CS184A/284A
AI in Biology and Medicine

Intro to Machine Learning, Data Visualization and Exploration

Machine Learning

Introduction to Machine Learning

Course Logistics

Data and Visualization

Supervised Learning

Artificial Intelligence (AI)

- Building "intelligent systems"
- Lots of parts to intelligent behavior

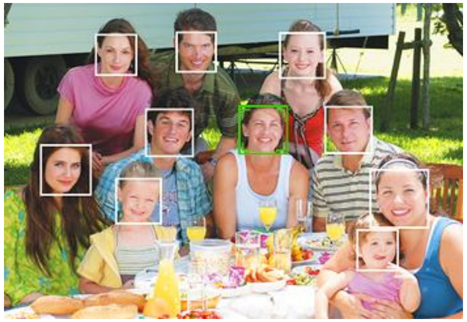
Darpa GC
(Stanley)

RoboCup

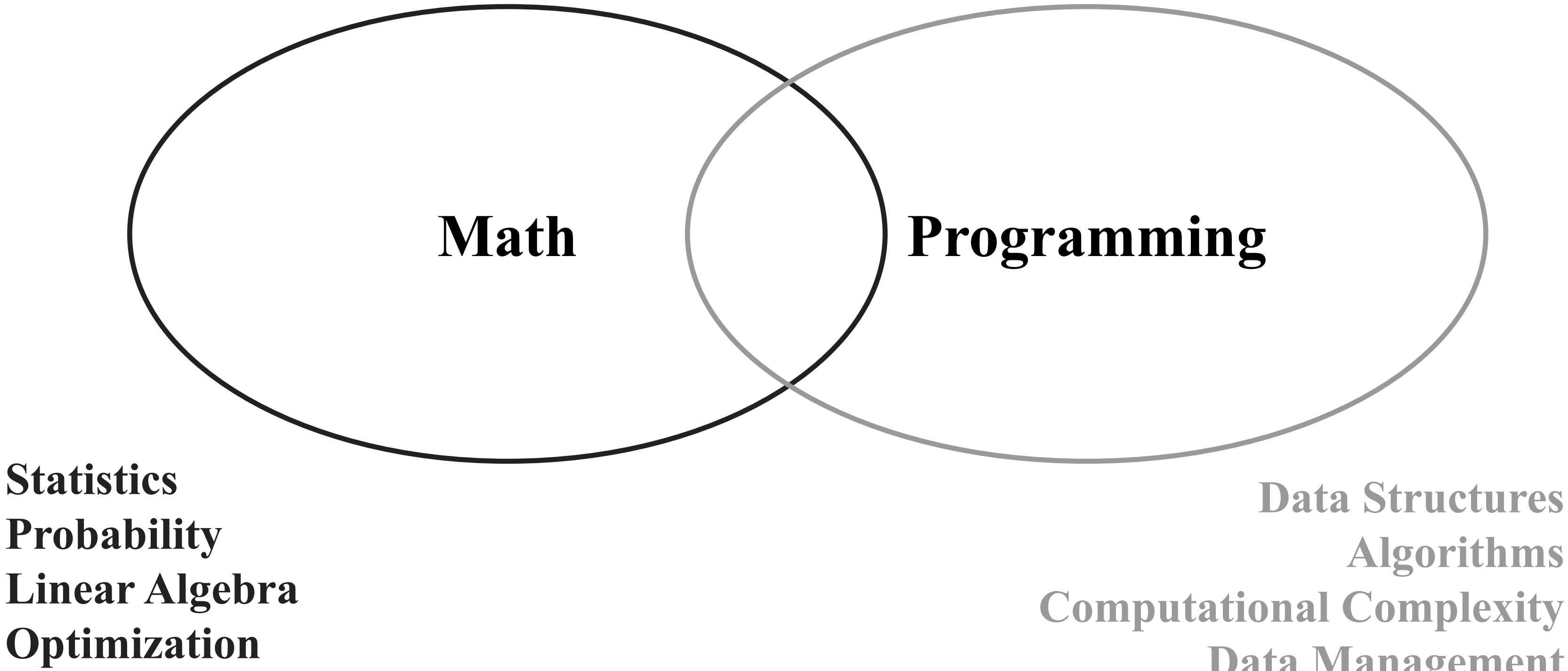
Chess (Deep Blue v. Kasparov)

Machine learning (ML)

- One (important) part of AI
- Making predictions (or decisions)
- Getting better with experience (data)
- Problems whose solutions are “hard to describe”



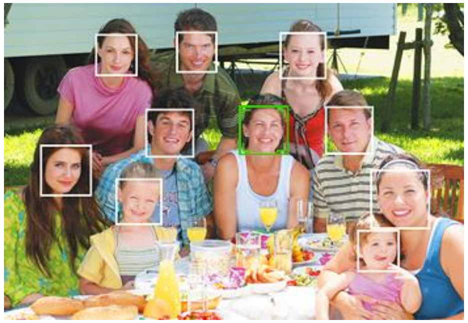
Machine Learning



Types of prediction problems

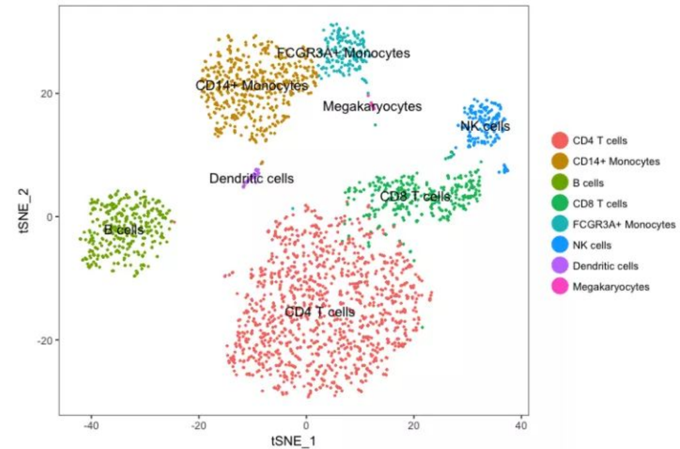
- Supervised learning
  - “Labeled” training data
  - Every example has a desired target value (a “best answer”)
  - Reward prediction being close to target

