Artificial Intelligence

Chapter 7 Machine Learning (Supervised Learning)

How does human learn?

Observation

Past Experience

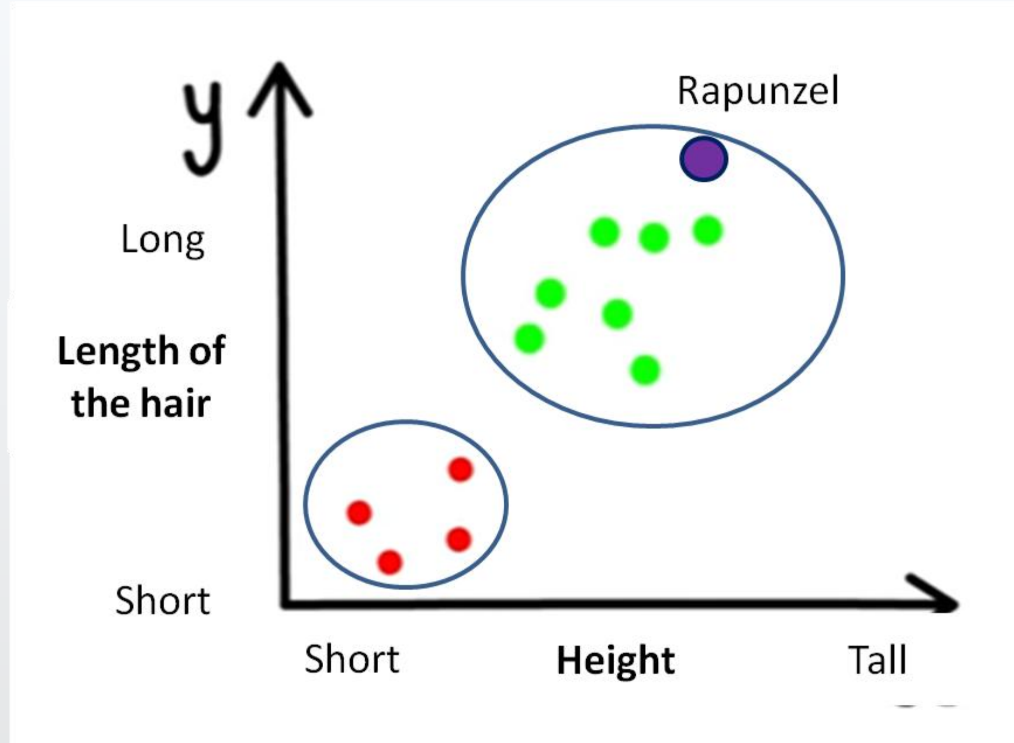
How does machine learn?

Machine Learning: Definition

- Machine Learning is a set of methods that can automatically **detect patterns** in data, and then **use the uncovered patterns to predict future data**, or to perform other kinds of decision making under uncertainty. *by Kevin P. Murphy*

Rapunzel:
Tall with long hair

My ex-gfs



Features/ independent variables | **Label**

Height	Length of the hair	Preferability (yes/ no)
1.70	12”	No
1.50	5”	Yes
1.75	20”	**?**

Prediction

Machine Learning (Example)

Google is Using Machine Learning to Predict the Likelihood of a Patient's Death – with 95% Accuracy!

PRANAV DAR, JUNE 19, 2018

Overview

- The AI research team at Google has developed a model that can predict the likelihood of a patient's death
- The AI is powered by neural networks and uses a ton of variables like the patient's old medical history, age and combines that with scribbled doctor's notes and PDFs
- Google tested the final model on 200,000+ patients and used over 46 billion data points
- The final model came up with an almost 95% accuracy when predicting patient outcomes

Features:

- Gender
- Age
- Previous diagnosis
- Present signs
- Lab results

What are the possible features?

Machine Learning (example)

What are these letters?

Optical Character Recognition (OCR)

Concepts in Machine Learning

- Type of Machine Learning
- Overfitting
- Features
- Assessing classification performance

Type of Machine learning

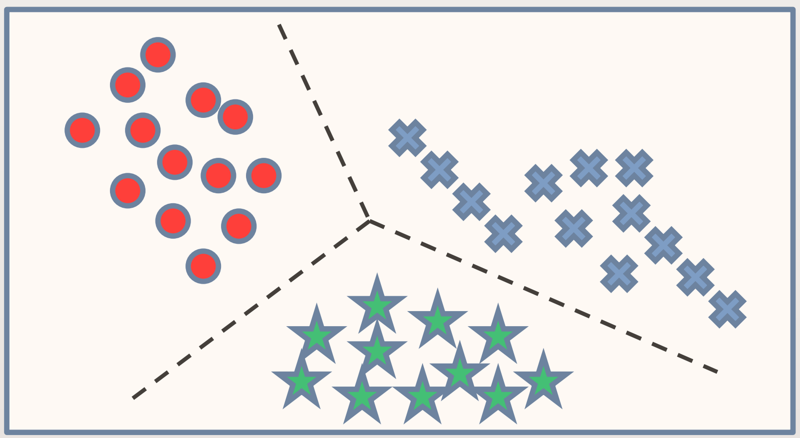
- Usually divided into two main types:
  - Supervised
  - Unsupervised

Will be covered

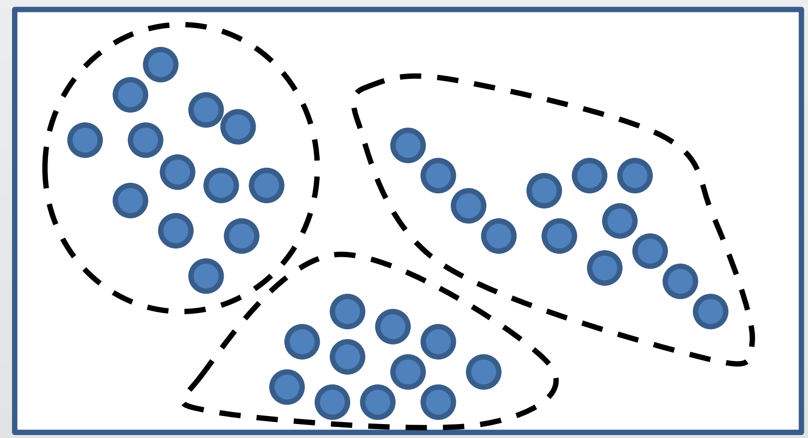
- Uncommon types:
  - Semi-supervised
  - Reinforcement learning

