_i$$

$$argmin_{w,\xi}\left(\lambda\left\|w\right\|^2 + hinge(wx,y)\right)$$

$$max\left\{0, 1 - ywx\right\}$$



Y-DATA SCHOOL OF DATA SCIENCE

Yandex

Soft SVM - Supporting error data points

$$argmin_{w,\xi} \left(\lambda \left\| w \right\|^2 + C \frac{1}{m} \sum_{i=1}^{m} \xi_i \right)$$

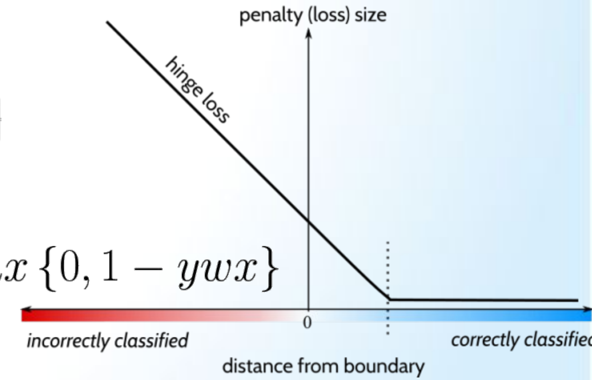
$$s.t\ \forall y_i(wx_i) >= 1 - \xi_i$$

$$argmin_{w,\xi} \left(\lambda \left\| w \right\|^2 + hinge(wx, y) \right)$$

regularization objective

But came from problem definition

$$max\,\{0, 1 - ywx\}$$



How we can solve it?

Gradient descent

Quadratic Programing

SGD for solving Soft-SVM

goal: Solve $\mathrm{argmin}_{\mathbf{w}} \left(\frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{i=1}^{m} \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}_i \rangle\} \right)$

parameter: T

initialize: $\boldsymbol{\theta}^{(1)} = \mathbf{0}$

for $t = 1, \ldots, T$

Let $\mathbf{w}^{(t)} = \frac{1}{\lambda t}\boldsymbol{\theta}^{(t)}$

Choose i uniformly at random from $[m]$

If $(y_i \langle \mathbf{w}^{(t)}, \mathbf{x}_i \rangle < 1)$

Set $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + y_i \mathbf{x}_i$

Else

Set $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)}$

output: $\bar{\mathbf{w}} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}^{(t)}$

https://leon.bottou.org/publications/pdf/lin-2006.pdf

Homework

1. **Implementing SVM called Pegasos (2011) - 55 (+ 10 bonus)**
    1. Implement Class - 35
    2. Test - 10
    3. Analyze param - 5
    4. Analyze learning - 5
    5. Mini-batch bonus - 10*
2. **The effect of imbalance on SVM - 15**
3. **Practical SVM in scikit-learn & hypertune - 10**
4. **Using different Kernels - 20**

Another Way to Solve SVM

And other tricks

From Primal to Dual

Primal

$$min\ \ f_0(x)$$

$$s.t\ \ f_i(x) \leq 0 \quad \forall i = 1..m$$

Original SVM definition

From Primal to Dual

Primal

$$\min \; f_0(x)$$

$$s.t \;\; f_i(x) \leq 0 \quad \forall i = 1..m$$

Original SVM definition

Dual

$$\underset{\alpha \geq 0}{L}(x, \alpha) = f_0(x) + \sum_{i=1} \alpha_i f_i(x)$$

$$g(\alpha) = min_x \; L(x, \alpha)$$

Dual definition that solvable with **linear solvers**

From Primal to Dual

Primal

$$\min\ f_0(x)$$

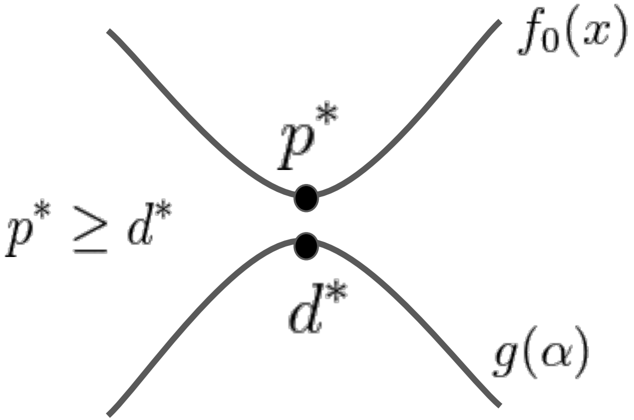
$$s.t\ \ f_i(x) \le 0 \quad \forall i = 1..m$$

