Fundamentals of Machine Learning

NONPARAMETRIC METHODS

Amit K Roy-Chowdhury

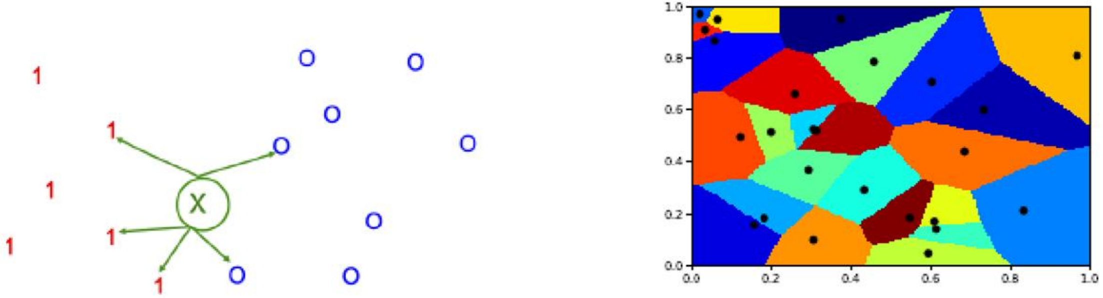
Acknowledgments: Adapted from slides at https://probml.github.io/pml-book/teaching1.html by Prof. Saw Shier Nee

Outline

- K Nearest Neighbor
- Kernel Density Estimation
- Support Vector Machine – Preliminaries
- Decision Trees

K Nearest Neighbor

$$p(y = c|\boldsymbol{x}, \mathcal{D}) = \frac{1}{K} \sum_{n \in N_K(\boldsymbol{x}, \mathcal{D})} \mathbb{I}(y_n = c)$$



(a) *(b)*

Figure 16.1: (a) Illustration of a K-nearest neighbors classifier in 2d for $K = 5$. The nearest neighbors of test point $\boldsymbol{x}$ have labels $\{1, 1, 1, 0, 0\}$, so we predict $p(y = 1|\boldsymbol{x}, \mathcal{D}) = 3/5$. (b) Illustration of the Voronoi tessellation induced by 1-NN. Adapted from Figure 4.13 of [DHS01]. Generated by knn_voronoi_plot.ipynb.

KNN Example

Example dataset: **CIFAR-10**
10 labels
50,000 training images
10,000 test images.

For every test image (first column), examples of nearest neighbors in rows

Slide credit: Andrej Karpathy

KNN Error Rate

$$P(Y=i|X=x) = \frac{p(x|Y=i)P(y=i)}{p(x)} = \frac{p(x|Y=i)P(Y=i)}{\sum_j p(x|Y=j)P(Y=j)} = q_i(x)$$

We pick the label with the maximum posterior probability.

$$q_0 = p(x|Y=0)P(Y=0)$$

$$q_1 = p(x|Y=1)P(Y=1)$$

What is the probability of error?

Risk: $r(x) = min(q_0(\mathbf{x}), q_1(\mathbf{x}))$ (we choose the lowest value but can make a mistake)

$\text{Bayes Error} = E[r(X)]$ (lower limit of the error with any classifier)

KNN comes within a factor of 2 of Bayes error

Outline

- K Nearest Neighbor
- Kernel Density Estimation
- Support Vector Machine
- Decision Trees

Kernel Density

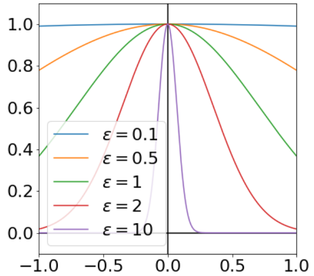
Density Kernel: $\mathcal{K} : \mathbb{R} \rightarrow \mathbb{R}_+$ such that $\int \mathcal{K}(x)dx = 1$ and $\mathcal{K}(-x) = \mathcal{K}(x)$

Example: Gaussian Kernel $\mathcal{K}(x) = \frac{1}{(2\pi)^{\frac{1}{2}}} e^{-x^2/2}$

Radial Basis Function $\mathcal{K}_h(x) = \frac{1}{h^D (2\pi)^{D/2}} \prod_{d=1}^{D} \exp(-\frac{1}{2h^2} x_d^2)$ where $\mathcal{K}_h(x) \triangleq \frac{1}{h}\mathcal{K}(\frac{x}{h})$