Extra reading and searching

Types of Machine learning



Supervised learning

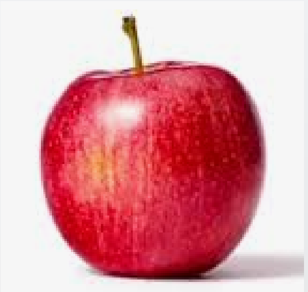


Unsupervised learning

Part 1: Supervised learning

- Supervised Learning
  - The data are labelled with pre-defined classes. It is like that a “teacher” gives the classes (supervision).
  - Goal: learn a mapping from inputs $\boldsymbol{x}$ to outputs y, given a labeled set of input-output pairs. $\mathcal{D}$ is the training set and n is the number of training examples.
    - $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$

Machine doesn’t know what fruits are these.



X (Features)

y (Label)

Features?

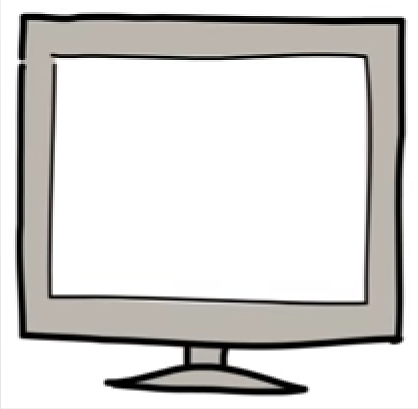
Label?

Color	Size	Fruit
Red	Big	Apple
Orange	Big	Orange
Red	Small	Grapes
Red	Big	Apple
Orange	Big	Orange

Supervised Learning

Fruit X

Features
- Red
- Big



Model

Apple

Labelled Data

- Classification when the output is categorical / nominal (discrete labels)

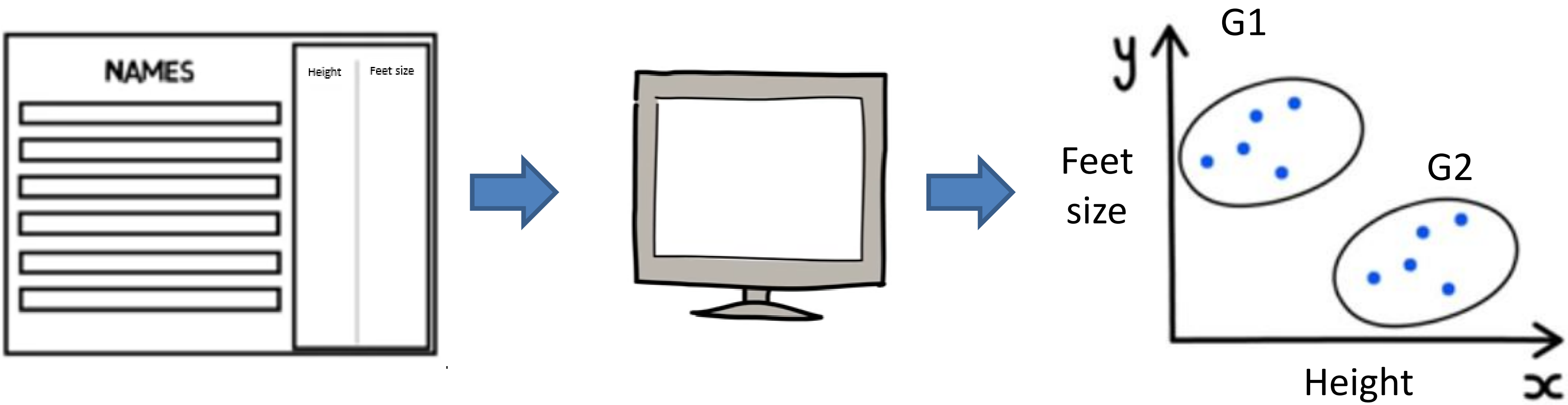
- Regression when the output is real values

- Algorithms: Linear regression , K-nearest neighbor, Support Vector Machine, Artificial Neural Network

Part 2: Unsupervised Learning

- Unsupervised Learning
  - Class labels of the data are unknown.
  - Goal: Given a set of data, the task is to establish the existence of classes or clusters in the data.
    - $\mathcal{D} = \{(\boldsymbol{x}_i)\}_{i=1}^{n}$
- Clustering
  - Finding association (in features)
  - Dimension reduction
  - Sometimes called knowledge discovery
- Algorithms: K-means, Mean Shift, Gaussian Mixture Model