Dual

$$\underset{\alpha \ge 0}{L(x, \alpha)} = f_0(x) + \sum_{i=1} \alpha_i f_i(x)$$

$$g(\alpha) = \min_x\ L(x, \alpha)$$

Original SVM definition



Dual definition that solvable with **linear solvers**

From Primal to Dual

Primal

$$\min \ f_0(x)$$

$$s.t \ f_i(x) \leq 0 \quad \forall i = 1..m$$

SVM definition

$$\min \ \|w\|^2$$

$$s.t \quad y_i w x_i \geq 1 \ \ \forall i = 1..m$$

Dual

$$\underset{\alpha \geq 0}{L(x, \alpha)} = f_0(x) + \sum_{i=1}^{m} \alpha f_i(x)$$

$$g(x) = \min_x \ L(x, \alpha)$$

From Primal to Dual

Primal

$$min\ \ f_0(x)$$

$$s.t\ \ f_i(x) \leq 0 \quad \forall i = 1..m$$

Dual

$$\underset{\alpha \geq 0}{L(x, \alpha)} = f_0(x) + \sum_{i=1}^{m} \alpha f_i(x)$$

$$g(x) = min_x\ \ L(x, \alpha)$$

SVM definition

$$min\ \ \|w\|^2$$

$$s.t \quad y_i w x_i \geq 1 \ \ \forall i = 1..m$$

Modify

$$min\ \frac{1}{2}\|w\|^2$$

$$s.t \quad 1 - y_i w x_i \leq 0 \ \ \forall i = 1..m$$

From Primal to Dual

Primal

$$\min \; f_0(x)$$

$$s.t \; f_i(x) \le 0 \quad \forall i = 1..m$$

SVM definition

$$\min \; \|w\|^2$$

$$s.t \quad y_i w x_i \ge 1 \;\; \forall i = 1..m$$

Modify

$$\min \; \frac{1}{2}\|w\|^2$$

$$s.t \quad 1 - y_i w x_i \le 0 \;\; \forall i = 1..m$$

Dual

$$\underset{\alpha \ge 0}{L(x, \alpha)} = f_0(x) + \sum_{i=1}^{m} \alpha f_i(x)$$

$$g(x) = \min_x \; L(x, \alpha)$$

$$\underset{\alpha \ge 0}{L(x, \alpha)} = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i (y_i w x_i - 1)$$

$$g(\alpha) = \min_x \; \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i (y_i w x_i - 1)$$

From Dual to Primal

$$\underset{\alpha \geq 0}{L(x, \alpha)} = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i (y_i w x_i - 1)$$

$$g(\alpha) = min_x \ \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i (y_i w x_i - 1)$$

From Dual to Primal

$$\underset{\alpha \geq 0}{L(x, \alpha)} = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{m} \alpha_i (y_i w x_i - 1)$$

$$g(\alpha) = min_x \ \frac{1}{2} \|w\|^2 - \sum_{i=1}^{m} \alpha_i (y_i w x_i - 1)$$

$$\frac{\partial L}{\partial w} = 0$$

From Dual to Primal

$$\underset{\alpha\geq 0}{L(x,\alpha)} = \frac{1}{2}\left\|w\right\|^2 - \sum_{i=1}^{m}\alpha_i(y_i w x_i - 1)$$

$$g(\alpha) = min_x \; \frac{1}{2}\left\|w\right\|^2 - \sum_{i=1}^{m}\alpha_i(y_i w x_i - 1)$$

$$\frac{\partial L}{\partial w} = 0$$

$$w - \sum_{i=1}^{m}\alpha_i y_i x_i = 0$$

$$w* = \sum_{i=1}^{m}\alpha_i y_i x_i$$

$$\alpha_i \geq 0$$

Y-DATA SCHOOL OF DATA SCIENCE

Yandex

Lagrange coefficients for each sample in the training set

All possible pairs in the training set

Optimization Definition

$$max_{\alpha_k} \sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l y_k y_l x_k x_l$$

Constraints

$$s.t \quad 0 \leq \alpha_k \leq C \quad \sum_{k=1}^{R} \alpha_k y_k = 0$$

Y-DATA SCHOOL OF DATA SCIENCE

Yandex

Optimization Definition	$max_{\alpha_k} \sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l y_k y_l x_k x_l$
Constraints	$s.t \quad 0 \leq \alpha_k \leq C \quad \sum_{k=1}^{R} \alpha_k y_k = 0$
Back to Primal	$w = \sum_{k=1}^{R} \alpha_k y_k x_k$

Y-DATA SCHOOL OF DATA SCIENCE

Yandex

Optimization Definition	$$max_{\alpha_k} \sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l y_k y_l x_k x_l$$
Constraints	$$s.t \quad 0 \leq \alpha_k \leq C \quad \sum_{k=1}^{R} \alpha_k y_k = 0$$
Back to Primal	$$w = \sum_{k=1}^{R} \alpha_k y_k x_k$$
Predict	$$f(x, w) = sign(w, x)$$