(value depends on distance from origin)

$e^{-(\varepsilon r)^2}$

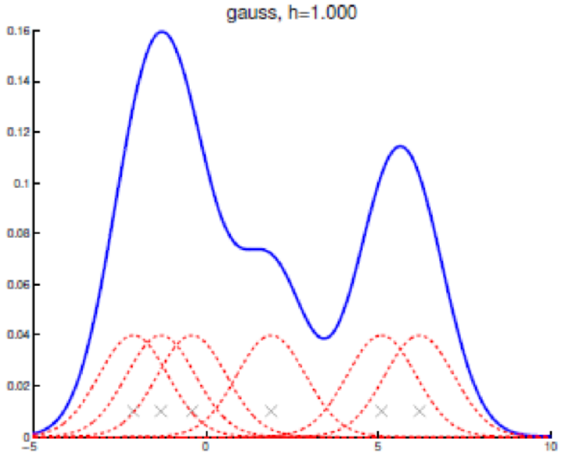


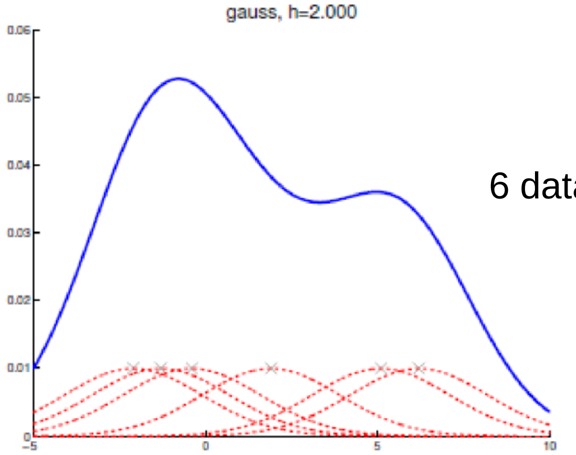
Credits: Wikipedia

Parzen Window Density Estimator

Allocate one cluster center per data point. Using Gaussian densities:

$$p(x|\theta) = \frac{1}{N}\sum_{n=1}^{N}\mathcal{N}(x|x_n,\sigma^2\mathbf{I}) \Rightarrow p(x|\mathcal{D}) = \frac{1}{N}\sum_{n=1}^{N}\mathcal{K}_h\left(x - x_n\right)$$





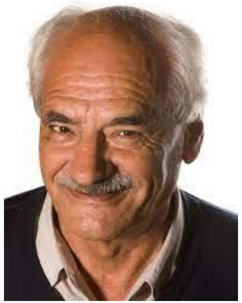
6 data points

Outline

- K Nearest Neighbor
- Kernel Density Estimation
- Support Vector Machine
- Decision Trees

Support Vector Machine

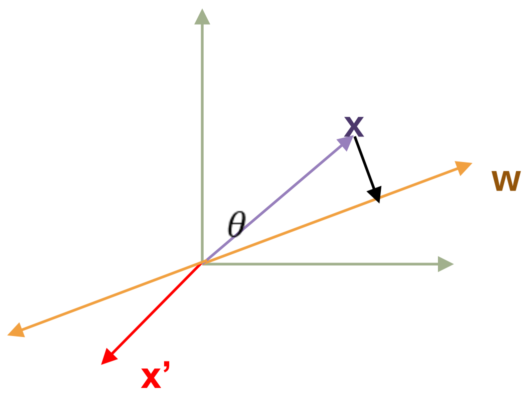
- Supervised learning models with associated learning algorithms that analyze data for classification and regression analysis.
- The original SVM algorithm was invented by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963



Support Vector Machine

- Robust algorithm for classification problems
  - Medical problems
  - Text and hypertext categorization
  - Image classification
- Advantages:
  - SVM depends on convex optimization
  - Easy to use
  - Excellent performance on different type of datasets

