1 Maschine Learning

Algorithm learns class of tasks, measured by loss function, from experience.

supervised learning learn $h : \Delta^* \to \Sigma^*, h = t$; example: $(x, y) \in \Delta^* \times \Sigma^*$, $t(x) = y$.

unsupervised learning learn $h : \Delta^* \to \Sigma^*, \ker(h) = \ker(t)$; example: $x \in \Delta^*$.

reinforcement learning learn strategy based on feedback from environment.

2 Supervised Learning

- model function $t : \mathcal{M} \to \mathcal{R}$
- $supp(t) = \{m \in \mathcal{M} \mid t(m) \neq 0\}$
- $\bar{m} \in supp(t) \Leftrightarrow t(\bar{m}) = 1$

Hypothesis of A: potential result of A

Hypothesis space $\mathcal{H}_A$ of A: set of all hypotheses

h fits D if $h(x_i) = y_i$ for all $(x_i, y_i) \in D$

Version space $\mathcal{V}_A(D)$ of A: all hypotheses that fit D

Inductive bias of A: set of assumptions that A uses to predict outputs of unseen data

2.1 Conjunctive Clause

$\theta = (\theta_1, ..., \theta_k), \theta_i \in M_i \cup \{\star, \perp\}$
- $\theta_\perp = (\perp, ..., \perp)$ most specific
- $\theta_\star = (\star, ..., \star)$ most general
- $supp(h_{\theta_\perp}) = \emptyset, supp(h_{\theta_\star}) = \mathcal{M}$
- $h_{\theta_\perp} = h_{(\theta_1,...,\perp,...,\theta_k)=...}$

induced hypothesis $h_\theta(m_1, ..., m_k) = 1$ if $\forall i : \theta_i \in \{m_i, \star\}$ else 0

$h \preceq h'$ if $supp(h) \subseteq supp(h')$. h is more specfic (less general) than h'

Find-S Algorithm finds most specific conjunctive clause that fits D

1. Start with $\theta_\perp = (\perp, ..., \perp)$
2. iterate over POSITIVE examples
3. min-generalize θ to fit example
4. $\perp \to a, a \to \star$

- maximal general hypothsis:
  1. start at $\theta_\star = (\star, ..., \star)$
  2. exclude every negative example
  3. $(\star, ...) \to \{(b, ...), (c, ...)...\}$
- If $V_A(D) \neq \emptyset$, Find-S finds $h \in V_A(D)$

disjunctive normal form $\Theta = \{\theta_1, ..., \theta_m\}$

'finite set of conjunctive clauses'

induced hypothesis $h_\Theta(\bar{m}) = 1$ if $\exists \theta \in \Theta : h_\theta(\bar{m}) = 1$ else 0
- $supp(\Theta) = \bigcup_{\theta \in \Theta} supp(h_\theta)$
- can represent all boolean functions

Boundary sets of version space

maximally general hypotheses $V_A^\top(D) = \{h \in V_A(D) \mid \nexists h' \in V_A(D) : h \prec h'\}$

maximally specific hypotheses $V_A^\perp(D) = \{h \in V_A(D) \mid \nexists h' \in V_A(D) : h' \preceq h\}$

-$h \in V_A^\top$ maximal, weil: $\forall x \in M \setminus supp(h) : supp(h) \cup \{x\} \notin supp(V_A(D))$

Theorem: $V_A(D) = \{h \in H_A | \exists h_\top \in V_A^\top(D), \exists h_\perp \in V_A^\perp(D) : h_\perp \preceq h \preceq h_\top\}$

-> $V_A(D)$ det. by $V_A^\top(D)$ and $V_A^\perp(D)$

- only 1 lower bound (in $V_A^\perp(D)$), potentially multiple upper bounds (in $V_A^\top(D)$)

Candidate Elimination Algorithm

Output: DNF for $V_A^\top(D)$ and $V_A^\perp(D)$

1. $S_\perp = \{\theta_\perp\}, S_\top = \{\theta_\star\}$
2. for $1 \leq i \leq n : y_i = 1$ (pos. xmpls)
   1. keep only fitting h from $S_\top$
   2. $\forall \theta \in S_\perp : h_\theta(x_i) = 0$
      - remove θ, add all min generalizations θ' of θ that fit x_i to $S_\perp$
   3. keep only most specific h in $S_\perp$
3. for $1 \leq i \leq n : y_i = 0$ (neg. xmpls)
   1. keep only fitting h from $S_\perp$
   2. $\forall \theta \in S_\top : h_\theta(x_i) = 1$
      - remove θ, add all min specializations θ' of θ that fit x_i to $S_\top$, for which a more specific $\theta_\perp \in S_\perp$ exists!
   3. keep only most general h in $S_\top$

