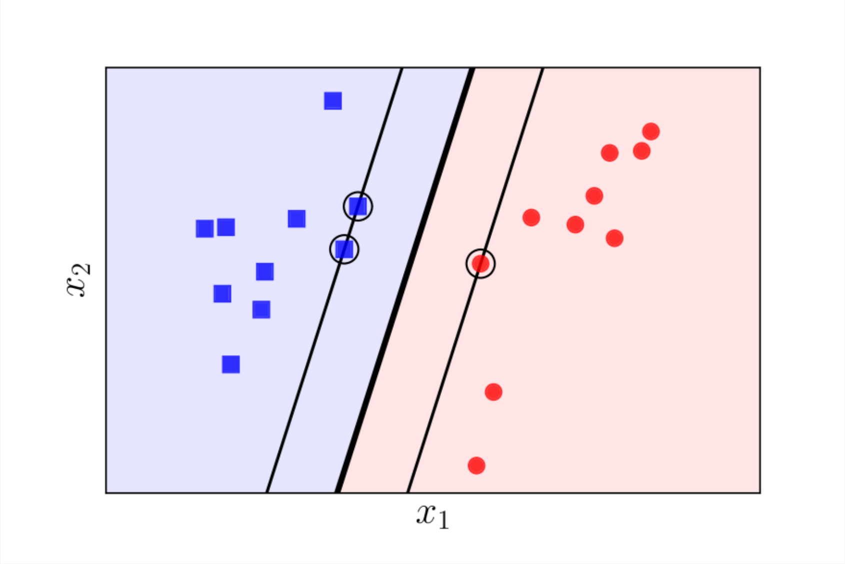
KERNEL SUPPORT VECTOR MACHINE

SUPPORT VECTOR MACHINE

- SVM là một thuật toán giám sát, nó có thể sử dụng cho cả việc phân loại hoặc đệ quy. Tuy nhiên nó được sử dụng chủ yếu cho việc phân loại
- SVM khi không sử dụng đến kernel thì nó có thể phân loại được cái dữ liệu phân biệt tuyến tính hoặc gần như tuyến tính bằng cách sử dụng các bài toán tối ưu như gradient descent hay bài toán đối ngẫu Lagrange



KERNEL TRICK

- Để có thể phân loại trên dữ liệu phi tuyến tính thì ta phải sử dụng đến kernel trick
- Để tối ưu cho SVM chúng ta phải giải quyết bài toán đối ngẫu có dạng như sau

$$\lambda = \arg\max_{\lambda} \sum_{n=1}^{N} \lambda_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \lambda_n \lambda_m y_n y_m \mathbf{x}_n^T \mathbf{x}_m$$

- Giả sử rằng ta có thể tìm được hàm số Φ () sao cho sau khi được biến đổi sang không gian mới, mỗi điểm dữ liệu x trở thành Φ (x)
- Để tối ưu cho SVM chúng ta phải giải quyết bài toán đối ngẫu có dạng như sau

