AAI复习

Lecture 2 Basic Search

Basic Search Methods: Performance

b – maximum # successors of any node in search tree.
d – depth of the least-cost solution.
m – maximum length of any path in the state space.

PF Metric	Breadth-first Search	Uniform-cost Search	Depth-first Search	Depth-limited Search	Iterative Deepening	Di-directional Search
Complete?	Yes*, if b is finite.	Yes*, if step costs$\geq \epsilon$.	No, infinite loops can occur.	No. (Eps. $l < d$)	Yes	Yes* if BFS is used for both search.
Optimal?	Yes*, if costs on the edge are non-negative.	Yes	No,	No	Yes*, if costs on the edge are non-negative.	Yes* if BFS is used & paths have a uniform cost.
Time?	$O(b^d)$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(b^m)$	$O(b^l)$	$O(b^d)$	$O(b^{d/2})$
Space?	$O(b^d)$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(bm)$	$O(bl)$	$O(bd)$	$O(b^{d/2})$

Lecture 3 Heuristic Search

Q: Waht is the difference between an evaluation function and a heuristic function?

A: Heuristic is a component of an evaluation function. / Evluation function consists of heuristic(s).

A* Search

Expand the node s that has the minmal $f(s) = h(s) + g(s)$

- $g(s)$: cost from Start to s.
- $h(s)$: estimated cost from s to Goal.

  > sometimes using the straight-line distance or mahattan distance

- $f(s)$: estimated total cost of path from Start through s to Goal.

A*: Pseudo-code

```
function A-STAR-SEARCH(initialState, goalTest)
    returns SUCCESS or FAILURE :  /* Cost f(n) = g(n) + h(n)  */

    frontier = Heap.new(initialState)
    explored = Set.new()

    while not frontier.isEmpty():
        state = frontier.deleteMin()
        explored.add(state)

        if goalTest(state):
            return SUCCESS(state)

        for neighbor in state.neighbors():
            if neighbor not in frontier ∪ explored:
                frontier.insert(neighbor)
            else if neighbor in frontier:
                frontier.decreaseKey(neighbor)

    return FAILURE
```

Summary: Search Methods with $f(s)$

- Uniform-cost search: $f(s) = g(s)$
- Greedy best-first search: $f(s) = h(s)$
- A* search: $f(s) = g(s) + h(s)$

> g(s): the path cost from Start to noed s.
>
> h(s): the estimated cost from nodes to Goal.

Generate admissible heuristics:

- from relaxed problems
- from sub-problem
- from experience

Exercise

Heuristic Path Algorithm is a type of Best First Search and its evaluation function is $f(n) = (2 - w)g(n) + w^*h(n)$

1. Suppose h is admissable. How to set w to make it optimal?

$$
\begin{gathered}
(2-w)g(n) + w \cdot h(n) = g(n) + h(n) \\
2g(n) - wg(n) + w \cdot h(n) = g(n) + h(n) \\
2g(n) - wg(n) = g(n) \quad \text{and} \quad w \cdot h(n) = h(n) \\
2 - w = 1 \quad \text{and} \quad w = 1
\end{gathered}
$$

   We get making it optimal when w=1.

2. When w=0, w=1, and w=2, which search algorithm does it represent? Explain your answer.

   当 $w = 0$ 时:

   - 评估函数 $f(n) = (2 - 0)g(n) + 0 \cdot h(n) = 2g(n)$。
   - 这代表了Dijkstra算法，因为Dijkstra算法只考虑从起点到当前节点的实际代价 $g(n)$，而不考虑启发式估计 $h(n)$。此时，$f(n)$ 只与 $g(n)$ 有关，且系数为2（在Dijkstra算法中，系数不影响搜索策略的本质）。

   当 $w = 1$ 时:

   - 评估函数 $f(n) = (2 - 1)g(n) + 1 \cdot h(n) = g(n) + h(n)$。
   - 这代表了A*算法，因为A*算法的评估函数就是 $f(n) = g(n) + h(n)$。

当 $w = 2$ 时：

- 评估函数 $f(n) = (2-2)g(n) + 2 \cdot h(n) = 2h(n)$。
  - 这代表了贪心最佳优先搜索（Greedy Best - First Search）算法，因为贪心最佳优先搜索只考虑启发式估计 $h(n)$，而不考虑从起点到当前节点的实际代价 $g(n)$。此时，$f(n)$ 只与 $h(n)$ 有关，且系数为2（在贪心最佳优先搜索中，系数不影响搜索策略的本质）。

Lecture 4 MetaHuristic

What is Evaluation Algorithm?

It is the study of computational systems which use ideas and get inspirations from natural evolution.

Generate-and-Test: Description of steps

1. Generate the initial solution at random and denote it as the current solution.
2. Generate the **next solution** from the current one by **perturbation.**

   > Perturbation can be `Crossover`, `Mutation`

3. Test whether the newly generated solution (next solution) is acceptable;
   - Accepted it as the current solution if yes;
   - Keep the current solution unchanged otherwise.
4. Go to Step 2 if the current solution is unsatisfactory, stop otherwise.

A Simple Evolutionary Algorithm (EA)

1 Generate the initial population $P(0)$ at random
2 $i \leftarrow 0$ *// Generation counter*
3 **WHILE** halting criteria are not satisfied
3.1 Evaluate the fitness of each individual in $P(i)$
3.2 Select parents form $P(i)$ based on their fitness in $P(i)$
3.3 Generate offspring from the parents using crossover and mutation to form $P(i+1)$
3.4 $i \leftarrow i + 1$

Simple consturct

1. Initialize
2. Evaluation
3. Repeat until get done(find the optimal solution)

Lab

Modified Order Crossover(MOX)

Current tour	Canonic tour	Ordinal representation
1 2 5 6 4 3 8 7	1 2 3 4 5 6 7 8	1
1 2 5 6 4 3 8 7	2 3 4 5 6 7 8	1 1
1 2 5 6 4 3 8 7	3 4 5 6 7 8	1 1 3
1 2 5 6 4 3 8 7	3 4 6 7 8	1 1 3 3
1 2 5 6 4 3 8 7	3 4 7 8	1 1 3 3 2
1 2 5 6 4 3 8 7	3 7 8	1 1 3 3 2 1
1 2 5 6 4 3 8 7	7 8	1 1 3 3 2 1 2
1 2 5 6 4 3 8 7	7	1 1 3 3 2 1 2 1

