Machine Learning

Lecture 5

Common Algorithms for Supervised Learning

Vincent Adam & Vicenç Gómez

2023-2024

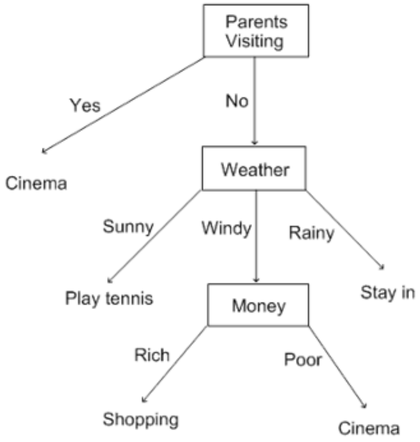
Content

Content

Definition

- Wikipedia: “A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences [...]”

Example


Parents Visiting
Yes
No
Cinema
Weather
Sunny
Windy
Rainy
Play tennis
Money
Stay in
Rich
Poor
Shopping
Cinema

Components

- Internal node: represents a test on a given feature
- Branch: represents a possible outcome of a given test
- Leaf node: represents a label or decision

Advantages

- Easy to understand and interpret!
- Useful even when little data is available
- Allow experts to express knowledge about a given problem

Relationship to Machine Learning

- A decision tree can represent a hypothesis $h(x)$!
- Input set $\mathcal{X} = \mathcal{X}^1 \times \cdots \times \mathcal{X}^d$
- Input $x \in \mathcal{X}$: vector of feature values
- Output: Label $h(x)$ on the leaf node reached by resolving the test at each internal node

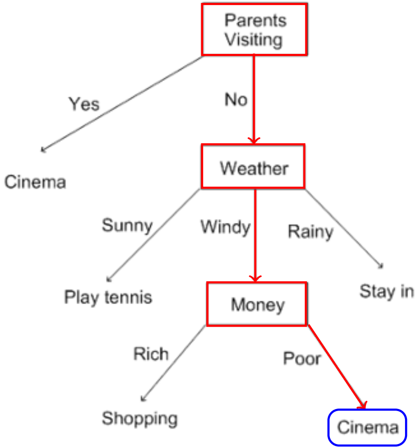
Example

- Input set $\mathcal{X} = \mathcal{X}^1 \times \mathcal{X}^2 \times \mathcal{X}^3$

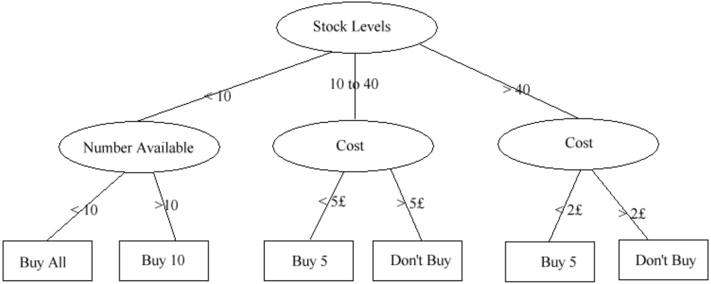
$$
\begin{array}{ll}
\mathcal{X}^1 = \{\text{Yes, No}\} & \text{(parents visiting)} \\
\mathcal{X}^2 = \{\text{Sunny, Windy, Rainy}\} & \text{(weather)} \\
\mathcal{X}^3 = \{\text{Rich, Poor}\} & \text{(money)}
\end{array}
$$

- Target set: $\mathcal{Y} = \{\text{Cinema, Play tennis, Shopping, Stay in}\}$
- Example input: $x = (\text{No, Windy, Poor}) \in \mathcal{X}$

Example

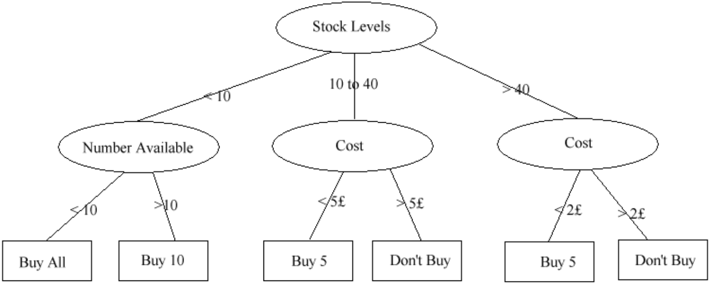

Parents Visiting
Yes
No
Cinema
Weather
Sunny
Windy
Rainy
Play tennis
Money
Stay in
Rich
Poor
Shopping
Cinema