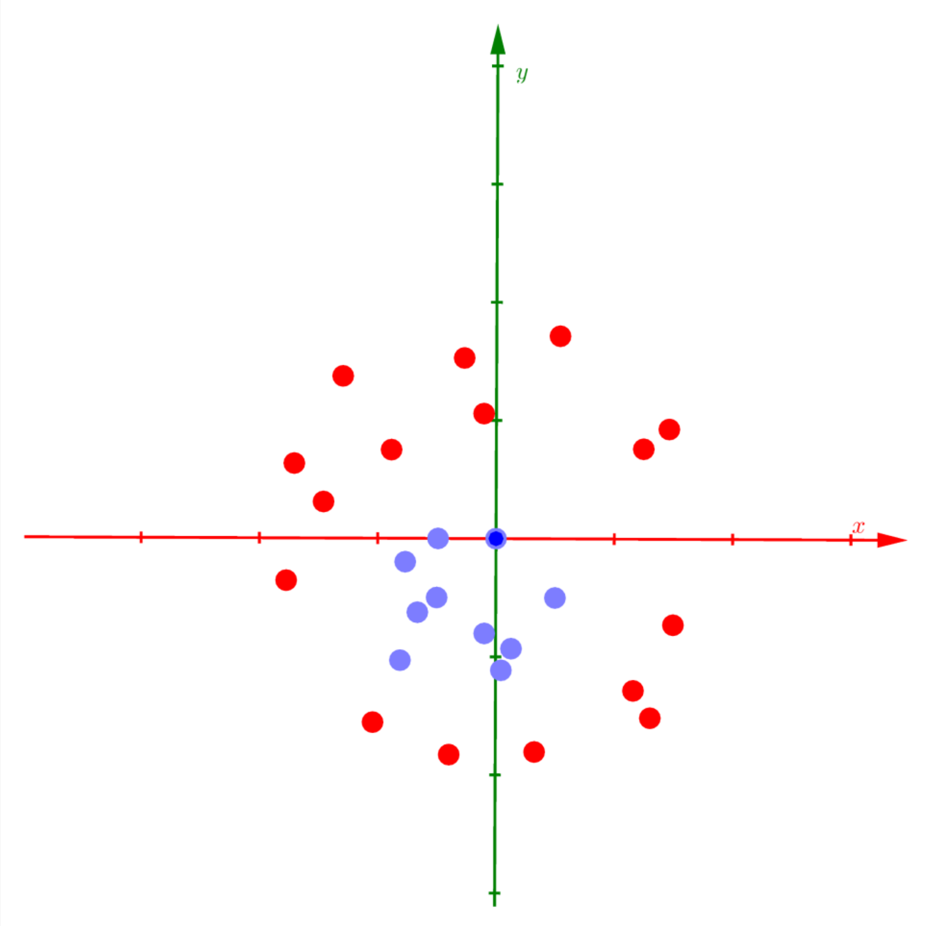
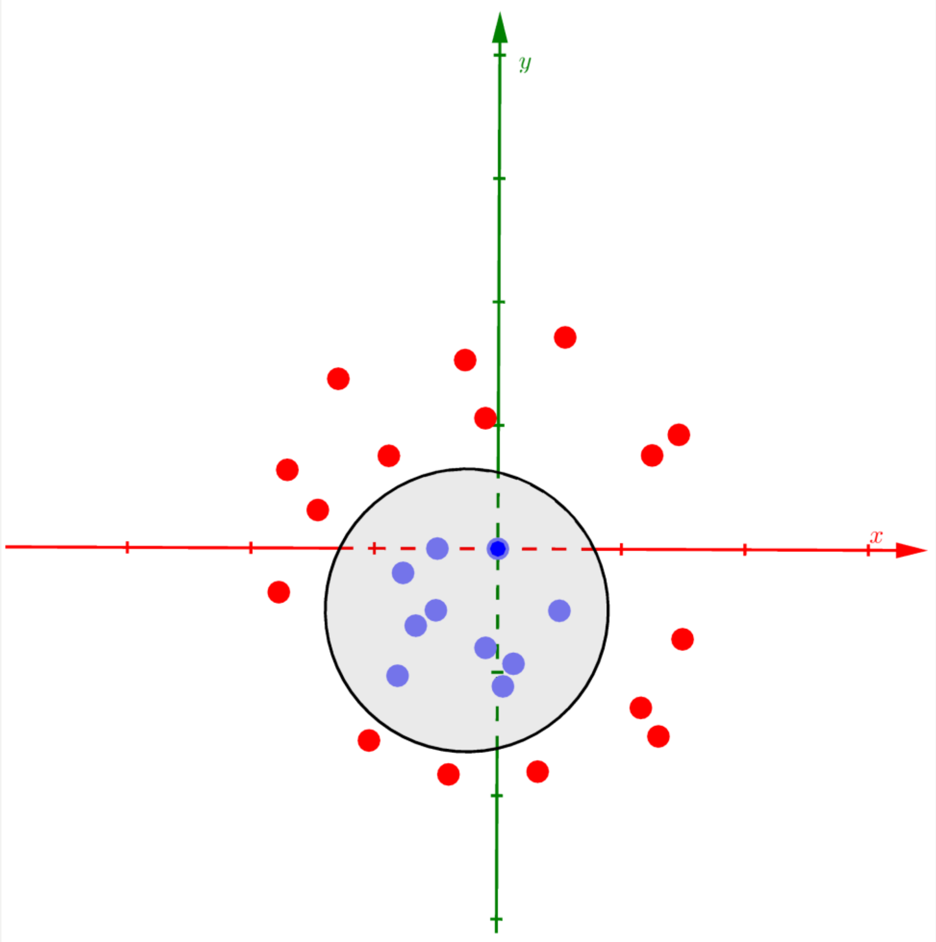
$$\lambda = \arg\max_{\lambda} \sum_{n=1}^{N} \lambda_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \lambda_n \lambda_m y_n y_m \Phi(\mathbf{x}_n)^T \Phi(\mathbf{x}_m)$$

KERNEL TRICK

- Lúc này, bằng cách định nghĩa hàm kernel k (x , z) = Φ (x) T Φ (z) , ta có thể viết lại bài toán

$$\lambda = \arg\max_{\lambda} \sum_{n=1}^{N} \lambda_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \lambda_n \lambda_m y_n y_m k(\mathbf{x}_n, \mathbf{x}_m)$$

- Và ta chỉ cần tính được hàm kernel là có thể giải quyết được bài toán mà không cần quan tâm đến số chiều của dữ liệu


y
x

y
x