Introduction to Data Science

DSA1101

Semester 1, 2018/2019
Week 8

Classification methods: Decision Trees

Decision Trees

- Recall from last lecture that *decision trees* are built based on measures such as entropy reduction (or equivalently information gain) in selecting the decision variables as well as their split points
- In today's lecture, we will look at a few more examples of *decision trees* in `R` and also take a look at the decision or prediction surface that arise from fitting *decision trees*

Classification methods: Decision Trees

Source: *The Straits Times*

- The goal of this illustrative example is to predict whether to play golf given factors such as weather outlook, temperature, humidity, and wind.

Decision Trees

- First, we read in the dataset `DTdata.csv` (which has been posted to IVLE) and load the R packages ‘rpart’ and ‘rpart.plot’.
-
```
library("rpart") # load libraries
library("rpart.plot")

play_decision <- read.table("DTdata.csv",header=
    TRUE,sep=",")
```

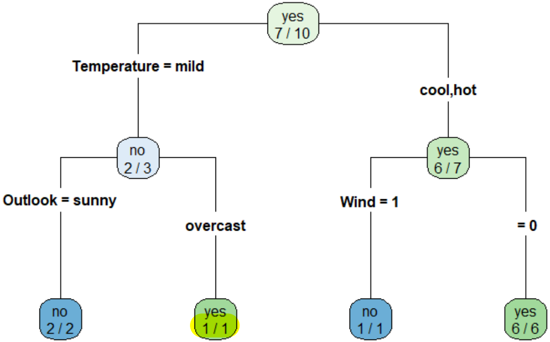
Decision Trees

- The CSV file contains five attributes: Play, Outlook, Temperature, Humidity, and Wind.
- Play would be the output variable (or the predicted class), and Outlook, Temperature, Humidity, and Wind would be the input variables.
-

```
> head(play_decision)
  Play  Outlook Temperature Humidity  Wind
1  yes    rainy        cool   normal FALSE
2   no    rainy        cool   normal  TRUE
3  yes overcast         hot     high FALSE
4   no    sunny        mild     high FALSE
5  yes    rainy        cool   normal FALSE
6  yes    sunny        cool   normal FALSE
```

Decision Trees

- We will build a *decision tree* to predict golf play based on feature variables such as weather outlook, temperature, humidity, and wind, using entropy reduction (or information gain) to determine the split variables.
- ```
  fit <- rpart(Play ~ Outlook + Temperature +
      Humidity + Wind,
  method="class",
  data=play_decision,
  control=rpart.control(minsplit=1),
  parms=list(split='information'))

  rpart.plot(fit, type=4, extra=2)
  ```

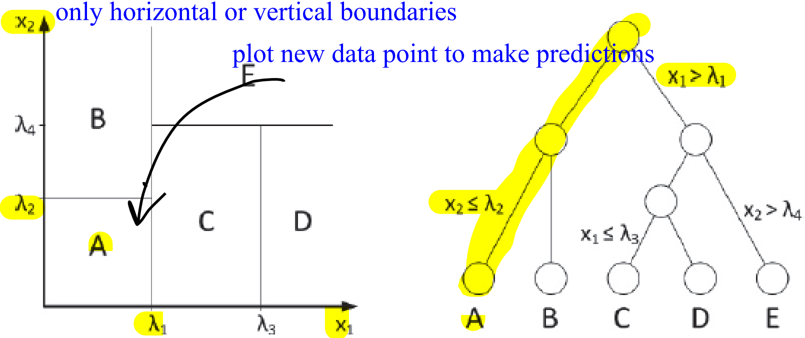
Minimum observation to split to a new tree
  ```

Decision Trees



Decision Trees

- The decision tree can be used to predict outcomes for new datasets.
- Consider a testing set that contains the following record: Outlook="rainy", Temperature="mild", Humidity="high", Wind=FALSE
- The goal is to predict the play decision of this record. The following code loads the data into R as a data frame newdata.
-

```
newdata <- data.frame(Outlook="rainy", Temperature
    ="mild",
Humidity="high", Wind=FALSE)

> newdata
  Outlook Temperature Humidity  Wind
1   rainy        mild     high FALSE
```

Decision Trees

- Next, use the predict function to generate predictions from a fitted rpart object. The format of the predict function as as follows:
- ```
  > predict(fit,newdata=newdata,type="prob")
    no yes
  1  1   0
  > predict(fit,newdata=newdata,type="class")
   1 class with highest probability
  no
  Levels: no yes
  ```
  ```

Decision Trees

- In a decision tree, the decision regions are rectangular surfaces.
- The next figure shows an example of five rectangular decision surfaces (A, B, C, D, and E) defined by four values $(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ of two attributes or feature variables $(x_1 and x_2)$.

