Data Mining

Chapter 6: Models and Patterns

Fall 2011

Ming Li

Department of Computer Science and Technology
Nanjing University
Models vs. Patterns

• Models

A model is a high level, global description of a data set.

• Patterns

Local characteristics of the data, perhaps holding for only a few records or a few variables.

• Type of Models & Patterns

– Descriptive: summarizing the data in a convenient and concise way

– Predictive: allowing one to make some statement about the population from which the data were drawn or about likely future values.
Modeling

• Model structures

Model structure is the functional form of the model, which contains a set of parameters.

• Parameters

Parameters is the “variable” of a special functional form, a specific parameter instantiate a concrete model.

\[ Y = f(X|\theta) = aX + b \quad \theta = \{a, b\} \]

Modeling is essential to data mining, whose goal is to determine a specific parameter of a pre-defined model structure in order to capture the relationships that you concern in data.
Predictive modeling

**Classification**
*Predicts categorical labels*

- Model the boundaries that separate the data of different labels in $d$-dim space.

**Regression**
*Predicts real-valued labels*

- Model the data points spread in $(d+1)$-dim space.

**Prediction**
*Tasks*

**Modeling**
Linear models

- **Linear model for regression**

  Linear model for regression is a \((d+1)\)-dimensional hyperplane

  \[
  \hat{Y} = f(X|\theta) = a_0 + \sum_{j=1}^{d} a_j X_j \quad \theta = \{a_0, \ldots, a_d\}
  \]

- **Linear model for classification**

  Linear model for regression is a \(d\)-dimensional hyperplane

  \[
  \hat{Y} = f(X|\theta) = \text{sign}(a_0 + \sum_{j=1}^{d} a_j X_j) \quad \theta = \{a_0, \ldots, a_d\}
  \]
Linear models

• Pros
  – Simple. Linear
  – Easy to interpret the “individual contribution” of each attribute

• Cons
  – Unable to reflex the non-linear relationship in the data which is common in many applications
  – The individual contribution makes no sense if attributes are highly correlated with each other.
From linearity to non-linearity

- Typical ways to transform linear models to non-linear models.

  - Non-linear mappings + linear models

  - “Stacks” many simple non-linear models to achieve flexible non-linearity.

  - Local Piecewise linear models

  - Memory-based local models

  - ...
Non-linear mapping-based linear model

- Extending the linear models for non-linearity by transforming the predictor variables $X_1, X_2, \ldots$ and / or the response variables

$$h(\hat{Y}) = a_0 + \sum_{j=1}^{d} a_j g_j(X_j)$$

Smooth, non-linear function

**Limitations:** the appropriate function forms of $g()$ and $h()$ have to be determined manually beforehand

- e.g., polynomial regression is a special case if $g$ is a polynomial function

$$Y = \alpha + \beta X + \chi X^2 + \delta X^3 + \ldots$$
“Stacks” many simple non-linear models

- Stacks simple non-linear models to achieve more flexible non-linearity.

**e.g., multi-layer perceptron**

✓ Each perceptron is a simple non-linear model, where $h^{-1}$ is a sigmoid function

$$h^{-1}(x) = \frac{1}{1 + e^{-x}}$$

✓ These simple non-linear models are stacked to gain arbitrary approximation of any (non-linear) function
Local piecewise models

- Combines a set of different linear models, each covering a region of the input space disjoint from the others.

  *e.g., classification / regression trees*
Local piecewise models

• Remarks
  – “Piecewise linear” is a very important idea in data mining and machine learning
  – Building simple model or patterns locally, and combines them for complex global structures.
  – Locality provides a framework for decomposing a complex model into simple local patterns.
  – **Problem**: difficult to decide the size of the region. Need to balance the approximation ability and number of data points for estimating the parameters.
“Memory-based” local models

- **A lazy approach:**
  - Retain all the data points
  - Leave the estimation of the predicted value $Y$ until the time at which a prediction is actually required.

- **Key idea:**
  Use the labels of stored examples similar to the instance to be predicted to make the prediction.

- **How to find similar instances?**
  - **Hard version:** neighbors contribute all
  - **Soft version:** everyone contribute some
“Memory-based” local models

- Hard version: $k$-nearest neighbor classification and regression.
  - Compute the distance between the current instance to all the instance stored in the data set.
  - Find $k$ nearest neighbors.
  - Use majority vote for classification and average for regression.

**Blue or Red?**

$k = 3$

$k = 5$
“Memory-based” local models

• Soft version: Kernel estimate
  – Select a single peak kernel function $K$ with a bandwidth $\sigma$
  – Evaluate the kernel function over the current instances and others stored in the data set
  – Use the weighted vote for classification and weighted average for regression, where weight is the function value
Model complexity: the more complex the better?

Model complexity

The expressive power of the model structure increases as the model becomes more complex

e.g.,

– Piecewise linear model: The number of pieces determines the expressive power

– Multi-layer perceptron: the number of hidden units determines the expressive power

– Classification and regression tree: the size of the tree determines the expressive power
Model complexity: the more complex the better?

Both simple model and complex model can fit the training data very well. The complex model may fit the training data better than the simple model.
**Overfitting**

*Overfitting*: a model describes random error or noise or the characteristics of each data point instead of the underlying relationship.

