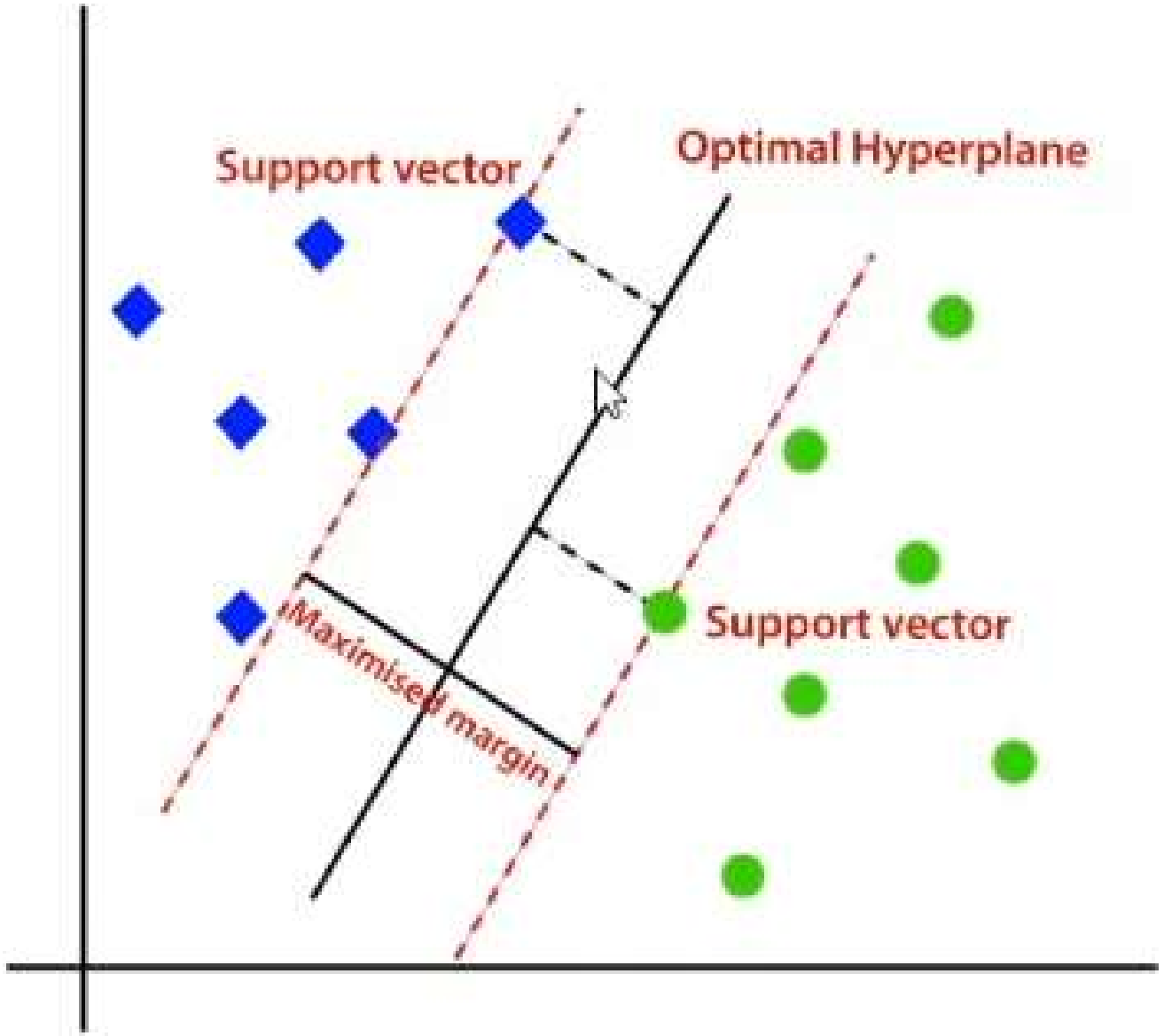

Support vector
Optimal Hyperplane
Maximised margin
Support vector

Support Vector Machine (SVM)

- many tasks can not be solved by classical programming or no mathematical model of the problem is available
- construction of machines capable of learning from experience
- this approach to problem solving as learning methodology
- requires dual representations, feature spaces, learning theory and optimization theory
- form the basis for the SVM concept and subfield of machine learning
- Applications:
- hand written character recognition - train computer to recognize from examples - the way humans learn to read
- finding genes in a DNA sequence, filtering email,
- detecting or recognizing objects in machine vision