Input set



- Finite set: test outcome is a specific value
- Real numbers: test outcome is an interval

Input set



- Finite set: test outcome is a specific value
- Real numbers: test outcome is an interval

How to learn trees from data?

Entropy

- Measures information content
- Large entropy $\Rightarrow$ unpredictable outcome
- For a random variable X, the entropy is expressed as

$$H(X) = \mathbb{E}[I(X)] = \mathbb{E}[-\log(P(X))]$$

  - $I(X)$: information content of X (itself a random variable)
  - $P(X)$: probability mass function
- Unsupervised training set $S = (x_1, \ldots, x_m)$:

$$H(S) = -\sum_{i=1}^{m} P(x_i) \log P(x_i)$$

Entropy of input-label pairs

- Training set $S = ((x_1, y_1), \ldots, (x_m, y_m))$ of input-label pairs
- For each $y \in \mathcal{Y}$, $S(\mathcal{Y}, y) \equiv \{(x_i, y_i) \in S : y_i = y\} \subseteq S$
- For each $y \in \mathcal{Y}$, $P(y) \equiv \frac{|S(\mathcal{Y},y)|}{|S|}$
- Entropy $H(S) = -\sum_{y \in \mathcal{Y}} P(y) \log P(y)$

Information gain

Expected information gain $=$ change in entropy

- $IG(S, \mathcal{X}^i) = H(S) - H(S|\mathcal{X}^i)$
- $H(S|\mathcal{X}^i) = \sum_{v \in \mathcal{X}^i} \frac{|S(\mathcal{X}^i, v)|}{|S|} H(S(\mathcal{X}^i, v))$

Learning Decision Trees

- Start with a single root node
- For each feature $\mathcal{X}^i$ not used in tests, compute $IG(S, \mathcal{X}^i)$
- Let $\mathcal{X}^* = \arg\max_{\mathcal{X}^i} IG(S, \mathcal{X}^i)$
- Split the node on $\mathcal{X}^*$ and distribute data points to new leaves
- Repeat at leaves until no more information gain is possible

Disadvantages

- Greedy: can get stuck in local minima
- Does not take into account correlation among attributes
- Overfitting: overly large decision tree
- Difficult to apply to real-valued attributes (where to split?)

Bootstrap Aggregating (Bagging)

- Given a training set $S = ((x_1, y_1), \ldots, (x_m, y_m))$, generate k smaller training sets of size $m' < m$ by sampling from S
- Learn a separate decision tree for each of the k training sets
- Aggregate the output of each decision tree
  - Regression: average the outputs
  - Classification: output by voting

Content

Support vector machine

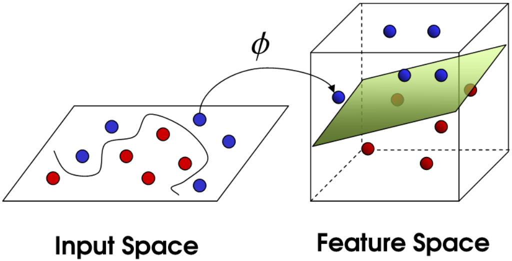
Support vector machine (SVM)

- Model for supervised learning
- Basic algorithm: binary linear classification ($\mathcal{Y} = \{-1, +1\}$)

Intuition

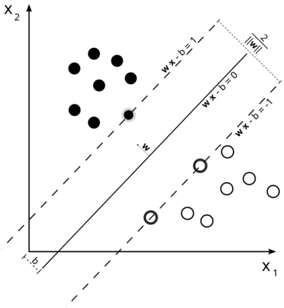
- data is not linearly separable? $\rightarrow$ increase the number of features
- Use kernels to avoid computational blowup and overfitting.