Decision Trees

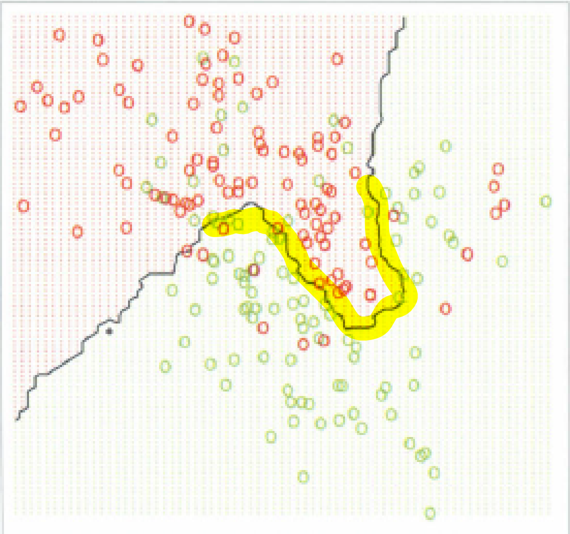


Prediction surface for *decision tree*. Source: *Data Science & Big Data Analytics* hard to trace for >=3 features

Decision Trees

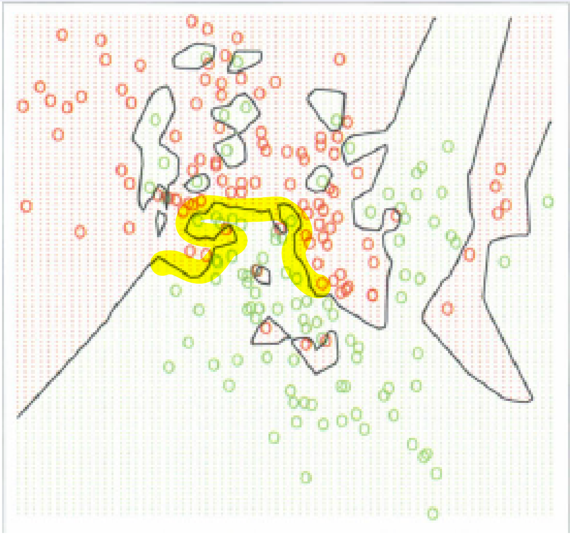
- Decision or prediction surface corresponds to a leaf node of the tree, and it can be reached by traversing from the root of the tree following by a series of decisions according to the value of an attribute.
- The decision surface can only be axis-aligned for the decision tree.
- Contrast this decision surface to that of other predictive methods, such as k-nearest neighbor classifier

k-nearest neighbor classification



Prediction by majority vote with 15 nearest neighbors. Source: *The Elements of Statistical Learning*, Hastie et al.

k-nearest neighbor classification

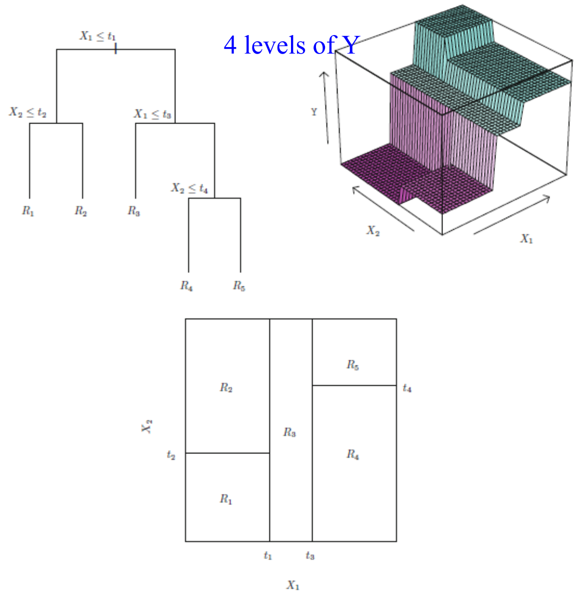


Prediction by majority vote with one nearest neighbor. Source: *The Elements of Statistical Learning*, Hastie et al.

Decision Trees

- Another example of a prediction or decision surface based on *decision trees*

Decision Trees



Prediction surface for *decision tree*. Source: *Introduction to Statistical*

Classification methods: The Naïve Bayes Classifier

Naïve Bayes Classifier

- *Naïve Bayes* is a probabilistic classification method based on Bayes' theorem (or Bayes' law) with a few tweaks
  improvements
- Bayes' theorem gives the relationship between the probabilities of two events and their conditional probabilities.

Naïve Bayes Classifier

Source: *Wikipedia*

- Bayes' law is named after the English mathematician Thomas Bayes.