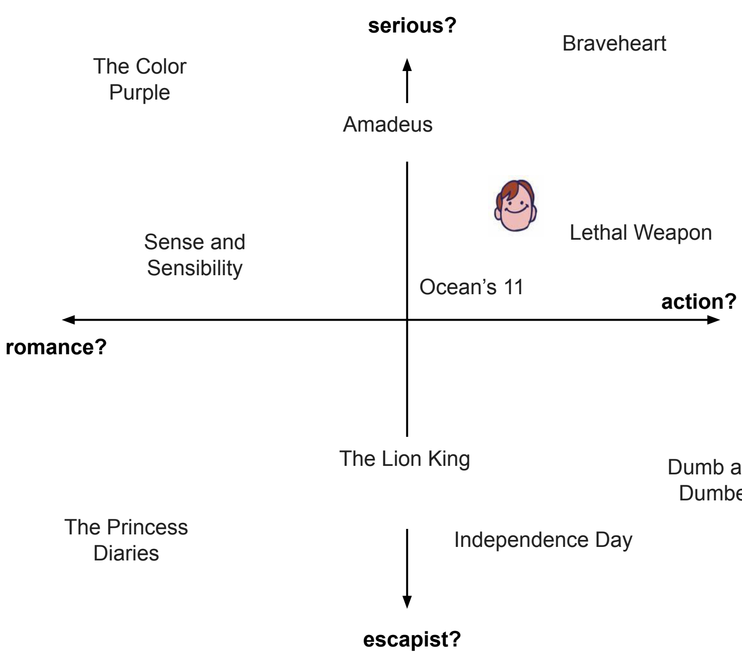
  - **Classification:** a discrete-valued prediction (often: action / decision)
  - **Regression:** a continuous-valued prediction

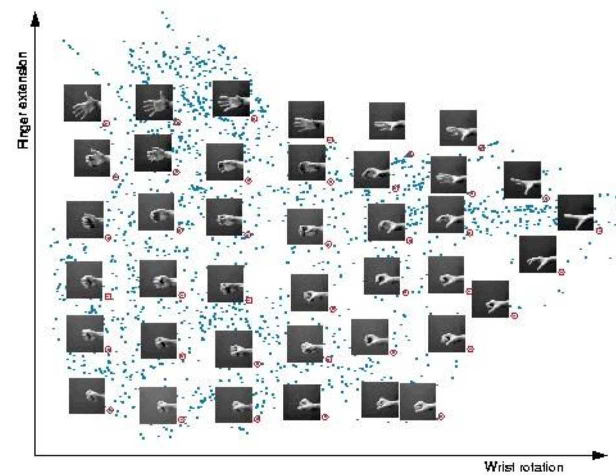


Types of prediction problems

- Supervised learning
- Unsupervised learning
  - No known target values
  - No targets = nothing to predict?
  - Reward “patterns” or “explaining features”
  - Often, data mining







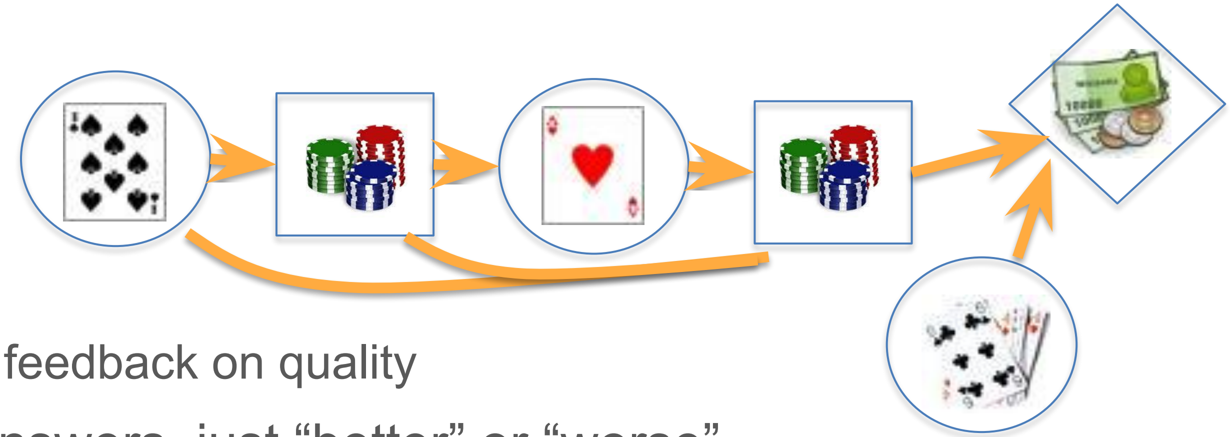
Types of prediction problems

- Supervised learning
- Unsupervised learning
- Semi-supervised learning
  - Similar to supervised
  - some data have unknown target values

- Ex: medical data
  - Lots of patient data, few known outcomes
- Ex: image tagging
  - Lots of images on Flikr, but only some of them tagged

Types of prediction problems

- Supervised learning
- Unsupervised learning
- Semi-supervised learning
- Reinforcement learning



- "Indirect" feedback on quality
  - No answers, just "better" or "worse"
  - Feedback may be delayed

Summary

What is machine learning?

- Computer science + Math (Optimization & Statistics)
- How do we learn from data to improve performance

Types of machine learning

- Supervised learning
- Unsupervised learning
- Semi-supervised learning
- Reinforcement learning

Machine Learning

Introduction to Machine Learning

Course Logistics

Data and Visualization

Supervised Learning

Data exploration

- Machine learning is a data science
  - Look at the data; get a “feel” for what might work

- What types of data do we have?
  - Binary values? (spam; gender; …)
  - Categories? (home state; labels; …)
  - Integer values? (1..5 stars; age brackets; …)
  - (nearly) real values? (pixel intensity; prices; …)

- Are there missing data?

- “Shape” of the data? Outliers?

Scientific software

- **Python**
  - **Numpy, MatPlotLib, SciPy…**
- Matlab
  - Octave (free)
- R
  - Used mainly in statistics
- C++
  - For performance, not prototyping
- And other, more specialized languages for modeling…

Representing data

- Example: Fisher's "Iris" data
  http://en.wikipedia.org/wiki/Iris_flower_data_set

- Three different types of iris
  - "Class", y

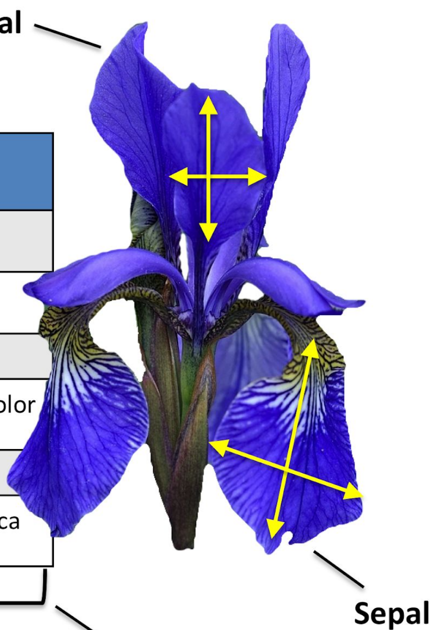
- Four "features", $x_1,\ldots,x_4$
  - Length & width of sepals & petals