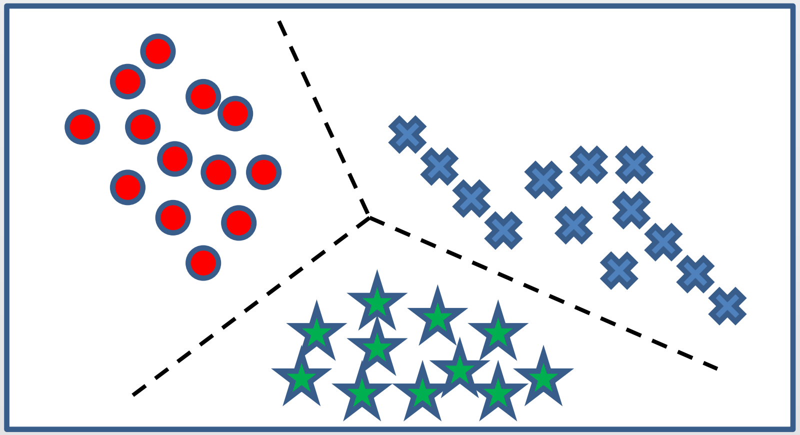
Unsupervised Learning



NO Labelled Data

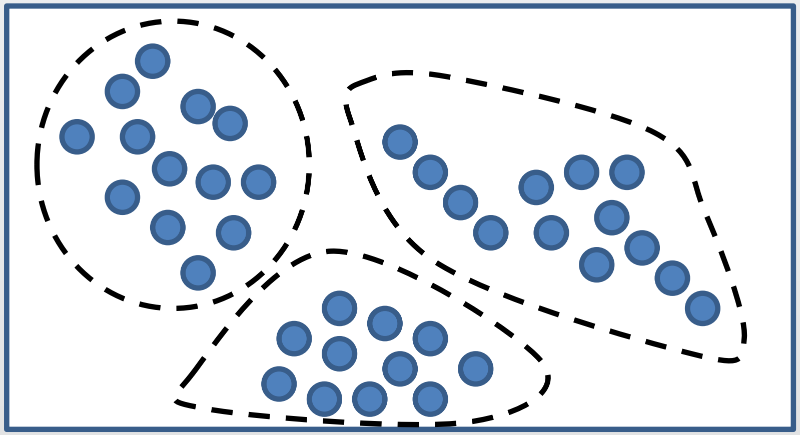
Categorized into two groups.