Optimization Definition

$$max_{\alpha_k} \sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l y_k y_l x_k x_l$$

Constraints

$$s.t \quad 0 \leq \alpha_k \leq C \quad \sum_{k=1}^{R} \alpha_k y_k = 0$$

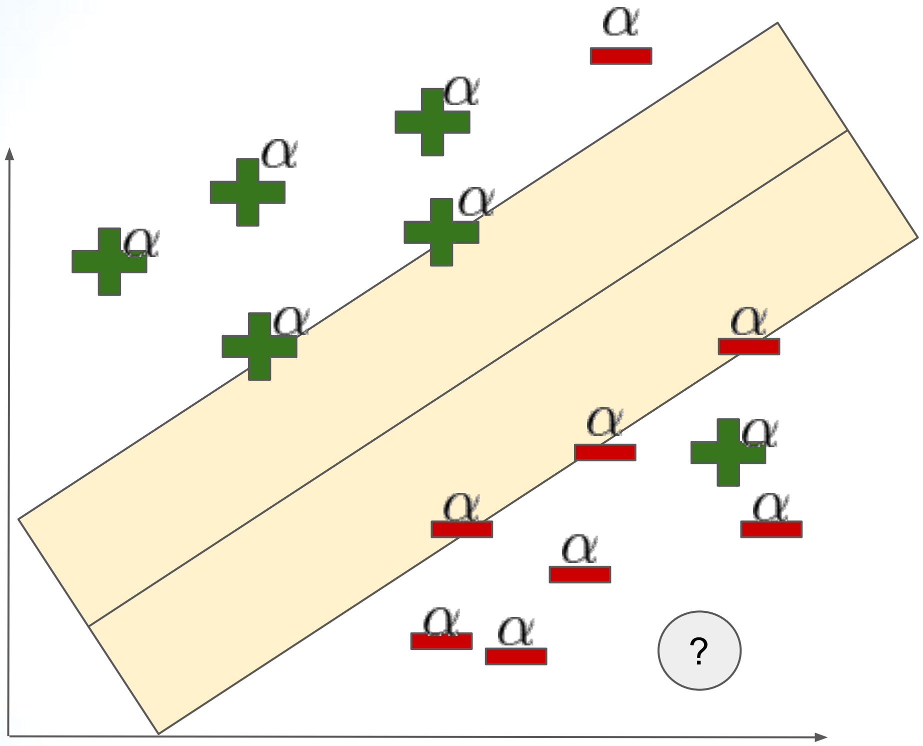
Back to Primal

$$w = \sum_{k=1}^{R} \alpha_k y_k x_k$$

Predict

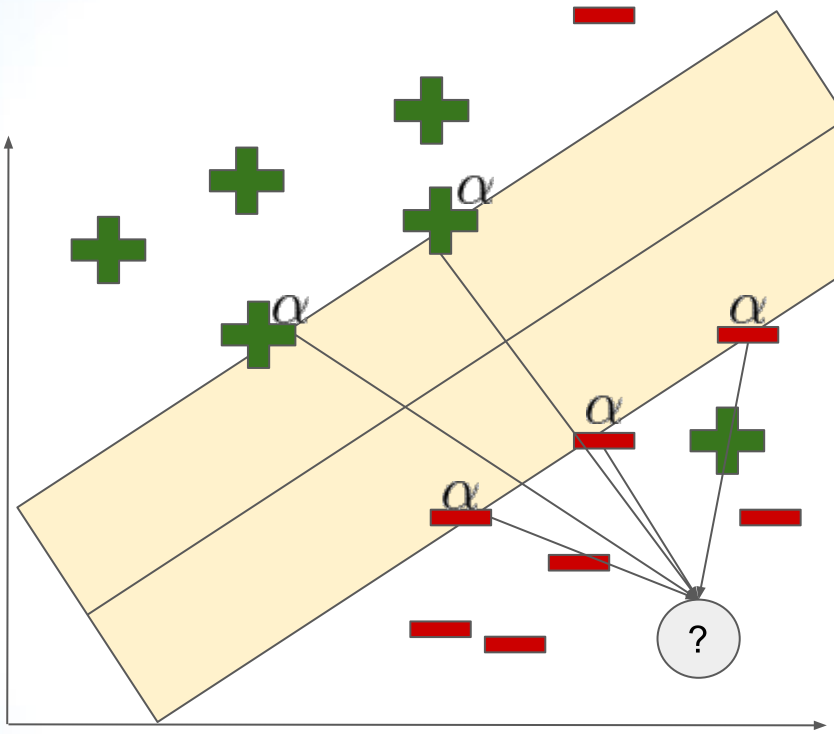
$$f(x, w) = sign(w, x) \Rightarrow f(x, w) = sign(\sum_{k=1}^{R} \alpha_k y_k x_k x)$$

Applying



$$f(x,w) = sign(\sum_{k=1}^{R} \alpha_k y_k x_k x)$$

Applying

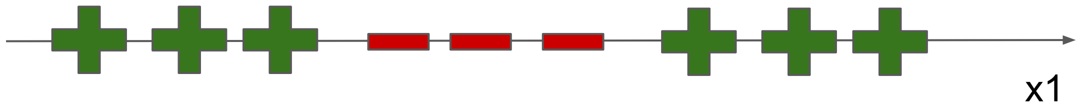


$$f(x, w) = sign(\sum_{k=1}^{R} \alpha_k y_k x_k x)$$

Reminds you something?