- 150 examples (data points)

Intro to Basic Terminology and Notations

Samples
(instances, observations)

	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
			...		
50	6.4	3.5	4.5	1.2	Versicolor
			...		
150	5.9	3.0	5.0	1.8	Virginica



Features
(attributes, measurements, dimensions)

Class labels
(targets)

Representing the data in Python

- Have m observations (data points)

$$\left\{x^{(1)} \dots, x^{(m)}\right\}$$

- Each observation is a vector consisting of n features

$$x^{(j)} = [x_1^{(j)} x_2^{(j)} \dots x_n^{(j)}]$$

- Often, represent this as a “data matrix”

$$\underline{X} = \begin{bmatrix} x_1^{(1)} & \dots & x_n^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(m)} & \dots & x_n^{(m)} \end{bmatrix}$$

```
import numpy as np        #import numpy
iris = np.genfromtxt("data/iris.txt",delimiter=None)
X = iris[:,0:4]    # load data and split into features, targets
Y = iris[:,4]
print(X.shape)  # 150 data points; 4 features each
   (150, 4)
```

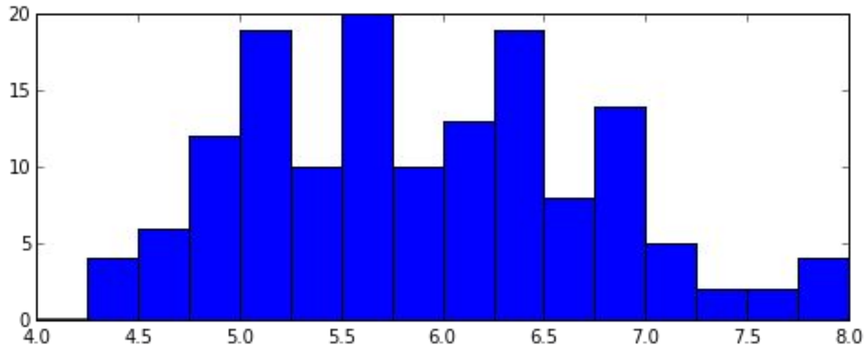
Basic statistics

- Look at basic information about features
  - Average value? (mean, median, etc.)
  - “Spread”?   (standard deviation, etc.)
  - Maximum / Minimum values?

```
print(np.mean(X, axis=0))       # compute mean of each feature
  [ 5.8433    3.0573    3.7580    1.1993 ]
print(np.std(X, axis=0))     #compute standard deviation of each feature
  [ 0.8281    0.4359    1.7653    0.7622 ]
print(np.max(X, axis=0))               # largest value per feature
  [  7.9411    4.3632    6.8606    2.5236 ]
print(np.min(X, axis=0))          # smallest value per feature
  [  4.2985    1.9708    1.0331    0.0536  ]
```

Histograms

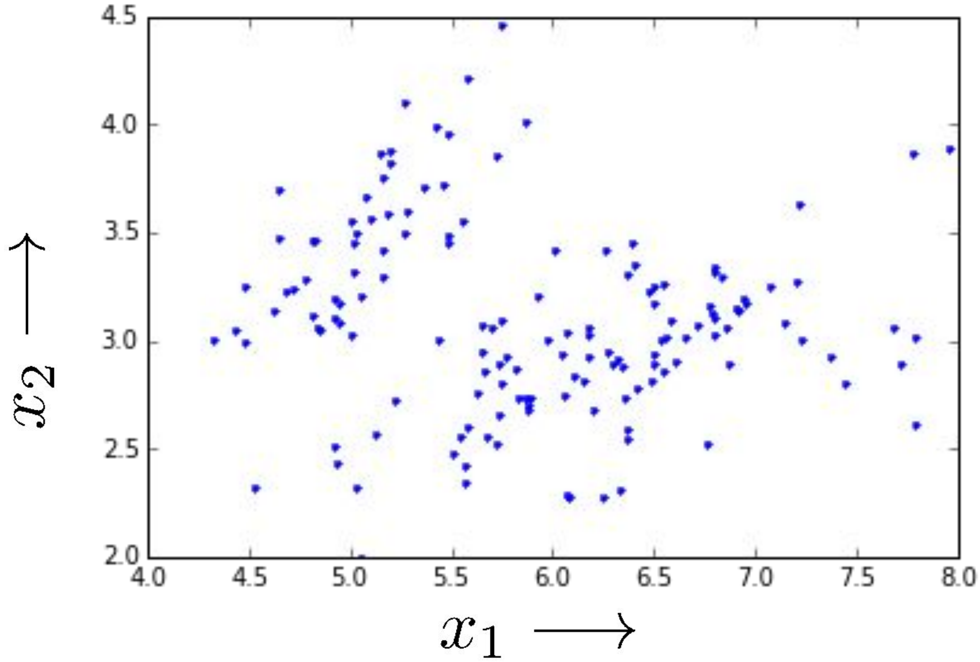
- Count the data falling in each of K bins
  - "Summarize" data as a length-K vector of counts (& plot)
  - Value of K determines "summarization"; depends on # of data
    - K too big: every data point falls in its own bin; just "memorizes"
    - K too small: all data in one or two bins; oversimplifies




```
% Histograms in MatPlotLib
import matplotlib.pyplot as plt
X1 = X[:,0]                 # extract first feature
Bins = np.linspace(4,8,17)  # use explicit bin locations
plt.hist( X1, bins=Bins )   # generate the plot
```