Why Learning Difficult

- method required to derive output from a set of inputs explicitly translate the method into a sequence of instructions
- no known method for computing the desired output from a set of inputs or computation may be very expensive
- for example, modeling a complex chemical reaction, where the precise interactions of the different reactants are not known
- classification of protein types based on the DNA sequence from which they are generated
- classification of credit applications - who will default and who will repay
- can not be solved by traditional programming
- can not be specified the precise method by which the correct output can be computed from the input data

Supervised Learning

- learn the input/output functionality from examples
- approach of using examples to synthesize programms
- examples are input/output pairs and called training data
- functional relationship mapping inputs to output difficult if output is corrupted by noise
- when an underlying function from inputs to outputs exists - referred as the target function
- the estimate of the target function which is learned or output by the learning algorithm is known as the solution of the learning problem
- in case of classification this function is referred to as the decision function
- the solution is chosen from a set of candidate functions which map from the input space to output domain

Candidate Functions: Hypotheses

- chosen a particular set or class of candidate functions known as hypotheses before trying to learn the correct function
- the algorithm which takes the training data as input and selects a hypothesis from the hypothesis space is referred to as the learning algorithm
- learning approach with binary output is referred as binary classification
- finite number of categories as multi class classification
- for real valued output the problem becomes known as regression
- unsupervised learning - no output values available
- learning includes density estimation, learning the support of a distribution, clustering

Models of Learning

- query to the environment - query learning
- query the environment about the output associated with a particular input
- reinforcement learning - learner has a range of actions at their disposal
- which they can take to attempt to move towards states where they can expect high rewards
- batch learning - all data at the start of learning
- on-line learning - one example at a time
- Complexity of Learning Algorithm
- the algorithm that runs in time polynomial in the size of the input
- if the class is too large the complexity of learning from examples can become prohibitive

Issues in Learning

- function that maps may not have a simple representation
- may not be verified training data is noisy - no guarantee that underlying function will map the data correctly
- e.g. credit checking, classification of web pages into categories – no exact science
- hypothesis is consistent for the training data but may not make correct classification for unseen data
- the ability of a hypothesis to correctly classify data not in the training set
- known as its generalisation this property need to be optimized

Why SVM

- learning algorithms - have common problems:
- case of local minima, limited number of training data,
- size of output hypothesis very large and impractical overfitting, poor generalisation
- learning algorithm controlled by a large number of parameters chosen by tuning heuristics make system difficult and unreliable to use
- SVM are learning systems that use a hypothesis space of linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory introduced by Vapnik
- SVM needs: hypothesis space and its representation, learning bias, learning algorithm

Linear Learning Machines

- In supervised learning, the learning machine is given a training set of examples (inputs) with associated labels (output values)
- The examples are in the form of attribute vectors, so that the input space is a subset of $\mathcal{R}^n$
- Once the attribute vectors are available, a number of sets of hypotheses could be chosen for the problem
- Linear classification
- Binary classification is frequently performed by using a real valued function $f : X \subseteq \mathcal{R}^n \rightarrow \mathcal{R}$
- the input $\mathbf{x} = (x_1, \ldots, x_n)'$ is assigned to the positive class, if $f(\mathbf{x}) \geq 0$ and otherwise to the negative class

Linear Learning Machines

- $f(\mathbf{x})$ is a linear function of $\mathbf{x} \in X$; can be written as

$$f(\mathbf{x}) = \langle \mathbf{w} \cdot \mathbf{x} \rangle + b = \sum_{i=1}^{n} w_i x_i + b$$

- $(\mathbf{w}, b) \in \mathcal{R}^n \times \mathcal{R}$ are the parameters that control the function and the decision rule is given by $\text{sgn}(f(\mathbf{x}))$
- The learning methodology implies that these parameters must be learned from the data
- A geometric interpretation of such hypothesis is that the input space X is split into two parts by the hyperplane defined by the equation

$$\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$$

Linear Learning Machines

- A hyperplane is an affine subspace of dimension $n-1$ which divides the space into two half spaces which correspond to the inputs of the two distinct classes
- hyperplane is the line with the positive region above and the negative region below
- the vector $\mathbf{w}$ defines a direction perpendicular to the hyperplane, varying the value of b moves the hyperplane parallel to itself
- It is therefore clear that a representation involving $n+1$ free parameters is necessary, if one wants to represent all possible hyperplanes in $\mathcal{R}^n$
- classifier called linear discriminants developed by Fisher
- $\mathbf{w}$ weight vector and b bias

