CZ4041 - Machine Learning

1 Bayesian Classifiers

Probabilities

Sum Rule

$P(A) = \sum_B P(A,B)$
$P(A) = \sum_B \sum_C P(A,B,C)$

Product Rule

$P(A,B) = P(B|A) \times P(A) = P(A|B) \times P(B)$

Bayes Theorem

$$P(A|B) = \frac{P(A,B)}{P(B)} = \frac{P(B|A)P(A))}{P(B)}$$

(Generalised case)

$$P(A_1 ... A_k | B_1 ... B_p) = \frac{P(B_1 ... B_p, A_1 ... A_k)}{P(B_1 ... B_p)}$$

Bayesian Classifiers

Bayesian classifiers aim to find the mapping $f : \mathbf{x} \Rightarrow y$ for supervised learning in the form of conditional probability $P(y|\mathbf{X})$ via Bayes rule.

$$P(y|\mathbf{X}) = \frac{P(y,\mathbf{X})}{P(\mathbf{X})} = \frac{P(\mathbf{X}|y)P(y)}{P(\mathbf{X})}$$

For a classification with C classes, given a data instance $\mathbf{x}^*$:

$$y^* = c^* \, if \, c^* = \underset{c}{\operatorname{argmax}} P(y = c|\mathbf{x}^*)$$

Applying Bayes rule,

$$P(y = c|\mathbf{x}^*) = \frac{P(\mathbf{x}^*|y = c)P(y = c)}{P(\mathbf{x}^*)}$$

Therefore,

$$y^* = \underset{c}{\operatorname{argmax}} \frac{P(\mathbf{x}^*|y = c)P(y = c)}{P(\mathbf{x}^*)}$$

$$= \underset{c}{\operatorname{argmax}} P(\mathbf{x}^*|y = c)P(y = c)$$

2 Bayesian Decision Theory

Incorporating cost of misclassification on top of simple Bayesian Classifiers.

Loss/Cost

Actions: a_c, i.e., predict $y = c$
Define λ_{ij} as the cost of a_i when optimal action is a_j. E.g.:
$\lambda_{00} = 0$ (predict correctly)
$\lambda_{11} = 0$ (predict correctly)
$\lambda_{01} = 10$ misclassify 1 as 0
$\lambda_{00} = 1$ misclassify 0 as 1

Expected Risk

Expected risk for taking action a_i:

$$R(a_i|\mathbf{x}) = \sum_{c=0}^{C-1} \lambda_{ic} P(y = c|\mathbf{x})$$

To classify, for all actions, calculate expected risk, then choose the action with the minimum risk.

Special Case: 0/1 loss

$$\lambda_{ij} = \begin{cases} 0 \text{ if } i = j \\ 1 \text{ if } i \neq j \end{cases}$$

$\therefore R(a_i|\mathbf{x}) = 1 - P(y = i|\mathbf{x})$
In this case,

$$\text{Choose } a_i \text{ if } R(a_i|\mathbf{x}) = \min_{a_c} R(a_c|\mathbf{x})$$

Is equivalent to:

$$\text{Predict } y = c^*$$

$$\text{if } P(y = c^*|\mathbf{x}) = \max_c P(y = c|\mathbf{x})$$

3 Naïve Bayes Classifiers

Independence

A is **independent** of B, if:
$P(A,B) = P(A|B) \times P(B) = P(A) \times P(B)$
$P(A,B) = P(B|A) \times P(A) = P(A) \times P(B)$
Or,
$P(A|B) = P(A)$
$P(B|A) = P(B)$

Conditional Independence

A is **conditionally independent** of B, given C if:
$P(A|B,C) = P(A|C)$

Naïve Bayes Classifier

1. Assumption: conditional independence of features given label

$$p(\mathbf{x}|y = c) = P(x_1, ..., x_d|y = c)$$

$$= P(x_1|y = c)P(x_2|y = c)...P(x_d|y = c)$$

$$= \prod_{i=1}^{d} P(x_i|y = c)$$

To classify a test record $\mathbf{x}^*$, compute the posteriors for each class:

$$p(y = c|\mathbf{x}^*) = \frac{(\prod_{i=1}^{d} P(x_i^*|y = c))P(y = c)}{P(\mathbf{x}^*)}$$

Since $P(\mathbf{x}^*)$ is constant for each class c, it is sufficient to choose the class that maximises the numerator term.

$$y^* = \underset{c}{\operatorname{argmax}}(\prod_{i=1}^{d} P(x_i^*|y = c))P(y = c)$$

Estimating Cond Prob (Discrete)

$$P(x_i = k|y = c) = \frac{|(x_i - k) \wedge (y = c)|}{|y = c|}$$

Estimating Cond Prob (Continuous)

$$P(x_i|y = c) = \frac{1}{\sqrt{2\pi\sigma_{ic}^2}} e^{-\frac{(x_i - \mu_{ic})^2}{2\sigma_{ic}^2}}$$

Supposing there are N_c instances in class c,
Sample mean:

$$\mu_{ic} = \frac{1}{N_c} \sum_{j=1}^{N_c} x_{ij}$$

Sample variance:

$$\sigma_{ic}^2 = \frac{1}{N_c - 1} \sum_{j=1}^{N_c} (x_{ij} - \mu_{ic})^2$$

Laplace Estimate

Alternative prob estimation for discrete features.