Scatterplots

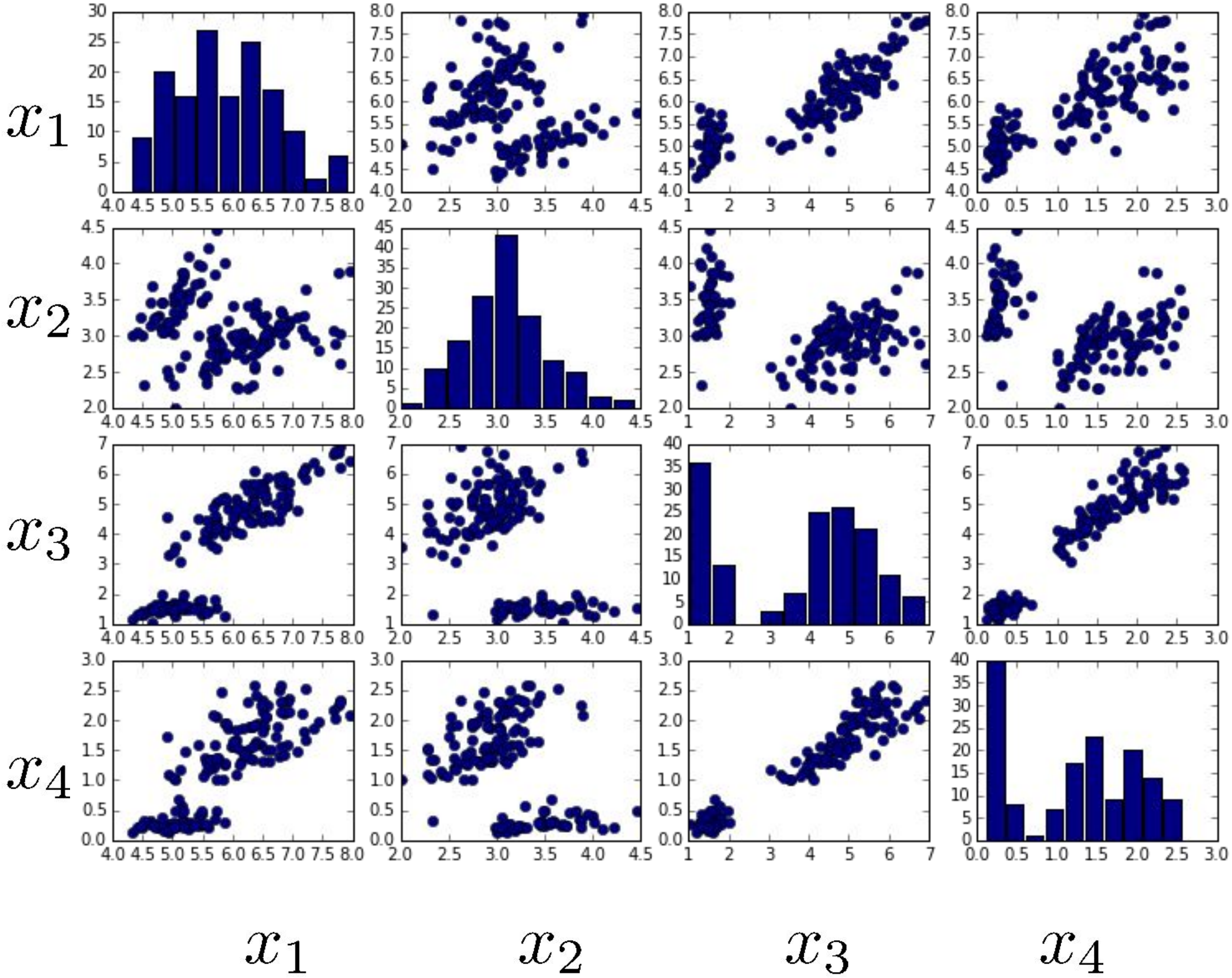
- Illustrate the relationship between two features




```
% Plotting in MatPlotLib
plt.plot(X[:,0], X[:,1], 'b.');   % plot data points as blue dots
```

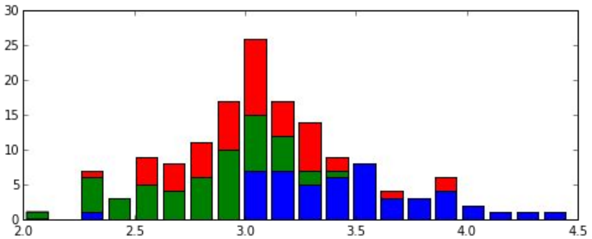
Scatterplots

- For more than two features we can use a pair plot:

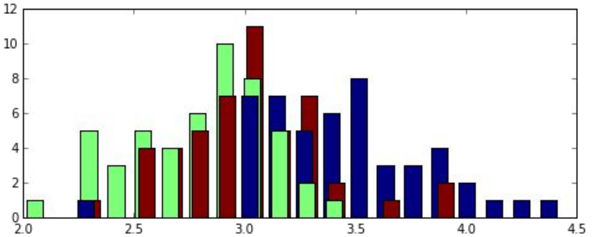


Supervised learning and targets

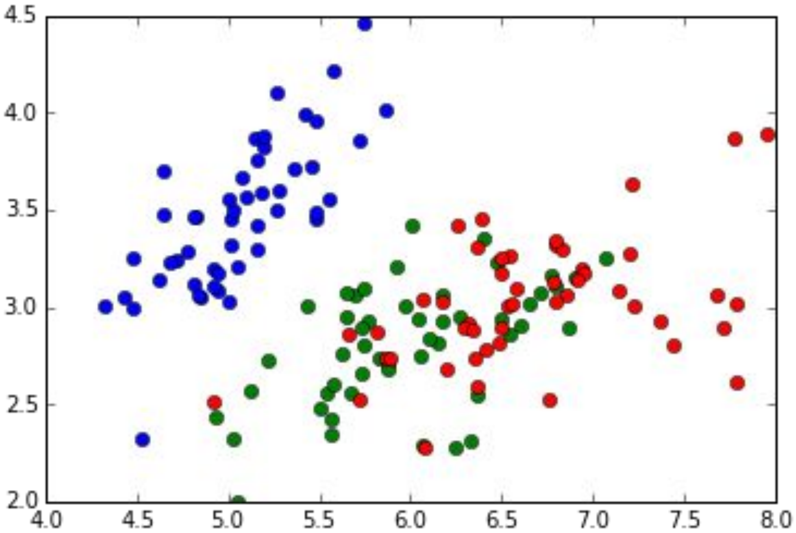
- Supervised learning: predict target values
- For discrete targets, often visualize with color




```
plt.hist( [X[Y==c,1] for c in np.unique(Y)] ,
            bins=20, histtype='barstacked')
```




```
ml.histy(X[:,1], Y, bins=20)
```




```
colors = ['b','g','r']
for c in np.unique(Y):
    plt.plot( X[Y==c,0], X[Y==c,1], 'o',
              color=colors[int(c)] )
```