Partially-mapped crossover(PMX)

- Crossover operators preserving the absolute position
  - Partially-mapped crossover (PMX) *(Goldberg and Lingle [1])*

    In the following figure, the offspring is created by first replacing the substring 236 in parent 2 by the substring 564.
    - Hence, city 5 replaces city 2, city 6 replaces city 3, and city 4 replaces city 6 (step 1).
    - Since cities 4 and 5 are now duplicated in the offspring, the inverse replacement is applied outside of the cut points. Namely, city 2 replaces city 5, and city 3 replaces city 4 (step 2).

parent 1	:	**1**	**2**	| **5**	**6**	**4** |	**3**	**8**	**7**
parent 2	:	1	4	| 2	3	6 |	5	7	8
offspring									
(step 1)	:	1	4	**5**	**6**	**4**	5	7	8
(step 2)	:	1	3	**5**	**6**	**4**	2	7	8

Exercise

Consider a TSP problem illustrated in the following figure.

- Exercise 1 Design an appropriate crossover operator and justify your design.

  > PMX / MOX
  >
  > All chromosomes carry exactly the same values and differ only in the ordering of these values.

- Exercise 2 Design an appropriate mutation operator and justify your design.

  > swap the cities at positions i and j.
  >
  > All chromosomes carry exactly the same values and differ only in the ordering of these values.

Two random cities in the path are selected and the positions of these two cities are swapped. **The crossover operation is to generate better offspring by preserving good gene characteristics, which is essentially to find a better solution among the existing local high-quality solutions.**

Mutation is to change a small part of individual genes, which can jump out of the original local search space and guide the algorithm to explore and find new solution space.

> The crossover can preserve good genes, which can find a better solution among the existing solutions
>
> The mutation can change a small part of th individual genes, which can guide algorithm to explore new solution space.

Lecture5 MetaheuristicsII

浮躁了，没看下去

Lecture6 Supervised Learning(I)

Machine Learning

- Supervised learning: Training data include both inputs and outputs
  - Classification, regression

  Given: training data of inputs X_l and corresponding outputs y_l.

  Goal: predict a '**correct**' output for a new input.
- Unsupervised learning: Training data do not include outputs.
  - Clustering

  Given: only unlabeled data of inputs X_u

  Goal: learn some structure of X_ur relationship among $X_u's$.
- Semi-supervised learning: Some training data are with output labels and some without.

  Given: A small portion of ($\mathcal{X}l$, $\mathcal{Y}l$) and large portion of $\mathcal{X}u$.

  Goal: prediction (classification).
- Reinforcement learning:

  Given: Training data do not include output labels, but do have a scalar feedback.

  Goal: learn a sequence of actions that maximize some cumulative rewards.

Supervised Learning

- Using past experiences to improve future performance on some task.
- Experience: the training examples or training data.
- What does it mean to improve performance? Learning is guided by an objective, e.g. a loss function to be minimized.

Generalization: Prediction Ability

Generalization (泛化): the ability to produce reasonable outputs for inputs not encountered during the training process.

Cross Validation

- The hold out method
- K-fold cross validation
- Leave-one-out(LOO) cross validation

Hypothesis Space $\mathcal{H}$ for Curve Fitting

Unferfitting: High training error and high test error.

> Use a more complex $\mathcal{H}$.

Overfitting: Low training error but high test error.

> Use a less comples $\mathcal{H}$

> Regularization (正则化): penalize certain parts of the parameter space or introduce additional constraints to constrain the hypothesis space.
>
> Get more training data

Computational Learning Theory

Bias-variance Trade-off

- The loss function consists of three terms
  - Irreducible error: σ^2
    - Bound on the algorithm performance
    - Uncertainty in the data
  - Squared bias: $\text{bias}^2 \hat{f}(x)$
    - Error due to simplification in the model
    - Difference between $f(x)$ and $E\hat{f}(x)$
    - Performance on the training data
  - Variance: $\text{var}\ \hat{f}(x)$
    - How much the estimate "jumps" around its mean
    - How well the method generalizes on different testing data

Complexity Choice

- Optimal model complexity
- Underfitting: to the left
- Overfitting: to the right

- Trade-off!

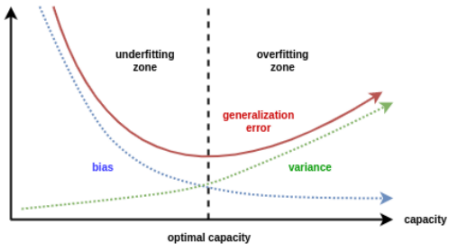


Image source: https://djsaunde.wordpress.com/2017/07/17/the-bias-variance-tradeoff/

- Bad performance on the training set (high bias)

  More complex model, different model, change hyperparameters, normalize inputs, train longer, change starting points, more complex optimization procedure, ...

  > 偏差是指模型预测值与真实值之间的差异，高偏差意味着模型对训练数据的拟合程度低，即模型过于简单，无法捕捉到数据中的规律。

- Good performance on the training set, bad performance on the validation set (high variance)

  Simpler model, more data in the training set, regularization, feature selection, ...

  > 方差是指模型对不同训练数据集的敏感程度，高方差意味着模型对训练数据过度拟合，模型过于复杂，导致在新数据（如验证集）上的表现很差。

Gradient Descent

$$Err = \underbrace{\frac{1}{N}\sum_{\mu}(\overbrace{y^{\mu}}^{\text{Ground truth}} - \underbrace{(W_1 X^{\mu} + W_0)}_{\text{Predicted values}})^2}_{\text{MSE}}$$

$$\frac{\partial}{\partial W_0} = -\frac{2}{N}\sum_{\mu}(y^{\mu} - (W_1 X^{\mu} + W_0))$$

$$\frac{\partial}{\partial W_1} = -\frac{2}{N}\sum_{\mu} X \times (y^{\mu} - (W_1 \times X^{\mu} + W_0))$$

$$W_0 = W_0 - \eta(\frac{\partial}{\partial W_0})$$

$$W_1 = W_1 - \eta(\frac{\partial}{\partial W_1})$$

Lecture7 Supervise Learning(II)

