Kernel methods

Narayana Santhanam

EE 645
Jan 27, 2026

University of Hawai'i Mānoa

Linear $\rightarrow$ Non-linear

Two approaches:

Kernel methods

Linear $\rightarrow$ Non-linear

Two approaches:

Kernel methods

guarantees, well understood

Linear → Non-linear

Two approaches:

Kernel methods

guarantees, well understood

computationally intensive (small data)

Neural networks

less well understood

Linear $\rightarrow$ Non-linear

Two approaches:

Kernel methods

guarantees, well understood

computationally intensive (small data)

Neural networks

less well understood

computationally cheap (large data)

Next couple of weeks

Kernel methods

Next couple of weeks

Kernel methods

High level picture

Next couple of weeks

Kernel methods

High level picture
Support Vector classification and regression

Next couple of weeks

Kernel methods

High level picture
Support Vector classification and regression
Kernel PCA

Next couple of weeks

Kernel methods

High level picture
Support Vector classification and regression
Kernel PCA
Random Fourier Futures

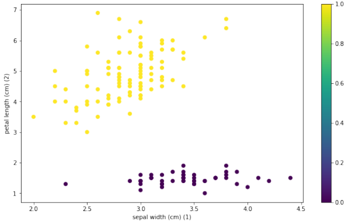
Next couple of weeks

Kernel methods

High level picture
Support Vector classification and regression
Kernel PCA
Random Fourier Futures
Credit risk, Power data

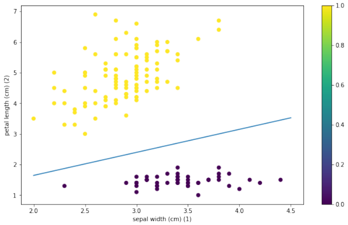
Classification

Linearly separable



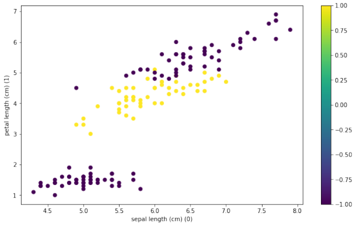
Classification

Linearly separable



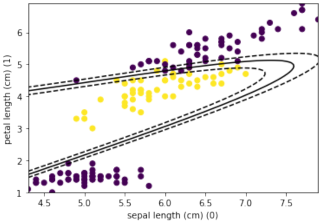
Classification

Not linearly separable



Classification

Not linearly separable



Boundaries not linear..

but are linear in a higher dimensional space!

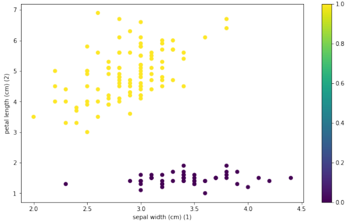
Closer look

Principled approach for linear $\rightarrow$ nonlinear
Powerful, yet generalizes well
Often explainable
    at least more than other state of art
New advances increase reach (more data)
    but not as much as NN

Stepping back

Linearly separable

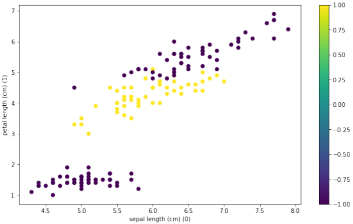
Where would you draw a *linear* classifier?



Stepping back

Not linearly separable

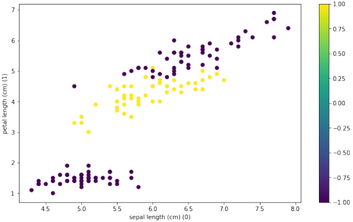
Where would you draw a *linear* classifier?



Stepping back

Not linearly separable

Where would you draw a *linear* classifier?

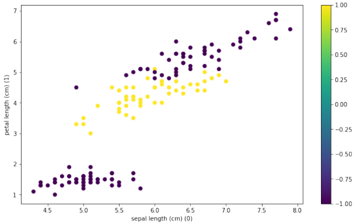


Minimum number of mistakes?

Stepping back

Not linearly separable

Where would you draw a *linear* classifier?



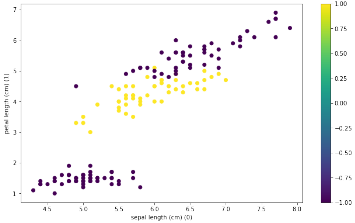
Minimum number of mistakes?

infeasible, NP-hard

Stepping back

Not linearly separable

Where would you draw a *linear* classifier?



Minimum number of mistakes?

infeasible, NP-hard

Instead: variation of ℓ_1 loss, sum of margins

Setting up linear classification

Distance of $\mathbf{z}$ from a plane $\mathbf{w}^T\mathbf{x} - b = 0$?

