CS 2750 Machine Learning

Lecture 1

Machine Learning

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

people.cs.pitt.edu/~milos/courses/cs2750/

Administration

Instructor:

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

TA:

Yanbing Xue
yax14@cs.pitt.edu
5324 Sennott Square, x4-8455

Office hours: TBA

Who am I?

- **Milos Hauskrecht –Professor of Computer Science**
- **Secondary affiliations:**
  - Intelligent Systems Program (ISP),
  - Department of Biomedical Informatics (DBMI)

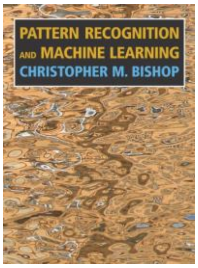
- **Research work:**
  - Machine learning, Data mining, Outlier detection, Probabilistic modeling, Time-series models and analysis

  Applications to healthcare:
  - EHR data analysis, Patient monitoring and alerting, Patient safety

Administration

Study material

- **Handouts, your notes and course readings**
- **Primary textbook:**



Chris. Bishop. Pattern Recognition and Machine Learning. Springer, 2006.

- **Additional text/readings**:

  Hal Daume III. A Course in Machine Learning. http://ciml.info

Administration

Study material

- **Other ML books:**
  - K. Murphy. Machine Learning: A probabilistic perspective, MIT Press, 2012.
  - J. Han, M. Kamber. Data Mining. Morgan Kauffman, 2011.
  - Friedman, Hastie, Tibshirani. Elements of statistical learning. Springer, 2nd edition, 2011.
  - Koller, Friedman. Probabilistic graphical models. MIT Press, 2009.
  - Duda, Hart, Stork. Pattern classification. 2nd edition. J Wiley and Sons, 2000.
  - T. Mitchell. Machine Learning. McGraw Hill, 1997.

Administration

- **Homework assignments (40%): weekly**
  - **Programming tool**: Matlab (free license, CSSD machines and labs)
  - **Matlab Tutorial:** recitations
- **Exams (35%):**
  - **Midterm** – March 7, 2019
  - **Final exam**– week of April 15 -19, 2019
- **Term projects (15%)**
  - **Presentations scheduled for final exam week**
- **Lectures (10%):**
  - **Attendance and Activity**
  - **Short quizzes**

Tentative topics

- **Introduction to ML**
  - Basic concepts
- **Density estimation:**
  - Basic parametric distribution
  - Exponential family distributions
  - Non-parametric density estimation methods
- **Supervised Learning**
  - Linear models for regression and classification.
  - Multi-layer neural networks.
  - Support vector machines. Kernel methods.
  - Decision trees
  - Non-parametric classification models

Tentative topics (cont)

- **Probabilistic graphical models**
  - Bayesian belief networks
  - Learning, expectation-maximization
- **Clustering:**
  - K-means clustering
  - Hierarchical clustering
- **Ensemble methods**
  - Mixture models
  - Bagging and boosting
- **Dimensionality reduction**
  - Feature selection
  - Principal component analysis (PCA)
- **Reinforcement learning**

Machine Learning

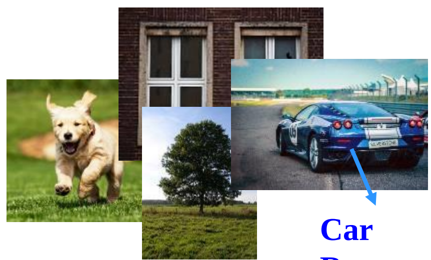
- The field of **machine learning** studies the design of computer programs (agents) capable of learning from past experience or adapting to changes in the environment

- The need for building agents capable of learning is everywhere.