Linear Model

Closed-form Solution

1. Closed-form Solution (闭式解)

- Optimization: $\mathcal{L}(w_0, w_1) = \frac{1}{2}\sum_{n=1}^{N}\left[y^{(n)} - \left(w_1 x^{(n)} + w_0\right)\right]^2$.
- First-order equations:

$$\frac{\partial}{\partial w_0}\mathcal{L}(w_0, w_1) = 0, \ \frac{\partial}{\partial w_1}\mathcal{L}(w_0, w_1) = 0.$$

- Solution:

$$w_1 = \frac{N\left(\sum_n x^{(n)} y^{(n)}\right) - \left(\sum_n x^{(n)}\right)\left(\sum_n y^{(n)}\right)}{N\left(\sum_n (x^{(n)})^2\right) - \left(\sum_n x^{(n)}\right)^2}$$

$$w_0 = \frac{\sum_n y^{(n)} - w_1 \sum_n x^{(n)}}{N}.$$

$$W = (X^T X)^{-1} X^T y$$

Iterative Solution: Advanced

- Batch GD: update $\boldsymbol{w}$ once with all training samples.

  > 在每次更新模型参数时，会使用整个训练数据集来计算梯度。也就是说，对于一个包含m个训练样本的数据集，在计算梯度时，会对这m个样本的损失函数求和，然后计算梯度。

  Guarantee global optimum but slow.

- Stochastic GD: update $\boldsymbol{w}$ N times with one training data for one update.

  > 它在每次更新模型参数时，仅使用一个随机选择的训练样本的梯度来更新参数。
  >
  > 其梯度估计的方差较大，可能需要更多的迭代次数才能收敛。

  Fast but do not guarantee global optimum with a fixed α.

  Online/offline settings

- Mini-batch SGD: update $\boldsymbol{w}$ several times with a subset of $\mathcal{D}$ for one update.

  > Mini-batch Gradient Descent 是 Batch GD 和 Stochastic GD 的一种折衷。它在每次更新参数时，使用一小部分训练样本（称为一个 mini-batch）来计算梯度。

- Optimization:

$$min_w \mathcal{L}(w) = \frac{1}{2}\sum_{n=1}^{N}[y^{(n)} - w^T x^{(N)}]^2$$

- Gradient descent (GD):

$$w_i \leftarrow w_i + \alpha \sum_n (y_i^{(n)} - w^T x^{(n)}) x_i^{(n)}$$

  α: learning rate, positive

Regularized Objective for MLR

- Regularized Objective:

$$min_w \mathcal{L}_{tr}(w) + \lambda \Omega(w)$$

$\Omega(w)$: regularization

Multivariate Linear Classification(MLC)多元线性分类

- Logistic regression model: $h_{\boldsymbol{w}}(\boldsymbol{x}) = \sigma(\boldsymbol{w}^T \boldsymbol{x})$, where $\sigma(z) = \frac{1}{1+e^{-z}}$.

Soft threshold function: $\sigma(z) = s(z) = \frac{1}{1+e^{-z}}$

- s(z) : sigmoid function.
- Differentiable: $s'(z) = s(z)[1 - s(z)]$

Decision Tree

Tree model: a function mapping feature vector $\boldsymbol{x}$ to a decision y via a sequence of tests.

> 通过简单的垂直和水平分割

Greedy Divide-and-conquer Strategy

- Approach: Greedy divide-and-conquer strategy-heuristic search.
  - Start from empty tree.
  - Decide the **best feature** based on heuristics.
  - Divide the problem into smaller subproblems;
  - Repeat (2)~(3) until stopping criteria.

> Heuristics: Pick the feature that maximizes information gain (信息增益).

How to measure the goodness of a feature formally?

- Information gain

Preliminary: Entropy(熵)

- Entropy: $\mathcal{H}(Y) = -\sum_k p(y_k) \log_2 p(y_k)$
- Larger entropy, more uncertainty.
  - High entropy: Y ~ uniform or flat distribution → less predictable
  - Low entropy: Y ~ peak/valley distribution → more predictable

Preliminary: Conditional Entropy

• Conditional entropy:

$$\mathcal{H}(Y|X) = \sum_j p(X = x_j) H(Y|X = x_j)$$

- Conditional entropy:

$$\mathcal{H}(Y|X) \triangleq \sum_j p(X = x_j) \cdot \mathcal{H}(Y|X = x_j).$$

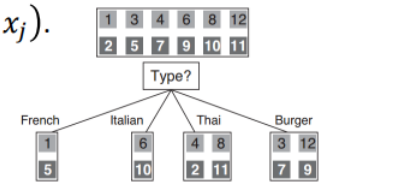


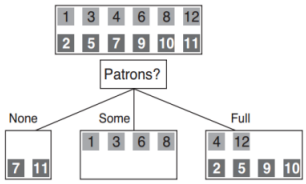
Image source: Figure 18.13 of the AI book by S. Russell & P. Novig.

- Example 2: $Y \sim label \ \& \ X \sim Type$
  - $p(X = French) = p(X = Italian) = \frac{2}{12}$;
  - $p(X = Thai) = p(X = Burger) = \frac{4}{12}$;
  - $\mathcal{H}(Y|X = French\ or\ Italian)$: $\frac{1}{2}\log\left(\frac{1}{2}\right) - \frac{1}{2}\log\left(\frac{1}{2}\right) = -\log(\frac{1}{2})$;
  - $\mathcal{H}(Y|X = Thai\ or\ Burger)$: $\frac{2}{4}\log\left(\frac{2}{4}\right) - \frac{2}{4}\log\left(\frac{2}{4}\right) = -\log(\frac{1}{2})$;
  - $\mathcal{H}(Y|X) = -\left[\frac{2}{12} \cdot \log\left(\frac{1}{2}\right) \cdot 2 + \frac{4}{12} \cdot \log\left(\frac{1}{2}\right) \cdot 2\right] = -\log\left(\frac{1}{2}\right) = 1.$