*Overfitting* occurs when the model is expressively complex (have too many freedom) in relation to the amount of data available.

- Complex model
- *Relative* small amount of data

*The overfitted model usually achieves good performance over the training data but cannot generalize well for the underlying distribution.*
Selecting appropriate model complexity

In general, design a **good score function** to indicate the performance of the predictive model over the whole distribution instead of the training data can help to achieve compromise between simplicity and complexity of the model structure.

Widely used approaches (to avoid overfitting)
- “goodness-of-fit” + “model complexity penalization”
- Cross validation
- Early stop
- Bayesian prior
- …
Descriptive modeling

- **Goals:**
  - Produces summary or description of the data

- **Model structures**
  - The model structures attempts to describe the data by modeling the distributions of the data
  - The data are *generated* from the (probabilistic) model structure
  - Such models are usually referred to as “generative model”

- **Major classes of distribution or density models**
  - Parametric
  - Non-Parametric
A mixture model is a probabilistic model for density estimation using a mixture distribution.

\[ p(x) = \sum_{k=1}^{K} p_k(x|\theta_k)\pi_k, \text{ where } \sum_{k=1}^{K} \pi_k = 1 \]

The data generative story:

1. Randomly pick a mixture component according to the probability \( \pi_k \)
2. Generate a data point \( x \) from the probability density \( p_k(x|\theta_k) \)
Mixtures of Parametric Models

• A widely used mixture model -- Gaussian Mixture models (GMM)

If $p_k(x)$ is Normal distribution, the mixture model is called Gaussian Mixture Model
Computing the joint distribution

• Joint distribution is usually difficult to compute
  – More parameters to be estimated than marginal distributions
  – Sensitive to dimensionality (especially for the nominal data)
  – Need more data to achieve reliable estimate of the parameters

Can we find an effective way to compute the joint distribution?

Factorize the joint distribution into simpler models based on some assumption of independence between variables
Factorization of density functions (I)

• **Independence assumption:**
  
  Individual variables are independent to each other.

\[
p(x) = p(x_1, x_2, ..., x_d) = \prod_{k=1}^{d} p_k(x_k)
\]

\[
x_1 \quad x_2 \quad ...... \quad x_d
\]

• **Conditional independence assumption:** (Naïve Bayes)

  Individual variables are independent to each other given a (latent) variable

\[
p(x) = \sum_{k=1}^{K} \left( \prod_{j=1}^{d} p_k(x_j | y = k) \right) p(y = k)
\]

\[
x_1 \quad x_2 \quad ...... \quad x_d
\]
**Factorization of density functions (II)**

- **Markov assumption:**

  Individual variables are dependent on the preceding variables

  \[
  p(x) = p_1(x_1) \prod_{k=2}^{d} p_k(x_k|x_1, \ldots, x_{k-1})
  \]

- **Markov Chain**

  Individual variables are only dependent on the immediately preceding variable

  \[
  p(x) = p_1(x_1) \prod_{k=2}^{d} p_k(x_k|x_{k-1})
  \]
Modeling the structure data

• In many real-world applications, the collected data have certain structures.
  – Sequential data (e.g., time series, sentences, audios, …)
  – Spatial data (e.g., maps, images, …)
  – …

• The structure information between data need to be modeled explicitly.
  – Sequential data: HMM, Conditional Random Fields, …
  – Spatial data: Markov Random Fields, …
Hidden Markov model

- **Model specification:**
  - Besides the observed random variable $Y$, there exists a hidden state random variable $X$ (discrete variable of $m$ values) at each time $t$

- **The generative story**
  1. The state of time $t$ is drawn according to $p(x_t|x_{t-1})$.
  2. The observation $y_t$ is determined according to the distribution $p(y_t|x_t)$.

$$p(y_1, \ldots, y_T, x_1, \ldots, x_T) = p(x_1)p(y_1|x_1) \prod_{t=2}^{d} p(y_t|x_t)p(x_t|x_{t-1})$$

![Diagram of Hidden Markov Model]
Pattern structures

• Unlike the model structure describing the whole data set, pattern structure aims to characterize some local aspect of the data.

• Two important things for pattern structure
  – Syntax of the pattern: how to describe the pattern
  – Semantics of the pattern: the information the pattern structures convey
Patterns for data matrices

• **Syntax**
  
  – **Primitive patterns:** \( X_k = c \)
  
  – **Complex patterns** using logical connections:
    
    \[ X_k = 1 \land X_j = 1 \]
  
  – **Pattern class:** a set of legal patterns, which consists of all primitive patterns and legal combination of primitives using logical connections.

  – **Form of the pattern structures**
    
    \[ \rho \rightarrow \phi \]
Patterns for data matrices

- **Pattern discovery:**
  - Find all patterns from the class that satisfy certain conditions with respect to the data set
  - Informativeness, novelty and understandability should be considered in the discovery process.

- **Association rule:** Accuracy and frequency measure should be associated with the rule to describe how accurate of the rule

\[
\{A_1, \ldots, A_k\} \Rightarrow \{B_1, \ldots, B_h\} \quad (p_s, p_c)
\]

- **Functional dependency:** the rule should be accurate

\[
A_1, \ldots, A_{k-1} \Rightarrow A_k
\]
Let’s move to Chapter 7