Outline

- Instance-based learning
- Naïve Bayes algorithm
- Bayesian networks
Instance-based learning

Machine learning process

Formulating the task → Getting data → Building **Instance-based learning predictor** → Final evaluation → Predictor to use
Instance-based learning

Key idea

- IBL methods initially store the training data (that is why IBL methods are often referred to as "lazy" methods).
- For a new instance, prediction is based on local similarity, i.e. a set of similar instances are retrieved and used for prediction.
- IBL methods can construct a different approximation of a target function for each distinct test instance.
- Both classification and regression.
Instance-based learning

Key points

1. A distance metric
2. How many nearby neighbours look at?
3. A weighting function
4. How to fit with local points?
Instance-based learning

Distance metric

The most common ones

- **Euclidean distance**
  \[ E(x_i, x_j) = \sqrt{\sum_{r=1}^{m} (x_{ir} - x_{jr})^2} \]  
  \[ (1) \]

- **Manhattan distance**
  \[ M(x_i, x_j) = \sum_{r=1}^{m} |x_{ir} - x_{jr}| \]  
  \[ (2) \]
Instance-based learning
Learning algorithms

- k-Nearest Neighbour
- Distance weighted k-NN
- Locally weighted linear regression
- ...
Instance-based learning

\textit{k-Nearest Neighbour algorithm}

1. A distance metric: Euclidian (most widely used)
2. How many nearby neighbours look at? \textit{k}
3. A weighting function: unused
4. How to fit with local points?

- \textit{k-NN classification}

\[
h(x) = \arg\max_{v \in Y} \sum_{i=1}^{k} \delta(v, y_i),
\]

where \(\delta(a, b) = 1\) if \(a = b\), otherwise 0

- \textit{k-NN regression}

\[
h(x) = \frac{\sum_{i=1}^{k} y_i}{k}
\]
Instance-based learning
Distance-weighted $k$-NN algorithm

1. **A distance metric**: Euclidian (most widely used)
2. **How many nearby neighbours look at?** $k$
3. **A weighting function**: greater weight closer neighbours

$$w_i(x) = \frac{1}{d(x, x_i)^2}$$

4. **How to fit with local points?**
   - Classification
     $$h(x) = \arg\max_{v \in Y} \sum_{i=1}^{k} w_i(x) \delta(v, y_i)$$  \hspace{1cm} (5)
   - Regression
     $$h(x) = \frac{\sum_{i=1}^{k} w_i(x)y_i}{\sum_{i=1}^{k} w_i(x)}$$ \hspace{1cm} (6)
Instance-based learning
Distance-weighted $k$-NN algorithm

Shepard’s method

- Classification

$$h(x) = \arg\max_{v \in Y} \sum_{i=1}^{n} w_i(x) \delta(v, y_i)$$  \hspace{1cm} (7)

- Regression

$$h(x) = \frac{\sum_{i=1}^{n} w_i(x) y_i}{\sum_{i=1}^{n} w_i(x)}$$  \hspace{1cm} (8)
Instance-based learning
Locally weighted linear regression

1. A distance metric: Euclidean (most widely used)
2. How many nearby neighbours look at? $k$
3. A weighting function: $w_i(x)$
4. How to fit with local points?

$$\Theta^* = \arg\min_{\Theta} \sum_{i=1}^{k} w_i(x)(\Theta^T x_i - y_i)^2$$ (9)
Instance-based learning
Locally weighted linear regression
Instance-based learning
LW linear regression vs. simple regression
Naïve Bayes classifier
Bayes theorem

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

\[ P(A|B) = \frac{P(A) \times P(B|A)}{P(B)} \]

- \( P(A) \) is the prior probability (marginal) probability of \( A \). It does not take into account any information about \( B \).
- \( P(A|B) \) is the conditional probability of \( A \), given \( B \). So called the posterior probability because it depends upon the specified value of \( B \); \( P(B|A) \) is the conditional probability of \( B \) given \( A \).
- \( P(B) \) is the prior (marginal) probability of \( B \), and acts as a normalizing constant.
Naïve Bayes classifier
Bayes theorem

\[
\Pr(Y \mid A_1, \ldots, A_m) = \frac{\Pr(Y) \times \Pr(A_1, \ldots, A_m \mid Y)}{\Pr(A_1, \ldots, A_m)} \tag{12}
\]

posterior = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}} \tag{13}
Let $X$, $Y$ and $Z$ be three discrete random variables. We say that $X$ is \textit{conditionally independent} of $Y$ given $Z$ if

$$\forall x_i, y_j, z_k, x_i \in \text{Values}(X), y_j \in \text{Values}(Y), z_k \in \text{Values}(Z) :$$

$$\Pr(X = x_i | Y = y_j, Z = z_k) = \Pr(X = x_i | Z = z_k)$$  \hspace{1cm} (14)$$

I.e., $P(X|Y, Z) = P(X|Z)$. 