- Conditional entropy:

$$\mathcal{H}(Y|X) \triangleq \sum_j p(X = x_j) \cdot \mathcal{H}(Y|X = x_j).$$

- Example 3: $Y \sim label \ \& \ X \sim Patrons$
  - $p(X = None) = \frac{2}{12}$; $p(X = Some) = \frac{4}{12}$; $p(X = Full) = \frac{6}{12}$;
  - $\mathcal{H}(Y|X = None)$: $\frac{2}{2}\log\left(\frac{2}{2}\right) = 0$;
  - $\mathcal{H}(Y|X = Some)$: $\frac{4}{4}\log\left(\frac{4}{4}\right) = 0$;
  - $\mathcal{H}(Y|X = Full)$: $\frac{2}{6}\log\left(\frac{2}{6}\right) + \frac{4}{6}\log\left(\frac{4}{6}\right) = -0.9183$;
  - $\mathcal{H}(Y|X) = -\left[\frac{2}{12} \cdot 0 + \frac{4}{12} \cdot 0 + \frac{6}{12} \cdot (-0.9183)\right] = 0.4591.$



Information Gain（信息增益）

Information gain: Decrease in entropy after splitting

$$IG(X) = \mathcal{H}(Y) - \mathcal{H}(Y|X)$$

- X: input feature,
- Y: classification label.

Example 4: $Y \sim label \ \& \ X \sim type/patrons$
- $IG(Type) = \mathcal{H}(Y) - \mathcal{H}(label|Type) = 1 - 1 = 0.$
- $IG(Patrons) = \mathcal{H}(Y) - \mathcal{H}(label|Patrons) = 1 - 0.4591 = 0.541.$

$IG\ Patrons > IG\ Type \Rightarrow$ Patrons is better.

- When the x is **continuous**
  - [Question] How to decide the possible $\{t\}$ for Est?
  - [Concern] Search through $\mathbb{R}^1 \Rightarrow$ Too hard!
  - [Answer] Only a finite number of values are useful:
    - Sort values of Est into $\{x_1, \cdots, x_m\}$ with non-duplicated values;
    - Consider candidates $\left\{t_i = x_i + \frac{x_{i+1} - x_i}{2} \middle| \ i = 1, \cdots, m-1\right\}$.

Best t^* for Est and its Information Gain

- Take the best t from $\{t\}$: Denote $X \sim Est$,
  - (1) Define $\mathcal{H}(Y|X:t) = p(X < t) \cdot \mathcal{H}(Y|X < t) + p(X \geq t) \cdot \mathcal{H}(Y|X \geq t)$;
  - (2) Compute $IG(Y|X:t_i) = \mathcal{H}(Y) - \mathcal{H}(Y|X:t_i)\ \forall t_i$;
  - (3) Choose $t^* = \arg\max_{t_i} IG(Y|X:t_i)$

- Use: $IG^*(Est) = IG(Y|X:t^*) = \max_{t_i} IG(Y|X:t_i)$.

Tree Overfitting

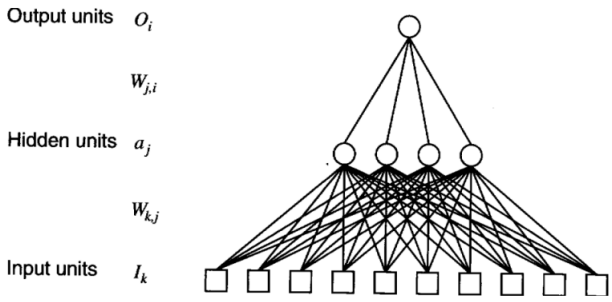
- More #feature, more likely overfitting; more #(train data), less likely overfitting.
- Decision tree pruning:
  - Build a fully grown tree.
  - Choose a node that has only leaf nodes as children.
  - Testing the feature 'relevance' for this node:
    - relevant→reserve this node.
    - irrelevant: replace it based on its leaf nodes.

Decision Tree for Regression

- Leaf node: Linear regression model on the examples in each leaf node.

Neural Network

Multilayer Neural Network



> $w_{i,j} \rightarrow w_{k,j}$， 中间还要包括一个非线性层，Sigmod，Relu

K-Nearest Neighbor

k-Nearest neighbor method:

- For classification: find k nearest neighbors of the testing point and take a vote.

  > vote: 根据这 k 个最近邻中各个类别的数量多少来进行投票决策。

- For regression: take mean or median of the k nearest neighbors, or do a local regression on them.

Distance metric: $\ell_p(x_j, x_q) = \left(\sum_i |s_{j,i} - x_{q,i}|^p\right)^{\frac{1}{p}}$

- Advantage:
  - Training is very fast.
  - Learn complex target functions.

- Do not lose information.
- Disadvantage:
  - Slow at query time.
  - Easily fooled by irrelevant attributes.

Exercises

1. Could you classify iris with multi-class linear regression classifier?

   选取iris的特征，构造`MLR`model，Loss Function选取MSE，迭代训练

2. Could you classify iris with multi-class SVM classifier?

   > 优化目标是最小化$\frac{1}{2}\|w\|^2$（w是超平面的法向量），同时满足约束条件$y_i(w^T x_i + b) \geqslant 1$，其中$x_i$是数据点，$y_i$是数据点的类别标签（$\pm 1$），$b$是偏置项。

Lecture8 Ensemble Learning

What is an Ensemble?

An ensemble indicates a collection of individual learning machines.

> 它结合多个学习器来解决一个问题，通常能获得比单个学习器更好的性能

Given K base learners, whose outputs are oj , $j = 1, 2, \dots, K$, a simple ensemble output could be

$$O = \sum_{j=1}^{K} w_j o_j$$

- **[Question 1] Is the ensemble much less likely to misclassify x than using a single learner?**
- **[Possible Answer]**
  - **Suppose each $f \in \mathcal{F}$ misclassifies a randomly chosen example with probability ε (called error) and the errors of these K learners are independent of each other.**
  - **Consider the case that there are 2 classes, then the error that the ensemble misclassifies an example is:**

$$\varepsilon_{ens} = \sum_{i=floor\left(\frac{K}{2}\right)+1}^{K} \binom{K}{i} \varepsilon^i (1-\varepsilon)^{K-i}$$

When is an Ensemble Better?

- The errors of base learners should be independent of each other.
- The base learners should do better than random guessing (i.e., with $\varepsilon < 0.5$).

BUT, True independence of errors from different base learners is hard to achieve because of, e.g.

- [Possible Solutions], deal with the problem of independent base learners.
  - Use different learners to reduce the positive correlation between their errors.
  - Different supervised learning methods.
  - Different parameters and weights.
  - Different base learners.