- $V_A^\top = \{h_\theta | \theta \in S_\top\}, V_A^\perp = \{h_\theta | \theta \in S_\perp\}$
- Concept indentified if: $S_\perp = S_\top$ and $|S_\top| = 1$. $V_A(D) = \emptyset$ if $S_\perp = \emptyset \vee S_\top = \emptyset$

2.2 Decision Trees

Splitting $\Pi = \{M - 1, ..., M_p\}$ is finite partition of (sub)feature Space $\mathcal{M}'$
- induces splitting of $\{1, ..., n\}$ into $I_{D'}(M_1), ..., I_{D'}(M_p)$ (sets of indices)
- monothetic splits: based on 1 feature
- simple split: monothetic, into all realizations $M = \{\bar{m} \in M | m_1 = a(, b...)\}$
- binary split: monothetic, into 2 sets $M = \{\bar{m} \in M | m_1 \in A\} \cup \{\bar{m} \in M | m_1 \notin A\}$
- induced hypothesis $h_T(\bar{m}) = T(v)$, where v is unique leaf s.t. $\bar{m} \in M_v$
- simple decision trees can represent all hypotheses

Decision Tree Quality Measures

- Number of leaves
- Height (max number of constraints to check)
- External path length (sum of all path lengths from root to leaf)
- Weighted external path length (sum of all path lengths from root to leaf, weighted by number of examples classified in that leaf)

Theorem: Given D and bound b, its NP hard to decide existance of decision tree T s.t. h_T fits D and T has ext. p.l. $\leq b$

- Majority Class $Maj_D(M')$ maj. $r \in R$

- Number of Misclassifications: $Err_D(M', r)$ in feature subspace M' with majority class r
- $Err_D(T)$: sum up all $Err_D(M_v, T(v))$

Pure Node v if $Err_D(M_v, T(v)) = 0$

- class distribution $p_D^{M'}(r)$: $p(r)$ in M'
- **Impurity Function** $\iota : [0,1]^R \to \mathbb{R}$ if
  - $\iota(p)$ is minimal $\forall p : p(r) = 1$
  - ι symmetric in classes
  - ι is maximal for uniform distr.

gets propability distribution as input

1. $\bar{\iota}(p) = 1 - \max_{r \in R} p(r)$

2Entropy $H(p) = -\sum_{r \in R} p(r) \log_2 p(r)$

3.Gini Impurity $G(p) = 1 - \sum_{r \in R} p(r)^2$

- Impurity of M' is $\iota_D(M') = \iota(p_D^{M'})$

Impurity Reduction of splitting $\Pi = \{M_1', ..., M_p'\}$ of M' is:

$\iota_D(\Pi) = \iota_D(M') - \sum_{i=1}^p \frac{|I_D(M_i')|}{|I_D(M')|} \iota_D(M_i')$

Tree Construction for $M' \subseteq M$

1. if no elements in M': new leave v: $T(v) = Maj_D(M)$
2. if $\iota(M') \leq \epsilon$: new leaf v: $T(v) = Maj_D(M')$
3. else: select split Π of M' with maximal impurity reduction

- strict imp. fct.: concave at every point
- $\iota_D(\Pi) \geq 0 \; \forall \; \Pi$ and strict imp. fct. ι

ID3 simple D.T., monothetic simple splits, impurity function: entropy.

inductive bias: local optimization (greedy)

CART D.T., binary splits, impurity function: Gini impurity

- true loss of $h \in H_A$: misclassifications: $l^*(h) = \sum_{\bar{m} \in M} (1 - \delta_{h(\bar{m}), t(\bar{m})})$
- h **overfits** D if $\exists h' \in H_A$: $l(h, D) < l(h', D)$ and $l^*(h) > l^*(h')$

when: training data: noisy, small, biased

- Training Data: optimize loss here
- Validation: optimize hyperparameters
- Test Data: final estimation (true loss)
- h **overfits** (D, D_V) if $\exists h' \in H_A$: $l(h, D) < l(h', D) \wedge l(h', D_V) < l(h, D_V)$

true loss of h estimated by $l(h, D_V)$

Countermeasures to overfitting:

- increase data quality/quantity
- early stopping (no more splits)
  thrhld large -> omits useful splits
  thrhld small -> Large Tree
- regularization (penalize model complexity in training process)

Pruning: turn inner node v into leave with label $Maj(M_v)$

D.T. pruning Algorithm

1. given fully trained D.T.
2. prune every inner node als long as pruning doesnt increase validation loss: $l(h_T', D_V) \leq l(h_T, D_V)$

loss functions (and derivatives)

- $l(h, D) = \sum_{i=1}^{n}(1 - \delta_{y_i, h(x_i)})$
- $\delta_{ij} = 1$ if $i = j, 0$ otherwise.
- asd