Projection Concept Recap



x = Datapoint
w = projection vector

To compute x projection on w,
assuming intercept = 0

$$w^T \cdot x = |w||x|cos\theta$$

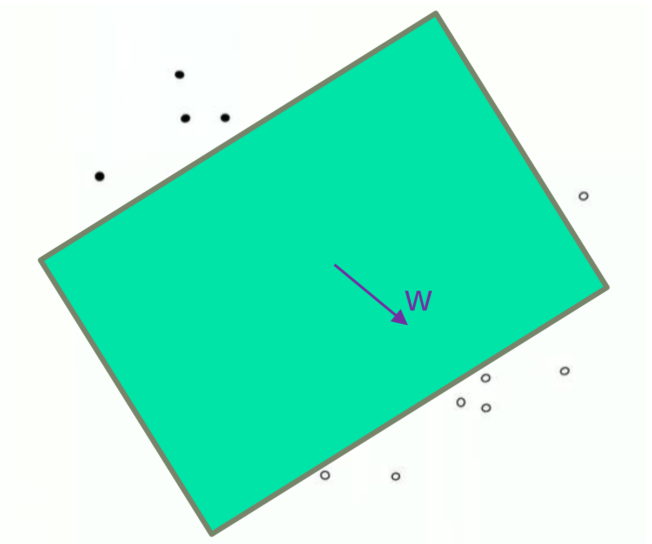
If $cos\theta > 0$ → Same side as w

$$w^T \cdot x = |w||x|cos\theta$$

If $cos\theta < 0$ → Opposite side as w

If $w^T \cdot x$ is +ve, same side as w
If $w^T \cdot x$ is –ve, opposite side as w

Linear Classifier



Assuming we have these points, we want to classify into black and white points.

We need a hyperplane, which is defined by the

- normal vector to the plane, w
- Intercept, b

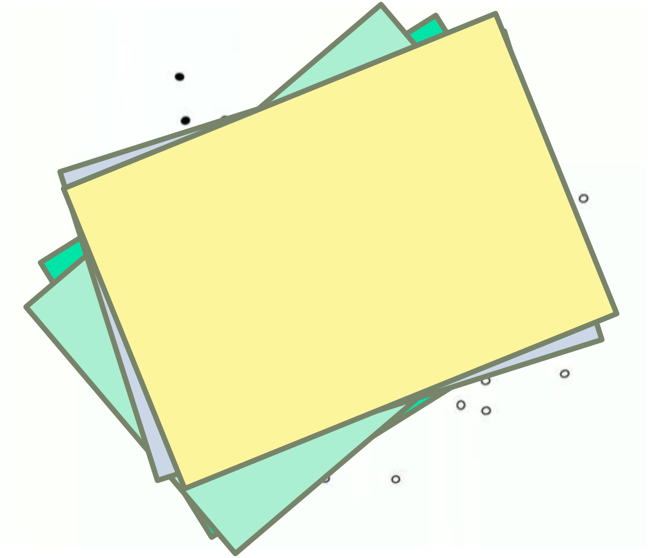
We want to have a classifier with a function of w and b.

$$x \longrightarrow \boxed{f\,(w,b)} \longrightarrow y$$

x is a data point, y is {-1, 1}
Black point = -1 ; white point = 1

$$y = f(x; w, b) = \operatorname{sgn}(w^T \cdot x + b)$$

Linear Classifier



Questions:

1. We can have different hyperplanes.

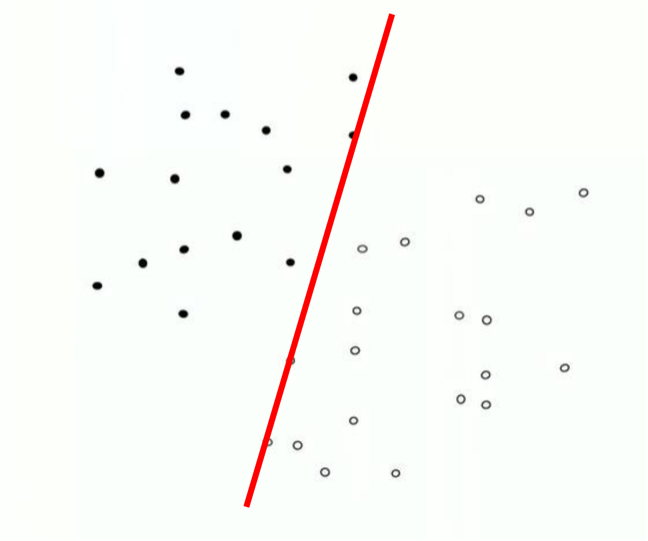
1. How do we choose the w and b so that we have an optimum classifier?