- Instead of pursuing mutual independence of errors of base learners, we can go one step further and encourage negative correlation of errors of base learners.

Negative Correlation Learning

Instead of creating an ensemble of unbiased individual networks whose errors are uncorrelated, NCL can produce individual networks whose errors are negatively correlated.

> 通过引入负相关机制来改进模型的性能

Other Methods for Constructing an Ensemeble Classifier

Bagging

Bootstrap Aggregating，常简称为 Bagging，是一种集成学习（Ensemble Learning）方法。它的主要思想是通过对训练数据集进行有放回的抽样（这种抽样方法被称为 Bootstrap 抽样），生成多个不同的训练子集，然后使用这些子集分别训练多个基学习器（Base learner），最后将这些基学习器的预测结果进行综合（通常是简单平均或多数投票）来得到最终的预测结果。

- Improves the generalization error by reducing the variance of the base classifiers.

> 通过减少基分类器的方差，可以提高模型的稳定性和泛化能力

Boosting

将多个弱分类器组合成强分类器，通过迭代自适应改变训练样本分布，为每个训练样本分配权重，每次迭代增加错误分类样本权重、降低正确分类样本权重并归一化，最后聚合训练好的基分类器作为最终集成模型。

- Two core components:
  - Weights:
    - Used as a sampling distribution when creating bootstraps.
    - Used by the base classifier to learn a model which is biased towards the examples that are hard to classify (ones with higher weights).
- Final ensemble = weighted-majority/weighted-average combination

AdaBoost

输入带标签样本集，初始化样本权重为 $\frac{1}{N}$，每次迭代根据权重采样生成自助样本，训练基模型，统计错误分类样本并更新权重（错误分类样本权重乘以），最后归一化权重，计算基模型权重，通过加权多数投票组合基模型。

Random Forest

由多个决策树组成，每个树基于独立随机向量生成，随机向量从固定概率分布生成（与 AdaBoost 的自适应方法不同），Bagging 是决策树的一个特殊情况。

> Many decision tree, More stable, better generalization

- Bagging using DTs is a special case of random forest.

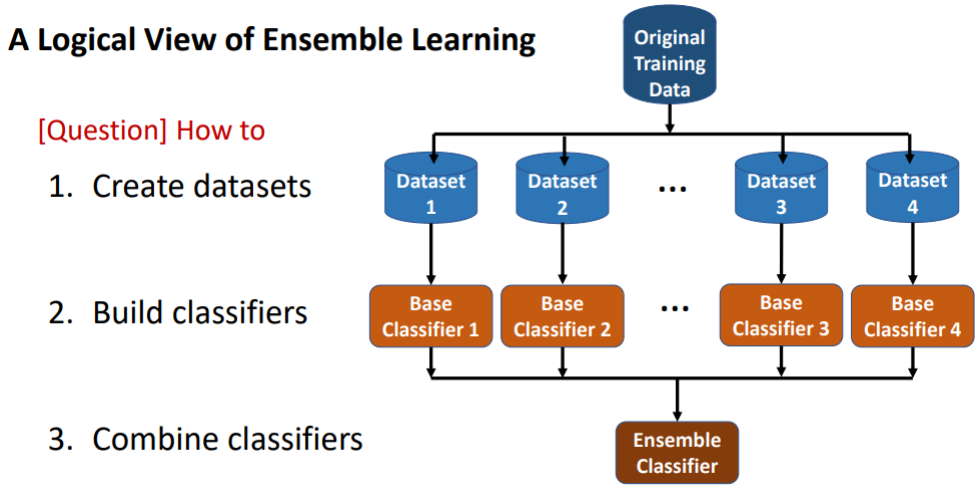
- [Question] What can you observe from the following bound?

$$Generalization\_Error \leq \frac{\bar{\rho}(1-s^2)}{s^2}$$

  - $\bar{\rho}$ denotes the average correlation among the DTs,
  - s is a quantity that measures the strength of the DT classifiers.
- [Observation]
  - **If the DTs are more correlated, the upper bound of the generalization error increases.**
    - ***Randomisation helps to reduce the correlation!***
  - **If the strength of the ensemble decreases, the upper bound of the generalization error increases.**

Current Status

- 回归问题中，在一定假设下集成性能不劣于单个学习器；分类问题中，虽无严格证明，但有大量经验证据表明集成优于单个学习器。
- 分类问题中，虽无严格证明，但有大量经验证据表明集成优于单个学习器。



Simple Ensemble Approaches

- Train different models on the same dataset, then vote (classification) or average (regression) of predictions of multiple trained models

> 不同的model，相同的Dataset

- Train the same model multiple times on different data sets generated from the original data set.

> 相同的model，不同的dataset

- Train different models on multiple datasets.

> model 和 datasets都不一样

  - **Dataset level:**
    - **By manipulating the training set:**
      - **Create multiples data sets by sampling the original data.**
      - **Methods differ in the sampling distribution.**
      - **Examples: *Boosting*, *Bagging*.**
    - **By manipulating the input features:**
      - **Methods differ in the feature selection.**
      - **Example: *Random Forest*.**
    - **By manipulating the class labels:**
      - **Methods differ in the class partition.**
      - **Example: *error-correcting output coding*.**
  - **Classifier level:**
    - **By manipulating the learning algorithms: depends on the classifier used.**

- Majority voting
- Weighted voting
- Averaging
- Weighted averaging

	Bagging	Boosting	Random Forest
Sampling distribution	Uniform random distribution, with replacement	Adaptive weighted distribution	Non-adaptive weighted distribution
#Feature	Same as original data set	Same as original data set	Fewer/More
Parallel?	Yes	No	No
Suitable problems /classifiers	Regression Classification	Classification Regression	Only applied to decision trees

Lab

stacking algorithm

Stacking 算法，即堆叠泛化算法（Stacked Generalization），是一种用于机器学习模型融合的技术，通过组合多个基模型的预测结果来创建一个更强大的元模型，以提升模型的预测性能。

