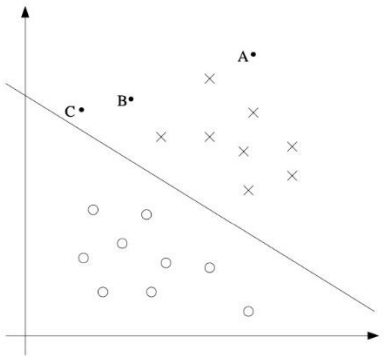
Support Vector Machines

https://aman.ai/cs229/svm/

1 Motivation

- **SVM**: is to find the **"best" margin** (distance between the line and the support vectors) that separates the classes and this reduces the risk of error on the data
- **Motivation**: consider a hyperplane $h(x) = w^T x + b$ to classify a set of samples as positive/negative. Suppose a sample x_i will be predicted as $y_i = 1$ if $w^T x_i + b > 0$ (e.g., the sigmoid function with linear hyperplane). In particular, if $w^T x_i + b \gg 0$, we are more confident to predict $y_i = 1$. For example, in the following figure, we are more confident to predict A as positive compared to C.



2 Notations

- Considering a linear classifier for a binary classification problem with labels y and features x. Note that we consider $y \in \{-1,1\}$.
- We use parameters w, b to represent the linear classifier, i.e.,

$$h_{w,b}(x) = g\left(w^T x + b\right)$$

Here, g(z)=1 if z⩾0, and g(z)=−1 otherwise. This "w,b" notation allows us to explicitly treat the intercept term b separately from the other parameters.

3 Functional and Geometric Margins

- **Function margin**: Given a training example $(x(i), y(i))$, we define the functional margin of (w, b) with respect to the training example

$$\hat{\gamma}^{(i)} = y^{(i)}(w^T x + b)$$

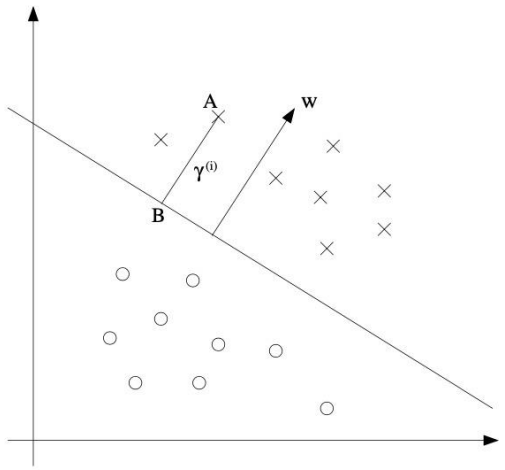
if $y_i(w^T x + b) > 0$, then our prediction on this example is correct. Hence, a large functional margin represents a confident and a correct prediction.

- **Geometric margins**: the distance between A and B is shown as

$$\gamma^{(i)} = \frac{w^T x^{(i)} + b}{\|w\|} = \left(\frac{w}{\|w\|}\right)^T x^{(i)} + \frac{b}{\|w\|}$$

More generally, the ***geometric margin*** of (w,b) with respect to a training example (x^i, y^i) to be

$$\gamma^{(i)} = y^{(i)} \left(\left(\frac{w}{\|w\|}\right)^T x^{(i)} + \frac{b}{\|w\|} \right)$$



4 Optimal Margin Classifier

- **Goal**: maximize the minimum geometric margin, saying,

  - Original form

$$\rho = \max_{(w,b)} \min_{i\in[m]} \frac{|\langle w, x_i\rangle + b|}{\|w\|_2} \quad (12)$$

$$\text{s.t. } y_i(\langle w, x_i\rangle + b) > 0 \quad \forall i \quad (13)$$

  - Alternative form 1

$$\rho = \max_{(w,b)} \min_{i\in[m]} \frac{y_i(\langle w, x_i\rangle + b)}{\|w\|_2} \quad (14)$$

  - Alternative form 2

$$\rho = \max_{(w,b):\ \min_{i\in[m]} y_i(\langle w,x_i\rangle+b=1} \frac{1}{\|w\|_2} \quad (15)$$

$$= \max_{(w,b):\ y_i(\langle w,x_i\rangle+b\geq 1} \frac{1}{\|w\|_2} \quad (16)$$

- The transferred Quadratic problem (QP) is shown below, where QP software allows convex quadratic objectives and linear constraints.

- Alternative form 3: Quadratic programming (QP)

$$\begin{aligned} \min_{(w,b)} \ & \frac{1}{2}\|w\|_2^2 \\ \text{s.t.} \ & y_i(\langle w, x_i \rangle + b) \geq 1, \quad \forall i \in [m] \end{aligned} \tag{18}$$

which is a constrained optimization problem that can be solved by standard QP packages

- **Exercise**: solve this problem by commercial quadratic programming (QP) code.