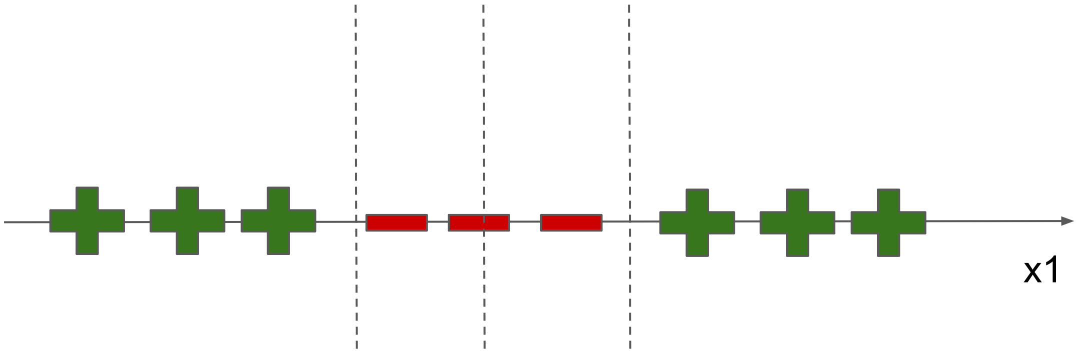
Dealing with non-linearity

How to separate with linear classifier?



Dealing with non-linearity

How to separate with linear classifier?