Classification



Supervised learning

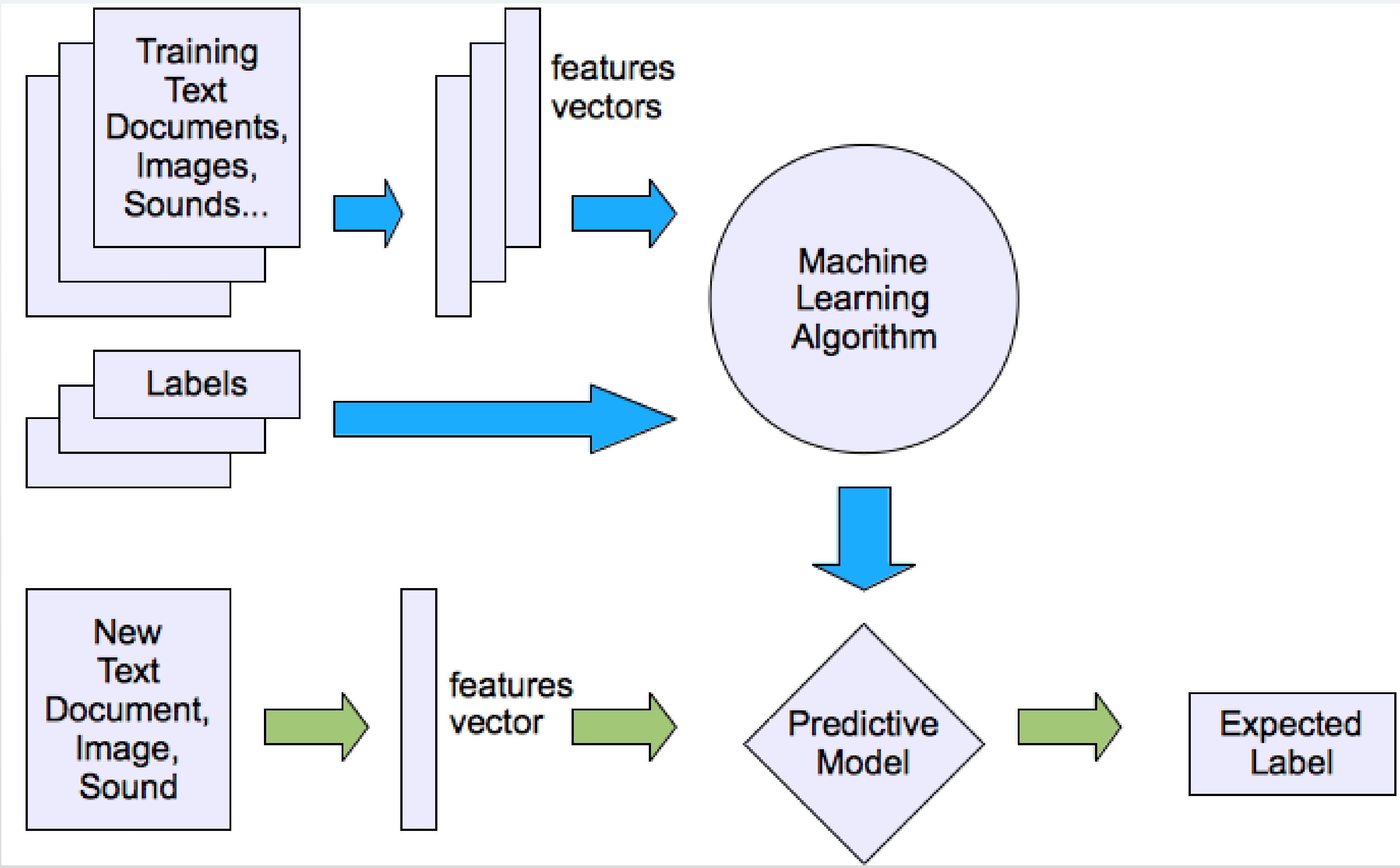
Clustering



Unsupervised learning

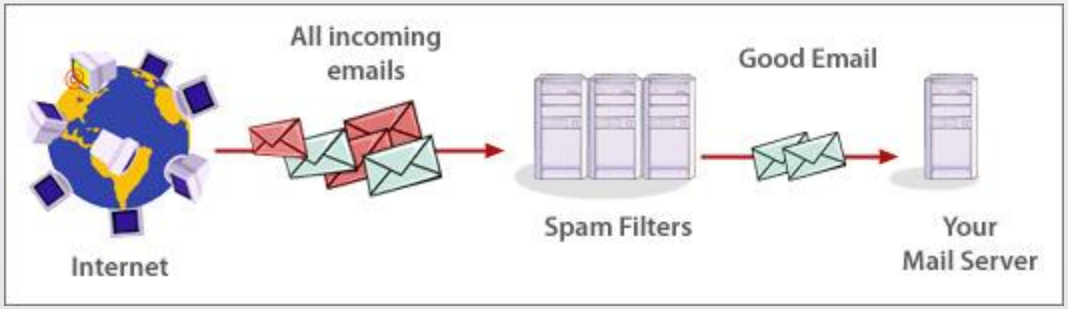
Part 1: Supervised learning

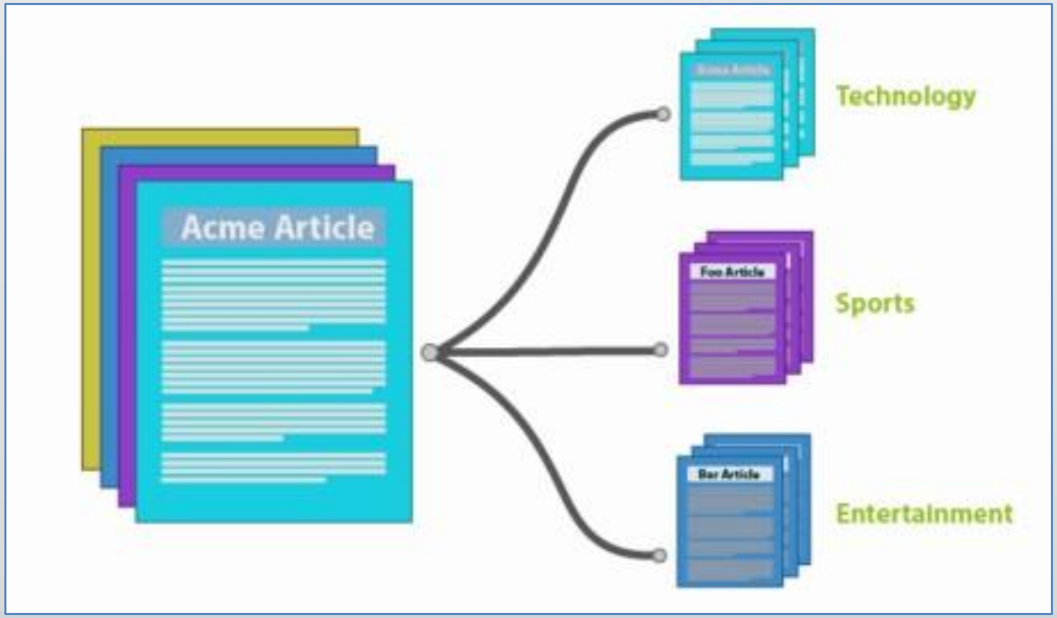
General of Supervised Learning



Real World Application (1)

- Document classification and email spam filtering



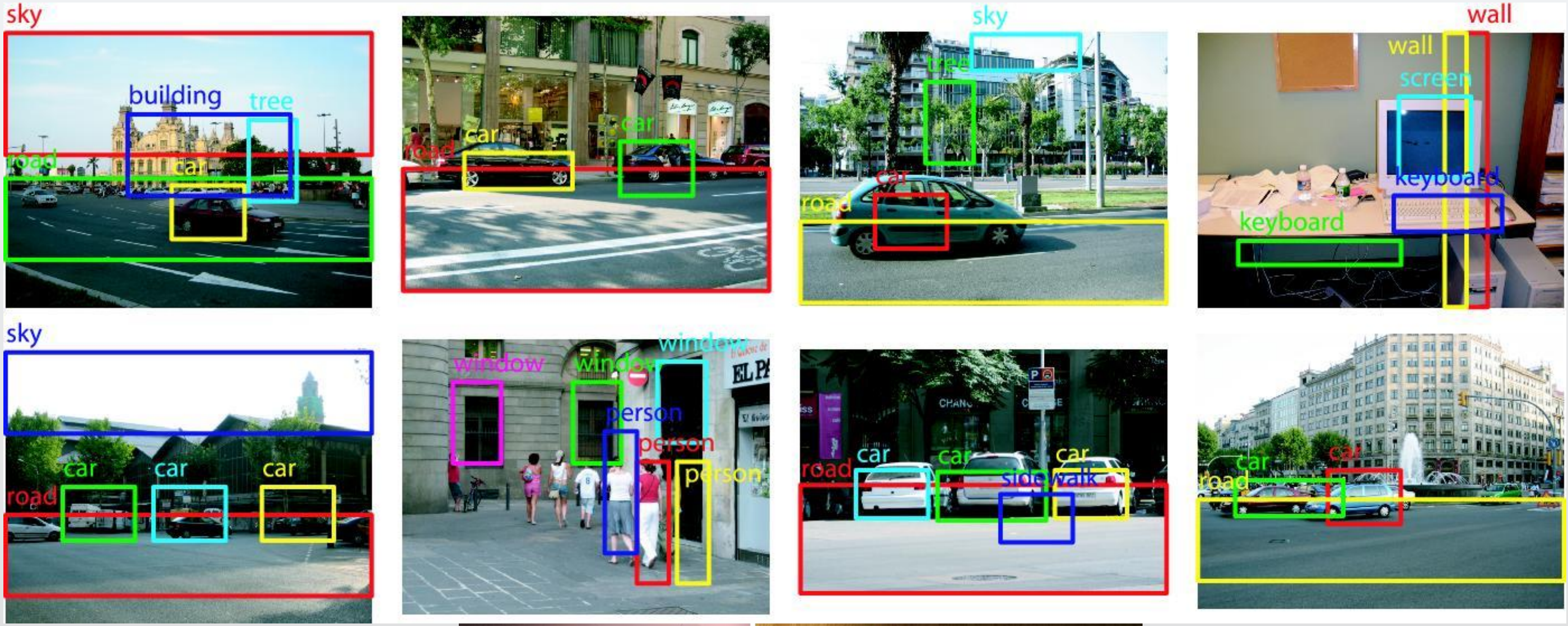


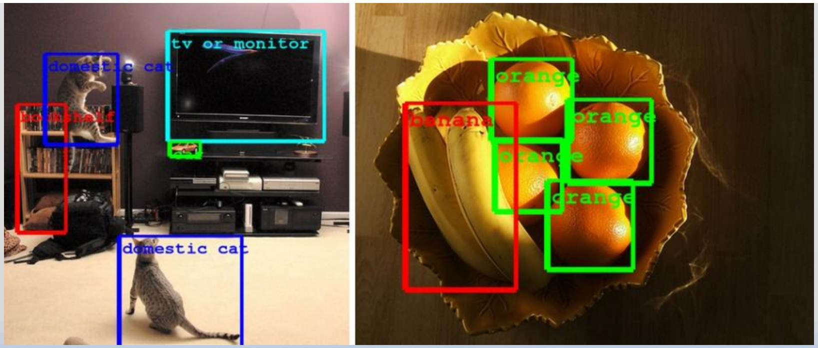
Real World Application (2)