5 Lagrange duality and KKT

- The original optimization problem

$$\begin{aligned} \min_w \ & f(w) \\ \text{s.t.} \ & g_i(w) \leq 0, \quad i = 1, \ldots, k \\ & h_i(w) = 0, \quad i = 1, \ldots, l \end{aligned}$$

- To solve it, we start by defining the generalized Lagrangian

$$\mathcal{L}(w, \alpha, \beta) = f(w) + \sum_{i=1}^{k} \alpha_i g_i(w) + \sum_{i=1}^{l} \beta_i h_i(w)$$

- Then the dual problem satisfies

$$d^* = \max_{\alpha,\beta:\alpha_i \geq 0} \min_w \mathcal{L}(w, \alpha, \beta) \leq \min_w \max_{\alpha,\beta:\alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta) = P^*$$

Where the left side is the dual problem, and the left side is the primal problem.

- **Strong duality**: $d^* = P^*$ when the slater's conditions hold, which is
  $g_i(w) < 0 \; if \; non \; linear, g_i(w) \leq 0 \; if \; it \; is \; linear$
- **KKT condition**:

$$\frac{\partial}{\partial w_i} \mathcal{L}(w^*, \alpha^*, \beta^*) = 0, \quad i = 1, \ldots, n \tag{3}$$

$$\frac{\partial}{\partial \beta_i} \mathcal{L}(w^*, \alpha^*, \beta^*) = 0, \quad i = 1, \ldots, l \tag{4}$$

$$\alpha_i^* g_i(w^*) = 0, \quad i = 1, \ldots, k \tag{5}$$

$$g_i(w^*) \leq 0, \quad i = 1, \ldots, k \tag{6}$$

$$\alpha^* \geq 0, \quad i = 1, \ldots, k \tag{7}$$

6 Optimal Margin Classifiers

- Recall the primal QP:

$$\min_{\gamma,w,b} \frac{1}{2}\|w\|^2$$
$$\text{s.t. } y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1, \quad i = 1, \ldots, m$$

- The Lagrangian for our optimization problem is

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i \left[y^{(i)}\left(w^T x^{(i)} + b\right) - 1\right]$$

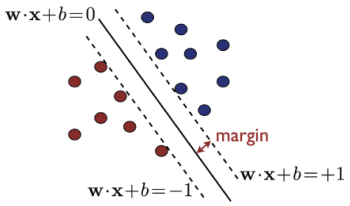
- Based on the KKT condition, we have

$$\nabla_w L = w - \sum_{i=1}^{m} \alpha_i y_i x_i = 0 \quad \Rightarrow \quad w = \sum_{i=1}^{m} \alpha_i y_i x_i$$
$$\nabla_b L = -\sum_{i=1}^{m} \alpha_i y_i = 0 \quad \Rightarrow \quad \sum_{i=1}^{m} \alpha_i y_i = 0$$
$$\forall i, \alpha_i (y_i(\langle w, x_i \rangle + b) - 1) = 0 \quad \Rightarrow \quad \alpha_i = 0 \text{ or } y_i(\langle w, x_i \rangle + b) = 1$$

- **Supporting vector**:

Consider the implication of the last equation in the previous page, $\forall i$

- $\alpha_i > 0$ and $y_i(\langle w, x_i \rangle + b) = 1$ or
- $\alpha_i = 0$ and $y_i(\langle w, x_i \rangle + b) \geq 1$



$$w = \sum_{i=1}^{m} \alpha_i y_i x_i \tag{27}$$

- Examples with $\alpha_i > 0$ are called **support vectors**
- In $\mathbb{R}^d$, $d + 1$ examples are sufficient to define a hyper-plane

And the parameter b^* based on the primal problem is shown below.

$$b^* = -\frac{\max_{i:y^{(i)}=-1} w^{*T} x^{(i)} + \min_{i:y^{(i)}=1} w^{*T} x^{(i)}}{2}$$

- **New prediction:** find only the inner products of features between x and the support vectors

$$w^T x + b = \left(\sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} \right)^T x + b \tag{12}$$

$$= \sum_{i=1}^{m} \alpha_i y^{(i)} \left\langle x^{(i)}, x \right\rangle + b \tag{13}$$