Lecture9 Multi-Objective Optimization and Learning

Pareto Front

由所有非支配解（在目标空间中不被其他解帕雷托优于的解）构成的集合，决策空间中的对应集合为帕雷托最优集。

帕累托前沿代表了这样一组解的集合：在这个集合中的解，在不使其他目标变差的情况下，无法再进一步优化任何一个目标了，也就是所有目标之间达到了一种权衡最优的状态。

Main Goal

- **收敛性（Convergence）**：找到尽可能接近帕雷托最优前沿的一组解。
- **多样性（Diversity）**：找到尽可能多样的一组解。

MOEAs

拥挤距离（Crowding Distance）：表示同一等级中包围特定解的最近邻域解的**长方体周长的一半**，用于密度估计。

NSGA-II

- 算法步骤
  - 合并父代和子代种群，选取前沿填充父代种群。
  - 对种群按非支配排序并选择前个元素。
  - 使用选择、交叉和变异创建新种群。
- 优缺点
  - **优点**：通过拥挤过程保持非支配解的多样性，无需额外多样性控制；精英主义保护已找到的帕雷托最优解不被删除。
  - **缺点**：当第一个非支配集成员超过N个时，可能丢弃一些帕雷托最优解。

Multi-Objective Learning

Multi-task learning: in short, multi-task learning is defined as learning multiple objective functions loss at the same time.

Two objectives:

- **Accuracy**:

$$\min \text{err}_k = \frac{1}{N}\sum_{i=1}^{N}(f_k^i - o^i)^2$$

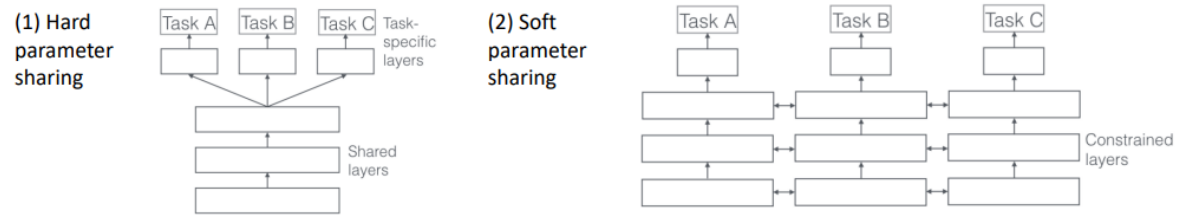
- **Diversity**:

$$\min \text{corr}_k = \sum_{i=1}^{N}(f_k^i - f^i)[\sum_{j\neq k, j=1}^{M}(f_j^i - f^i)]$$

- f^i is the ensemble's output for a training sample i;
- f_k^i is the k^{th} base learner's output for a training sample i;
- o^i is the desired output (true value) for a training sample i.

Basic framework of multi-task learning

- Hard parameter sharing: Most parameters are shared. For example, the bottom parameters are shared uniformly, and the top parameters are independent of each model.
- Soft parameter sharing: Each task has its own model and corresponding parameters. Compared with single task learning, different tasks restrict their model parameters through some way of information sharing.



> hard parametes共享是指多个任务之间共享全部或者部分模型参数。这种方式能够显著减少模型的参数总量，提高训练效率，降低过拟合风险。如果不同任务之间的差异较大，共享的参数可能无法很好地适应所有任务的需求，导致任务之间产生干扰，影响模型在某些任务上的表现。
>
> soft parameters共享并不直接共享模型的参数，而是通过某种机制使不同任务的参数之间相互影响，从而达到隐式共享特征的目的。软参数共享能够更好地适应不同任务之间的差异，每个任务可以根据自身的特点来调整参数，同时又能从其他任务中学习到一些有用的信息。由于每个任务都有自己独立的参数，模型的参数总量相对较大，训练时间和计算资源的消耗也会增加。

Lecture10 UnsupervisedLearning

Clustering is an unsupervised learning methods, because the labels are not given.

聚类是将相似对象或数据分组在一起的无监督学习方法，旨在发现数据中的隐含模式、属性和结构。其定义为给定 n 个对象的表示，基于相似性度量找到 k 个组，使同一组内对象相似度高，不同组间相似度低。

MInkowski distance(L_P-distance)

$$d(x_i, x_j) = (\sum_{l=1}^{N}|x_i^{(l)} - x_j^{(l)}|^p)^{\frac{1}{p}}$$

Chebyshev distance(L_∞-distance)

$$d(x_i, x_j) = \max_{1 \le l \le n} |x_i^{(l)} - x_j^{(l)}|$$

Hierarchical Algorithm

层次聚类算法以递归方式寻找嵌套的簇，要么采用凝聚模式（bottom-up, 从每个数据点自成一个簇开始，依次合并最相似的一对簇以形成簇层次结构）；要么采用分裂（top-down, 自上而下）模式（从所有数据点在一个簇开始，递归地将每个簇划分为更小的簇）。

Question

How do you represent a cluster of more than one point?

欧几里得空间用质心（centroid）表示，即数据点平均值；非欧几里得空间用聚类点（clustroid），即离其他点 “最近” 的点，其 “最近” 定义有多种，如最小最大距离、最小平均距离、最小距离平方和等。

How do you determine the “nearness/similarity” of clusters?

> Measure cluster distances by distances of centroids

可将聚类点视为质心计算簇间距离；或取两簇间任意两点距离最小值；或定义 “凝聚度” 概念，如最大距离，合并最凝聚的簇。凝聚度可通过簇直径（最大点间距离）、平均距离或密度相关方法衡量。

When to stop combining clusters?

- 基于稳定性

  观察合并过程中某些稳定性指标的变化，例如簇内方差、簇间距离的变化率等。如果在连续多次合并后，这些指标的变化很小或者不再有明显变化，认为聚类结构已经相对稳定，此时停止合并。

- 基于距离阈值

  设定一个距离阈值，当要合并的两个簇之间的距离（根据所采用的距离度量方式，如质心距离、最小距离、最大距离等）超过这个阈值时，停止合并

The Non-Euclidean Case

非欧几里得空间用聚类点（clustroid），即离其他点 “最近” 的点，其 “最近” 定义有多种，如:

- 最小最大距离,
- 最小平均距离,
- 最小距离平方和等。

实现与复杂度：朴素实现每次计算所有簇对距离再合并，复杂度为 O (N^3)；精心实现使用优先队列可降至 O (N^2 log (N))，但对于大数据集仍较昂贵。

Well-known Hierarchical Algorithms

- **单链聚类（Single-link Clustering）**：两簇相似度为最相似成员间相似度，关注簇最接近区域，具 “乐观” 性，步骤包括初始化、找最相似簇对、合并、更新矩阵、重复直至只剩一个簇。
- **全链聚类（Complete-link Clustering）**：两簇相似度为最不相似成员间相似度，考虑聚类整体结构，具 “悲观” 性，步骤与单链聚类类似，但更新矩阵时计算新簇与旧簇距离方式不同。

k-means and Other Cost Minimization Clustering

k - 均值聚类（k - means Clustering）

- **目标函数**：在欧几里得空间中，最小化所有点到其所属簇质心的平方距离之和，是 NP 难问题，贪心算法收敛于局部最优。
- **主要过程**：选择初始 k 个簇（如随机选一个点，再选 k-1 个最远点为质心，将各点分配到最近质心簇，更新质心，重复直至簇成员稳定）。
- **问题**：
  - 初始化影响结果；需选择合适距离度量；
  - 簇数量 k 选择缺乏理论支持，可通过观察平均距离随 k 变化来选择，距离下降快到合适 k 后变化小，k 过大平均距离改进小。
  - Average falls rapidly until right k, then changes little.

    > 聚类中心的均值下降很大然后下降很小的那个转折点就是最佳的k

Spectral Clustering

将数据点表示为加权图节点，边权重为点对相似度，目标是将节点划分为子集使割大小（连接不同子集节点边权重和）最小，但最小割算法常导致不平衡簇。

Lecture11 Feature Engineering

Feature Engineering

特征是描述问题且对预测或解决问题有用的信息。特征工程是确定哪些特征对训练模型有用，然后通过转换原始数据来创建这些特征的过程。

Why Normalize Numeric Features?

当同一特征值差异大或不同特征范围差异大时，归一化可避免训练问题，如梯度更新过大导致训练失败或梯度下降 “反弹” 影响收敛，可采用异质学习率解决。

Normalization Technique	Formula	When to Use
Linear Scaling	$x' = (x - x_{min})/(x_{max} - x_{min})$	When the feature is more-or-less uniformly distributed across a fixed range.
Clipping	if x > max, then x' = max. if x < min, then x' = min	When the feature contains some extreme outliers.
Log Scaling	x' = log(x)	When the feature conforms to the power law.
Z-score	x' = (x - μ) / σ	When the feature distribution does not contain extreme outliers.

分类数据转换

- One/Multi-hot encoding
- Hashing
- Embeddings: high-dimensional --> low-dimensional space

Three Typical Methods

- **包装器方法（Wrapper methods）**：模型相关，通过添加 / 删除特征导航特征子集，在验证集评估模型性能，重复直到精度无提高，优点是准确性高，缺点是计算昂贵且有过拟合风险。
- **过滤器方法（Filter methods）**：独立于学习模型，根据与 AI 任务相关性的启发式分数对特征排名并选择子集，优点是快速简单、泛化性好，缺点是未考虑特征间相互作用、准确性不如包装器方法，可作为包装器特征选择的预处理，如基于相关性的过滤器（假设好特征与输出高度相关且相互间不高度相关，通过相关性分数进行特征排名、去除冗余相关特征），相关性度量包括经典线性相

关（如 Pearson 相关，计算简单但不能捕获非线性相关且要求数值特征）和信息论方法（如信息增益，能捕获非线性相关但计算成本高且偏向值多的特征；对称不确定性，可补偿信息增益偏差并归一化值到 [0,1]）。

- **嵌入式方法（Embedded methods）**：特征选择是模型构建一部分，由学习过程引导特征搜索，利用算法返回模型结构获取相关特征集，优点是类似包装器方法但计算成本低且不易过拟合，如决策树（构建过程就是特征选择过程，树中未使用所有特征）。

> 嵌入到模型中，和模型一起训练

Regularization

正则化（Regularization）

- **基本思想**：模型中特征越多复杂度越大，正则化通过对复杂度引入惩罚来减少特征，使模型倾向于低复杂度（少特征），从贝叶斯观点看是给学习模型施加世界是简单的先验知识。

- Find $f \in \mathcal{F}$ minimizing

$$\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_{tr}\big(y_i, f(x_i)\big) + \lambda \cdot \Omega(f)$$

H
  - $\mathcal{F}$: a class of candidate functions
  - $\Omega(f)$: the complexity of a model f
  - $\lambda > 0$: a regularization parameter
- Question: How do we pick parameter λ?
- Answer: Cross validation.

- Ridge regression

$$J = \frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y_i)^2 + \lambda||w||_2^2$$

- Lasso regression

$$J = \frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y_i)^2 + \lambda||w||_1$$

> Lasso can be used for feature selection, but ridge regression cannot. In other words, Lasso is easier to make the weight become 0, while ridge is easier to make the weight close to 0.S

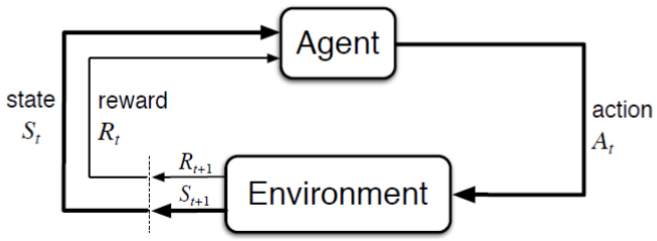
Summary

- Feature selection is a heuristic search problem.
- Use regularization on all possible features to prevent overfitting.

Lecture12 Markov Decision Process

强化学习是一种计算方法，智能体（agent）通过与环境交互，从奖励信号中学习如何将情境映射到行动，以最大化数值奖励。

- **An agent receives current state S_t and reward R_t from the environment, where t denotes the time step/tick.**



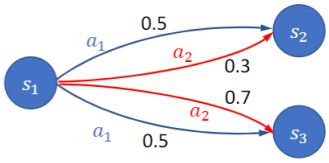
Markov property: Given the present, the future is independent of the history.

> 给定当前时刻，未来与历史状态无关

Transition Probability Function

- $\mathcal{P}: \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$, **transition probability function**
$$\mathcal{P}(s, a, s') = \mathbb{P}[S_{t+1} = s' | S_t = s, A_t = a]$$

1) Time-independent.