Illustration



Maximum-margin hyperplane

- Maximizes the margin, i.e. the distance from the data points
- Idea: consider two boundary hyperplanes



Algorithm

- Equation of a hyperplane: $\sum_{i=1}^{d} w_i x_i + b = \langle w, x \rangle + b = 0$
- Boundary hyperplanes: $\langle w, x \rangle + b = -1$ and $\langle w, x \rangle + b = 1$
- Distance between boundary hyperplanes: $2/\|w\|$
- Maximize distance $\Rightarrow$ minimize $\|w\|$ $\Rightarrow$ minimize $\frac{1}{2}\|w\|^2$

Constrained optimization

- Constraints:
  - For each x_i such that $y_i = +1$, $\langle w, x_i \rangle + b \geq 1$
  - For each x_i such that $y_i = -1$, $\langle w, x_i \rangle + b \leq -1$
- Rewrite as $y_i(\langle w, x_i \rangle + b) \geq 1$

Constrained optimization

- Constraints:
  - For each x_i such that $y_i = +1$, $\langle w, x_i \rangle + b \geq 1$
  - For each x_i such that $y_i = -1$, $\langle w, x_i \rangle + b \leq -1$
- Rewrite as $y_i(\langle w, x_i \rangle + b) \geq 1$
- Quadratic programming optimization problem:

$$\min_{w,b} \quad \frac{1}{2}\langle w, w \rangle$$

subject to $y_i(\langle w, x_i \rangle + b) \geq 1$ for each $1 \leq i \leq m$

Support vectors

- Solution is of the form $w = \sum_{i=1}^{m} \alpha_i y_i x_i$
- Support vectors: inputs x_i such that $\alpha_i > 0$
- Support vectors satisfy equality: $y_i(\langle w, x_i \rangle + b) = 1$

Dual form

- One can show that the dual optimization problem is defined as

$$\max_\alpha \quad \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m \alpha_i\alpha_j y_i y_j \langle x_i, x_j \rangle$$

$$\text{subject to} \begin{cases} \alpha_i \geq 0 \text{ for each } 1 \leq i \leq m, \\ \sum_{i=1}^m \alpha_i y_i = 0 \end{cases}$$

Soft margin

The basic algorithm presented requires data to be linearly separable.
How to deal with data that is not linearly separable?

Soft margin

- allow mislabelled examples
- Find hyperplane that splits the examples as cleanly as possible
- Idea: upper bound each α_i by a constant C
- Optimization problem not significantly more complex

Model complexity

- Vapnik: Model complexity proportional to $\|w\|$
- Intuition: if $\|w\|$ is small, the true error of the separating maximum-margin hyperplane is close to the training error
- Independent of the number of features!

Non-linear classification

- Basic algorithm performs binary linear classification
- In general, data not linearly separable
- Non-linear transformation from original space to new space
- Choose transformation such that data is (almost) linearly separable in transformed space

Inner product space

- Vector space that defines an inner product $\langle \cdot, \cdot \rangle$
- Vectors x and y orthogonal $\Leftrightarrow \langle x, y \rangle = 0$
- Inner product induces a norm $\|x\| = \sqrt{\langle x, x \rangle}$
- Generalizes the Euclidean space (inner product $=$ scalar product)

Kernel trick

- Map observations from a set $\mathcal{X}$ to an inner product space V
- Trick: in V, only use algorithms based on inner product
- Kernels compute inner product directly on elements in $\mathcal{X}$
- No need to compute the mapping from $\mathcal{X}$ to V explicitly!

$$\max_\alpha \quad \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

$$\text{subject to} \begin{cases} \alpha_i \geq 0 \text{ for each } 1 \leq i \leq m, \\ \sum_{i=1}^m \alpha_i y_i = 0 \end{cases}$$

Non-linear transformation

- Map original feature space to high-dimensional space
- Find maximum-margin hyperplane in transformed space
- Replace each dot product with a kernel $k(x_1, x_2) = \langle\phi(x_1), \phi(x_2)\rangle$
- A kernel intrinsically regularizes $\rightarrow$ avoids overfitting