Machine Learning

Introduction to Machine Learning

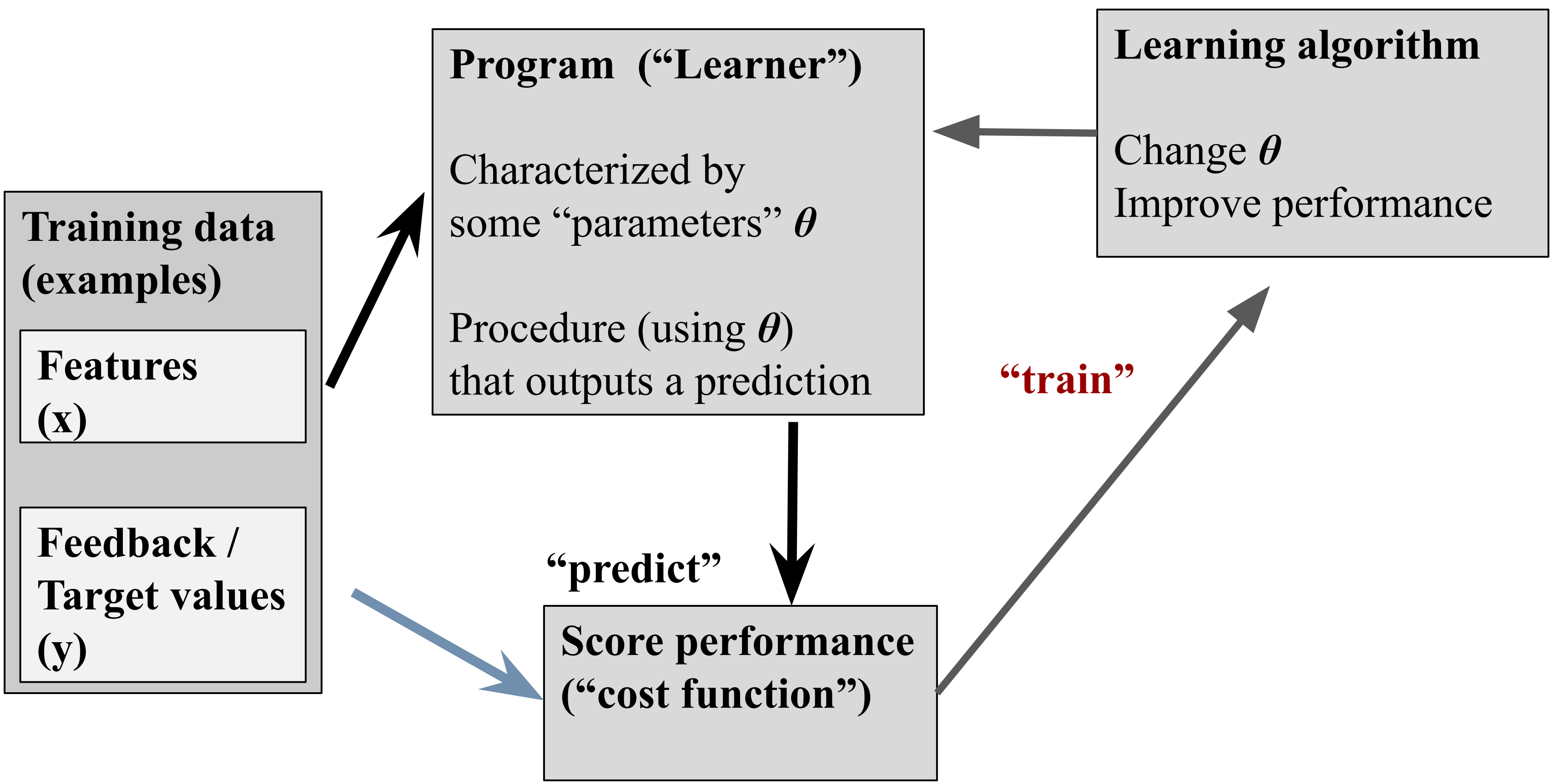
Course Logistics

Data and Visualization

Supervised Learning

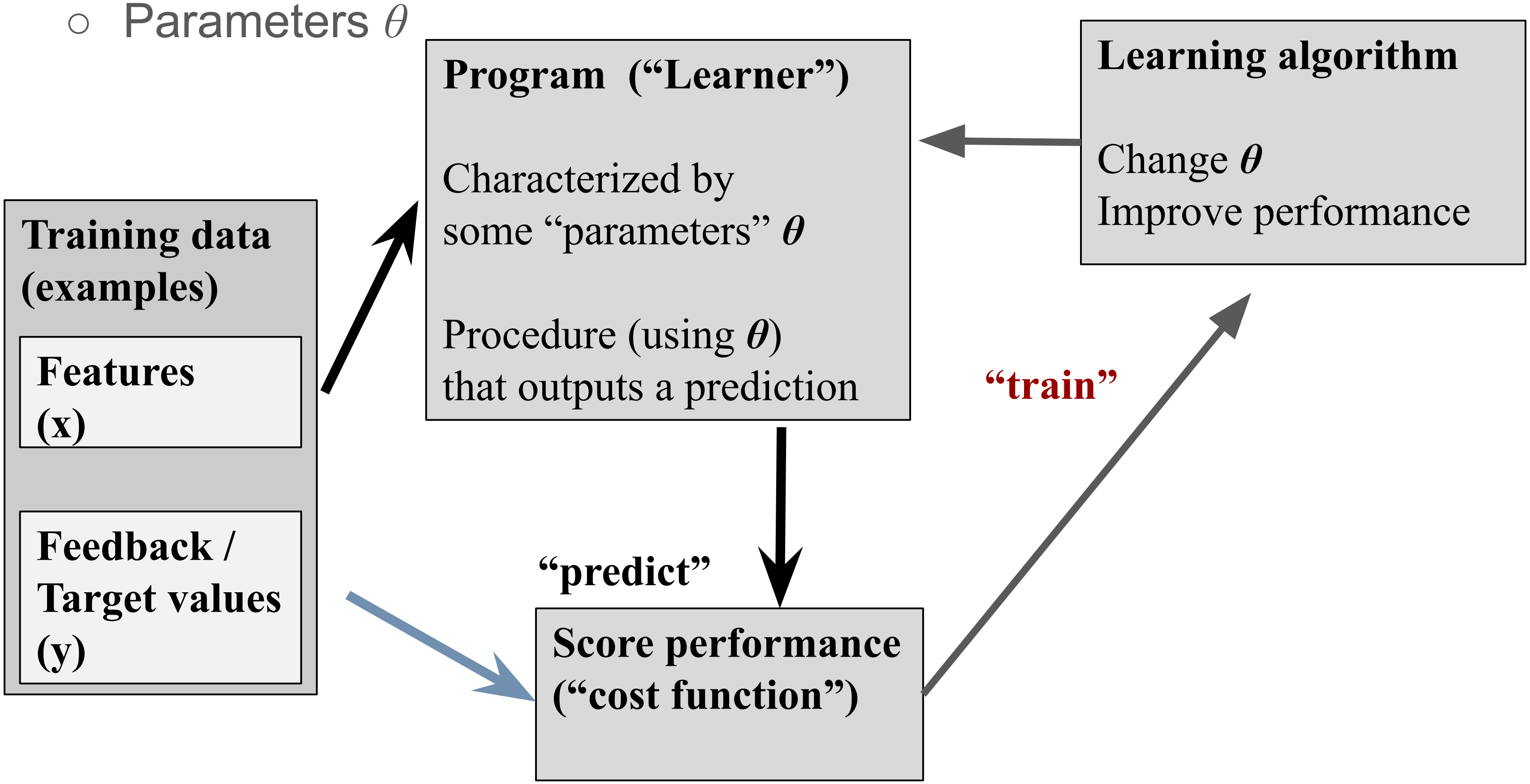
How does machine learning work?