  Examples:
  - text, web-page classification
  - web search
  - speech recognition

Machine Learning

- **Examples (cont.):**
  - Image/video classification, annotation and retrieval
  - adaptive interfaces
  - Route optimization
  - Game playing



Car
Race car

Learning process

  **Model of data**

Learner (a computer program) processes data ***D*** representing past experiences and tries to build a **model** that either:

– Generates appropriate response to future data, or

– Describes 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 occurrence of a disease for future patients

– describe the dependencies between diseases, symptoms

Types of learning problems

- **Supervised learning**
  - Takes data that consists of pairs (**x**,**y**)
  - Learns mapping ***f*:** **x** (input) → **y** (output, response)
- **Unsupervised learning**
  - Takes data that consist of vectors **x**
    - Learns relations **x** among vector components
    - Groups/clusters data into the groups
- **Reinforcement learning**
  - Learns mapping ***f*:** **x** (input) → **y** (desired output)
  - From (**x**,**y**,r) triplets where **x** is an input, **y** is a response chosen by the user/system, and r is a reinforcement signal
  - **Online:** see x, choose y and observe r
- **Other types of learning: Active learning, Transfer learning, Deep learning**

Supervised learning

Data: $D = \{d_1, d_2, .., d_n\}$ **a set of *n* examples**

$d_i = < \mathbf{x}_i, y_i >$

$\mathbf{x}_i$ is input vector, and *y* is desired output (given by a teacher)

Objective: learn the mapping $f : X \to Y$

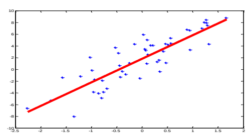
s.t. $y_i \approx f(x_i)$ for all $i = 1, .., n$

Two types of problems:

- **Regression:** X discrete or continuous $\to$
  Y is **continuous**
- **Classification:** X discrete or continuous $\to$
  Y is **discrete**

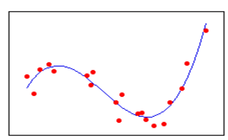
Supervised learning examples

- **Regression:** Y is **continuous**



Debt/equity
Earnings
Future product orders

$\longrightarrow$ Stock price



Data:

Debt/equity	Earnings	Future prod orders	Stock price
20	115	20	123.45
18	120	31	140.56
….			

Supervised learning examples

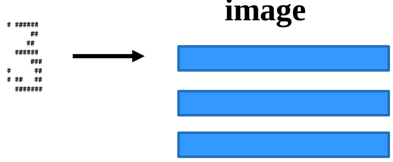
- **Classification:** Y is **discrete**

```
# ######
      ##
     ##
  ######
      ###
#      ##
# ##   ##
  #######
```

→ **Label "3"**

Handwritten digit (array of 0,1s)

Data: **image** **digit**



3
7
5
....

Unsupervised learning

- **Data:** $D = \{d_1, d_2, .., d_n\}$
  $d_i = \mathbf{x}_i$ vector of values
  No target value (output) y

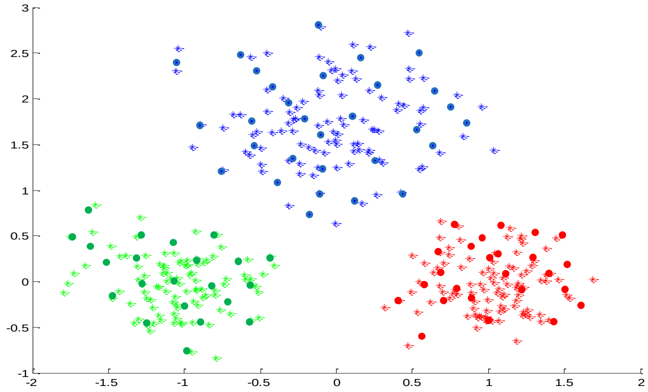
- **Objective:**
  – learn relations between samples, components of samples

Types of problems:

- **Clustering**
  Group together "similar" examples, e.g. patient cases
- **Density estimation**
  – Model probabilistically the population of samples

