PQHS 471
Lecture 6:
Decision Tree, Bayes Classifier, KNN

Classification problems

Here the response variable Y is *qualitative* — e.g. email is one of $\mathcal{C} = (\texttt{spam}, \texttt{ham})$ ($\texttt{ham}$=good email), digit class is one of $\mathcal{C} = \{\texttt{0}, \texttt{1}, \ldots, \texttt{9}\}$. Our goals are to:

- Build a classifier $C(X)$ that assigns a class label from $\mathcal{C}$ to a future unlabeled observation X.
- Assess the uncertainty in each classification
- Understand the roles of the different predictors among $X = (X_1, X_2, \ldots, X_p)$.

Motivating example

Fruit Identification.

Skin	Color	Size	Flesh	Conclusion
Hairy	Brown	Large	Hard	safe
Hairy	Green	Large	Hard	Safe
Smooth	Red	Large	Soft	Dangerous
Hairy	Green	Large	Soft	Safe
Smooth	Red	Small	Hard	Dangerous
...				

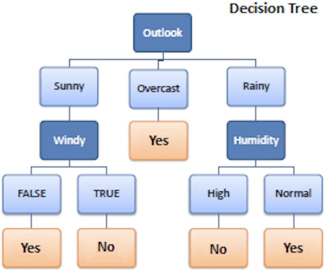
"At the edge of the world, statistical journey begins."

Decision Tree

- A flowchart tree-like structure that is made from training set.

Predictors | Target

Outlook	Temp.	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

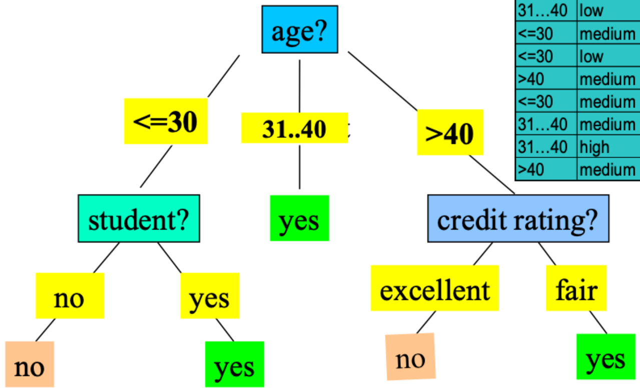


Decision Tree Induction (DTI)

- The method of learning the decision trees from the training set.
- Need to have a training dataset with **observations** and **class labels**.
- The tree structure has a root node, internal nodes or decision nodes, leaf node, and branches.
- The root node is the topmost node. It represents the best attribute selected for classification.
- Internal nodes of the decision nodes represent a test of an attribute of the dataset
- Leaf node or terminal node represents the classification or decision label.
- Some decision trees only have binary nodes (have exactly two branches of a node), while some are non-binary.

Decision Tree Induction: An example

- Training data set: Buys_computer
- Resulting tree:



age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Algorithm for DTI

- ID3 (Iterative Dichotomiser), C4.5,by Quinlan.
- CART (Classification and Regression Trees)

Algorithm for DTI

- ID3 (Iterative Dichotomiser), C4.5,by Quinlan.
- CART (Classification and Regression Trees)
- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes(predictors) are categorical (if continuous-valued, they are discretized in advance)
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)

Algorithm for DTI

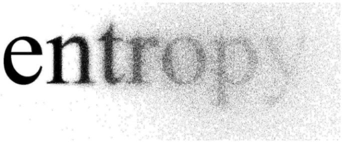
Conditions for stopping partitioning

- All samples for a given node belong to the same class
- There are no remaining attributes for further partitioning – **majority voting** is employed for classifying the leaf
- There are no samples left

Attribute(predictors) selection measures

- **Idea: select attribute that partition samples into homogeneous groups**
- Measures:
  - Information gain (ID3)
  - Gain ratio (C4.5)
  - Gini index (CART)
  - Variance reduction for continuous target variable (CART)

