Machine Learning

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square, x4-8845

http://www.cs.pitt.edu/~milos/courses/cs2750/

Administration

Instructor:
Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square, x4-8845

TA:
Tomas Singliar
tomas@cs.pitt.edu
5802 Sennott Square, x4-8832

Office hours: TBA
Administration

Study material
• **Handouts, your notes and course readings**
• **Primary textbook:**
• **Recommended book:**
• **Other books:**

CS 2750 Machine Learning

Administration

• **Lectures:**
  – **Random** short quizzes testing the understanding of basic concepts from previous lectures
• **Homeworks: weekly**
  – **Programming tool:** Matlab (CSSD machines and labs)
  – **Matlab Tutorial:** next week
• **Exams:**
  – **Midterm** (March)
• **Final project:**
  – **Proposals** (March)
  – **Written report + Oral presentation**
    (end of the semester)
Tentative topics

- Learning.
- Density estimation.
- Linear models for regression and classification.
- Learning Bayesian networks.
- Clustering. Latent variable models.
- Dimensionality reduction. Feature extraction.
- Hidden Markov models.
- Reinforcement learning

Machine Learning

- The field of **machine learning** studies the design of computer programs (agents) capable of learning from past experience or adapting to changes in the environment

- The need for building agents capable of learning is everywhere
  - predictions in medicine,
  - text and web page classification,
  - speech recognition,
  - image/text retrieval,
  - commercial software
Learning

Learning process:
Learner (a computer program) processes data $D$ representing past experiences and tries to either develop an appropriate response to future data, or describe in some meaningful way the data seen.

Example:
Learner sees a set of patient cases (patient records) with corresponding diagnoses. It can either try:
- to predict the presence of a disease for future patients
- describe the dependencies between diseases, symptoms

Types of learning

- **Supervised learning**
  - Learning mapping between input $x$ and desired output $y$
  - Teacher gives me $y$’s for the learning purposes
- **Unsupervised learning**
  - Learning relations between data components
  - No specific outputs given by a teacher
- **Reinforcement learning**
  - Learning mapping between input $x$ and desired output $y$
  - Critic does not give me $y$’s but instead a signal (reinforcement) of how good my answer was
- **Other types of learning:**
  - Concept learning, explanation-based learning, etc.
Supervised learning

Data: \[ D = \{d_1, d_2, \ldots, d_n\} \] a set of \( n \) examples
\[ d_i = \langle x_i, y_i \rangle \]
x\(_i\) is input vector, and \( y \) is desired output (given by a teacher)

Objective: learn the mapping \( f: X \rightarrow Y \)
\[ \text{s.t. } y_i \approx f(x_i) \text{ for all } i = 1, \ldots, n \]

Two types of problems:
- **Regression:** X discrete or continuous \( \rightarrow \)
  Y is continuous
- **Classification:** X discrete or continuous \( \rightarrow \)
  Y is discrete

Supervised learning examples

- **Regression:** Y is continuous
  
  Debt/equity
  Earnings
  Future product orders \( \rightarrow \) company stock price

- **Classification:** Y is discrete
  
  Handwritten digit (array of 0,1s) \( \rightarrow \) Label “3”
Unsupervised learning

- **Data:** \( D = \{d_1, d_2, \ldots, d_n\} \)
  \( d_i = x_i \) vector of values
  No target value (output) \( y \)

- **Objective:**
  - learn relations between samples, components of samples

Types of problems:
- **Clustering**
  Group together “similar” examples, e.g. patient cases
- **Density estimation**
  - Model probabilistically the population of samples

Unsupervised learning example.

- **Density estimation.** We want to build the probability model of
  a population from which we draw samples \( d_i = x_i \)
Unsupervised learning. Density estimation

- A probability density of a point in the two dimensional space
  - Model used here: **Mixture of Gaussians**

Reinforcement learning

- We want to learn: \( f : X \to Y \)
- We see samples of \( x \) but not \( y \)
- Instead of \( y \) we get a feedback (reinforcement) from a **critic** about how good our output was

- The goal is to select outputs that lead to the best reinforcement
Learning

• Assume we see examples of pairs \((x, y)\) and we want to learn the mapping \(f : X \rightarrow Y\) to predict future \(y\)s for values of \(x\)
• We get the data what should we do?

Learning bias

• Problem: many possible functions \(f : X \rightarrow Y\) exists for representing the mapping between \(x\) and \(y\)
• Which one to choose? Many examples still unseen!
Learning bias

- Problem is easier when we make an assumption about the model, say, \( f(x) = ax + b + \varepsilon \)
  \[ \varepsilon = N(0, \sigma) \] - random (normally distributed) noise
- Restriction to a linear model is an example of learning bias

\[
\begin{array}{c}
  \text{y} \\
  \text{x}
\end{array}
\]

Learning bias

- **Bias** provides the learner with some basis for choosing among possible representations of the function.
- **Forms of bias:** constraints, restrictions, model preferences
- **Important:** There is no learning without a bias!

\[
\begin{array}{c}
  \text{y} \\
  \text{x}
\end{array}
\]
Learning bias

- Choosing a parametric model or a set of models is not enough
  Still too many functions \( f(x) = ax + b + \varepsilon \quad \varepsilon = N(0, \sigma) \)
  - One for every pair of parameters \( a, b \)

Fitting the data to the model

- We are interested in finding the best set of model parameters

**Objective:** Find the set of parameters that:
- reduces the misfit between the model and observed data
- Or, (in other words) that explain the data the best

**Error function:**

**Measure of misfit between the data and the model**

**Examples of error functions:**
- Average square error \( \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 \)
- Average misclassification error \( \frac{1}{n} \sum_{i=1}^{n} 1_{y_i \neq f(x_i)} \)

Average # of misclassified cases
Fitting the data to the model

• **Linear regression**
  - Least squares fit with the linear model
  - minimizes \( \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 \)

![Graph showing linear regression](image)

Typical learning

**Three basic steps:**

• **Select a model** or a set of models (with parameters)
  
  E.g. \( y = ax + b \)

• **Select the error function** to be optimized
  
  E.g. \( \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 \)

• **Find the set of parameters optimizing the error function**
  
  – The model and parameters with the smallest error represent the best fit of the model to the data

But there are problems one must be careful about …
Learning

Problem
• We fit the model based on past experience (past examples seen)
• But ultimately we are interested in learning the mapping that performs well on the whole population of examples

Training data: Data used to fit the parameters of the model

Training error: \[ \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 \]

True (generalization) error (over the whole unknown population):
\[ E_{(x,y)}[(y - f(x))^2] \] Mean squared error

Training error tries to approximate the true error !!!!
Does a good training error imply a good generalization error ?