Naïve Bayes Classifier

- A *naïve Bayes* classifier assumes that the presence or absence of a particular feature of a class is unrelated to the presence or absence of other features.
- For example, an object can be classified based on its attributes such as shape, color, and weight.
- A reasonable classification for an object that is spherical, yellow, and less than 60 grams in weight may be a tennis ball.
- Even if these features depend on each other or upon the existence of the other features, a *naïve Bayes* classifier considers all these properties to contribute independently to the probability that the object is a tennis ball.

Naïve Bayes Classifier

- The input variables are generally categorical, but variations of the algorithm can accept continuous variables.
- There are also ways to convert continuous variables into categorical ones. This process is often referred to as the discretization of continuous variables.
- For example, weight can be discretized to the categorical values $\leq 1kg$, $1kg <$ & $\leq 5kg$, and $> 5kg$.
- The output typically includes a class label and its corresponding probability score.

Naïve Bayes Classifier

- Because *naïve Bayes* classifiers are easy to implement and can execute efficiently even without prior knowledge of the data, they are among the most popular algorithms for classifying text documents.
- Spam filtering is a classic use case of *naïve Bayes* text classification.
- *Naïve Bayes* classifiers can also be used for fraud detection.
- In the domain of auto insurance, for example, based on a training set with attributes such as driver's rating, vehicle age, vehicle price, historical claims by the policy holder, police report status, and claim genuineness, *naïve Bayes* can provide probability-based classification of whether a new claim is genuine

Naïve Bayes Classifier: Bayes' Theorem

- The conditional probability of event C occurring, given that event A has already occurred, is denoted as $P(C|A)$, which is defined as

$$P(C|A) = \frac{P(A \cap C)}{P(A)} = \frac{P(A|C) \times P(C)}{P(A)},$$

  where where $P(A \cap C)$ is the probability that both events A and C occur.

- Bayes' theorem is significant because quite often $P(C|A)$ is much more difficult to compute than $P(A|C)$ and $P(C)$ from the training data.
- By using Bayes' theorem, this problem can be circumvented; we will illustrate this with a few numerical examples.

Naïve Bayes Classifier: Bayes' Theorem

- The first example concerns computing the probability that a patient carries a disease based on the result of a lab test.
- The test returns a positive result in 95% of the cases in which the disease is actually present, and it returns a positive result in 6% of the cases in which the disease is not present.

TPR = 95%, FPR = 6%

- Furthermore, 1% of the entire population has this disease.

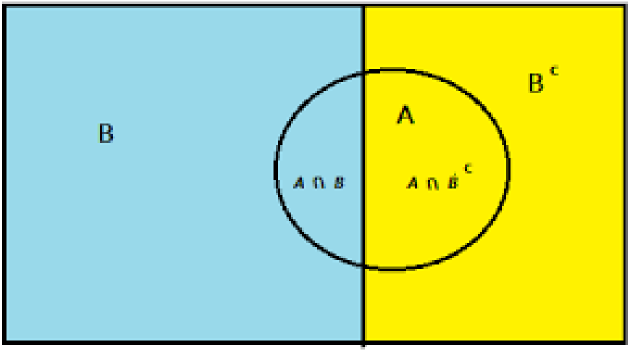
Naïve Bayes Classifier: Bayes' Theorem

- Suppose that a patient took a lab test for a certain disease and the result came back positive.
- What is the probability that the patient actually has the disease, given that the test result is positive?
- Define the events $C = \{\text{having the disease}\}$ and $A = \{\text{positive test result}\}$
- We wish to compute the conditional probability $P(C|A)$.

Naïve Bayes Classifier: Bayes' Theorem

- Let $\neg\mathcal{M}$ denote the negation of the event $\mathcal{M}$
- Based on the problem description, we have $P(C) = 0.01$, $P(\neg C) = 0.99$, $P(A|C) = 0.95$ and $P(A|\neg C) = 0.06$
- $$\begin{aligned} P(C|A) &= \frac{P(A|C)P(C)}{\boxed{P(A)}} \\ &= \frac{P(A|C)P(C)}{\boxed{P(A \cap C) + P(A \cap \neg C)}} \\ &= \frac{P(A|C)P(C)}{\boxed{P(C) \times P(A|C) + P(\neg C)P(A|\neg C)}} \\ &= \frac{0.95 \times 0.01}{0.01 \times 0.95 + 0.99 \times 0.06} \approx 0.1379 \end{aligned}$$

Decision Trees



Law of total probability

Naïve Bayes Classifier: Bayes' Theorem

- That means that the probability of the patient actually having the disease given a positive test result is 13.79%.
- Without any test result, the probability of the patient actually having the disease is only 1%
- The probability of being labelled as having the disease (Y) increases after incorporating the feature variable of test result (X)
- We will study the *naïve Bayes* classifier in more detail next week.