Brief Review of Entropy



- Entropy (Information Theory)
  - A measure of uncertainty associated with a random variable
  - Calculation: For discrete random variable Y taking m distinct values $y_1, y_2, ..., y_m$

$$H(Y) = -\sum_{i=1}^{m} p_i log(p_i)$$

    where $p_i = P(Y = y_i)$
- Interpretation:
  - Higher entropy $\rightarrow$ higher uncertainty
  - Lower entropy $\rightarrow$ lower uncertainty
- Conditional Entropy: $H(Y|X) = \sum_x p(x) H(Y|X = x)$

Attribute selection measure: Information Gain (ID3/C4.5)

- **Idea: select the attribute with the highest information gain**
- Let p_i be the probability that an arbitrary tuple (observation + label) in D belongs to class C_i, estimated by $|C_{i,D}|/|D|$
- Expected information (entropy) needed to classify a tuple in D:

$$Info(D) = -\sum_{i=1}^{m} p_i log(p_i)$$

- Information needed (after using A to split D into v partitions) to classify D:

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

- Information gained by branching on attribute A:

$$Gain(A) = Info(D) - Info_A(D)$$

Attribute selection: Information Gain

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

- Class P: buys computer = "yes"
- Class N: buys computer = "no"

$$Info(D) = I(9,5) = -\frac{9}{14}log(\frac{9}{14}) - \frac{5}{14}log(\frac{5}{14}) = 0.940$$

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

Attribute selection: Information Gain

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

$$Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2) = 0.694$$

Here, we have $\frac{5}{14}I(2,3)$ because the "age≤30" group has 5 out of 14 samples, with 2 yes and 3 no.

Hence:

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

Attribute selection: Information Gain

Similarly:

$$Gain(age) = 0.246$$

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(creditrating) = 0.048$$

Use "age" as the first(root) node for Decision Tree

Computing information gain for continuous valued attributes

- Let attribute A be a continuous-valued attribute
- Must determine the best split point for A
  - Sort the value A in increasing order
  - Typically, the midpoint between each pair of adjacent values is considered as a possible split point
    $\star(a_i + a_{i+1})/2$ is the midpoint between the values of a_i and a_{i+1}
  - The point with the *minimum expected information requirement* for A is selected as the split-point for A
- Split: $D1$ is the set of tuples in D satisfying $A \leq$ split-point, and $D2$ is the set of tuples in D satisfying $A >$ split-point

Gain Ratio for attribute selection (C4.5)

- Information gain measure is biased towards attributes with a large number of values
- C4.5 (a successor of ID3) uses gain ratio to overcome the problem (normalization to information gain)

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

  - GainRatio(A) = Gain(A)/SplitInfo(A)
- Ex. $SplitInfo_{income}(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right) = 1.557$
  - gain_ratio(income) = 0.029/1.557 = 0.019
- The attribute with the maximum gain ratio is selected as the splitting attribute

Gini Index (CART, IBM IntelligentMiner)

- If a dataset D contains examples from n classes, *gini index*, $gini(D)$ is defined as:

$$gini(D) = 1 - \sum_{j=1}^{n} p_j^2$$

  where p_j is the relative frequency of class j in D

- If a dataset D split on A into two subsets D_1 and D_2, the *gini index* $gini(D)$ is defined as:

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

- Reduction in Impurity:

$$\Delta gini(A) = gini(D) - gini_A(D)$$

- The attribute provides the largest reduction in impurity (or smallest $gini_{split}(D)$) is chosen to split the node

Computation of Gini Index

- Ex. D has 9 tuples in buys_computer = "yes" and 5 in "no"

$$gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- Suppose the attribute income partitions D into 10 in D_1: {low, medium} and 4 in D_2

$$gini_{income \in \{low,medium\}}(D) = \left(\frac{10}{14}\right) Gini(D_1) + \left(\frac{4}{14}\right) Gini(D_2)$$

$$= \frac{10}{14}\left(1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2\right) + \frac{4}{14}\left(1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2\right)$$

$$= 0.443$$

$$= Gini_{income \in \{high\}}(D).$$

$Gini_{\{low,high\}}$ is 0.458; $Gini_{\{medium,high\}}$ is 0.450. Thus, split on the {low,medium} (and {high}) since it has the lowest Gini index