- "Meta-programming"
  - Predict – apply rules to examples
  - Score – get feedback on performance
  - Learn – change predictor to do better

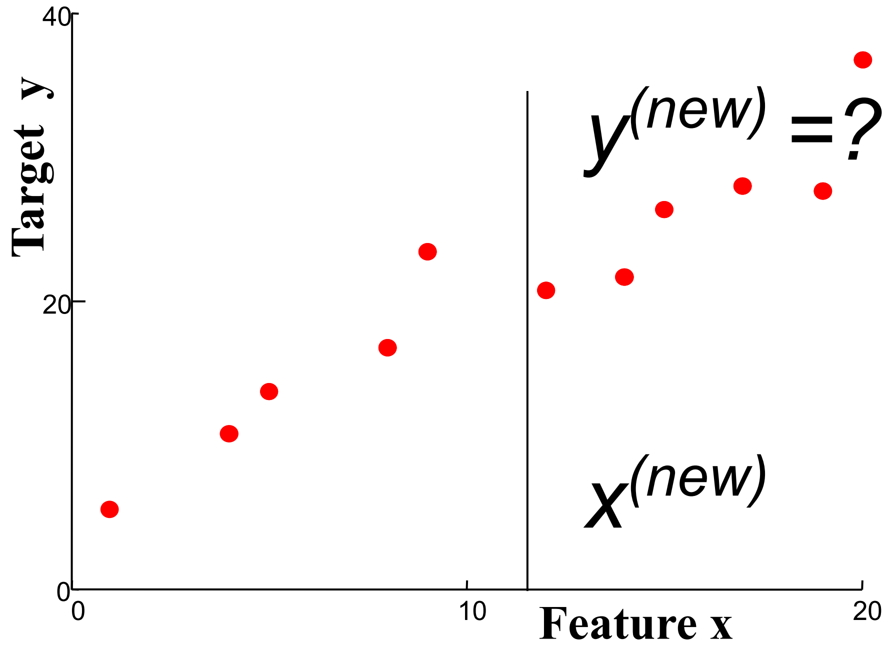


How does machine learning work?

- Notation
  - Features x
  - Targets y
  - Predictions $\hat{y} = f(x\,;\,\theta)$
  - Parameters θ

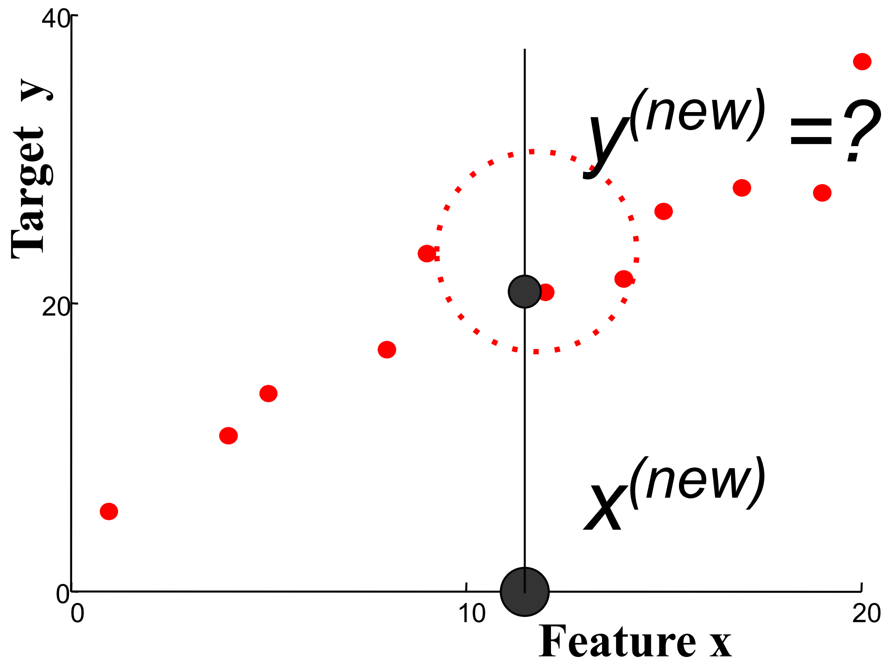


Regression; Scatter plots



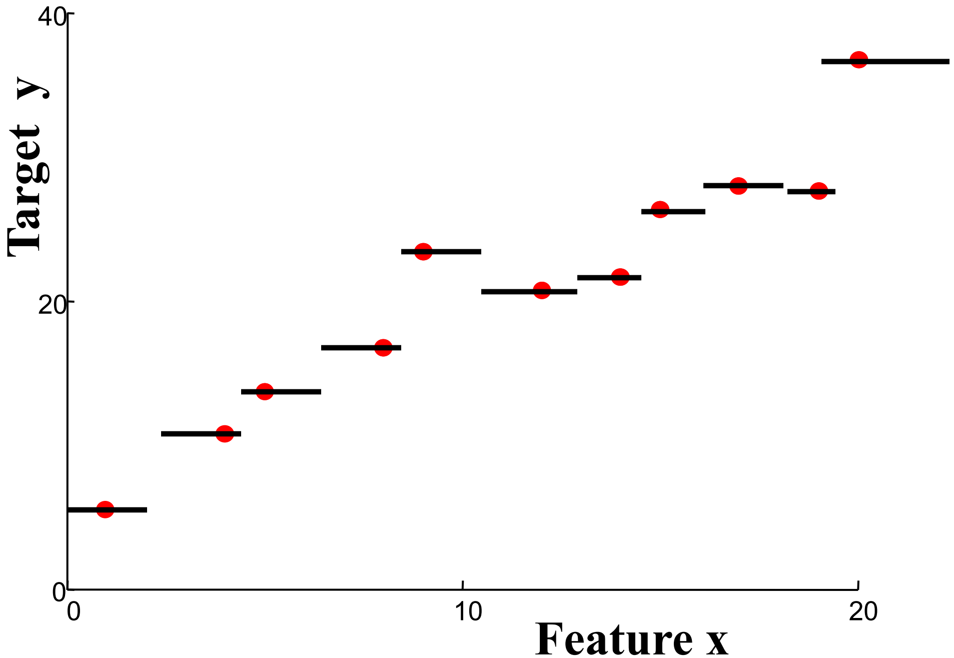
- Suggests a relationship between x and y
- *Prediction*: new x, what is y?