Non-linear transformation

- Map original feature space to high-dimensional space
- Find maximum-margin hyperplane in transformed space
- Replace each dot product with a kernel $k(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle$
- A kernel intrinsically regularizes $\rightarrow$ avoids overfitting

Example

$$\phi(x) = [1, \sqrt{2}x, x^2]$$

$$\langle \phi(x_1), \phi(x_2) \rangle = \phi(x_1)^T \phi(x_2) = 1 + 2x_1 x_2 + x_1^2 x_2^2 = (1 + x_1 x_2)^2$$

Common kernels

$k(x_1, x_2) = x_1^\top x_2$:	dot product (Euclidean)
$k(x_1, x_2) = (x_1^\top x_2)^q$:	polynomial homogeneous
$k(x_1, x_2) = (x_1^\top x_2 + 1)^q$:	polynomial inhomogeneous
$k(x_1, x_2) = \exp(-\gamma \|x_1 - x_2\|^2)$:	Gaussian radial basis, $\gamma > 0$
$k(x_1, x_2) = \tanh(\kappa x_1^\top x_2 - c)$:	Hyperbolic tangent, $\kappa, c > 0$

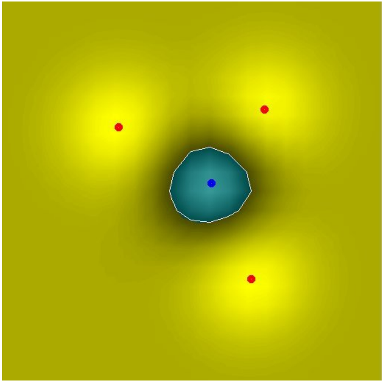
Classification

- Solution in V is of the form $w = \sum_{i=1}^{m} \alpha_i y_i \phi(x_i)$
- Classification of new example x:

$$\langle w, \phi(x) \rangle = \sum_{i=1}^{m} \alpha_i y_i \langle \phi(x_i), \phi(x) \rangle = \sum_{i=1}^{m} \alpha_i y_i k(x_i, x)$$

- In general, there is no w' in $\mathcal{X}$ such that $\langle w, \phi(x) \rangle = k(w', x)$

Intuition of Gaussian radial basis



Properties

- Generalizes the perceptron
- Simultaneously minimizes classification error and maximizes geometric margin

Disadvantages

- Highly dependent on the kernel and the kernel parameters
- Highly dependent on the soft margin constant C
- Uncalibrated class membership probabilities
- Only directly applicable to two-class tasks
- Solved model difficult to interpret

Common use

- Use a Gaussian radial basis kernel (single parameter γ)
- Grid search to find best combination of γ and C
- Use cross validation to test each parameter choice
- Final model trained on complete data set using chosen parameters

Multi-class SVM

- Generalize the algorithm for binary classification
- Classify data with a finite number $L > 2$ of class labels
- Reduce to multiple binary classification problems

Content

Decision tree learning

- $\mathcal{X} = \mathcal{X}^1 \times \mathcal{X}^2 \times \mathcal{X}^3$ such that $\mathcal{X}^1 = \mathcal{X}^2 = \mathcal{X}^3 = \mathcal{Y} = \{0, 1\}$
- Each data point is on the format $(x, y) = (x^1, x^2, x^3, y)$
- $S = \{(0, 0, 0, 0), (0, 1, 0, 1), (1, 0, 1, 1), (1, 1, 0, 0)\}$
- Apply the Information gain maximization to learn a decision tree

Entropy of S

- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$
- $S(\mathcal{Y},0) = \{(0,0,0,0),(1,1,0,0)\}$,
  $S(\mathcal{Y},1) = \{(0,1,0,1),(1,0,1,1)\}$

Entropy of S

- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$
- $S(\mathcal{Y},0) = \{(0,0,0,0),(1,1,0,0)\}$,
  $S(\mathcal{Y},1) = \{(0,1,0,1),(1,0,1,1)\}$
- $P(0) = \frac{|S(\mathcal{Y},0)|}{|S|} = \frac{2}{4} = \frac{1}{2}$, $P(1) = \frac{|S(\mathcal{Y},1)|}{|S|} = \frac{2}{4} = \frac{1}{2}$