Comparing Attribute Selection Measures

These three measures, in general, return good results but

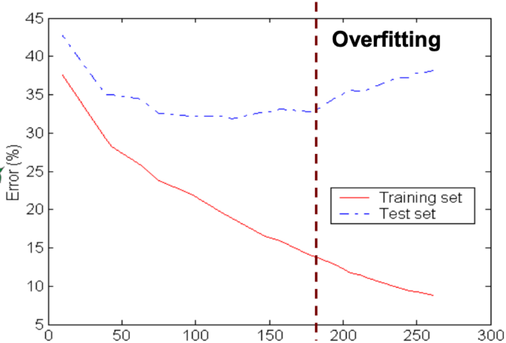
- **Information Gain**:
  biased towards multivalued attributes
- **Gain Ratio**:
  tends to prefer unbalanced splits in which one partition is much smaller than the others
- **Gini Index**:
  (1) biased to multivalued attributes
  (2) tends to favor tests that result in equal-sized partitions and purity in both partitions

Overfitting in DTI

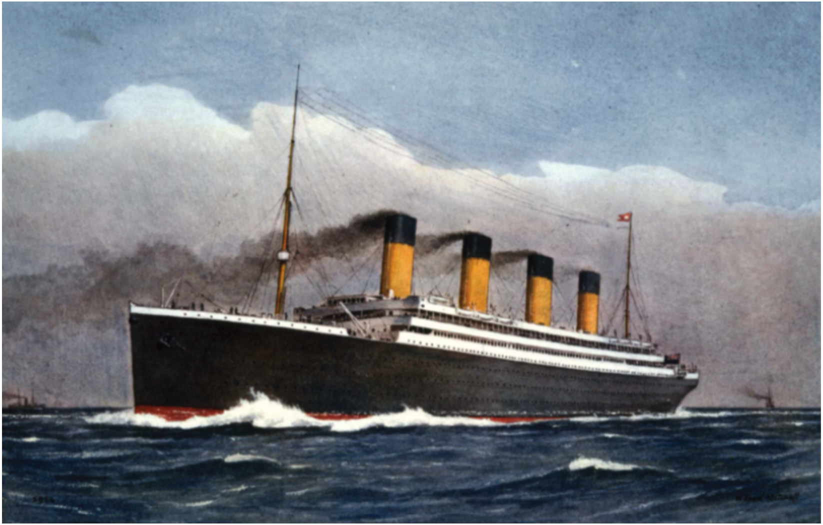
Overfitting: An induced tree may overfit the training data

- Too many branches, some may reflect anomalies and noises
- Poor accuracy for unseen sample

Underfitting: when model is too simple, both training and test errors are large



Would you survive the Titanic?



Build a predictive model: "what sorts of people were more likely to survive?" Kaggle Titanic ML Competition

Bayes Classification Methods

Bayesian Classification: Why?

- **The most fundamental statistical classifier.**
- Performs probabilistic prediction, i.e., predicts class membership probabilities.
- Foundation: Bayes' Theorem
- The best classifier as it *minimizes the expected classification error rate*.
- Often used as a reference in simulation study.
- Bayes classifier is often **unknown in practice.**

Bayes' Theorem: Basics

Total Probability Theorem

$P(B) = \sum_{i=1}^{M} P(B|A_i)P(A_i)$

Bayes' Theorem

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

- $P(A|B)$ and $P(B|A)$ are conditional probabilities.
- $P(A)$ and $P(B)$ are marginal probabilities.