Nearest neighbor regression



- Find training datum $x^{(i)}$ closest to $x^{(new)}$ Predict $y^{(i)}$

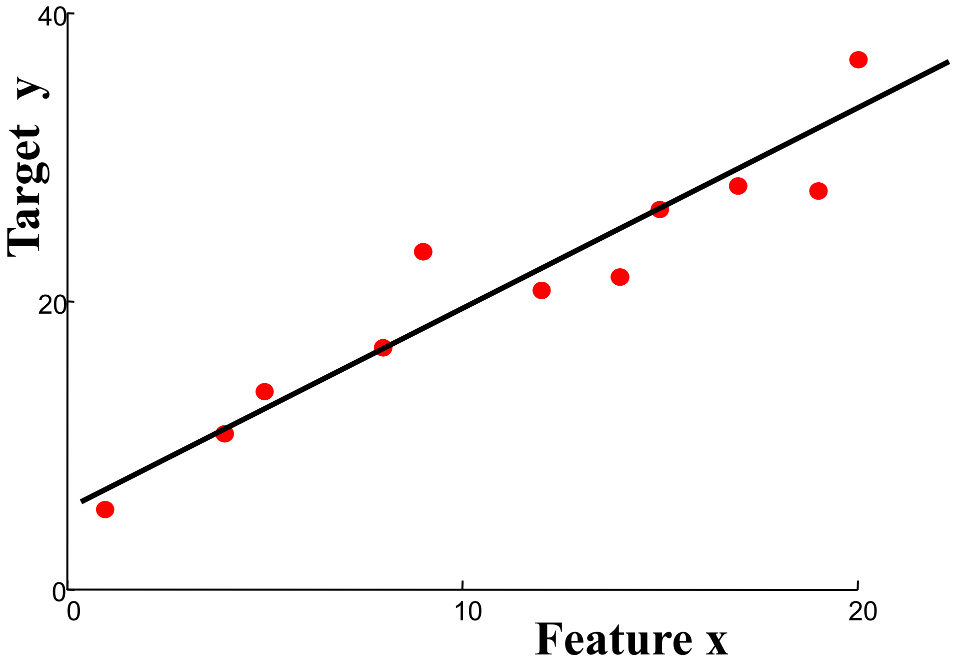
Nearest neighbor regression



"Predictor":
Given new features:
Find nearest example
Return its value

- Defines a function f(x) implicitly
- "Form" is piecewise constant

Linear regression



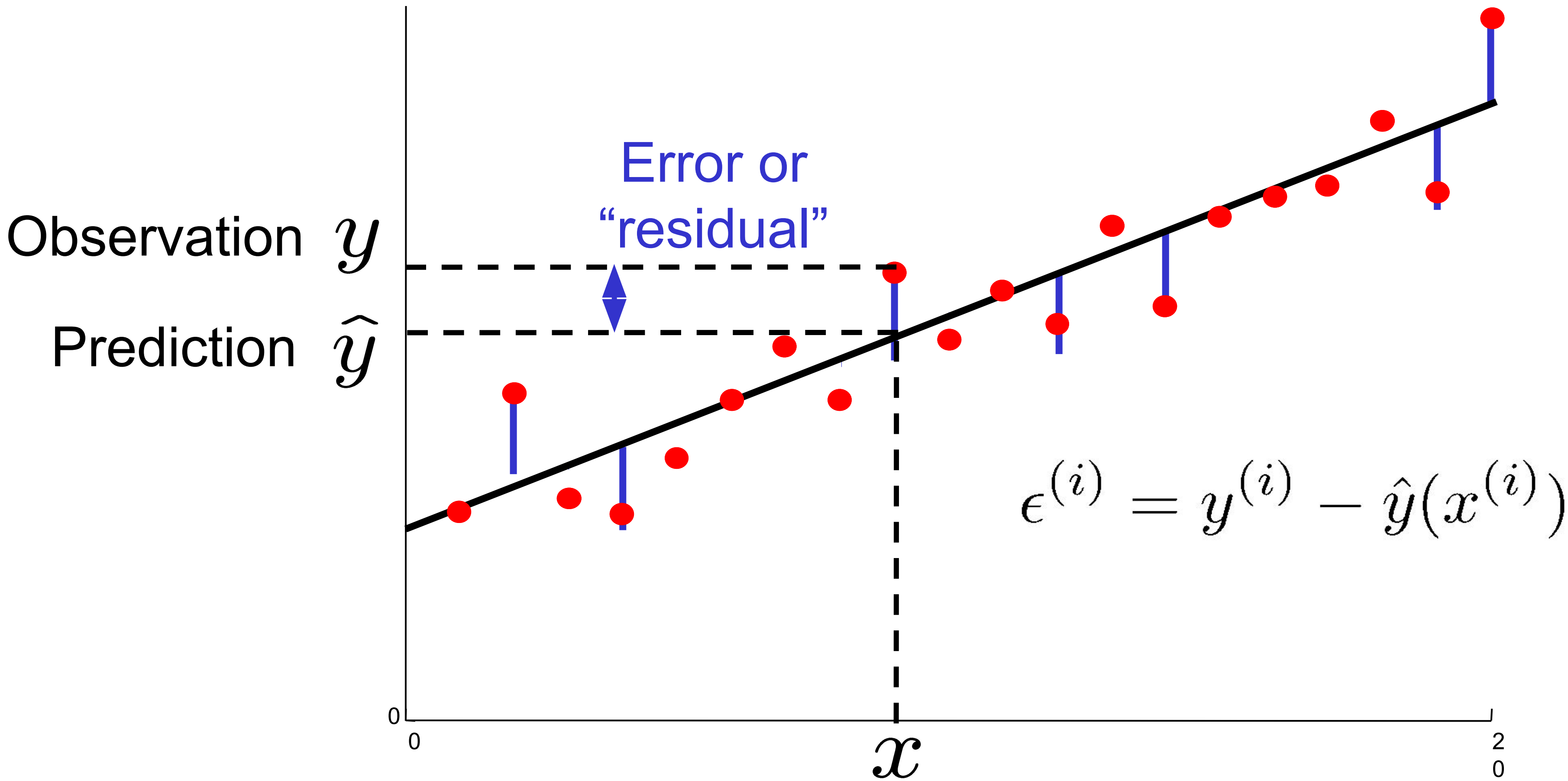
"Predictor":
Evaluate line:

$$r = \theta_0 + \theta_1 x_1$$

return r

- Define form of function f(x) explicitly
- Find a good f(x) within that family

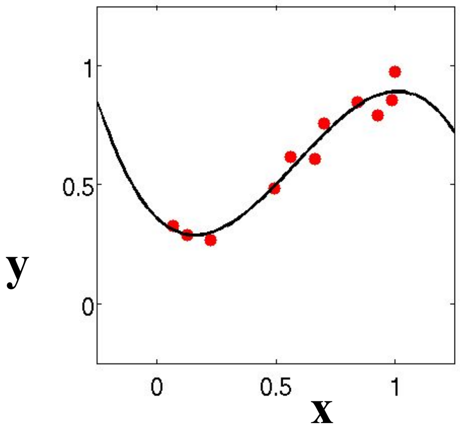
Measuring error



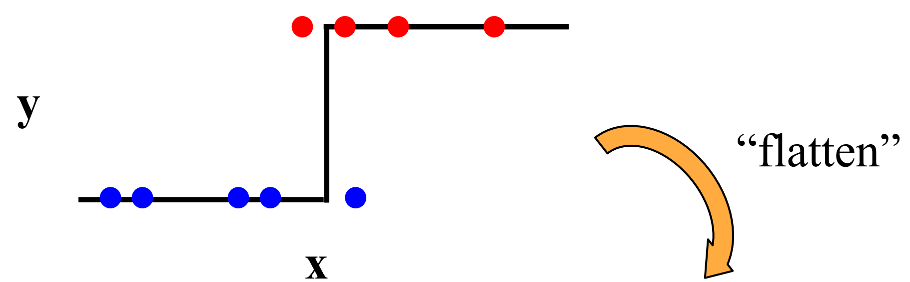
$$\mathrm{MSE} = \frac{1}{m} \sum_i \left(y^{(i)} - \hat{y}(x^{(i)}) \right)^2$$

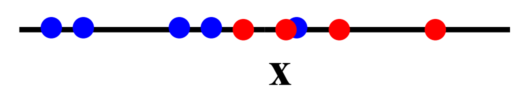
Regression vs. Classification

Regression



Classification



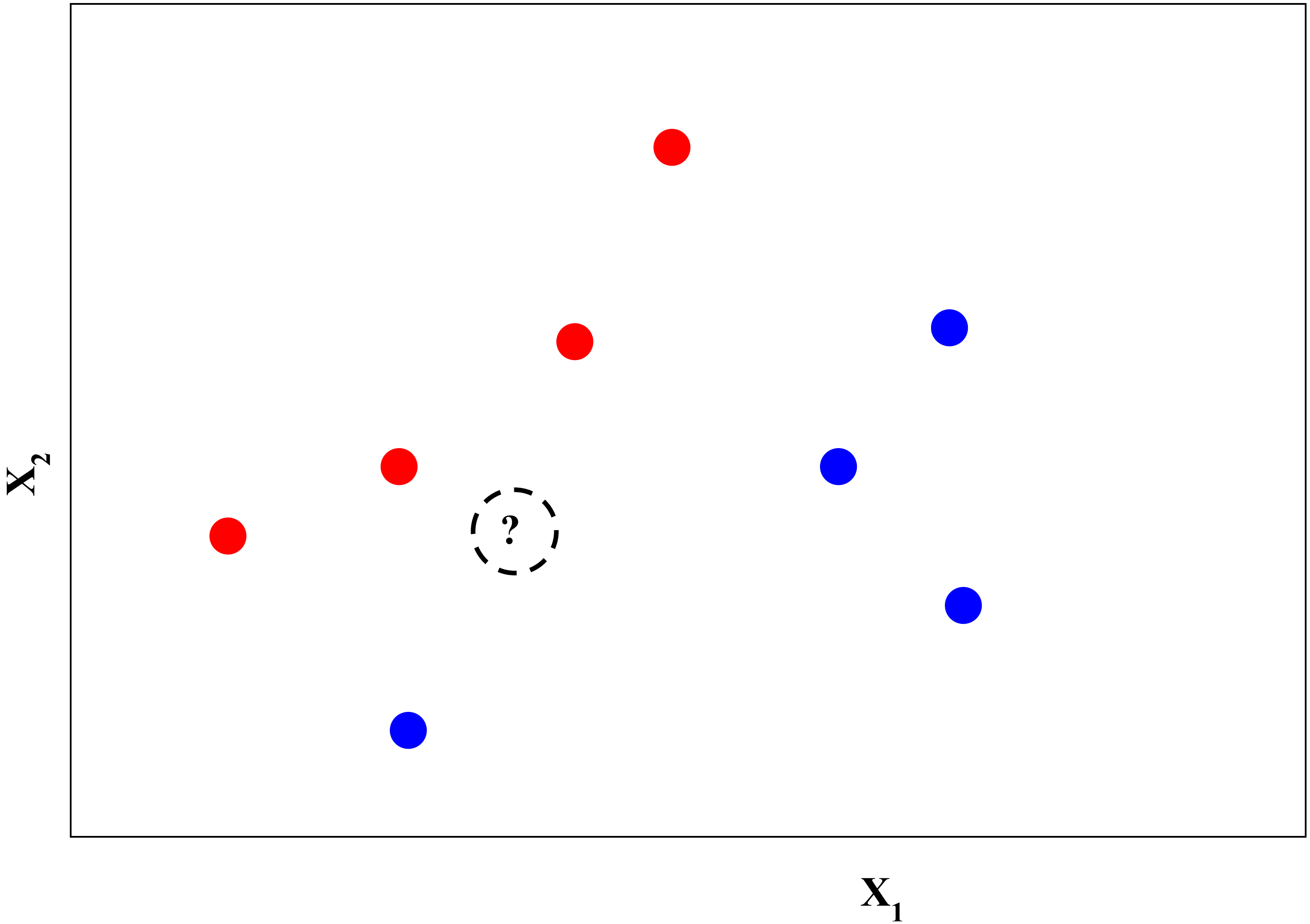


Features x
Real-valued target y

Predict continuous function ŷ(x)

Features x
Discrete class c
(usually 0/1 or +1/-1)
Predict discrete function ŷ(x)

Classification



Classification

$$\text{ERR} = \frac{1}{m}\sum_i \left[\, y^{(i)} \neq \hat{y}(x^{(i)}) \,\right]$$