- Classifying flower

Real World Application (3)

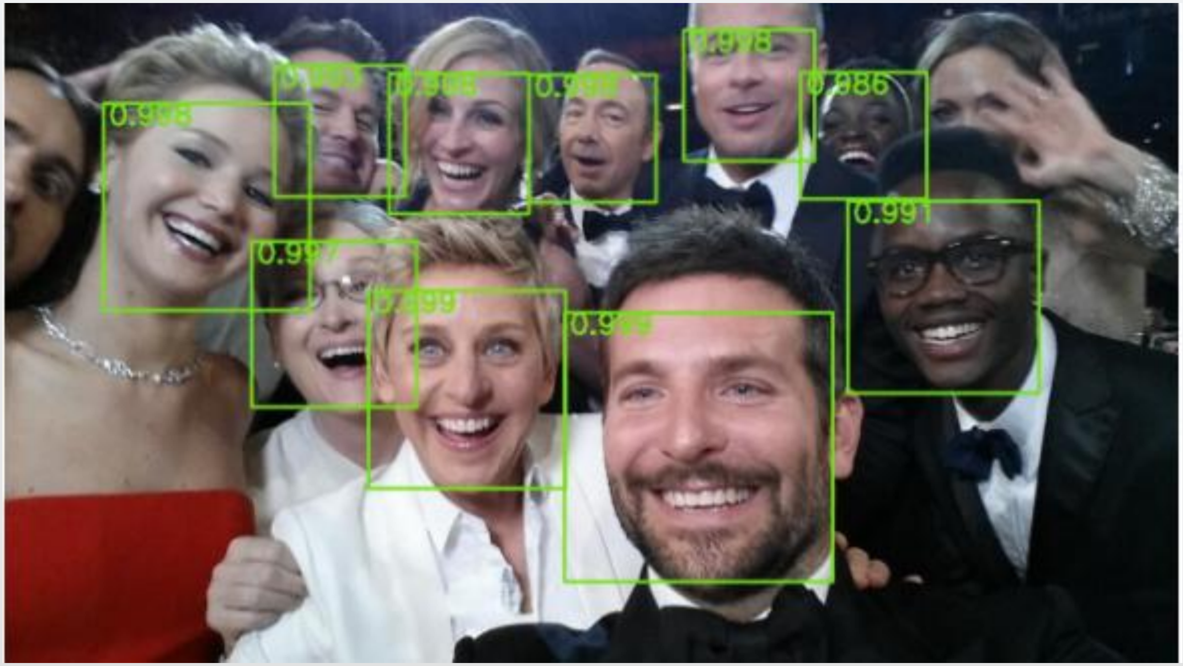
- Object classification





Real World Application (4)

- Face detection and recognition



Example of classification problem 1

- An emergency room in a hospital measures 17 variables (e.g., blood pressure, age, etc) of newly admitted patients.
- A decision is needed: whether to put a new patient in an intensive-care unit.
- Due to the high cost of ICU, those patients who may survive less than a month are given higher priority.
- Problem: to predict high-risk patients and discriminate them from low-risk patients.

Example of classification problem 2

- A credit card company receives thousands of applications for new cards. Each application contains information about an applicant:
  - age
  - Marital status
  - annual salary
  - outstanding debts
  - credit rating
  - etc.

- Problem: to decide whether an application should approved, or to classify applications into two categories, approved and not approved.

In general…

- Classification is like human learn from past experiences.

- Computer does not has **“experiences”** so it learns from data, which represent some **“past experiences”** of an application domain.

- **Our focus**: learn a **target function** that can be used to predict the values of a discrete class attribute, e.g., **approve or not-approved, and high-risk or low risk**.

An example: data (loan application)

Approved or not

ID	Age	Has_Job	Own_House	Credit_Rating	Class
1	young	false	false	fair	No
2	young	false	false	good	No
3	young	true	false	good	Yes
4	young	true	true	fair	Yes
5	young	false	false	fair	No
6	middle	false	false	fair	No
7	middle	false	false	good	No
8	middle	true	true	good	Yes
14	old	true	false	excellent	Yes
15	old	false	false	fair	No

$$\mathcal{D}_i = \{A_1, A_2, A_3, A_4, y_i\}$$

$$\mathcal{D}_1 = \{Age = 20, Has\,Job = 0, Own\,House = 0, Credit\,Rating = 5, Class = 0\}$$