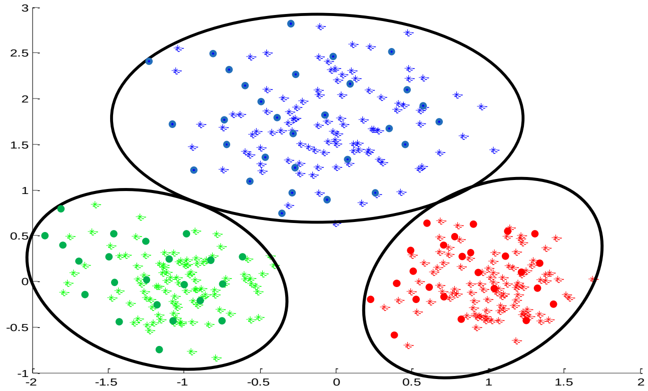
Unsupervised learning example

- **Clustering.** Group together similar examples $d_i = \mathbf{x}_i$



Unsupervised learning example

- **Clustering.** Group together similar examples $d_i = \mathbf{x}_i$

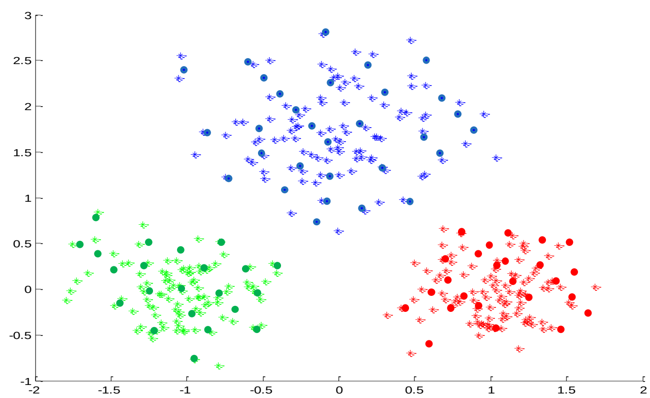


Unsupervised learning example

- **Clustering.** Group together similar examples

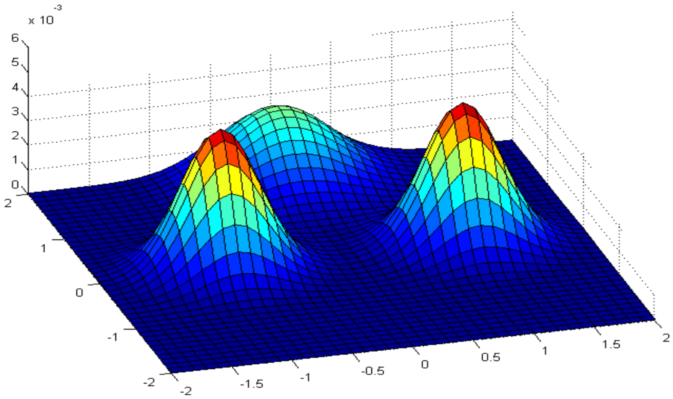
Unsupervised learning example

- **Density estimation.** We want to build a probability model P(**x**) of a population from which we drew examples $d_i = \mathbf{x}_i$



Unsupervised learning. Density estimation

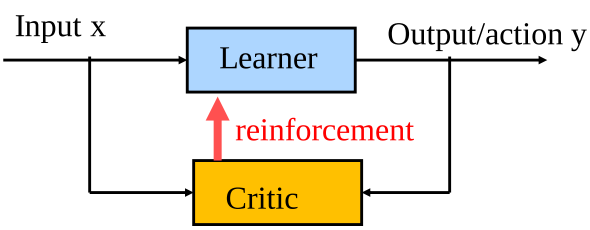
- A probability density of a point in the two dimensional space
  - Model used here: **Mixture of Gaussians**



Reinforcement learning

We want to learn: $f : X \rightarrow Y$

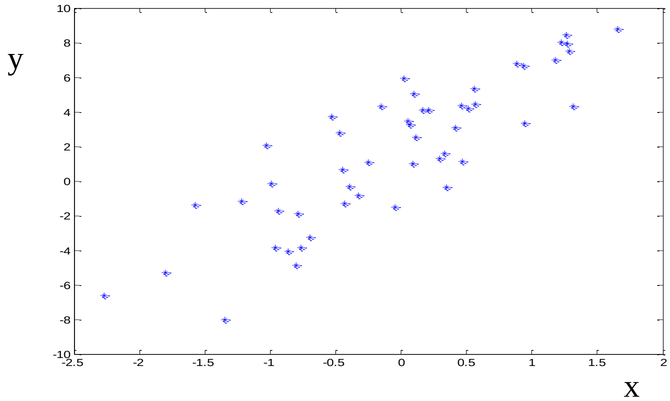
- We see examples of inputs **x** but not *y*
- We select y for observed x from available choices
- We get a feedback (reinforcement) from a **critic** about how good our choice of y was