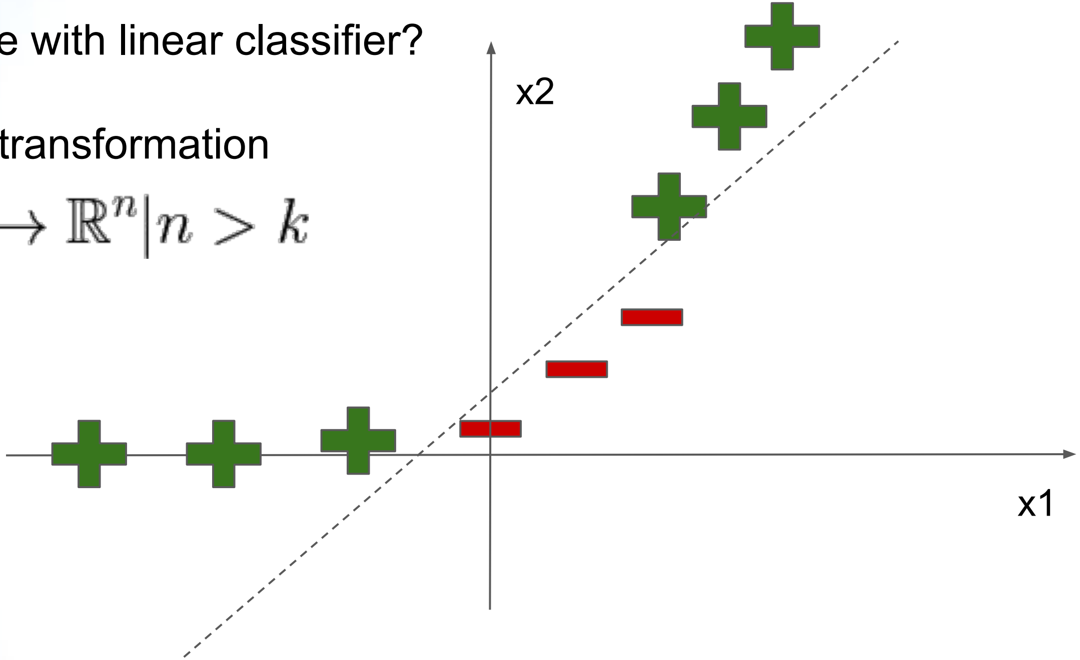


Dealing with non-linearity

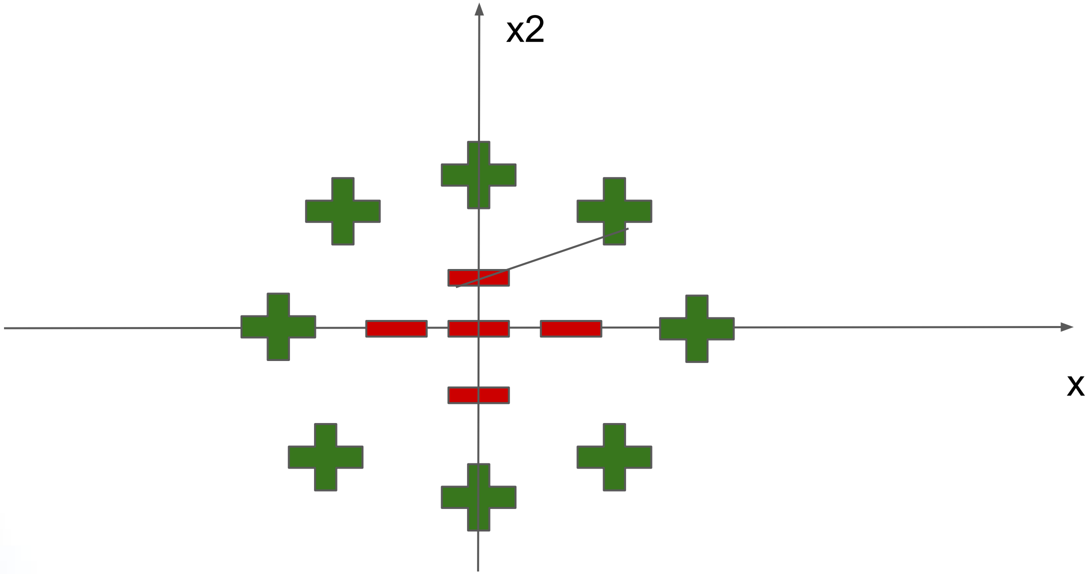
How to separate with linear classifier?