$$\mathcal{D}_1 = \{20, 0, 0, 5, 0\}$$

About Classification

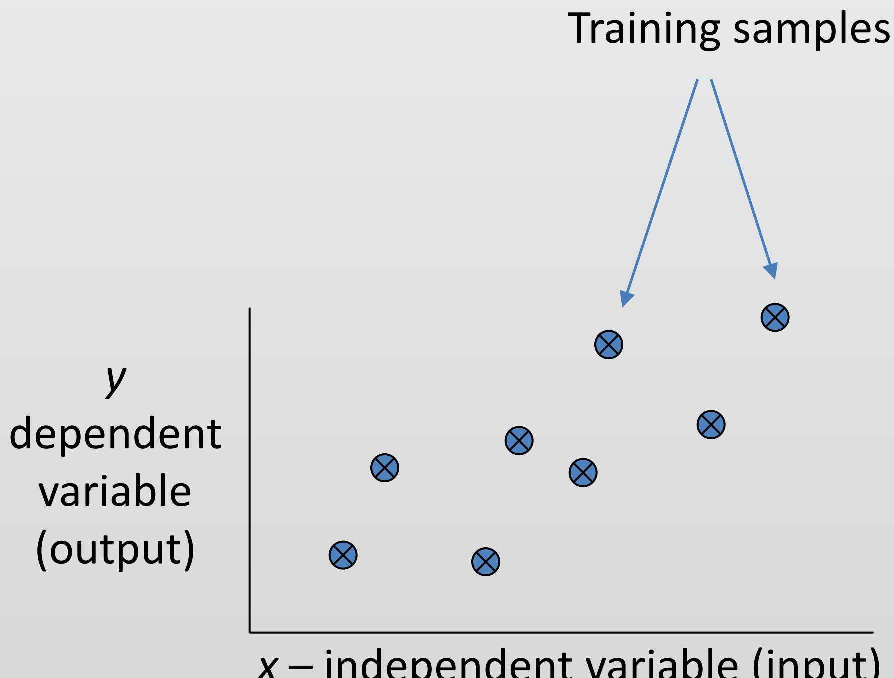
- Binary Classification
- Multiclass Classification
- Classification Methods
  - Regression
  - K- Nearest Neighbour (KNN)
  - Decision Tree (DT)
  - Support Vector Machine (SVM)
  - Naïve Bayesian Classifier
- Assessing classification performance

Classification Methods

1. Regression
2. K Nearest Neighbour
3. Decision Tree
4. Support Vector Machine
5. Bayesian Classification

1. Regression

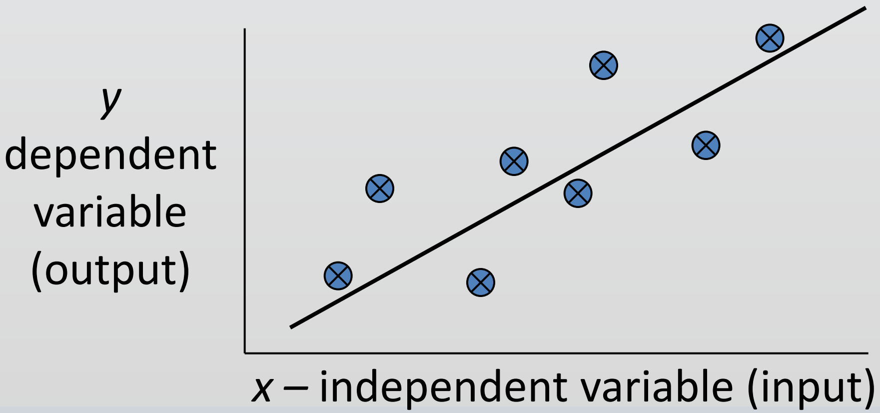
- In regression the output is continuous real value
  - Function Approximation



1. Regression

- In regression the output is continuous real value
  - Function Approximation

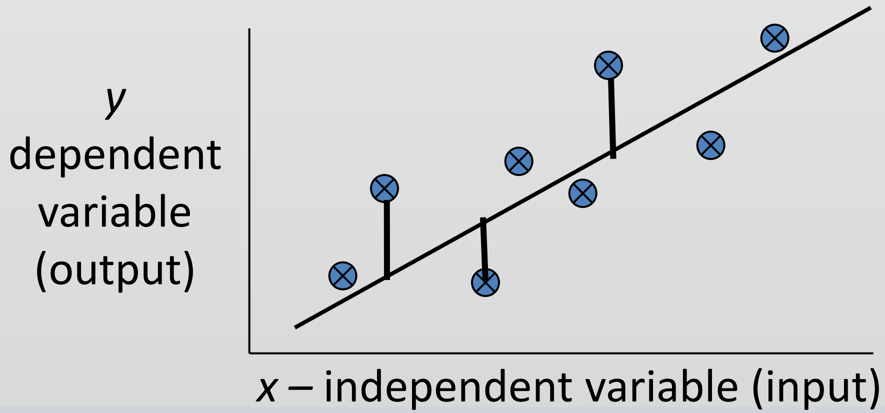
- Many models could be used – Simplest is **linear regression**
  - Fit data with the best hyper-plane which "goes through" the points



1. Regression

- In regression the output is continuous real value
  - Function Approximation

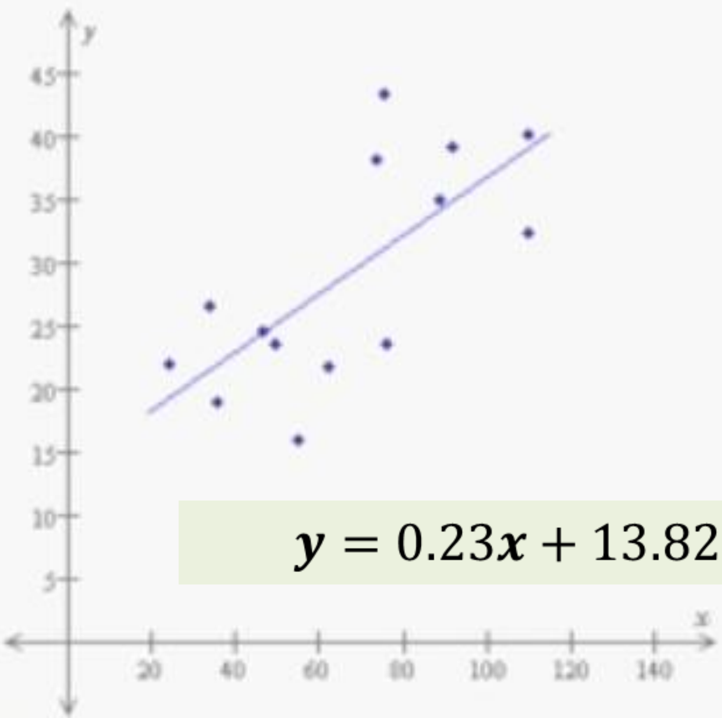
- Many models could be used – Simplest is **linear regression**
  - Fit data with the best hyper-plane which "goes through" the points
  - For each point the differences between the predicted point and the actual observation is the ***residue***