Posterior probability

Posterior $\propto$ Likelihood $\times$ Prior

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)}$$

$$p(\theta|x) \propto p(x|\theta)p(\theta)$$

$p(\theta)$ is the prior
$p(x|\theta)$ is the likelihood
$p(\theta|x)$ is the posterior

Bayes' theorem: cookie example

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

Red jar: 10 chocolate + 30 plain
Yellow jar: 20 chocolate + 20 plain
Pick a jar, and then pick a cookie
If it's a plain cookie, what's the probability the cookie was picked out of red jar?

Classification using posterior

- Let D be a training set of tuples and their associated class labels, and each tuple is represented by an $n - D$ attribute vector $\mathbf{X} = (x_1, x_2, ..., x_n)$
- Suppose there are m classes $C_1, C_2, ..., C_m$.
- Classification is to find the i s.t.

$$argmax_i P(C_i|\mathbf{X})$$

- By Bayes' theorem:

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

- Since $P(\mathbf{X})$ is constant for all classes:

$$P(C_i|\mathbf{X}) \propto P(\mathbf{X}|C_i)P(C_i)$$

(Naïve) Bayes Classifier

A simplified assumption: attributes are conditionally independent:

$$P(\mathbf{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i)$$

(Naïve) Bayes Classifier: Example

Class:

C1:buys_computer = 'yes'

C2:buys_computer = 'no'

Data to be classified:

X = (age <=30,

Income = medium,

Student = yes

Credit_rating = Fair)

age	income	student	credit_rating	com
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

(Naïve) Bayes Classifier: Example

age	income	student	credit_rating	com
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

- $P(C_i)$: P(buys_computer = "yes") = 9/14 = 0.643
  P(buys_computer = "no") = 5/14= 0.357
- Compute $P(X|C_i)$ for each class
  P(age = "<=30" | buys_computer = "yes") = 2/9 = 0.222
  P(age = "<= 30" | buys_computer = "no") = 3/5 = 0.6
  P(income = "medium" | buys_computer = "yes") = 4/9 = 0.444
  P(income = "medium" | buys_computer = "no") = 2/5 = 0.4
  P(student = "yes" | buys_computer = "yes) = 6/9 = 0.667
  P(student = "yes" | buys_computer = "no") = 1/5 = 0.2
  P(credit_rating = "fair" | buys_computer = "yes") = 6/9 = 0.667
  P(credit_rating = "fair" | buys_computer = "no") = 2/5 = 0.4
- **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

$P(X|C_i)$: P(X|buys_computer = "yes") = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X|buys_computer = "no") = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

$P(X|C_i)*P(C_i)$: P(X|buys_computer = "yes") * P(buys_computer = "yes") = 0.028
P(X|buys_computer = "no") * P(buys_computer = "no") = 0.007

Therefore, X belongs to class ("buys_computer = yes")

Avoiding the Zero-Probability Problem

- Naïve Bayes Classifier requires each conditional probability to be **non-zero**. Otherwise, the predicted prob. will be 0:

$$P(\mathbf{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i)$$

- E.g. A dataset with 1,000 tuples, income = low ($n = 0$), income = medium ($n = 990$), income = high ($n = 10$).
- Use **Laplacian correction**
  - Adding 1 to each case
    Prob(income=low) = 1/1003
    Prob(income=medium) = 991/1003
    Prob(income=high) = 11/1003
- The "corrected" prob. estimates are close to their "uncorrected" counterparts.