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

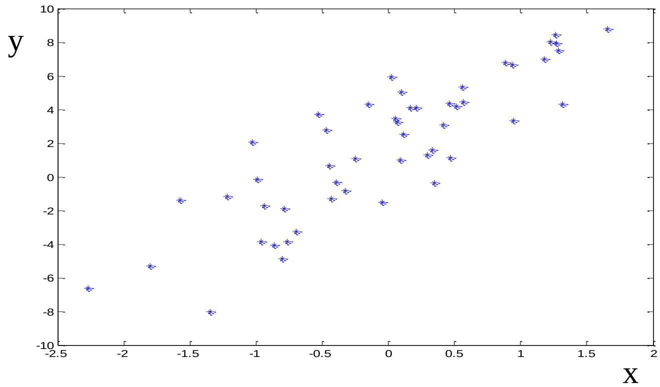
Learning: first look

- Assume we see examples of pairs (**x** , *y*) in *D* and we want to learn the mapping $f : X \to Y$ to predict y for some future **x**
- We get the data *D* - what should we do?



Learning: first look

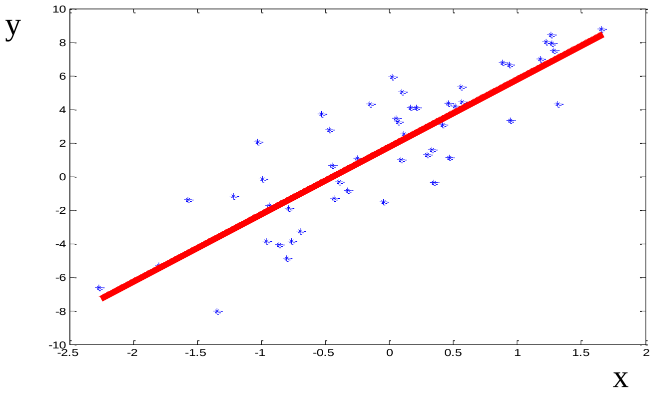
- **Problem:** many possible functions $f : X \to Y$ exists for representing the mapping between **x** and y
- Which one to choose? Many examples still unseen!



Learning: first look

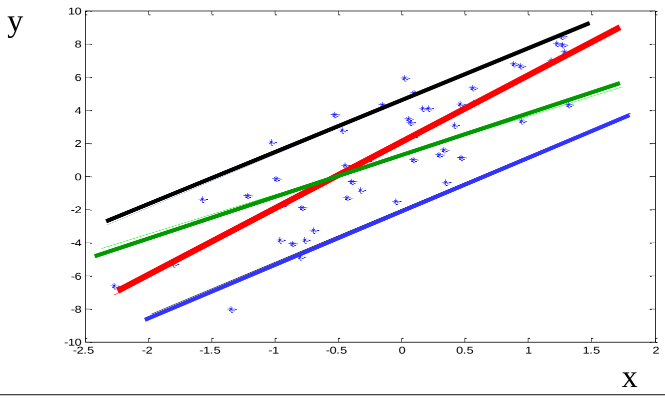
- **Solution:** make an assumption about the model, say,