What we need is a evaluation metric to determine a good classifier.

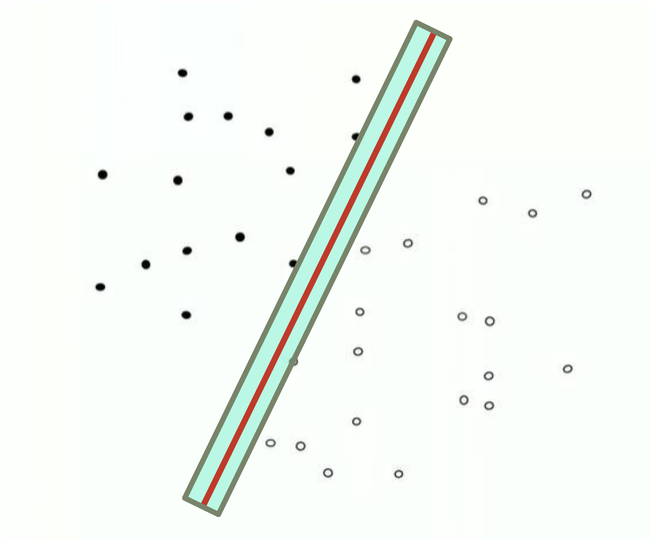
In SVM, we evaluate the margin - we want to maximize the margin between the points and the hyperplane.

Linear Classifier

Tight margin

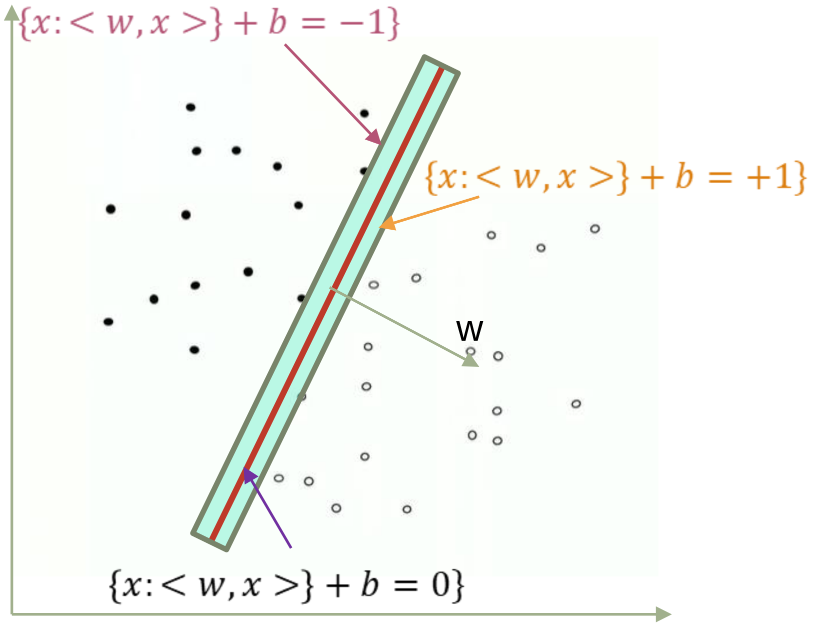


Maximize margin



To find which margin is the best, we need to measure the margin distance between points that touch the margin and the plane