(Naïve) Bayes Classifier: comments

Advantages:

- Easy to implement
- Good results obtained in most of the cases

Disadvantages:

- Assumption: class conditional independence, therefore loss of accuracy
- Practically, dependencies exist among variables
  E.g., hospitals: patients: Profile: age, family history, etc.
  Symptoms: fever, cough etc.,
- Dependencies among these cannot be modeled by Naïve Bayes Classifier

How to deal with these dependencies? Bayesian Belief Networks

(Naïve) Bayes Classifier: practice

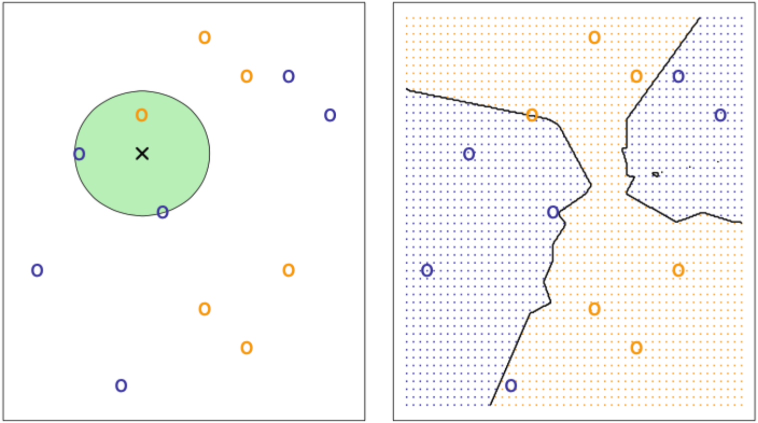
Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Predict whether this person will play golf or not for the following tuple: (Outlook = Sunny, Temp = Cool, Humidity = High, Windy = True)

k-nearest neighbors (KNN)

k-nearest neighbors (KNN)

KNN is a local non-parametric classification method.
k is a hyperparameter.
KNN example with $k = 3$.



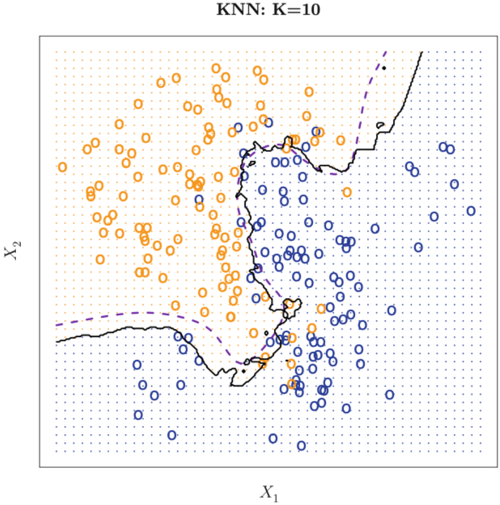
k-nearest neighbors (KNN)

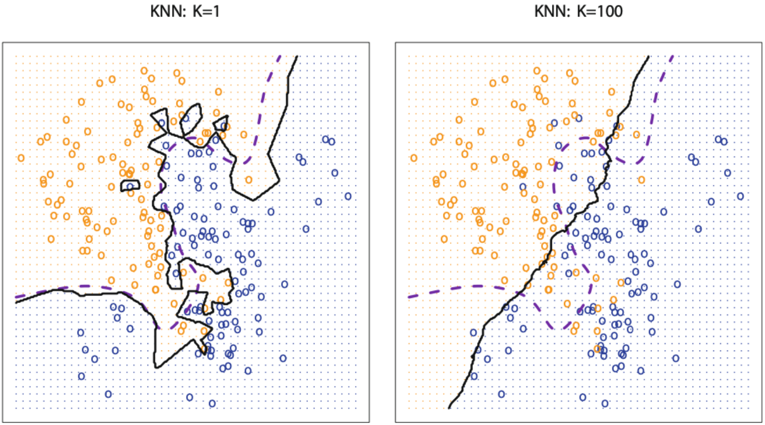
- Given a positive integer k and a test observation x_0
- First, identifies the k points in the training data that are closest to x_0, represented by $\mathcal{N}_0$.
- Then, estimates the conditional probability for class j as the fraction of points in $\mathcal{N}_0$ whose response values equal j.

$$Pr(Y = j|X = x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = j)$$

- Assign x_0 to the class with the largest probability.

k-nearest neighbors (KNN)




KNN: K=1
KNN: K=100

k-nearest neighbors (KNN)