$$f(x) = ax + b$$



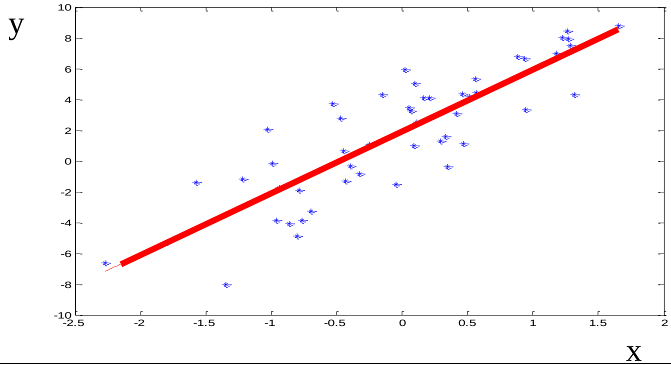
Learning: first look

- Choosing a parametric model or a set of models is not enough
  Still too many functions $f(x) = ax + b$
  - One for every pair of parameters *a*, *b*



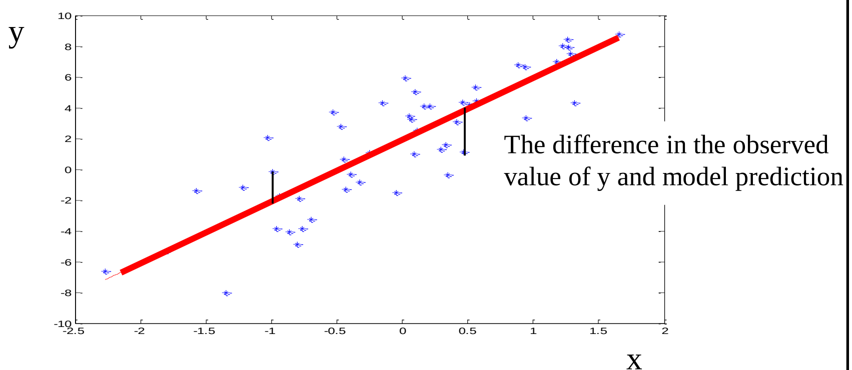
Learning: first look

- We want the **best set** of model parameters
  - reduce the misfit between the model **M** and observed data ***D***
  - Or, (in other words) explain the data the best
- **How to measure the misfit?**



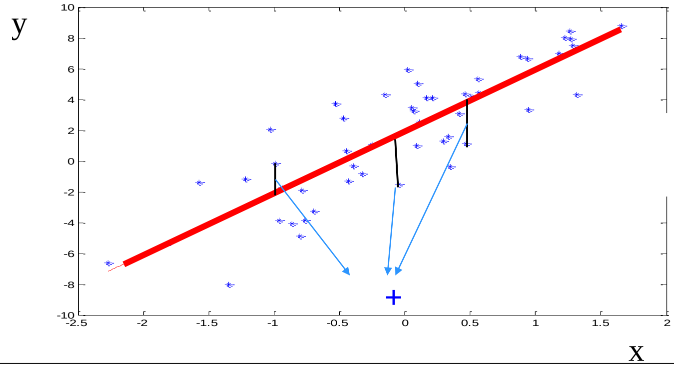
Learning: first look

- We want the **best set** of model parameters
  - reduce the misfit between the model **M** and observed data ***D***
  - Or, (in other words) explain the data the best
- **How to measure the misfit?**



Learning: first look

- We want the **best set** of model parameters
  - reduce the misfit between the model **M** and observed data ***D***
  - Or, (in other words) explain the data the best
- **How to measure the misfit?**



Learning: first look

- We want the **best set** of model parameters
  - reduce the misfit between the model **M** and observed data ***D***
  - Or, (in other words) explain the data the best
- **How to measure the misfit?**

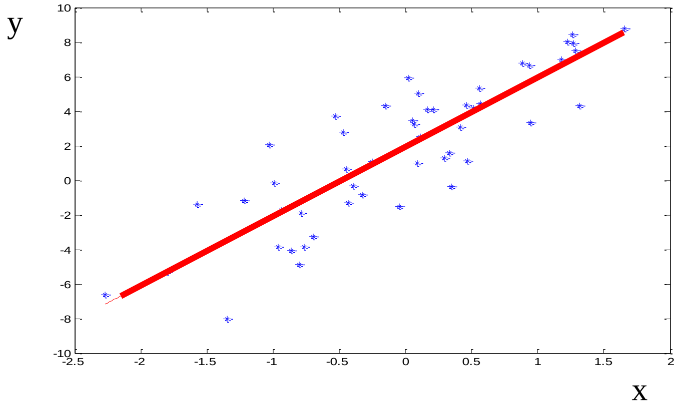
Objective function:

- **Error (loss) function: Measures the misfit between *D* and M**
- **Examples of error functions:**
  - Average Square Error $$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2$$
  - Average Absolute Error $$\frac{1}{n}\sum_{i=1}^{n}|y_i - f(x_i)|$$

Learning: first look

- **Linear regression**
- Minimizes the squared error function for the linear model

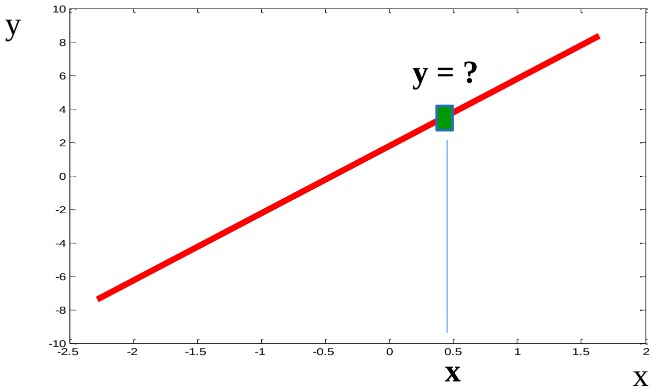
$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2$$



Learning: first look

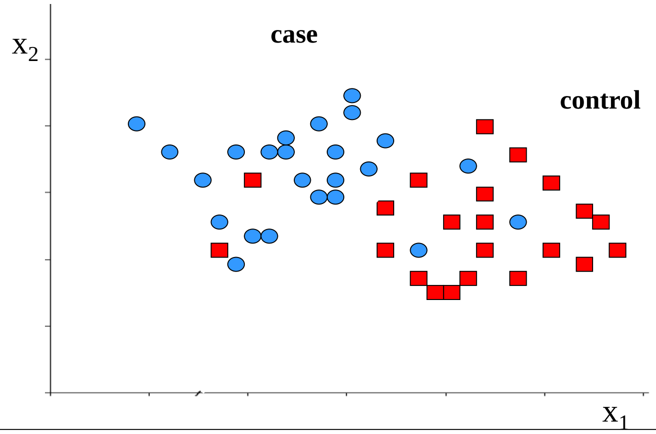
- **Application:** A new example **x** with unknown value y is checked against the model, and y is calculated

$$y = f(x) = ax + b$$



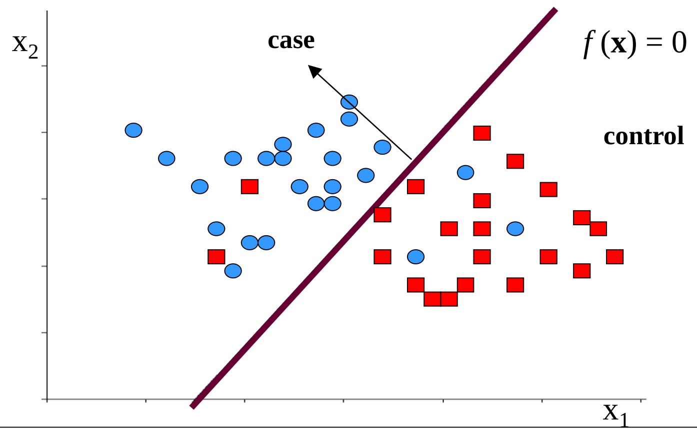
Supervised learning: Classification

- **Data** D: pairs (**x** , *y*) where y is a class label:
  y examples: patient will be readmitted (case) or no (control),
  patient has disease (case) or no (control)