Use non-linear transformation

$$\phi(X) : \mathbb{R}^k \to \mathbb{R}^n | n > k$$

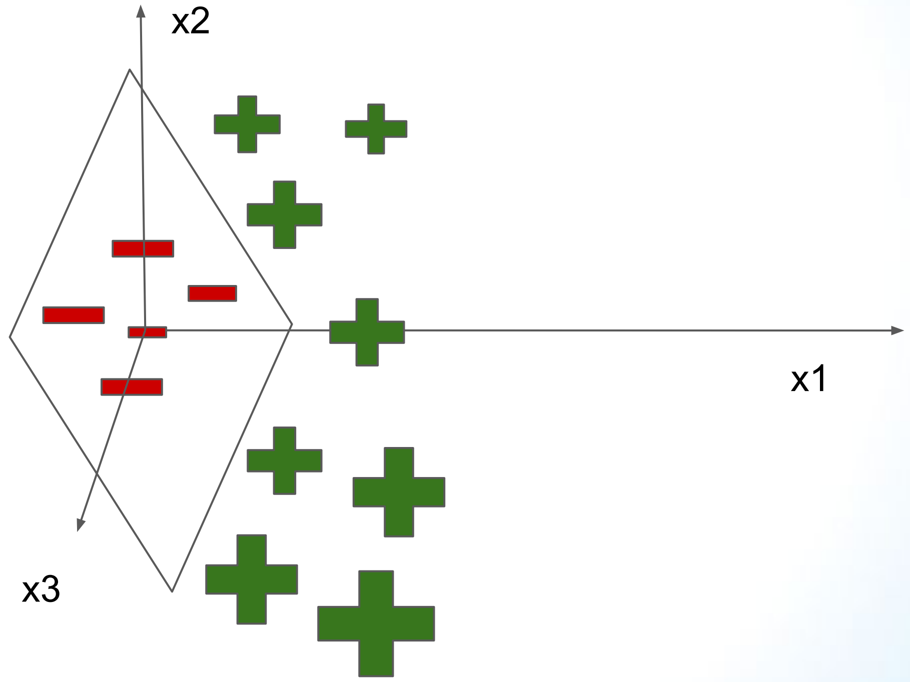


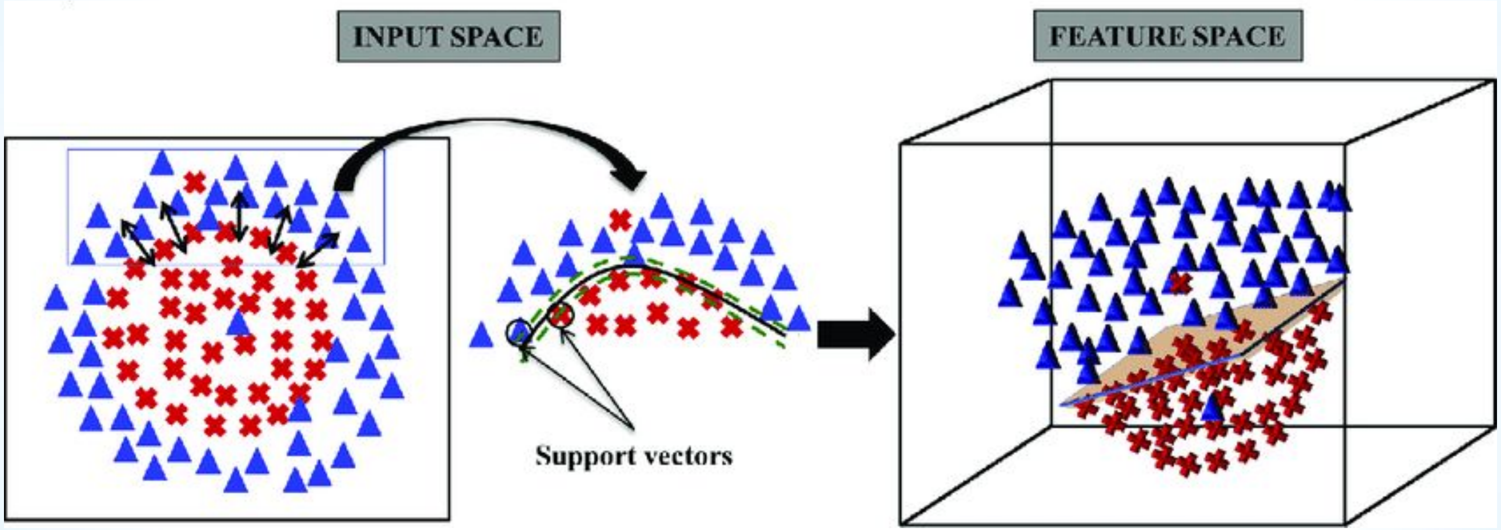
Dealing with non-linearity



Dealing with non-linearity

$$\phi(X) : \mathbb{R}^k \rightarrow \mathbb{R}^n | n > k$$




INPUT SPACE
FEATURE SPACE
Support vectors

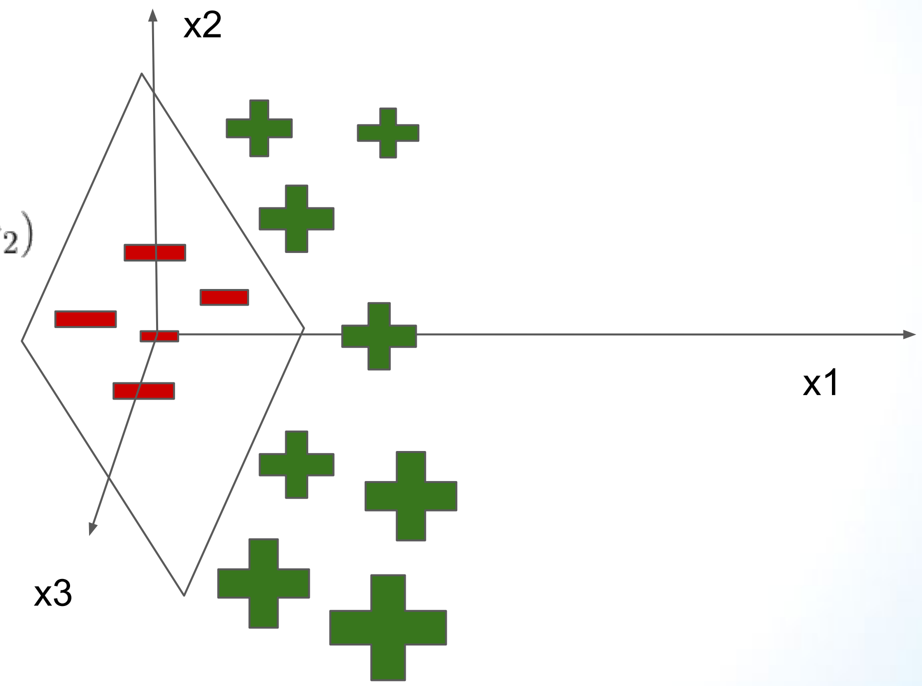
Dealing with non-linearity

How to separate with linear classifier?

$$\phi(X) : \mathbb{R}^k \rightarrow \mathbb{R}^n | n > k$$

$$(x_1, x_2) \mapsto \theta(X) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

A capital X = a datapoint



The Kernel Trick

To support this transformation We would need to compute all these features for each sample

The Kernel Trick

To support this transformation We would need to compute all these features for each sample

The Solution: the Kernel trick!
Use a dot product in feature space can be computed as kernel function