1. Regression

$$y = ax + b$$

	Player payroll, x (in \$1,000,000s)	Mean attendance, y (in thousands)
Anaheim	46.6	24.69
Baltimore	73.4	38.15
Boston	109.6	32.47
Chicago White Sox	62.4	21.85
Cleveland	92.0	39.26
Detroit	49.8	23.70
Kansas City	35.6	19.01
Minnesota	24.4	21.98
New York Yankees	109.8	40.25
Oakland	33.8	26.54
Seattle	75.7	43.33
Tampa Bay	55.0	16.05
Texas	88.5	34.94
Toronto	75.8	23.70



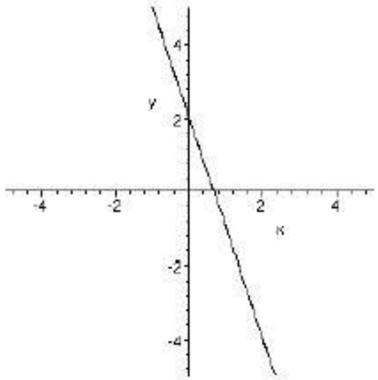
$$y = 0.23x + 13.82$$

Figure 1

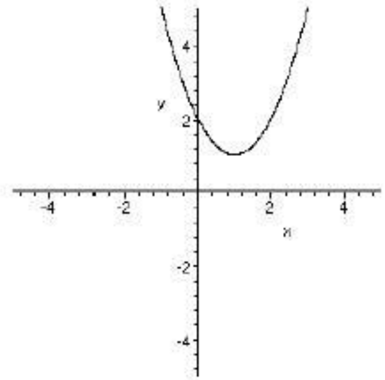
$$x = 33.8, \text{predict } y$$

$$y = 21.6$$

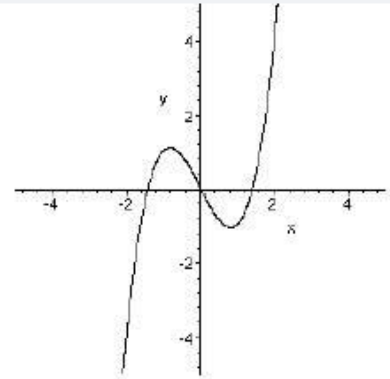
1. Regression



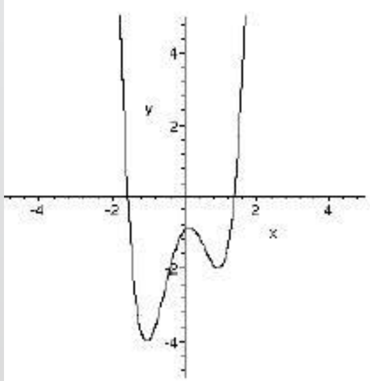
$f(x) = -3x + 2$



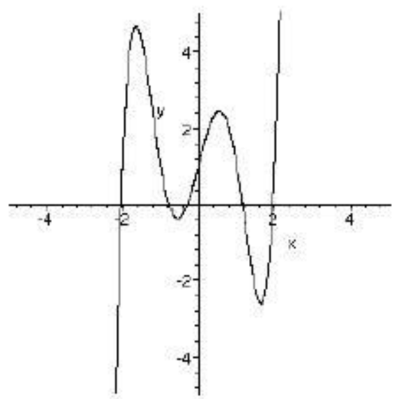