Setting up linear classification

Distance of $\mathbf{z}$ from a plane $\mathbf{w}^T\mathbf{x} - b = 0$?

$$\frac{\mathbf{w}^T\mathbf{z} - b}{||\mathbf{w}||}$$

Formulation of Support Vector Classification

Analyze this in some detail

Formulation of Support Vector Classification

Analyze this in some detail

One of the key advantages: interpretability

- Comes through formulation

Formulation of Support Vector Classification

Analyze this in some detail

One of the key advantages: interpretability

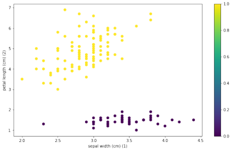
- Comes through formulation

Primal/Dual formulation is a key optimization idea

- accelerations of training neural networks
- resource allocation/optimization in economics,
  - urban planning

Formulation

Linearly separable points:



Training data:

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$$

Margin (closest distance from separating boundary)

$$\gamma(\mathbf{w}, b) = \min_{\mathbf{x}_i} \frac{|\mathbf{w}^T \mathbf{x}_i - b|}{||\mathbf{w}||}$$

Redundant parameterization scaling $\mathbf{w}, b$ by any number does not change the margin

Formulation

Support vector machine formulation:

$$\mathbf{w}^*, b^* = \arg\max_{\mathbf{w},b} \gamma(\mathbf{w}, b)$$

subject to $y_i(\mathbf{w}^T\mathbf{x}_i - b) \geq 0$ (for all training $(\mathbf{x}_i, y_i)$)

Redundant parameterization: scaling $\mathbf{w}, b$ changes nothing

- only the directions/intercepts matter
- represent by scaling of $\mathbf{w}, b$ that ensures

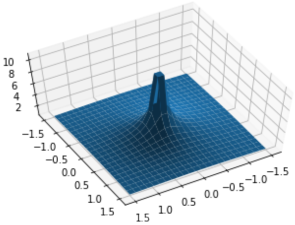
$$\min_{\mathbf{x}_i} |\mathbf{w}^T\mathbf{x}_i - b| = 1$$

Formulation

Support vector machine formulation
(removing the redundant formulation)

$$\mathbf{w}^*, b^* = \arg\max_{\mathbf{w}, b} \frac{1}{||\mathbf{w}||}$$

subject to $y_i(\mathbf{w}^T \mathbf{x}_i - b) \geq 1$ (for all training $(\mathbf{x}_i, y_i)$)



Formulation

$\frac{1}{||\mathbf{w}||}$ is convex, but...

maximizing (convex domain) isn't convex optimization

minimize convex/maximize concave in convex domain

But it is easy to come with a convex formulation

$$\mathbf{w}^*, b^* = \arg\min_{\mathbf{w}, b} \frac{1}{2} ||\mathbf{w}||^2$$

subject to $y_i(\mathbf{w}^T \mathbf{x}_i - b) \geq 1$ (for all pairs $(\mathbf{x}_i, y_i)$)

A bit on convex optimization

General concepts beyond support vector machine formulation

A bit on convex optimization

General concepts beyond support vector machine formulation

Lagrangian:

$$L(\mathbf{w}, b, \Lambda) = \frac{1}{2}||\mathbf{w}||^2 + \sum_i \lambda_i \left(1 - y_i(\mathbf{w}^T \mathbf{x}_i - b)\right)$$

$L(\mathbf{w}, b, \Lambda)$ is a key idea in any constrained optimization

Primal/dual formulation

Neat property of Lagrangians:

Optimal point

$\min_{\mathbf{w},b} \max_{\Lambda \geq 0} L(\mathbf{w}, b, \Lambda)$ (primal formulation)

Dual formulation (Nash, von Neumann)

$\max_{\Lambda \geq 0} \min_{\mathbf{w},b} L(\mathbf{w}, b, \Lambda)$ (dual formulation)

Primal/dual formulation

Neat property of Lagrangians:
Optimal point
$\min_{\mathbf{w},b} \max_{\Lambda \geq 0} L(\mathbf{w}, b, \Lambda)$ (primal formulation)

Dual formulation (Nash, von Neumann)
$\max_{\Lambda \geq 0} \min_{\mathbf{w},b} L(\mathbf{w}, b, \Lambda)$ (dual formulation)

Incidentally Nash won both the Nobel Prize in Econ and the Abel Prize

Primal/dual formulation

For free:

$$\min_{\mathbf{w},b} \max_{\Lambda \geq 0} L(\mathbf{w}, b, \Lambda) \geq \max_{\Lambda \geq 0} \min_{\mathbf{w},b} L(\mathbf{w}, b, \Lambda)$$

But equality in many cases

- in many convex formulations, including our current case
- so one could solve either version
- dual in kernel methods very insightful for explainability

Dual formulation: 2 key insights

- Setting gradient of $L(\mathbf{w}, b, \Lambda)$ to 0, optimal $\mathbf{w}^*$ satisfies

$$\mathbf{w}^* = \sum_i \lambda_i y_i \mathbf{x}_i$$

Representer theorem: solution $\mathbf{w}^*$ is linear combination of inputs

Dual formulation: 2 key insights

- Setting gradient of $L(\mathbf{w}, b, \Lambda)$ to 0, optimal $\mathbf{w}^*$ satisfies

$$\mathbf{w}^* = \sum_i \lambda_i y_i \mathbf{x}_i$$

Representer theorem: solution $\mathbf{w}^*$ is linear combination of inputs

- Finding Λs

$$\max_{\Lambda \geq 0} \sum \lambda_i - \frac{1}{2} \begin{bmatrix} \lambda_1 y_1 & \ldots & \lambda_n y_n \end{bmatrix} X X^T \begin{bmatrix} \lambda_1 y_1 \\ \vdots \\ \lambda_n y_n \end{bmatrix}$$

Data only shows up through dot products (XX^T)

Crux of Kernel approach to nonlinearity