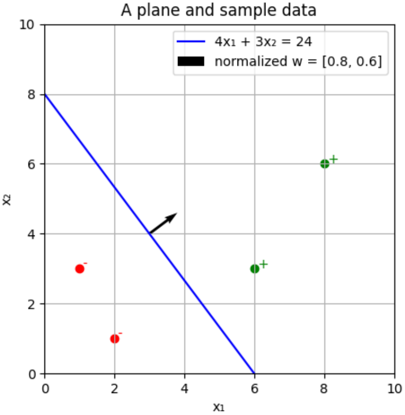
Week 12: Support Vector Machine (SVM)

Ekarat Rattagan

Support Vector Machine (SVM)

- Supervised learning algorithm used for classification and regression tasks.
- Given training data $\{(x_i, y_i)\}_{i=1}^{n}$, where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$. Finds the hyperplane, $w_1x_1 + w_2x_2 + b = 0$, that best separates data points of different classes
- Maximizes the margin, i.e. the distance between the separating hyperplane and the closest data points from either class.

How does SVM works? (1/3)

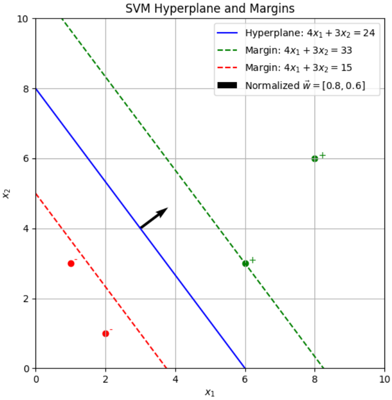


- Let $4x_1 + 3x_2 - 24 = 0$ be a plane
- Let $\vec{w} = \begin{bmatrix} 4 \\ 3 \end{bmatrix}$ and $b = -24$.
- Let dot product $\vec{w}^\top \vec{x_i}$

$$\begin{bmatrix} w_1 & w_2 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = w_1 x_1 + w_2 x_2$$

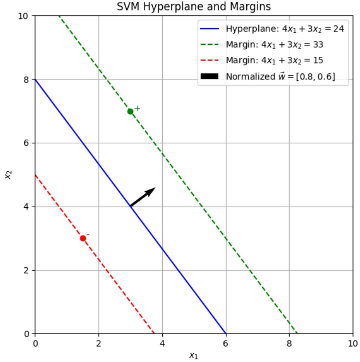
- Let define $\vec{w}^\top \vec{x_\oplus} + b \geq 0$ eq. 1

How does SVM works? (2/3)



- Let $\vec{w}^\top \vec{x_\oplus} + b \geq +1$
- Let $\vec{w}^\top \vec{x_\ominus} + b \leq -1$
- We've known $y_i = +1$ for $\vec{x_\oplus}$
- We've known $y_i = -1$ for $\vec{x_\ominus}$
- We got $y_i(\vec{w}^\top \vec{x_\oplus} + b) - 1 \geq 0$
- Only for support vectors, we obtain $y_i(\vec{w}^\top \vec{x_\ominus} + b) - 1 = 0 \quad$ eq. 2

How does SVM works? (3/3)



- Width of margin $= (\vec{x}_{\oplus} - \vec{x}_{\ominus}) \cdot \frac{\vec{w}}{\|\vec{w}\|}$
- $= \frac{(\vec{w}\vec{x}_{\oplus} - \vec{w}\vec{x}_{\ominus})}{\|\vec{w}\|}$
- $= \frac{(1-b)-(-b-1))}{\|\vec{w}\|}$
- $= \frac{2}{\|\vec{w}\|}$ eq. 3

Objective: Hard-margin SVM

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|\vec{w}\|^2 \\ \text{subject to} \quad & y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1, \quad \text{for } i = 1, \ldots, N \end{aligned}$$

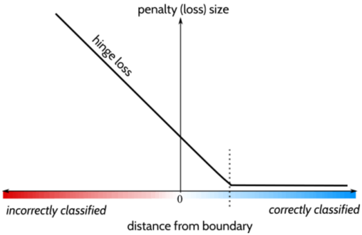
- Hard margin SVM.
  - It assumes the data is perfectly linearly separable.
  - No points are allowed to be misclassified or even touch the margin.