Compute the margin

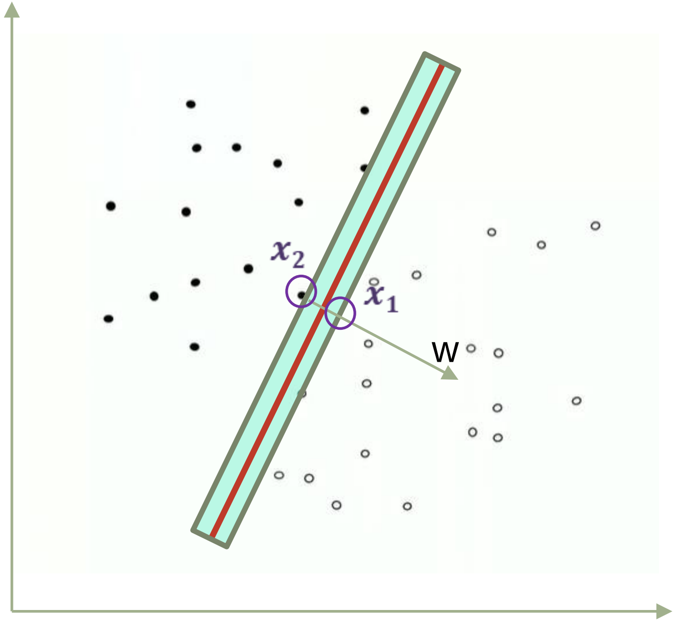


We first define our decision boundary function

$$y = f(x; w, b) = \mathrm{sgn}(w^T \cdot x + b)$$

What we know is
+ve value points are same as w
-ve value points are opposite as w
Zero value points are on the hyperplane

Maximizing the margin



$$w^T x_1 + b = 1$$

$$w^T x_2 + b = -1$$

$$w^T(x_1 - x_2) = 2$$

We want to maximize this $\frac{w^T(x_1 - x_2)}{|w|} = \frac{2}{|w|}$ (normalize)

Equivalent to minimizing the inverse, $\frac{|w|}{2}$

Or even better to minimizing the convex form, $\frac{|w|^2}{2}$

Objective function

Or even better to minimizing the convex form, $\frac{|w|^2}{2}$

Next thing to take note is to ensure the points sit on the correct side.

Add a constraint to the objective function

Objective function

3 scenarios:

1. Correct side but inside the margin

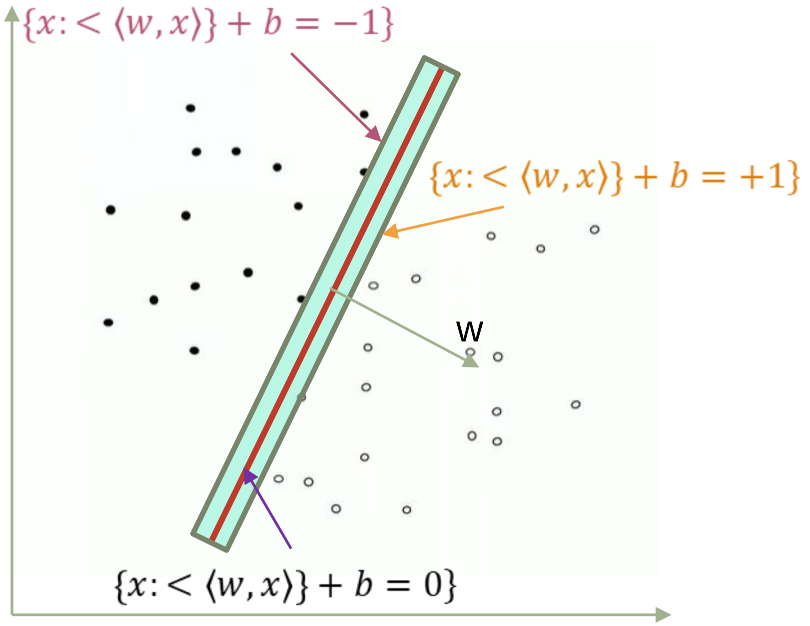
$$-1 < (\langle w, x_i \rangle + b) < 1$$

1. Correct side and outside the margin

$$\text{For } y_i = -1,\ (\langle w, x_i \rangle + b) < -1$$
$$\text{For } y_i = +1,\ (\langle w, x_i \rangle + b) > +1$$

1. Wrong side

$$\text{For } y_i = -1,\ (\langle w, x_i \rangle + b) > 0$$
$$\text{For } y_i = 1, (\langle w, x_i \rangle + b) < 0$$