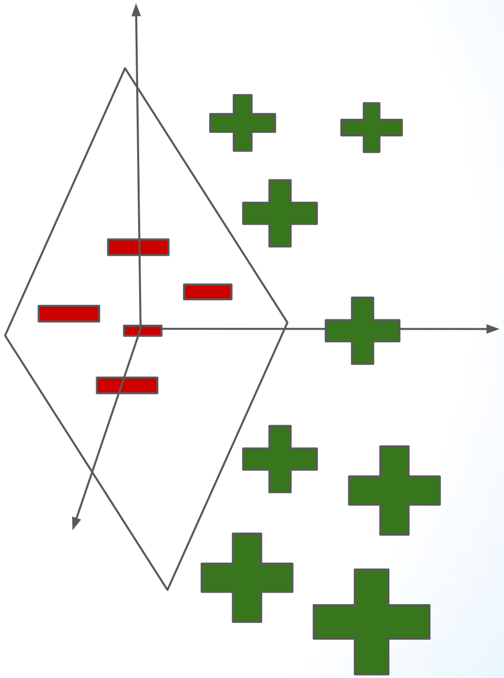
$$K(x_i, x_j) = \phi(x_i)\phi(x_j)$$

The Kernel Trick

$$(x_1, x_2) \mapsto \theta(X) = (x_1^2, x_2^2, \sqrt{2}x_1 x_2)$$

$$K(X_i, X_j) = \phi(X_i)\phi(X_j) = (X_i X_j)^2$$

$$= (X_{i_1}^2, X_{i_2}^2, \sqrt{2}X_{i_1}X_{i_2})(X_{j_1}^2, X_{j_2}^2, \sqrt{2}X_{j_1}X_{j_2})^T =$$
$$(X_{i_1}X_{j_1} + X_{i_2}X_{j_2})^2 = (X_i X_j)^2$$



The Kernel Trick

We would need to compute all these features!

The Solution: the Kernel trick!

Use a dot product in feature space can be computed as kernel function

$$K(x_i, x_j) = \phi(x_i)\phi(x_j)$$

Where do we have dot product in SVM?

Optimization Definition

$$max_{\alpha_k} \sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l y_k y_l x_k x_l$$

Constraints

$$s.t \quad 0 \leq \alpha_k \leq C \quad \sum_{k=1}^{R} \alpha_k y_k = 0$$

Back to Primal

$$w = \sum_{k=1}^{R} \alpha_k y_k x_k$$

Predict

$$f(x, w) = sign(w, x) \qquad f(x, w) = sign(\sum_{k=1}^{R} \alpha_k y_k x_k x)$$

Y-DATA SCHOOL OF DATA SCIENCE

Yandex

Optimization Definition

$$max_{\alpha_k} \sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l y_k y_l \boxed{x_k x_l}$$

Constraints

$$s.t \quad 0 \leq \alpha_k \leq C \quad \sum_{k=1}^{R} \alpha_k y_k = 0$$

Back to Primal

$$w = \sum_{k=1}^{R} \alpha_k y_k x_k$$

Predict

$$f(x, w) = sign(w, x) \qquad f(x, w) = sign(\sum_{k=1}^{R} \alpha_k y_k x_k x)$$

Optimization Definition

$$max_{\alpha_k} \sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l y_k y_l \, K(x_k x_l)$$

Constraints

$$s.t \quad 0 \leq \alpha_k \leq C \quad \sum_{k=1}^{R} \alpha_k y_k = 0$$

Back to Primal

$$w = \sum_{k=1}^{R} \alpha_k y_k x_k$$

Predict

$$f(x, w) = sign(w, x) \qquad f(x, w) = sign(\sum_{k=1}^{R} \alpha_k y_k x_k x)$$

Y-DATA SCHOOL OF DATA SCIENCE

Yandex

Optimization Definition	$$max_{\alpha_k} \sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l y_k y_l \, K(x_k x_l)$$
Constraints	$$s.t \quad 0 \leq \alpha_k \leq C \quad \sum_{k=1}^{R} \alpha_k y_k = 0$$
Back to Primal	$$w = \sum_{k=1}^{R} \alpha_k y_k x_k$$
Predict	$$f(x, w) = sign(w, x) \qquad f(x, w) = sign(\sum_{k=1}^{R} \alpha_k y_k x_k x)$$

We still need to compute all the kernels pairs, but we don't need to maintain the feature space

Good Kernel Functions for SVM

On top the polynomial kernel function there are more suitable ones:

Radial Basis Function (RBF):

$$K(X_i, X_j) = exp\left(-\frac{(X_i - X_j)^2}{2\sigma^2}\right)$$

Tanh Function (nn):

$$K(X_i, X_j) = tanh\left(\kappa X_i X_j - \delta\right)$$

Good Kernel Functions for SVM

On top the polynomial kernel function there are more suitable ones:

Radial Basis Function (RBF):

$$K(X_i, X_j) = exp\left(-\frac{(X_i - X_j)^2}{2\sigma^2}\right)$$

hypertune

Tanh Function (nn):

$$K(X_i, X_j) = tanh\left(\kappa X_i X_j - \delta\right)$$

SVM Pros & Cons

Good ➕

- Learn many non linear pattern
- Pick "conservative" hypotheses, that are less likely to overfit the data,
- Good for Small datasets with though target pattern
- “Only” 2 params - C, Kernel Function

SVM Pros & Cons

Good

- Learn many non linear pattern
- Pick “conservative” hypotheses, that are less likely to overfit the data,
- Good for Small datasets with though target pattern
- “Only” 2 params - C, Kernel Function

Bad

- O(n^2-3) runtime depends on C and the kernel (n number of datapoints)
- O(n^2) memory to compute all the pairwise kernels - 5-10K datapoints
- different values for the Kernel prams

Q&A

VC Dimension - https://www.youtube.com/watch?v=puDzy2XmR5c

https://winvector.github.io/margin/margin.pdf
https://youtu.be/LceLJvKMbBk?t=7311
https://youtu.be/fB47g3QM0sk?t=839

Shalev-Shwartz, S., Singer, Y., Srebro, N., & Cotter, A. (2011). Pegasos: Primal estimated sub-gradient solver for svm. Mathematical programming, 127(1), 3-30. [[pdf](http://www.ee.oulu.fi/research/imag/courses/Vedaldi/ShalevSiSr07.pdf)]

Schölkopf, B., Williamson, R. C., Smola, A. J., Shawe-Taylor, J., & Platt, J. C. (2000). Support vector method for novelty detection. In Advances in neural information processing systems (pp. 582-588). [[pdf](http://papers.nips.cc/paper/1723-support-vector-method-for-novelty-detection.pdf)]

Livni, R., Crammer, K. & Globerson, A.. (2012). A Simple Geometric Interpretation of SVM using Stochastic Adversaries. Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics, in PMLR 22:722-730. [[pdf](http://proceedings.mlr.press/v22/livni12/livni12.pdf)]

From Primal to Dual

$$\begin{array}{ll} \text{minimize} & (1/2)x^T P_0 x + q_0^T x + r_0 \\ \text{subject to} & (1/2)x^T P_i x + q_i^T x + r_i \leq 0, \quad i = 1, 2, \ldots, m \\ & Ax = b \end{array}$$

From Primal to Dual

$$\begin{array}{ll} \text{minimize} & (1/2)x^T P_0 x + q_0^T x + r_0 \\ \text{subject to} & (1/2)x^T P_i x + q_i^T x + r_i \le 0, \quad i = 1, 2, \ldots, m \\ & Ax = b \end{array}$$

Works in convex problems
Transform the problem from minimize to maximize
Use lagrange coefficients

$$\begin{array}{ll} \text{maximize} & -(1/2)q(\lambda)^T P(\lambda)^{-1} q(\lambda) + r(\lambda) \\ \text{subject to} & \lambda \succeq 0 \end{array}$$

where:

$$P(\lambda) = P_0 + \sum_{i=1}^{m} \lambda_i P_i, \quad q(\lambda) = q_0 + \sum_{i=1}^{m} \lambda_i q_i, \quad r(\lambda) = r_0 + \sum_{i=1}^{m} \lambda_i r_i$$

Y-DATA
SCHOOL OF DATA SCIENCE

Yandex

From Primal to Dual

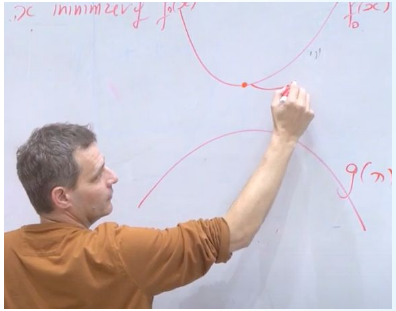
$$\begin{array}{ll} \text{minimize} & (1/2)x^T P_0 x + q_0^T x + r_0 \\ \text{subject to} & (1/2)x^T P_i x + q_i^T x + r_i \le 0, \quad i = 1, 2, \ldots, m \\ & Ax = b \end{array}$$

Works in convex problems
Transform the problem from minimize to maximize
Use lagrange coefficients

$$\begin{array}{ll} \text{maximize} & -(1/2)q(\lambda)^T P(\lambda)^{-1} q(\lambda) + r(\lambda) \\ \text{subject to} & \lambda \succeq 0 \end{array}$$

where:

$$P(\lambda) = P_0 + \sum_{i=1}^{m} \lambda_i P_i, \quad q(\lambda) = q_0 + \sum_{i=1}^{m} \lambda_i q_i, \quad r(\lambda) = r_0 + \sum_{i=1}^{m} \lambda_i r_i$$



https://youtu.be/LceLJvKMbBk?t=7311
https://youtu.be/fB47g3QM0sk?t=839