7 Kernel function

https://www.kaggle.com/code/prashant111/svm-classifier-tutorial/notebook

https://stats.stackexchange.com/questions/90736/the-difference-of-kernels-in-svm

- **Motivation**: replace the inner product $< x^i, x >$ by a kernel function $K(x_i, x)$ as

$$K(x_i, x) = \langle \Phi(x_i), \Phi(x) \rangle$$

- **Linear**: $K(x_i, x) = x_i^T x$, Linear kernel is used when the data is linearly separable.
- **Polynomial Kernels**: Polynomial kernel is very popular in Natural Language Processing.

$$K(\boldsymbol{x}, \boldsymbol{x}') = (\gamma \langle \boldsymbol{x}, \boldsymbol{x}' \rangle + c)^d, \forall \boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^n$$

Examples: Polynomial Kernels (II)

For the special case with $d = 2$, assume $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^2$ (let $\gamma = 1$ for simplicity)

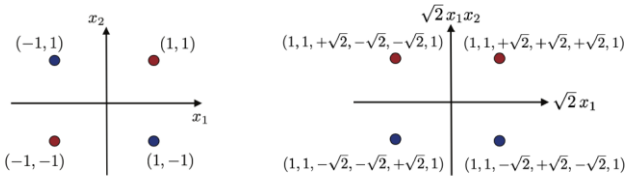
$$\begin{aligned}
K(\boldsymbol{x}, \boldsymbol{x}') &= (\langle \boldsymbol{x}, \boldsymbol{x}' \rangle + c)^2 && (55)\\
&= (x_1 x'_1 + x_2 x'_2 + c)^2 && (56)\\
&= x_1^2 x'^2_1 + x_1 x_2 x'_1 x'_2 + c x_1 x'_1 + x_1 x_2 x'_1 x'_2 \\
&\quad + x_2^2 x'^2_2 + c x_2 x'_2 + c x_1 x'_1 + c x_2 x'_2 + c^2 && (57)\\
&= x_1^2 x'^2_1 + x_2^2 x'^2_2 + 2 x_1 x'_1 x_2 x'_2 && (58)\\
&\quad + 2c x_1 x'_1 + 2c x_2 x'_2 + c^2 && (59)\\
&= [x_1^2, x_2^2, \sqrt{2} x_1 x_2, \sqrt{2c} x_1, \sqrt{2c} x_2, c] \begin{bmatrix} x'^2_1 \\ x'^2_2 \\ \sqrt{2} x'_1 x'_2 \\ \sqrt{2c} x'_1 \\ \sqrt{2c} x'_2 \\ c \end{bmatrix}
\end{aligned}$$

Exercise: Find out the $\Phi(\boldsymbol{x})$ function in $K(\boldsymbol{x}, \boldsymbol{x}') = (\langle \boldsymbol{x}, \boldsymbol{x}' \rangle + c)^3$

Let $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$, then

$$\Phi(x) = [x_1^2, x_2^2, \sqrt{2}x_1x_2, \sqrt{2c}x_1, \sqrt{2c}x_2, c]^\mathsf{T} \tag{60}$$

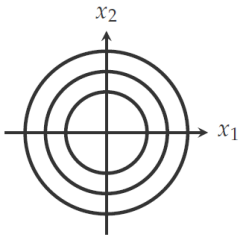
which maps a 2-D data point x into a 6-D space as $\Phi(x)$ Recall the XOR problem



- **Gaussian kernel**: It is used when we have no prior knowledge about the data.

For any constant $\gamma > 0$, a **Gaussian kernel** or **radial basis function** (RBF) is the kernel K defined over $\mathbb{R}^d$ by

$$K(x, x') = \exp\left(-\gamma \|x' - x\|_2^2\right) \tag{61}$$



- **Sigmoid kernel**: Sigmoid kernel has its origin in neural networks. We can use it as the proxy for neural networks.

$$k(x', x) = \tanh\left(\alpha\, x^T x' + r\right)$$

- **A valid Kernel**: the matrix $K(x^i, x)$ is positive semidefinite.

$$\boldsymbol{c}^\mathsf{T}\mathbf{K}\boldsymbol{c} \geq 0$$

8 non-separable cases

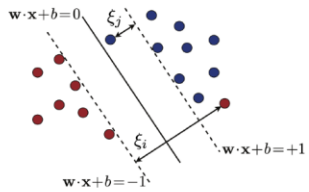
- **Motivation**: some samples do not satisfy the condition $y_i(w^T x_i + b) \geq 1$. Therefore, we consider a relaxed constraint.