Entropy of S

- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$
- $S(\mathcal{Y},0) = \{(0,0,0,0),(1,1,0,0)\}$,
  $S(\mathcal{Y},1) = \{(0,1,0,1),(1,0,1,1)\}$
- $P(0) = \frac{|S(\mathcal{Y},0)|}{|S|} = \frac{2}{4} = \frac{1}{2}$, $P(1) = \frac{|S(\mathcal{Y},1)|}{|S|} = \frac{2}{4} = \frac{1}{2}$

$$H(S) = -P(0)\log P(0) - P(1)\log P(1) =$$

$$= -\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2} =$$

$$= -\log\frac{1}{2} = -(-1) = 1$$

Splitting on $\mathcal{X}^1$

- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$
- $S_0 = S(\mathcal{X}^1, 0) = \{(0,0,0,0),(0,1,0,1)\}$,
  $S_1 = S(\mathcal{X}^1, 1) = \{(1,0,1,1),(1,1,0,0)\}$

Splitting on $\mathcal{X}^1$

- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$
- $S_0 = S(\mathcal{X}^1, 0) = \{(0,0,0,0),(0,1,0,1)\}$,
  $S_1 = S(\mathcal{X}^1, 1) = \{(1,0,1,1),(1,1,0,0)\}$
- $P_0(0) = \frac{1}{2}$, $P_0(1) = \frac{1}{2}$, $P_1(0) = \frac{1}{2}$, $P_1(1) = \frac{1}{2}$

Splitting on $\mathcal{X}^1$

- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$
- $S_0 = S(\mathcal{X}^1, 0) = \{(0,0,0,0),(0,1,0,1)\}$,
  $S_1 = S(\mathcal{X}^1, 1) = \{(1,0,1,1),(1,1,0,0)\}$
- $P_0(0) = \frac{1}{2}$, $P_0(1) = \frac{1}{2}$, $P_1(0) = \frac{1}{2}$, $P_1(1) = \frac{1}{2}$

$$H(S|\mathcal{X}^1) = \frac{|S_0|}{|S|}H(S_0) + \frac{|S_1|}{|S|}H(S_1)$$

$$H(S_0) = -\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2} = 1$$

$$H(S_1) = -\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2} = 1$$

$$IG(S,\mathcal{X}^1) = H(S) - H(S|\mathcal{X}^1) = 1 - \left(\frac{1}{2}\cdot 1 + \frac{1}{2}\cdot 1\right) = 0$$

Splitting on $\mathcal{X}^2$

- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$
- $S_0 = S(\mathcal{X}^2, 0) = \{(0,0,0,0),(1,0,1,1)\}$,
  $S_1 = S(\mathcal{X}^2, 1) = \{(0,1,0,1),(1,1,0,0)\}$

Splitting on $\mathcal{X}^2$

- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$
- $S_0 = S(\mathcal{X}^2, 0) = \{(0,0,0,0),(1,0,1,1)\}$, $S_1 = S(\mathcal{X}^2, 1) = \{(0,1,0,1),(1,1,0,0)\}$
- $P_0(0) = \frac{1}{2}$, $P_0(1) = \frac{1}{2}$, $P_1(0) = \frac{1}{2}$, $P_1(1) = \frac{1}{2}$

Splitting on $\mathcal{X}^2$

- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$
- $S_0 = S(\mathcal{X}^2, 0) = \{(0,0,0,0),(1,0,1,1)\}$,
  $S_1 = S(\mathcal{X}^2, 1) = \{(0,1,0,1),(1,1,0,0)\}$
- $P_0(0) = \frac{1}{2}$, $P_0(1) = \frac{1}{2}$, $P_1(0) = \frac{1}{2}$, $P_1(1) = \frac{1}{2}$

$$H(S|\mathcal{X}^2) = \frac{|S_0|}{|S|}H(S_0) + \frac{|S_1|}{|S|}H(S_1)$$

$$H(S_0) = -\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2} = 1$$

$$H(S_1) = -\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2} = 1$$

$$IG(S,\mathcal{X}^2) = H(S) - H(S|\mathcal{X}^2) = 1 - \left(\frac{1}{2}\cdot 1 + \frac{1}{2}\cdot 1\right) = 0$$