Why Feature Space

- X input space and Y output domain
- $X \subseteq \mathcal{R}^n$, for binary classification $Y = \{-1, 1\}$, for m class classification $Y = \{1, 2, \ldots, m\}$, for regression $Y \subseteq \mathcal{R}$
- training data $S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)) \subseteq (X \times Y)^l$
- l number of examples $\mathbf{x}_i$ as examples or instances y_i as labels i
- limited computational power of linear learning machines reported by Minsky and Papert in 1960
- complex real world applications required more expressive hypothesis spaces than linear functions
- i.e. the target concept can not be expressed as a simple linear combination of given attributes
- abstract features of the data be exploited

Kernel representation

- an alternative solution by projecting the data into a high dimensional feature space to increase the computational power of the linear learning machines
- the advantage of using the machines in the dual representation
- the attraction of the kernel method: the learning algorithms and theory can largely be decoupled from the specifics of the application area
- which must simply be encoded into the design of an appropriate kernel function
- kernel function main building block of SVM

Kernel function: Learning in Feature Space

- appropriately choosing kernel function,
- implicitly perform a non-linear mapping to a high dimensional feature space without increasing the number of tunable parameters,
- provided the kernel computes the inner product of the feature vectors corresponding to the two inputs
- complexity of the target function to be learned depends on the way it is represented and the difficulty of the learning task can vary
- ideally a representation that matches the specific learning problem should be chosen
- common strategy in machine learning involves changing the representation of the data

$$\mathbf{x} = (x_1, \ldots, x_n) \rightarrow \phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \ldots, \phi_N(\mathbf{x}))$$

- mapping the input space X into a new space $F = \{\phi(\mathbf{x}) | \mathbf{x} \in X\}$

Higher Dimensional Representation: Example

- example of target function

$$f(m_1, m_2, r) = C\frac{m_1 m_2}{r^2}$$

- Newton's law of gravitation
- expressing the gravitational force between two bodies with masses m_1, m_2 and separation r
- the law is expressed in terms of the observable quantities, mass and distance

$$(m_1, m_2, r) \rightarrow (x, y, z) = (\ln m_1, \ln m_2, \ln r)$$

gives the representation

$$g(x, y, z) = \ln f(m_1, m_2, r) = \ln C + \ln m_1 + \ln m_2 - 2\ln r = c + x + y - 2z$$

Feature Space and Dimensionality

$$\mathbf{x} = (x_1, \ldots, x_n) \rightarrow \phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \ldots, \phi_d(\mathbf{x})) \quad d < n$$

- computational and generalisation performance can degrade as the number of feature grows, referred as the curse of dimensionality
- difficulties with high dimensional feature spaces are unfortunate
- degradation of performance can be avoided in the SVM
- gravitation law $\mathbf{x} = (p_1^x, p_1^y, p_1^z, p_2^x, p_2^y, p_2^z, m_1, m_2)$
- mapping $\phi : \mathcal{R}^8 \rightarrow \mathcal{R}^3$

$$\mathbf{x} = (p_1^x, p_1^y, p_1^z, p_2^x, p_2^y, p_2^z, m_1, m_2) \rightarrow$$

$$\phi(\mathbf{x}) = \left(\sqrt{\sum_{i \in \{x,y,z\}} (p_1^i - p_2^i)^2}, m_1, m_2 \right)$$

Feature Space: Dimension Reduction

- detection of irrelevant features and subsequent elimination
- e.g. colour of two bodies and temperature, neither quantity affects the target output value
- principal components analysis provides a mapping of the data to a feature space
- in which the new features are linear functions of the original attributes
- and are sorted by the amount of variance that the data exhibit in each direction
- dimensionality reduction can be performed by removing features corresponding to directions in which the data have low variance,
- though there is no guarantee that these features are not essential for performing the target classification

Feature Space: Non-linear mapping

- implicit mapping into feature space
- to learn non-linear relations with a linear machine, need to select a set of non-linear features and
- to rewrite the data in the new representation equivalent to applying a fixed non-linear mapping of the data to a feature space in which the linear machine can be used
- the set of hypotheses will be functions of the type

$$f(\mathbf{x}) = \sum_{i=1}^{N} w_i \phi_i(\mathbf{x}) + b$$

- $\phi : X \to F$ is a non-linear map from the input space to some feature space
- build non-linear machines in two steps: first a fixed non-linear mapping transforms the data into a feature space F and
- then a linear machine is used to classify them in the feature space