Objective function

Finding the optimal hyperplane is an optimizing problem

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

subject to

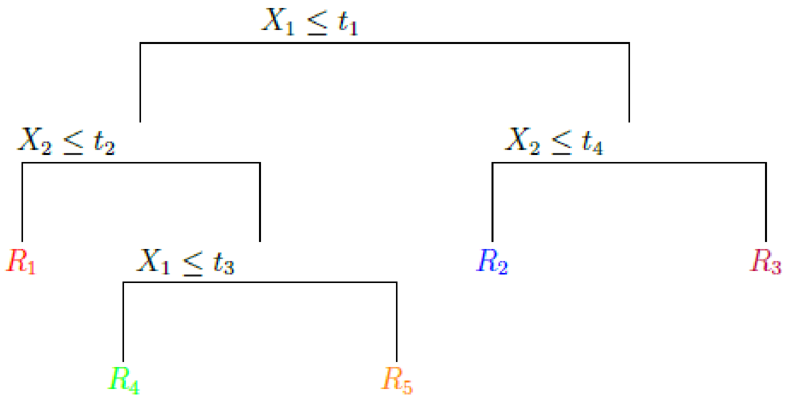
$$\left.\begin{aligned}\langle w, x_i\rangle + b \geq +1 \; when \; y_i = +1 \\ \langle w, x_i\rangle + b \leq -1 \; when \; y_i = -1\end{aligned}\right\} \Rightarrow y_i(\langle w, x_i\rangle + b) \geq 1, \; i = 1,2,3,\ldots, \mathrm{M}$$

- N+1 parameters (N: dimension of data)
- M constraints (M; number of datapoints)
- *Primal problem*

Outline

- K Nearest Neighbor
- Kernel Density Estimation
- Support Vector Machine
- Decision Trees

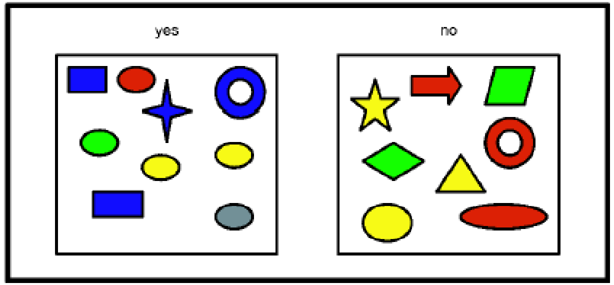
Classification and Regression Trees

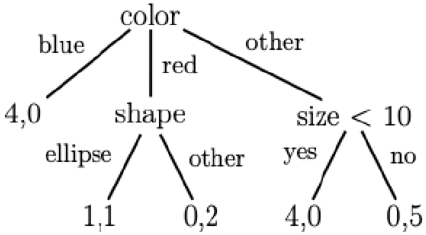


R_j is the region specified by the j'th leaf node

$$R_1 = [(d_1 \leq t_1), (d_2 \leq t_2)]$$

$$R_2 = [(d_1 \leq t_1), (d_2 > t_2), (d_3 \leq t_3)]$$

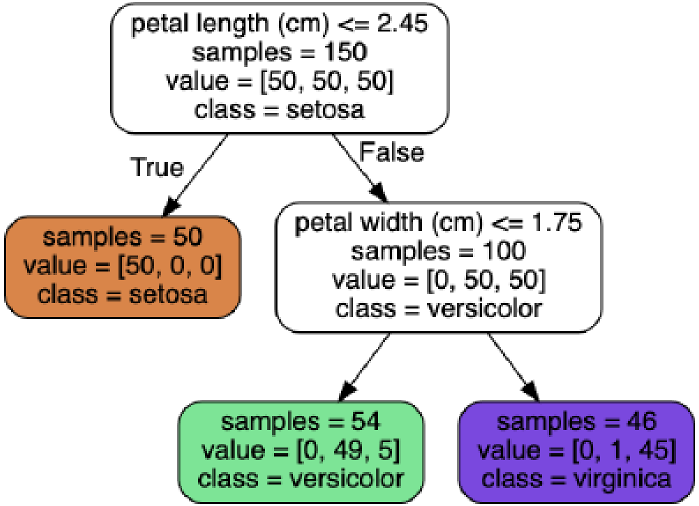




(n_1, n_0) $\quad$ n_1 *positive examples*

n_0 *negative examples*

Classification Tree Example



Ensemble Learning

$$f(y|x) = \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} f_m(y|x)$$

Reduce variance by averaging over multiple models

Bagging: Fit different base models to different randomly sampled (with replacement) versions of the data.

Random forest: Combines bagging and feature randomness to create a forest of decision trees. Learns trees based on a subset of input variables at each node of the tree.

Boosting: Combining weak learners to form a strong learner.