Splitting on $\mathcal{X}^3$

- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$
- $S_0 = S(\mathcal{X}^3, 0) = \{(0,0,0,0),(0,1,0,1),(1,1,0,0)\}$,
  $S_1 = S(\mathcal{X}^3, 1) = \{(1,0,1,1)\}$

Splitting on $\mathcal{X}^3$

- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$
- $S_0 = S(\mathcal{X}^3, 0) = \{(0,0,0,0),(0,1,0,1),(1,1,0,0)\}$,
  $S_1 = S(\mathcal{X}^3, 1) = \{(1,0,1,1)\}$
- $P_0(0) = \frac{2}{3}$, $P_0(1) = \frac{1}{3}$, $P_1(0) = 0$, $P_1(1) = 1$

Splitting on $\mathcal{X}^3$

- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$
- $S_0 = S(\mathcal{X}^3, 0) = \{(0,0,0,0),(0,1,0,1),(1,1,0,0)\}$,
  $S_1 = S(\mathcal{X}^3, 1) = \{(1,0,1,1)\}$
- $P_0(0) = \frac{2}{3}$, $P_0(1) = \frac{1}{3}$, $P_1(0) = 0$, $P_1(1) = 1$

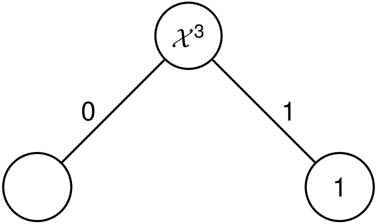
$$H(S|\mathcal{X}^3) = \frac{|S_0|}{|S|}H(S_0) + \frac{|S_1|}{|S|}H(S_1)$$

$$H(S_0) = -\frac{2}{3}\log\frac{2}{3} - \frac{1}{3}\log\frac{1}{3} \approx 0.27$$

$$H(S_1) = -0\log 0 - 1\log 1 = 0$$

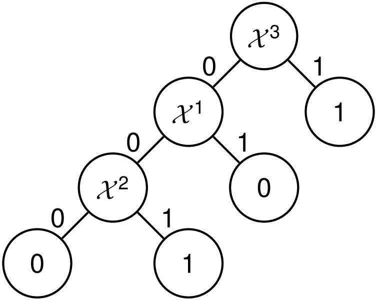
$$IG(S,\mathcal{X}^3) = H(S) - H(S|\mathcal{X}^3) = 1 - \left(\frac{3}{4}\cdot 0.27 + \frac{1}{4}\cdot 0\right) \approx 0.79$$

Resulting tree



- Split on left node with $S_0 = \{(0,0,0,0),(0,1,0,1),(1,1,0,0)\}$
- $S_1 = \{(0,1,0,1)\}$ cannot be split any further, so we predict 1

Final tree



- $S = \{(0,0,0,0),(0,1,0,1),(1,0,1,1),(1,1,0,0)\}$

Support vector machines

- Derive the dual optimization problem for SVMs

Support vector machines

- The constrained optimization problem is given by

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

Support vector machines

- The constrained optimization problem is given by

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

- The Lagrangian is given by

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2}\langle w, w\rangle + \sum_{i=1}^{m} \alpha_i(1 - y_i(\langle w, x_i\rangle + b))$$

Support vector machines

- The Lagrangian is given by

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2}\langle w, w\rangle + \sum_{i=1}^{m} \alpha_i(1 - y_i(\langle w, x_i\rangle + b))$$

Support vector machines

- The Lagrangian is given by

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2}\langle w, w\rangle + \sum_{i=1}^{m} \alpha_i (1 - y_i(\langle w, x_i\rangle + b))$$

- The KKT conditions are given by

$$\nabla \mathcal{L}(w, b, \alpha) = 0$$
$$\alpha_i (1 - y_i(\langle w, x_i\rangle + b)) = 0, \quad \forall i \in [m]$$