Objective: Soft-margin SVM

$$
\begin{aligned}
\text{minimize} \quad & \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{N}\xi_i \\
\text{subject to} \quad & y_i(\vec{w}\cdot\vec{x}_i + b) \geq 1 - \xi_i, \quad \forall i \\
& \xi_i \geq 0, \quad \forall i
\end{aligned}
$$

- ξ_i: slack variable for each data point i
- C: regularization parameter that controls the trade-off:
  - High C: prioritize classifying points correctly (small slack)
  - Low C: allow more flexibility (wider margin)
- Works even when data is **not linearly separable**
- Helps handle **outliers and noisy data**
- Prevents **overfitting** by allowing margin violations

Cost function: Hinge loss



$$\min_{\vec{w},b} \quad \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{N} \max(0,\ 1 - y_i(\vec{w}\cdot\vec{x}_i + b))$$

Gradient Descent of SVM

Given the objective function:

$$\min_{\vec{w},b} \quad \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{n} \max(0,\ 1 - y_i(\vec{w}\cdot\vec{x}_i + b))$$

The subgradients are:

$$\nabla_{\vec{w}} = \begin{cases} \vec{w} - Cy_i\vec{x}_i & \text{if } y_i(\vec{w}\cdot\vec{x}_i + b) < 1 \\ \vec{w} & \text{otherwise} \end{cases}$$

$$\nabla_b = \begin{cases} -Cy_i & \text{if } y_i(\vec{w}\cdot\vec{x}_i + b) < 1 \\ 0 & \text{otherwise} \end{cases}$$

Update rules:

$$\vec{w} \leftarrow \vec{w} - \alpha\nabla_{\vec{w}}$$

$$b \leftarrow b - \alpha\nabla_b$$

Kernel Trick

- Maps input features into higher-dimensional space to make data linearly separable.
- Use Sequential Minimal Optimization (SMO) to find the model.
- Common kernels:
  - Linear: $K(x, x') = x^T x'$
  - Polynomial: $K(x, x') = (x^T x' + c)^d$
  - Radial Basis Function (RBF): $K(x, x') = \exp(-\gamma \|x - x'\|^2)$

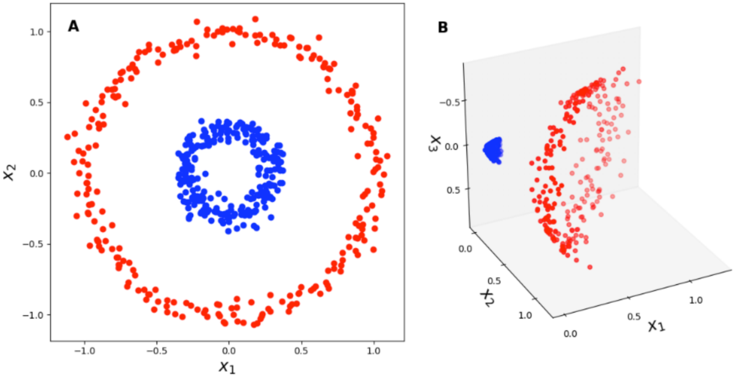


Figure 1: The "lifting trick". (a) A binary classification problem that is not linearly separable in $\mathbb{R}^2$. (b) A lifting of the data into $\mathbb{R}^3$ using a polynomial kernel, $\varphi([x_1 \;\; x_2]) = [x_1^2 \;\; x_2^2 \;\; \sqrt{2}x_1x_2]$.

https://gregorygundersen.com/blog/2019/12/10/kernel-trick/

Choosing the Right Kernel

- Linear kernel for linearly separable data.
- Polynomial kernel for data with curved boundaries.
- RBF kernel for complex boundaries; requires tuning γ.
- Use cross-validation to select the best kernel and parameters.

SVM in Practice

- Scale features to have zero mean and unit variance.
- Tune hyperparameters C and kernel parameters using cross-validation.
- SVMs can handle high-dimensional data effectively.

Advantages of SVM

- Effective in high-dimensional spaces.
- Works well when the number of dimensions exceeds the number of samples.
- Memory efficient as it uses a subset of training points (support vectors).

Limitations of SVM

- Not suitable for very large datasets due to high training time.
- Less effective when the data has a lot of noise and overlapping classes.
- Requires careful tuning of hyperparameters and selection of the appropriate kernel.

Classifier	**Kernel support**	**Optimization Method**	**Uses Gradient Descent**
`SVC`	Non-linear	SMO	No
`SGDClassifier`	Linear	Stochastic Gradient Descent	Yes
`LinearSVC`	Linear	Coordinate Descent	No

Table: Comparison of SVM Classifiers in scikit-learn