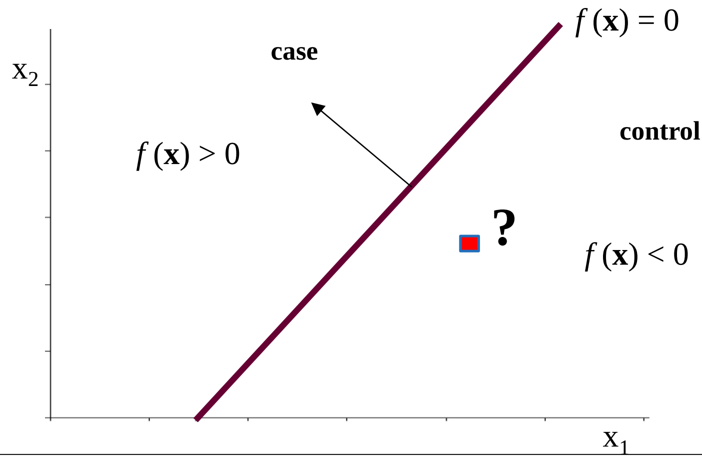
Supervised learning: Classification

- Find a model *f*: X → R, say $f(x) = ax_1 + bx_2 + c$ that defines a decision boundary *f* (**x**) = 0 that separates well the two classes
  - **Note that some examples may not be correctly classified**



Supervised learning: Classification

- A new example x with unknown class label is checked against the model, the class label is assigned



Learning: first look

1. Data: $D = \{d_1, d_2, .., d_n\}$

2. Model selection:

- **Select a model** or a set of models (with parameters)

  E.g. $y = ax + b$

3. Choose the objective function (loss function)

- **Squared error** $\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2$

4. Learning:

- **Find the set of parameters optimizing the error function**
  - The model and parameters with the smallest error

5. Application

- **Apply the learned model to new data**
- E.g. predict ys for new inputs **x** using learned $f(\mathbf{x})$

Learning: first look

1. Data: $D=\{d_1,d_2,..,d_n\}$

2. Model selecti

– **Selec**

E.g.

3. Choose the

– **Squared e**

4. Learning:

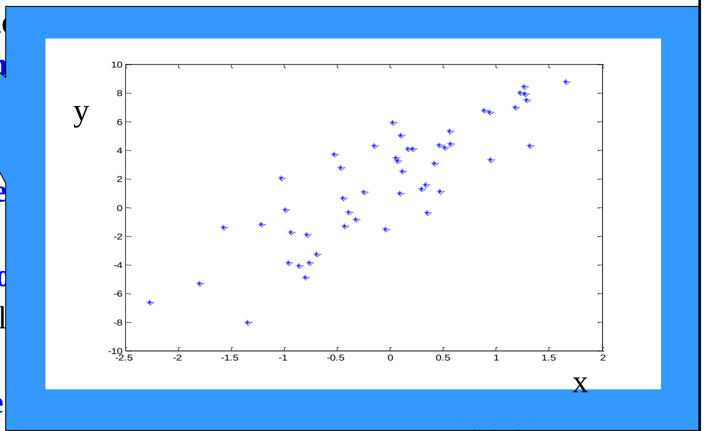
• **Find the set**

– The model

5. Application:

– **Apply the**

– E.g. predict ys for new inputs **x** using learned $f(\mathbf{x})$



A learning system: basic cycle

1. Data: $D=\{d_1,d_2,..,d_n\}$

2. Model selection:

– **Select a model** or a set of models (with parameters)