Assume three variables: Thunder, Rain, and Lighting. Thunder is conditionally independent of Rain given Lighting:

$$Pr(\text{Thunder} | \text{Rain, Lighting}) = Pr(\text{Thunder} | \text{Lighting})$$
Naïve Bayes classifier

Discriminative vs. generative classifiers

- **discriminative classifier** does not care about how the data was generated. It simply classifies a given example.

- **generative classifier** models how the data was generated in order to classify an example. It asks the question *Based on the generation assumptions, which class is most likely to generate this example?*
Naïve Bayes classifier

Discriminative vs. generative classifiers

\[
Pr(y | x) = ?
\]

- Logistic regression classifier is a **discriminative classifier**

  \[
h_\Theta(x) = p(y = 1 | x, \Theta)
\]

- Naïve Bayes classifier is a **generative classifier**

  1. Learn \( Pr(x | y) \) and \( Pr(y) \)
  
  2. Apply Bayes rule to get

  \[
  Pr(y | x) = \frac{Pr(x | y) Pr(y)}{Pr(x)} \sim Pr(x | y) Pr(y)
  \]

  3. \( \hat{y} = \arg\max_y Pr(y | x) = \arg\max_y Pr(x | y) Pr(y) \)
If we work with two features $A_1, A_2$ and we assume that they are conditionally independent given the target class $Y$, then

$$
\Pr(A_1, A_2 | Y) \overset{\text{product rule}}{=} \Pr(A_1 | A_2, Y) \times \Pr(A_2 | Y) \overset{\text{conditional independence assumption}}{=} \Pr(A_1 | Y) \times \Pr(A_2 | Y)
$$
Assume conditional independence of features $A_1, \ldots, A_m$ given $Y$. Thus

$$\Pr(x|y) = \Pr(x_1, x_2, \ldots, x_m|y) \overset{\text{chain rule}}{=} \prod_{j=1}^{m} \Pr(x_j|x_1, x_2, \ldots, x_{j-1}, y) \overset{\text{c. i. a.}}{=} \prod_{j=1}^{m} \Pr(x_j|y)$$

**Naïve Bayes classifier**

$$\hat{y} = \arg\max_{y_k \in Y} \Pr(y_k) \prod_{j=1}^{m} \Pr(x_j|y_k) \quad (15)$$
Naïve assumption of feature conditional independence given a target class is rarely true in real world applications. Nevertheless, Naïve Bayes classifier surprisingly often shows good performance in classification.
Naïve Bayes Classifier is a linear classifier.

NB classifier gives a method for predicting rather than an explicit classifier.

**Prediction function**

\[ \hat{y} = \text{argmax}_{y_k \in Y} \Pr(y_k) \prod_{j=1}^{m} \Pr(x_j | y_k) \]

We focus on **binary classification** \( Y = \{0, 1\} \) with binary features \( A_1, \ldots, A_m \).

We predict \( \hat{y} = 1 \) iff

\[
\frac{\Pr(y = 1) \prod_{j=1}^{m} \Pr(x_j | y = 1)}{\Pr(y = 0) \prod_{j=1}^{m} \Pr(x_j | y = 0)} > 1
\]
Naïve Bayes Classifier
is a linear classifier

Denote \( p_j = \Pr(x_j = 1|y = 1) \), \( q_j = \Pr(x_j = 1|y = 0) \)

Then

\[
\frac{\Pr(y = 1) \prod_{j=1}^{m} p_j^{x_j} (1 - p_j)^{1-x_j}}{\Pr(y = 0) \prod_{j=1}^{m} q_j^{x_j} (1 - q_j)^{1-x_j}} > 1
\]

\[
\frac{\Pr(y = 1) \prod_{j=1}^{m} (1 - p_j)(\frac{p_j}{1-p_j})^{x_j}}{\Pr(y = 0) \prod_{j=1}^{m} (1 - q_j)(\frac{q_j}{1-q_j})^{x_j}} > 1
\]

Take logarithm
Naïve Bayes Classifier
is a linear classifier

\[
\log \frac{\Pr(y = 1)}{\Pr(y = 0)} + \sum_{j=1}^{m} \log \frac{1 - p_j}{1 - q_j} + \sum_{j=1}^{m} (\log \frac{p_j}{1 - p_j} - \log \frac{q_j}{1 - q_j}) x_j > 0
\]

NB classifier as a linear classifier where

\[
\Theta_j = \log \frac{p_j}{1 - p_j} - \log \frac{q_j}{1 - q_j}
\]
Task: Will students fall asleep during the lecture?