Consider the relaxed constraint

$$y_i(\langle w, x_i \rangle + b) \geq 1 - \xi_i \tag{31}$$

and three cases of ξ_i

- $\xi_i = 0$
- $0 < \xi_i < 1$
- $\xi_i \geq 1$



- The modified optimization problem is shown below

$$\min_{(w,b)} \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{m} \xi_i^p$$
$$\text{s.t. } y_i(\langle w, x_i \rangle + b) \geq 1 - \xi_i, \quad \forall i \in [m]$$
$$\xi_i \geq 0$$

where $C \geq 0$, $p \geq 1$, and $\{\xi_i\}_{i=1}^m \geq 0$ are known as **slack variables** and are commonly used in optimization to define relaxed versions of constraints.

- Then the lagrange function is shown below

$$\mathcal{L}(w, b, \xi, \alpha, r) = \frac{1}{2}w^T w + C\sum_{i=1}^{m} \xi_i - \sum_{i=1}^{m} \alpha_i \left[y^{(i)} \left(x^T w + b \right) - 1 + \xi_i \right] - \sum_{i=1}^{m} r_i \xi_i$$

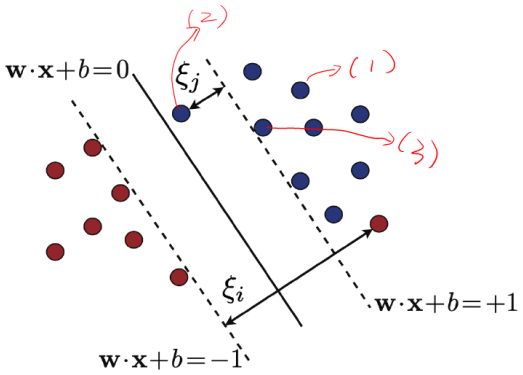
- Based on the KKT condition, the coefficients and support vectors are ($0 < a_i \leq c$ *or* $a_i = c$)

$$\begin{aligned} \alpha_i + \beta_i &= C \\ \alpha_i = 0 \quad &\text{or} \quad y_i(w^\top x_i + b) = 1 - \xi_i \\ \beta_i = 0 \quad &\text{or} \quad \xi_i = 0 \end{aligned}$$

(1) $\alpha_i = 0 \Rightarrow y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1$

(2) $\alpha_i = C \Rightarrow y^{(i)}\left(w^T x^{(i)} + b\right) \leq 1$

(3) $0 < \alpha_i < C \Rightarrow y^{(i)}\left(w^T x^{(i)} + b\right) = 1$



8 Solution: SMIO algorithm

- Key idea: iteratively optimize each component of the parameter a_i.
- Solution (SMO): To satisfy some constraint in the dual optimization, the SMO algorithm updated a pair of parameters each time.

$$\alpha_1 y^{(1)} + \alpha_2 y^{(2)} = -\sum_{i=3}^{m} \alpha_i y^{(i)}$$

9 Numerical results

- The numerical results showed that the **Gaussian Kernel** and **linear kernel** (Radial Basis Function kernel) functions better.

```
----------------------rbf------------------------------
Model accuracy score with default hyperparameters: 0.8328
----------------------linear---------------------------
Model accuracy score with default hyperparameters: 0.8102
----------------------Polymer--------------------------
Model accuracy score with default hyperparameters: 0.7940
----------------------sigmoid--------------------------
Model accuracy score with default hyperparameters: 0.3750
----------------------rbf,C=40-------------------------
Model accuracy score with default hyperparameters: 0.8520
----------------------linear,C=40----------------------
Model accuracy score with default hyperparameters: 0.7998
----------------------Polymer,C=40---------------------
Model accuracy score with default hyperparameters: 0.8224
----------------------sigmoid,C=40---------------------
```

- By adding a Regularization parameter $C = 40$, the accuracy can increases to 85%.

10 Logistic Regression VS SVM

https://medium.com/axum-labs/logistic-regression-vs-support-vector-machines-svm-c335610a3d16#:~:text=Difference%20between%20SVM%20and%20Logistic%20Regression&text=SVM%20is%20based%20on%20geometrical,regression%20is%20vulnerable%20to%20overfitting.

- LR: classification
- SVM: classification (main tasks) and regression
- **Deference between LR and SVM**
    a) SVM tries to find the **"best" margin** (distance between the line and the support vectors) that separates the classes and this reduces the risk of error on the data, while logistic regression does not, instead it can have different decision boundaries with different weights that are near the optimal point.
    b) SVM works well with unstructured and semi-structured data like text and images while logistic regression works with already identified independent variables.
    c) SVM is based on geometrical properties of the data while logistic regression is based on statistical approaches.
    d) The risk of overfitting is less in SVM, while Logistic regression is vulnerable to overfitting.

- Model selections:

    In general, starting with LR to evaluate the data. If LR does not work well, use SVM with linear regression instead.