Support vector machines

- Setting the gradient of the Lagrangian to 0 yields

$$\nabla \mathcal{L}(w,b,\alpha) = \begin{pmatrix} \frac{\partial \mathcal{L}(w,b,\alpha)}{\partial w_1} \\ \vdots \\ \frac{\partial \mathcal{L}(w,b,\alpha)}{\partial w_d} \\ \frac{\partial \mathcal{L}(w,b,\alpha)}{\partial b} \end{pmatrix} = \begin{pmatrix} w_1 - \sum_{i=1}^{m} \alpha_i y_i x_i^1 \\ \vdots \\ w_d - \sum_{i=1}^{m} \alpha_i y_i x_i^d \\ -\sum_{i=1}^{m} \alpha_i y_i \end{pmatrix} = 0$$

Support vector machines

- Setting the gradient of the Lagrangian to 0 yields

$$\nabla \mathcal{L}(w, b, \alpha) = \begin{pmatrix} \frac{\partial \mathcal{L}(w,b,\alpha)}{\partial w_1} \\ \vdots \\ \frac{\partial \mathcal{L}(w,b,\alpha)}{\partial w_d} \\ \frac{\partial \mathcal{L}(w,b,\alpha)}{\partial b} \end{pmatrix} = \begin{pmatrix} w_1 - \sum_{i=1}^{m} \alpha_i y_i x_i^1 \\ \vdots \\ w_d - \sum_{i=1}^{m} \alpha_i y_i x_i^d \\ -\sum_{i=1}^{m} \alpha_i y_i \end{pmatrix} = 0$$

- Solution given by $w^* = \sum_{i=1}^{m} \alpha_i y_i x_i$ and $\sum_{i=1}^{m} \alpha_i y_i = 0$

Support vector machines

- Setting the gradient of the Lagrangian to 0 yields

$$\nabla\mathcal{L}(w,b,\alpha) = \begin{pmatrix} \frac{\partial\mathcal{L}(w,b,\alpha)}{\partial w_1} \\ \vdots \\ \frac{\partial\mathcal{L}(w,b,\alpha)}{\partial w_d} \\ \frac{\partial\mathcal{L}(w,b,\alpha)}{\partial b} \end{pmatrix} = \begin{pmatrix} w_1 - \sum_{i=1}^m \alpha_i y_i x_i^1 \\ \vdots \\ w_d - \sum_{i=1}^m \alpha_i y_i x_i^d \\ -\sum_{i=1}^m \alpha_i y_i \end{pmatrix} = 0$$

- Solution given by $w^* = \sum_{i=1}^m \alpha_i y_i x_i$ and $\sum_{i=1}^m \alpha_i y_i = 0$
- For any $k \in [m]$ such that $\alpha_k > 0$ we have $1 = y_k(\langle w, x_k\rangle + b^*)$

$$\begin{aligned} \Leftrightarrow \quad & y_k = y_k^2(\langle w, x_k\rangle + b^*) = \langle w, x_k\rangle + b^* \\ \Leftrightarrow \quad & b^* = y_k - \langle w, x_k\rangle \end{aligned}$$

Support vector machines

- Inserting w^* and b^* into the Lagrangian yields the dual objective

$$
\begin{aligned}
g(\alpha) = \mathcal{L}(w^*, b^*, \alpha) = \frac{1}{2}\langle \sum_{i=1}^{m} \alpha_i y_i x_i, \sum_{j=1}^{m} \alpha_j y_j x_j \rangle + \sum_{i=1}^{m} \alpha_i \\
- \sum_{i=1}^{m} \alpha_i y_i \langle \sum_{j=1}^{m} \alpha_j y_j x_j, x_i \rangle - b^* \sum_{i=1}^{m} \alpha_i y_i \\
= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle + 0
\end{aligned}
$$

Support vector machines

- The dual optimization problem is given by

$$\begin{aligned}
\max_{\alpha} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \\
\text{s.t.} \quad & \alpha_i \geq 0, \quad \forall i \in [m] \\
& \sum_{i=1}^{m} \alpha_i y_i = 0
\end{aligned}$$