Linear Machine: Dual Representation

- important property of linear machines is that they can be expressed in a dual representation
- this means that the hypothesis can be expressed as a linear combination of the training points so that
- the decision rule can be evaluated using just inner products between the test points and the training points:

$$f(\mathbf{x}) = \sum_{i=1}^{l} \alpha_i y_i \langle \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}) \rangle + b$$

- if there is a way of computing the inner product $\langle \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}) \rangle$ in feature space directly as a function of the original input points,
- it becomes possible to merge the two steps needed to build a non-linear learning machine - call such a direct computation method a kernel function

Kernel Function

- A kernel is a function K such that for all $\mathbf{x}, \mathbf{z} \in X$

$$K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle$$

- due to dual representation the dimension of the feature space need not affect the computation
- the number of operations required to compute the inner product by evaluating the kernel function is not necessarily proportional to the number of features
- the use of kernels makes it possible to map the data implicitly into a feature space and to train a linear machine in such a space
- the only information about the training examples is their Gram matrix in the feature space, this matrix is also referred to as the kernel matrix

Kernel Function

- key to this approach is to find a kernel function that can be evaluated efficiently,
- aim is to introduce non-linearity into the feature space map
- once such a function is available, the decision rule can be evaluated by at most l equations of the kernel:

$$f(\mathbf{x}) = \sum_{i=1}^{l} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

- important fact is that no need to know the underlying feature map in order to be able to learn in the feature space
- example

$$K(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x} \cdot \mathbf{z} \rangle$$

- or feature map to be any fixed linear transformation $\mathbf{x} \rightarrow \mathbf{A}\mathbf{x}$

$$K(\mathbf{x}, \mathbf{z}) = \langle \mathbf{A}\mathbf{x} \cdot \mathbf{A}\mathbf{z} \rangle = \mathbf{x}'\mathbf{A}'\mathbf{A}\mathbf{z} = \mathbf{x}'\mathbf{B}\mathbf{z}$$

- $\mathbf{B} = \mathbf{A}'\mathbf{A}$ square symmetric positive semi-definite matrix

Example: obtaining non-linear map

$$(\mathbf{x} \cdot \mathbf{z})^2 = \left(\sum_{i=1}^{n} x_i z_i\right)^2 = \left(\sum_{i=1}^{n} x_i z_i\right)\left(\sum_{j=1}^{n} x_j z_j\right)$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j z_i z_j = \sum_{(i,j)=(1,1)}^{(n,n)} (x_i x_j)(z_i z_j)$$

- It is equivalent to an inner product between the feature vectors

$$\phi(\mathbf{x}) = (x_i x_j)_{(i,j)=(1,1)}^{(n,n)}$$

Kernel for General Feature Space

$$
\begin{aligned}
(\langle \mathbf{x} \cdot \mathbf{z} \rangle + c)^2 &\overline{\mathrm{I}} \left(\sum_{i=1}^{n} x_i z_i + c \right) \left(\sum_{j=1}^{n} x_j z_j + c \right) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j z_i z_j + 2c \sum_{i=1}^{n} x_i z_i + c^2 \\
&= \sum_{(i,j)=(1,1)}^{(n,n)} (x_i x_j)(z_i z_j) + \sum_{i=1}^{n} \left(\sqrt{2c} x_i \right) \left(\sqrt{2c} z_i \right) + c^2
\end{aligned}
$$

- features are all the monomials of degree upto 2
- This can be extended for degree $d \geq 2$

Making Kernels

- kernel function is an alternative computational short-cut
- create a complicated feature space, work out the inner product in feature space
- find a direct method computing that value in terms of the original inputs
- in practice, define a kernel function directly, hence implicitly defining the feature space,
- avoid feature space not only in the computation of inner products but also in the design of the learning machine itself
- defining a kernel function for an input space is more natural than creating a complicated feature space

Kernel Function Characteristics

- determine what properties of a function $K(\mathbf{x}, \mathbf{z})$ are necessary to ensure that it is a kernel for some feature space
- the function must be symmetric

$$K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle = \langle \phi(\mathbf{z}) \cdot \phi(\mathbf{x}) \rangle = K(\mathbf{z}, \mathbf{x})$$