$Attr = \{\text{It’s raining, They are tired, They were at the party last night}\}$

$Values(\text{It’s raining}) = \{\text{Yes, No}\}$
$Values(\text{They are tired}) = \{\text{Yes, No}\}$
$Values(\text{They were at the party last night}) = \{\text{Yes, No}\}$

$Y = \text{FallAsleep}, Values(\text{FallAsleep}) = \{\text{Yes, No}\}$
Naïve Bayes assumption
– features are conditionally independent given the value of the target class.
Bayesian belief networks

Motivation

... but

They were at the party last night.

It's raining

They are tired.

FallAsleep
Bayesian belief networks

Motivation

- Naïve Bayes classifier assumes that ALL features are conditionally independent given the value of the target class.
- A Bayesian network is a graphical model that encodes probabilistic relationships among attributes of interest.
- BBNs allow stating conditional independence assumptions that apply to SUBSETs of the attributes.
- Dependencies are modeled as graph where nodes correspond to attributes and edges go from cause to effect.
- BBNs combine prior knowledge with observed data.
- BBNs are less constraining than the global assumption by NB.
Bayesian belief networks
Settings

Consider an arbitrary set of random variables $X_1, X_2, ..., X_m$. Each variable $X_i$ can take on the set of possible values $Values(X_i)$.

We define the **joint space** of the variables $X_1, X_2, ..., X_m$ to be the cross product $Values(X_1) \times Values(X_2) \times Values(X_3) \times ... \times Values(X_m)$.

The probability distribution over the joint space is called the **joint probability distribution** $Pr(x_1, x_2, ..., x_m)$ where $x_1 \in Values(X_1), x_2 \in Values(X_2), ..., x_n \in Values(X_m)$.

BBN describes the joint probability distribution for a set of variables by specifying a set of conditional independence assumptions together with sets of local conditional probabilities.
Bayesian belief networks

Representation

1. A directed acyclic graph $G = (V, E)$
   - nodes are random variables
   - arcs between nodes represent probabilistic dependencies
   - arcs are drawn from cause to effect
   - $Y$ is a descendant of $X$ if there is a directed path from $X$ to $Y$.

2. The network arcs represent the assertion that the variable $X$ is conditionally independent of its nondescendants given its immediate predecessors $\text{Parents}(X)$; $\Pr(X|X_i)_{X_i \in \text{Parents}(X)}$

3. A set of tables for each node in the graph - a conditional probability table is given for each variable; it describes the probability distribution for that variable given the values of its immediate predecessors.
Building a Bayes net

1. Choose the variables to be included in the net: \( A, B, C, D, E \)
2. Add the links
3. Add a probability table for each root node $\Pr(X)$ and nonroot node $\Pr(X_i | X_{i \in \text{Parents}(X)})$
The join probability of any assignment of values \( x_1, x_2, \ldots, x_m \) to the tuple of network variables \( X_1, X_2, \ldots, X_m \) can be computed by the formula

\[
\Pr(x_1, x_2, \ldots, x_m) = \Pr(X_1 = x_1 \land X_2 = x_2 \land \cdots \land X_m = x_m) = \prod_{i=1}^{m} \Pr(x_i|\text{Parents}(X_i))
\]
Bayesian belief networks

Two components

1. A function for evaluating a given network based on the data.
2. A method for searching through the space of possible networks.

Learning the network structure

- searching through the space of possible sets of edges
- estimating the conditional probability tables for each set
- computing the quality of the network
This 'search and score' algorithm heuristically searches for the most probable belief-network structure given a training data.

It starts by assuming that a node has no parents, after which, in every step it adds incrementally the parent whose addition mostly increase the probability of the resulting structure. K2 stops adding parents to the nodes when the addition of a single parent cannot increase the probability of the network given the data.

In general, the BBNs deal with probability propagation that consists of updating the probability values of the variables in a dependence graph, given some variables that have been observed.
K2 algorithm

INPUT
- a set of $m$ nodes (i.e., attributes)
- an ordering on the nodes $X_1, X_2, \ldots, X_m$ (parents are before their kids)
- an upper bound $u$ on the number of parents a node may have,
- training data $D$, $|D| = n$

OUTPUT
- for each node, a printout of the parent nodes

Note on the initial nodes ordering: the Naïve Bayes Classifier is a network with an edge leading from the target feature to each other features. This network can be used as a starting point for the search.
Summary of Examination Requirements

- Key points of instance-based learning – distance metric, number of neighbours, weighting function, fitting local points
- $k$-NN (weighted) algorithm
- Locally weighted linear regression
- Discriminative and generative classifiers
- Naïve Bayes Classifier – conditional independence, linear decision boundary
- Bayesian belief networks – structure, conditional probabilities