2) $\sum_{s' \in \mathcal{S}} \mathcal{P}(s, a, s') = 1.$

3) $\sum_{a \in \mathcal{A},\, s' \in \mathcal{S}} \mathcal{P}(s, a, s')$ **can be greater than 1.**

Core Elements of RL

Policy

A policy defines an agent's behaviour (mapping from state to action), i.e. how the agent acts in the given circumstance.

> 策略定义智能体行为，是从状态到行动的映射。

- A stochastic policy π is a conditional probability distribution over actions given states
$$\pi(\pi|s) = \mathbb{P}(A_t = a|S_t = s)$$
- Both are stationary (time-independent) and historyindependent.

Value

- A value function: a prediction of future reward which is used to evaluate state(s) so as to select hopefully optimal action(s).

> 包含了未来reward的期望值，用来评估state的好坏

Value Functions

- Define state value function $v_\pi(s) : S \to \mathbb{R}$ of policy π as
$$v_\pi(s) = \mathbb{E}[G_t|\pi, S_t = s]$$
- Define state value function $v_\pi(s) : S \times \mathcal{A} \to \mathbb{R}$ of policy π as
$$q_\pi(s, a) = \mathbb{E}[G_t|\pi, S_t = s, A_t = a]$$

For deterministic $\pi(s)$, since $\mathbb{P}(S_{t+1} = s' | S_t = s) = \mathcal{P}(s, \pi(s), s')$, it becomes

$$v_\pi(s) = \sum_{s' \in S} \mathcal{P}(s, \pi(s), s') \big(\mathcal{R}(s, \pi(s), s') + \gamma v_\pi(s')\big).$$

For stochastic $\pi(a|s)$, it becomes

$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \sum_{s' \in S} \mathcal{P}(s, a, s') (\mathcal{R}(s, a, s') + \gamma v_\pi(s'))$$

- **The equation for $\pi(s)$ is a special case of that for $\pi(a|s)$.**
- **However, using $\pi(s)$ is sufficient in many cases.**

Bellman equation

$$v_\pi = R_\pi + \gamma P_\pi v_\pi$$

$$q_\pi(s, a) = \sum_{s' \in S} \mathcal{P}(s, a, s') \big(\mathcal{R}(s, a, s') + \gamma v_\pi(s')\big).$$

Compare

$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \sum_{s' \in S} \mathcal{P}(s, a, s') \big(\mathcal{R}(s, a, s') + \gamma v_\pi(s')\big)$$

and

$$q_\pi(s, a) = \sum_{s' \in S} \mathcal{P}(s, a, s') \big(\mathcal{R}(s, a, s') + \gamma v_\pi(s')\big)$$

we have

$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s)\, q_\pi(s, a)$$

Lecture13 Reinforcement Learning

Model-based RL

Model-based reinforcement learning: maintain an estimated MDP M("model") and use it as the input of the planning algorithms.

> 维护一个估计的MDP，用它来作为算法的输入

- (Vanilla) model-based RL algorithm
  - (0) Start with an arbitrary policy π and an estimated MDP $\widehat{M}$
  - (1) Interact with the environment using π, record transitions and rewards
  - (2) Update estimated MDP $\widehat{M}$
  - (3) Compute the optimal policy $\hat{\pi}^*$ of $\widehat{M}$ using PI/VI, update $\pi \leftarrow \hat{\pi}^*$
  - (4) Goto (1) until π converges

Exploration vs Exploitation

- **探索（Exploration）**：故意采取当前知识下看似非最优的行动以获取更多信息发现更好策略。
- **利用（Exploitation）**：采取当前认为最优的行动。

ϵ-Greedy

以概率ϵ选择随机行动，以概率$1-\epsilon$选择“最优”行动，ϵ常取小值。优点是易实现且多数时候采取最优行动（初始策略足够好时不易采取灾难性行动），缺点是智能体行为永不收敛（持续探索），某些情况下可能指数级低效（如示例中智能体可能因探索概率低而难发现高奖励状态）。

Rmax

假设$q_\pi(s,a)=R_{max}$MDP 最大即时奖励），除非行动在状态至少被采取次，迫使智能体多次尝试所有可能行动再做结论。优点是若足够大，在无限长学习过程中大概率最优（样本复杂度理论），缺点是探索过于激进，在一些实际应用（如机器人、自动驾驶、电站控制等）可能不合适，状态 / 行动空间大时可能花费太多时间探索而无合理计划（多数值为R_{max}）。

Choose a strategy

- 选择策略应根据应用需求，如现实损失可忽略（游戏等）可采用系统探索，实验昂贵（机器人等）则采用更保守（贪婪）探索。

Model-free RL

Model free的强化学习方法不尝试对环境进行显式建模，而是直接学习最优策略或价值函数

- Temporal difference estimation (TD)

$$\hat{v}^\pi(S_t) \leftarrow \hat{v}^\pi(S_t) + \alpha\left(R_{t+1} + \gamma\hat{v}^\pi(S_{t+1}) - \hat{v}^\pi(S_t)\right)$$

- Compare to MC:

$$\hat{v}^\pi(S_t) \leftarrow \hat{v}^\pi(S_t) + \alpha(G_t - \hat{v}^\pi(S_t))$$

- $R_{t+1} + \gamma\hat{v}^\pi(S_{t+1})$ is sometimes called “TD target”
  $R_{t+1} + \gamma\hat{v}^\pi(S_{t+1}) - \hat{v}^\pi(S_t)$ is called “TD error”

> G(t)为折扣的累加奖励回报

MC vs TD

MC: unbiased, but usually has a higher variance

TD: biased, but usually has a lower variance

- On-policy: estimated values $\hat{q}^\pi$ is aboutπ
  - TD, Sarsa, MC
- Off-policy: estimated values $\hat{q}^\pi$is NOT about π, but about some other (possibly better) policy
  - Q-learning

- The use of “model-based” & “model-free” is sometimes confusing
  - Some researchers use “model” to refer to estimated MDPs, as in previous pages
    - “model-based” → plan with estimated MDPs
    - “model-free” → plan without estimated MDPs
    - both need interaction for collecting information
  - Some use “model” to refer to full prior (ground truth) knowledge of MDPs
    - “model-based” → plan with full knowledge of MDPs, does not need interaction
    - “model-free” → plan without full knowledge, need interaction