- satisfy the inequalities that follow from the Cauchy-Schwarz inequality

$$\begin{aligned} K(\mathbf{x}, \mathbf{z})^2 &= \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle^2 \leq ||\phi(\mathbf{x})||^2 ||\phi(\mathbf{z})||^2 \\ &= \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{x}) \rangle \langle \phi(\mathbf{z}) \cdot \phi(\mathbf{z}) \rangle = K(\mathbf{x}, \mathbf{x}) K(\mathbf{z}, \mathbf{z}) \end{aligned}$$

- these conditions are not sufficient to guarantee the existence of a feature space

Support Vector Machines

- if $\mathbf{w}$ is the weight vector realising a functional margin of 1 on the positive point $\mathbf{x}^+$ and the negative point $\mathbf{x}^-$, compute its geometric margin as follows :
- A functional margin of 1 implies

$$\langle \mathbf{w} \cdot \mathbf{x}^+ \rangle + b = +1 \quad \langle \mathbf{w} \cdot \mathbf{x}^- \rangle + b = -1$$

- while to compute the geometric margin, normalize $\mathbf{w}$, geometric margin γ is then the functional margin of the resulting classifier

$$\begin{aligned} \gamma &= \frac{1}{2}\left(\left\langle \frac{\mathbf{w}}{\| \mathbf{w} \|_2} \cdot \mathbf{x}^+ \right\rangle - \left\langle \frac{\mathbf{w}}{\| \mathbf{w} \|_2} \cdot \mathbf{x}^- \right\rangle\right) \\ &= \frac{1}{2 \| \mathbf{w} \|_2}\left(\langle \mathbf{w} \cdot \mathbf{x}^+ \rangle - \langle \mathbf{w} \cdot \mathbf{x}^- \rangle\right) = \frac{1}{\| \mathbf{w} \|_2} \end{aligned}$$

Support Vector Machines

- the resulting margin will be equal to $\frac{1}{\|\mathbf{w}\|_2}$
- Given a linearly separable training sample

$$S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l))$$

- the hyperplane $(\mathbf{w}, b)$ that solves the optimization problem

$$\begin{array}{ll} \text{minimize}_{\mathbf{w},b} & \langle \mathbf{w} \cdot \mathbf{w} \rangle \\ \text{subject to} & y_i \left(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \right) \geq 1, i = 1, \ldots, l \end{array}$$

- realises the maximal margin hyperplane with geometric margin $\gamma = \frac{1}{\|\mathbf{w}\|_2}$
- transform this optimization problem into its corresponding dual problem

Support Vector Machines

- The primal Lagrangian is

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle - \sum_{i=1}^{l} \alpha_i [y_i(\langle \mathbf{w}_i \cdot \mathbf{x}_i \rangle + b) - 1]$$

  $\alpha_i \geq 0$ are the Lagrange multipliers

- The corresponding dual is found by differentiating with respect to $\mathbf{w}$ and b imposing stationarity

$$\frac{\partial L(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{l} y_i \alpha_i \mathbf{x}_i = 0$$

$$\frac{\partial L(\mathbf{w}, b, \alpha)}{\partial b} = \sum_{i=1}^{l} y_i \alpha_i = 0$$

- resubstituting the relations obtained

Support Vector Machines

$$\mathbf{w} = \sum_{i=1}^{l} y_i \alpha_i \mathbf{x}_i \quad 0 = \sum_{i=1}^{l} y_i \alpha_i$$

into the primal to obtain

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle - \sum_{i=1}^{l} \alpha_i [y_i(\langle \mathbf{w}_i \cdot \mathbf{x}_i \rangle + b) - 1]$$

$$= \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle - \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle + \sum_{i=1}^{l} \alpha_i$$

$$= \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle$$

Support Vector Machines

- the hypothesis can be described as a linear combination of the training points: the application of optimization theory naturally leads to the dual representation
- Consider a linearly separable training sample

$$S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l))$$

and suppose the parameters α^* solve the following quadratic optimization problem:

$$\begin{array}{ll} \text{maximize} & \mathbf{W}(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle \\ \text{subject to} & \sum_{i=1}^{l} y_i \alpha_i = 0 \;\; \alpha_i \geq 0 \; i = 1, \ldots, l \end{array}$$

Support Vector Machines

- the weight vector $\mathbf{w}^* = \sum_{i=1}^{l} y_i \alpha_i^* \mathbf{x}_i$ realises the maximal margin hyperplane with geometric margin $\gamma = 1/\| \mathbf{w}^* \|_2$
- the value of b does not appear in the dual problem and so b^* must be found making use of the primal constraints

$$b^* = -\frac{\max_{y_i=-1}(\langle \mathbf{w}^* \cdot \mathbf{x}_i \rangle) + \min_{y_i=1}(\mathbf{w}^* \cdot \mathbf{x}_i \rangle)}{2}$$