E.g. $y=ax+b$

3. Choose the

– **Squa**

4. Learning:

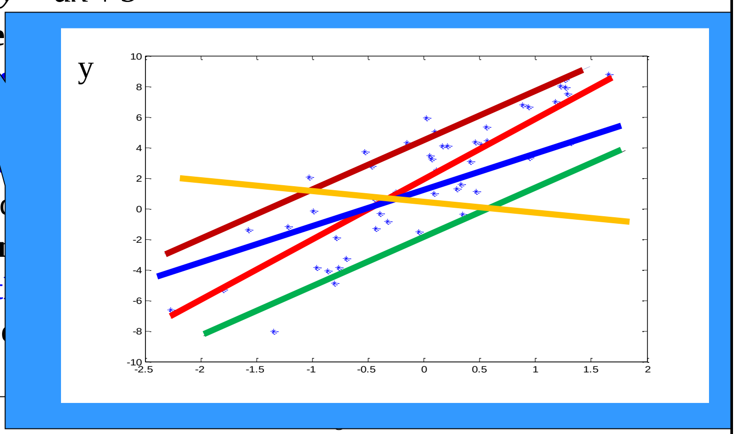
• **Find the se**

– The mod

5. Applicatio

– **Apply t**

– E.g. pre



Learning: first look

1. Data: $D = \{d_1, d_2, .., d_n\}$

2. Model selection:

- **Select a model** or a set of models (with parameters)

  E.g. $y = ax + b$

3. Choose the objective function (loss function)

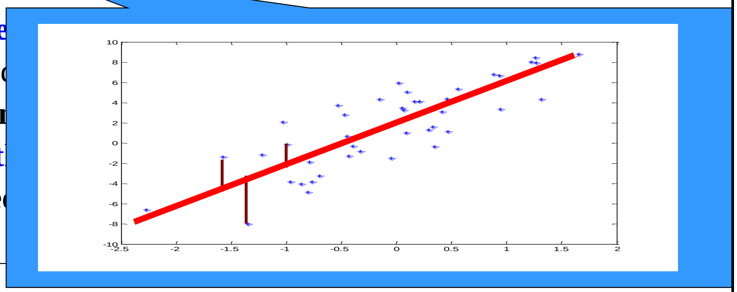
- **Squared error** $\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2$

4. Learning:

- **Find the se**
  - The mo

5. Applicatio

- **Apply t**
- E.g. pre



Learning: first look

1. Data: $D =$

2. Model sele

- **Select a**

  E.g. y

3. Choose the

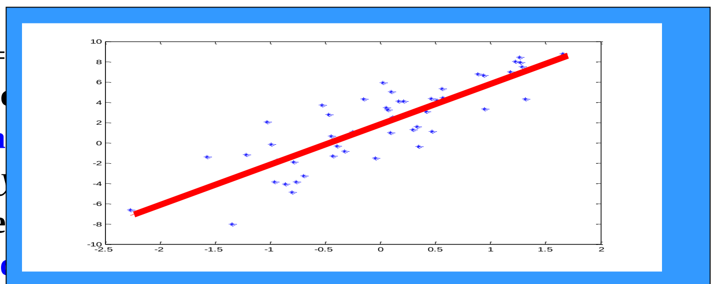
- **Square**

4. Learning:

- **Find the set of parameters optimizing the error function**
  - The model and parameters with the smallest error

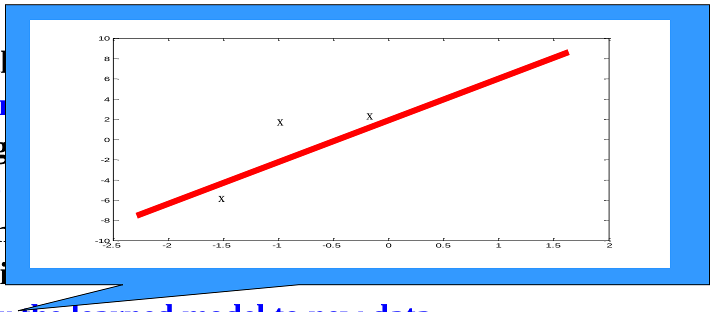
5. Application:

- **Apply the learned model to new data**
- E.g. predict ys for new inputs **x** using learned $f(\mathbf{x})$



Learning: first look

1. **Data:** $D = \{d_1, d_2, .., d_n\}$
2. **Model selection:**
   – **Select a model** or a set of models (with parameters)
     E.g.
3. **Choose t**
   – **Squar**
4. **Learning**
• **Find the**
   – The m
5. **Applicati**



   – **Apply the learned model to new data**
   – E.g. predict ys for new inputs **x** using learned $f(\mathbf{x})$