$$P(x_i = k|y = c) = \frac{|(x_i - k) \wedge (y = c)| + 1}{|y = c| + n_i}$$

where n_i is #distinct values of x_i. In extreme cases with no training data, $P(x_i = k|y = c) = \frac{1}{n_i}$

M-estimate

A more general estimation:

$$P(x_i = k|y = c) = \frac{|(x_i - k) \wedge (y = c)| + m\tilde{P}}{|y = c| + m}$$

Where m is a hyperparameter and $\tilde{P}$ is prior information of $P(x_i = k|y = c)$. (e.g., domain knowledge)
Extreme case with no training data: $P(x_i = k|y = c) = \tilde{P}(x_i = k|y = c)$

4 Bayesian Belief Networks

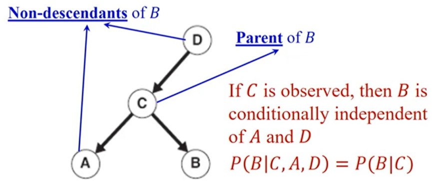
Suppose all features are **discrete** (if there are continuous and discrete, estimation is much more difficult)

Two key elements:
1. A directed acyclic graph (DAG) encoding dependence relationships between a set of variables

2. A probability table associating each node to immediate parent nodes

DAG: Conditional Independence

A node in a Bayesian network is conditionally independent of its non-descendants, **if its parents are known**.



IMPORTANT! If A and B are conditionally independent given C, we have:
1. $P(A|B,C) = P(A|C)$
2. $P(A,B|C) = P(A|C)P(B|C)$

Important! Using BBN for Inference

Given a BBN, and an inference(prediction) task:
1. Translate problem into probabilisitc language
2. If the probabilities to be estimated cannot be obtained from the probability tables of the BBN, then
A. Identify a subgraph which captures the dependence between input variables (features) and output variable (class)

B. Based on the network topology, apply product rule, sum rule and the properties of conditional independence and independence to induce equivalent forms of the probabilities until all probabilities can be found from the probability tables.

5 Decision Trees

- Greedy strategy, split records based on feature test that optimises certain criterion
Key issues:
1) How to split the records?
- Specifying feature test condition
- Determining best split
2) When to stop splitting?

Determining Test Conditions

Splitting based on binary features

2 Possible outcomes (e.g. Yes/No)

Splitting based on discrete features

- Multi-way split: Use as many partitions as distinct values
e.g.: Marital Status $\Rightarrow$ [Single], [Divorced], [Married]

- Binary split: Divides possible values as 2 subsets, need to find optimal partitioning
e.g.: Marital Status $\Rightarrow$ [Single, Divorced], [Married]

Splitting based on continuous features

- Binary split: $(x_j < v)$ or $(x_j \geq v)$

- Multi-way split (Discretization)

Consider all possible splits and find the best cut
Can be very computationally intensive

Determining Best Split

Using measure of node impurity – favour split with low degree of impurity

Measure of Impurity: Entropy

Entropy at a given node t:

$$E(t) = -\sum_c P(y = c; t) log_2 P(y = c; t)$$

Information Gain

$\Delta_{info} = E(\text{parent}) - E(\text{children})$
To get entropy for children, get entropy of all children nodes and normalize by # of training examples in each child node.
Suppose a parent node t is split into P partitions (children),
$\Delta_{info} = E(t) - \sum_{j=1}^{P} \frac{n_j}{n} E(j)$

Disadvantage: Tends to prefer splits that result in large number of partitions.

Penalizing large number of partitions (Gain Ratio)

$$\Delta_{\text{InfoR}} = \frac{\Delta_{\text{info}}}{\text{SplitINFO}}$$

$$\text{SplitINFO} = -\sum_{i=1}^{P} \frac{n_i}{n} log_2(\frac{n_i}{n})$$

Stopping Criterias

1. All data belong to same class
2. Stop expanding when all data have similar feature vals
3. Early termination (avoid overfitting)

6 Generalisation

Overfitting: Test error rate increase when training error decrease
Underfitting: Model too simple, both training and test error large

Training errors: error on training set, $e(T)$
Generalisation errors: error on previously unseen testing set, $e'(T)$

Estimating Generalisation Errors

Optimistic Estimate

Assume training set is good representation of overall data
$e'(T) = e(T)$
Decision tree induction algo select model with lowest training error rate.

Occam's Razor

Include information of model complexity when evaluating a model.
$e'(T) = e(T) + N \times k$
where N is the number of leaf nodes and k is a hyperparameter $k > 0$

Using Validation Set

Divide training data to 2 subsets, 1 for training and 1 for estimating generalisation error.

Addressing overfitting

Pre-Pruning

- Stop if number of instances is less than user-specified threshold
- Stop if expanding current node does not improve generalisation errrors

Post-Pruning

- Grow tree to its entirety
- Trim nodes in bottom-up fashion
- If generalisation error improves after trimming, replace sub-tree by new leaf