- The Karush-Kuhn-Tucker complementarity conditions provide useful information about the structure of the solution
- the conditions state that the optimal solutions α^*, $(\mathbf{w}^*, b^*)$ must satisfy

$$\alpha_i^* [y_i(\langle \mathbf{w}^* \cdot \mathbf{x}_i \rangle + b^*) - 1] = 0 \quad i = 1, \ldots, l$$

- this implies that only for inputs $\mathbf{x}_i$ for which the functional margin is one and that therefore lie closet to the hyperplane are the corresponding α_i^* non-zero

Support Vector Machines

- All the other parameters α_i^* are zero
- In the expression for the weight vector only these points are involved
- for this reason they are called support vectors (sv)
- optimal hyperplane can be expressed in the dual representation in terms of this subset of the parameters

$$\begin{aligned} f(\mathbf{x}, \alpha^*, b^*) &= \sum_{i=1}^{l} y_i \alpha_i^* \langle \mathbf{x}_i \cdot \mathbf{x} \rangle + b^* \\ &= \sum_{i \in \mathsf{SV}} y_i \alpha_i^* \langle \mathbf{x}_i \cdot \mathbf{x} \rangle + b^* \end{aligned}$$

- the Lagrange multipliers associated with each point become the dual variables, giving them an intuitive interpretation quantifying how important a given training point is in forming the final solution

Video Authentication Using Relative Correlation Information and SVM

- The authenticity of the video data is of paramount interest: presented as evidence in many criminal cases, Video surveillance, forensics, law enforcement, and content ownership
- Video authentication is a process which ascertains that the content in a given video is authentic and exactly same as when captured
- It should detects the type and location of malicious tampering
- An intelligent video authentication algorithm using SVM
  - does not require the computation and storage of secret key or embedding of watermark
  - computes the local relative correlation information and classifies the video as tampered or non-tampered
  - not affected by video processing operations such as compression and scaling

Video authenticity

- Compression and scaling - affect the content and properties of the video data but acceptable and not considered as tampering
- Video authentication algorithms should be able to differentiate between intentional tampering and acceptable operations
- Video authentication algorithms: classified into digital signature based, watermarking based, and other methods
- Digital signature based authentication schemes - authentication data is separately stored either in user defined field such as in the header of MPEG sequence, or in a separate file
- Watermarking embeds the authentication data into the primary multimedia sources without changing the video content

Video authenticity

- Digital signature is generated from image blocks and used as the watermark (edge features of the image to generate the digital signature).
- Encrypting the feature point positions in an image/video
- In watermarking, any manipulation on the watermarked data also changes the content of the embedded watermark.
- Watermarking based authentication algorithm examines the variations in the extracted watermark to verify the integrity of multimedia data.
- one video frame is watermarked in another frame using a specific sequence and a key- authentication data is embedded at GOP level
- Digital signature - if compromised then it is easy to deceive the authentication system
- Watermarking: Embedding may alter the content of video which is not permissible in the court of law

Other approaches

- Motion trajectory and cryptographic secret sharing techniques are used.
- Different shots are segmented from a given video and the key frames in a shot are selected based on the motion trajectory.
- A secret frame is constructed and used as the secret key of a particular shot.
- A master key for the entire video is then generated using different secret keys computed for all the shots.
- Authenticity of a video is determined using the computed master key.
- Special hash functions are used to authenticate the edited video.

SVM based video authentication algorithm

- Authentication algorithm which computes the salient local information in digital video frames and establishes a relationship among the frames.
- This relationship is termed as the relative correlation information and is used to authenticate the video data.
- SVM based learning algorithm is used to classify the video as tampered or non-tampered.
- Algorithm does not require computation and storage of any key or embedding of secret information in the video data.
- The algorithm uses inherent video information for authentication, thus making it useful for real world applications.

SVM based Video authentication algorithm

I

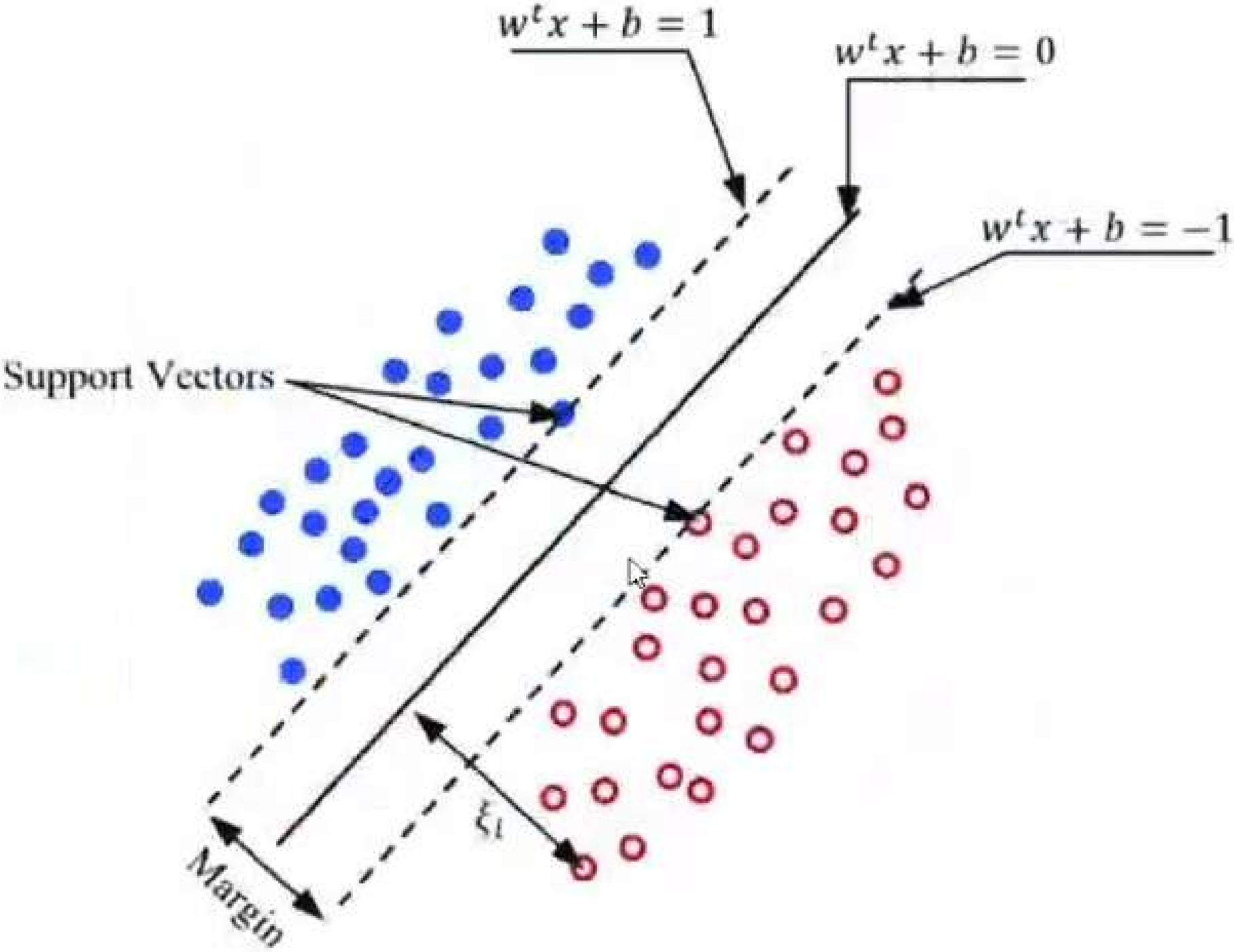
- Algorithm computes the correlation information between two video frames.
- This information is computed locally using corner detection algorithm and then classification is performed using SVM.
- The algorithm is divided into two stages: (1) SVM training and (2) tamper detection and classification using SVM.
- SVM Training - classify the tampered and non-tampered video data.
- SVM based learning and classification - can differentiate between attack and acceptable operations
- Training is performed using a manually labeled training video database.
- If the video in the training data is tampered, then it is assigned the label -1 otherwise (if it is not tampered) the label is $+1$.

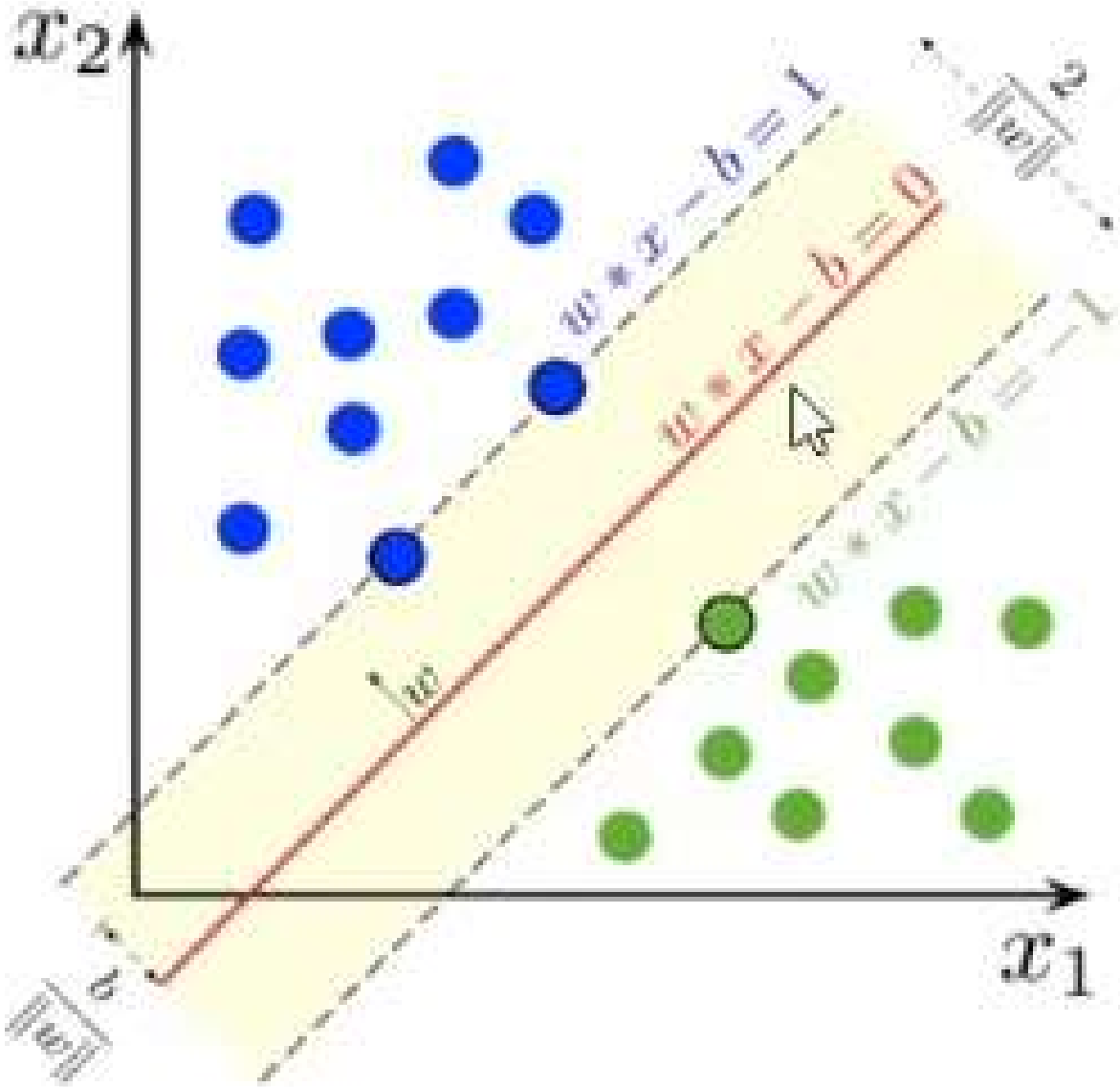
SVM based Video authentication algorithm

- From the training videos, relative correlation information are extracted.
- This labeled information is then used as input to the SVM which performs learning and generates a non-linear hyperplane that can classify the video as tampered and non-tampered.

SVM based Video authentication algorithm

- The window size for the second frame is greater in order to provide tolerance to the errors which may occur during corner point computation.
- One to many local correlation is performed on both the frames. Every window of the first frame is correlated with every window of the second frame.
- Window pairs that provide the maximum correlation are then selected.
- To handle incorrect corner pairs, those pairs are selected that have similar coordinate positions and the correlation value is greater than pre-selected value.
- Let the local correlation between two frames be L_i and $i = 1, 2, \ldots, m$. m is the number of corresponding corner points in the two frames.


$w^t x + b = 1$
$w^t x + b = 0$
$w^t x + b = -1$
Support Vectors
ξ_i
Margin


x_2
x_1
$w * x - b = 1$
$w * x - b = 0$
$w * x - b = -1$
w
$\frac{2}{\|w\|}$
$\frac{b}{\|w\|}$

CREDITS SOME CS GUY!