$g(x) = x^2 - 2x + 2$



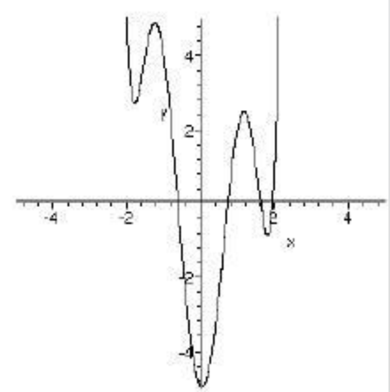
$h(x) = x^3 - 2x$



$F(x) = 2x^4 - 4x^2 + x - 1$



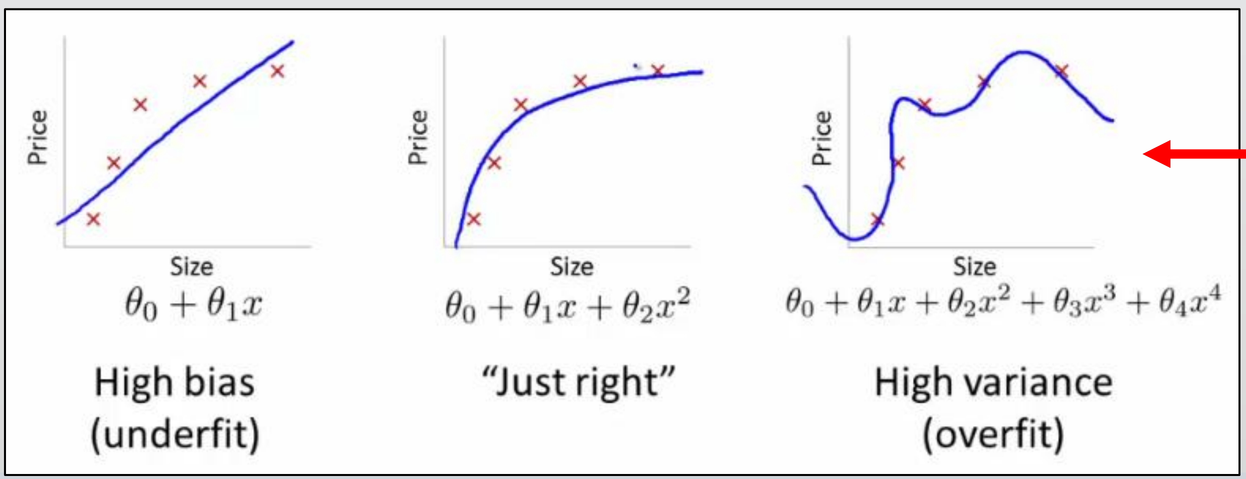
$G(x) = x^5 - 5x^3 + 4x + 1$



$H(x) = x^6 - 7x^4 + 14x^2 - x - 5$

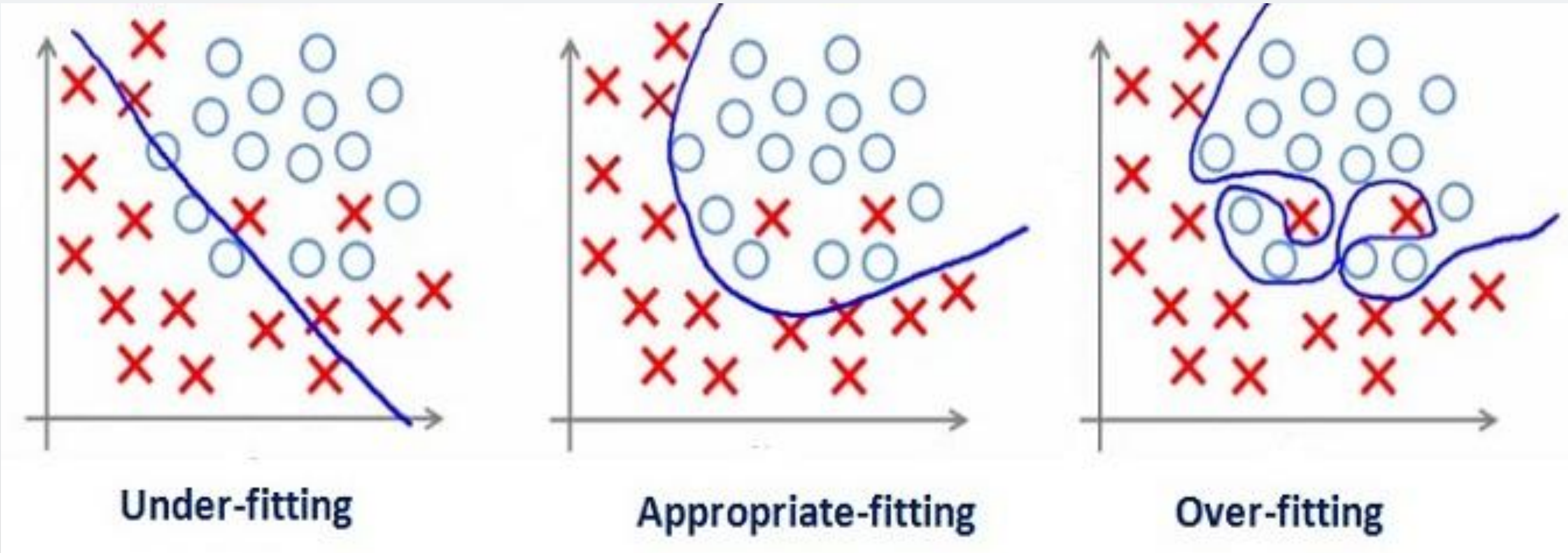
1. Regression

- To avoid overfitting:
  - Number of parameters estimated from the data must be considerably less than the number of data points
  - Advisable to choose the degree of polynomial as low as possible – often a simple linear relationship is assumed.



Fail to generalize

Overfitting

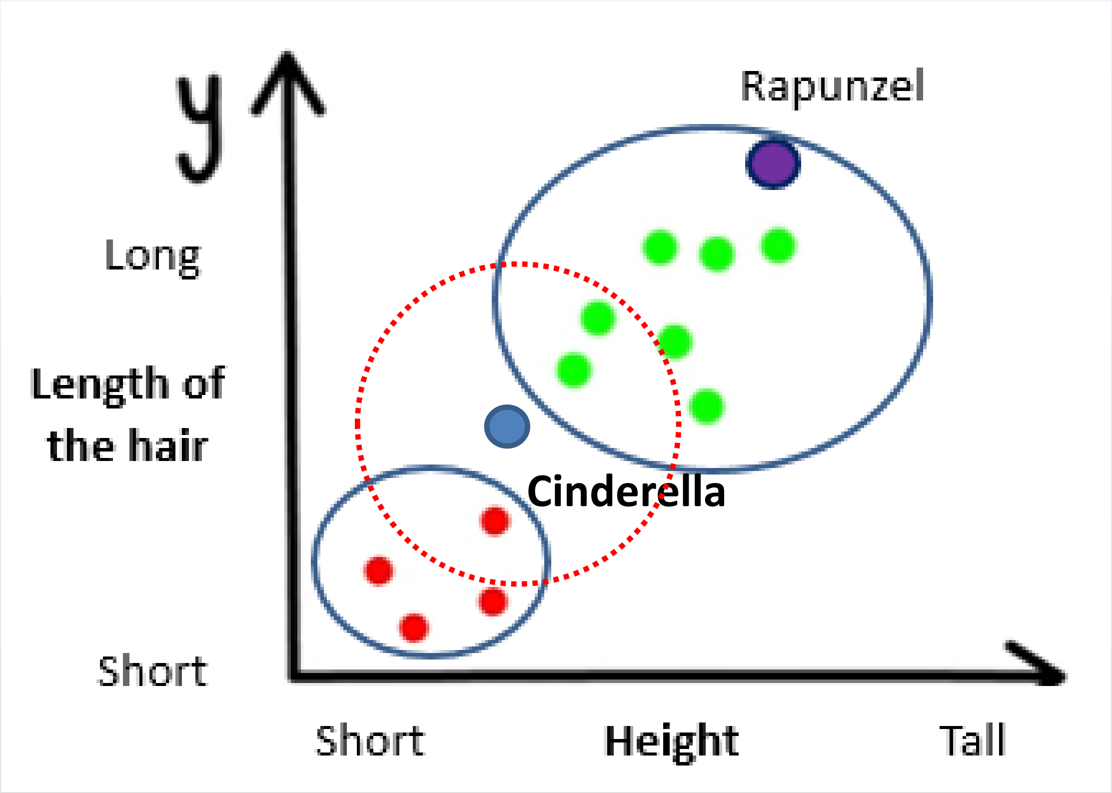


2. K-Nearest Neighbour

- K-Nearest Neighbour (KNN)
  - Distance-based classifier

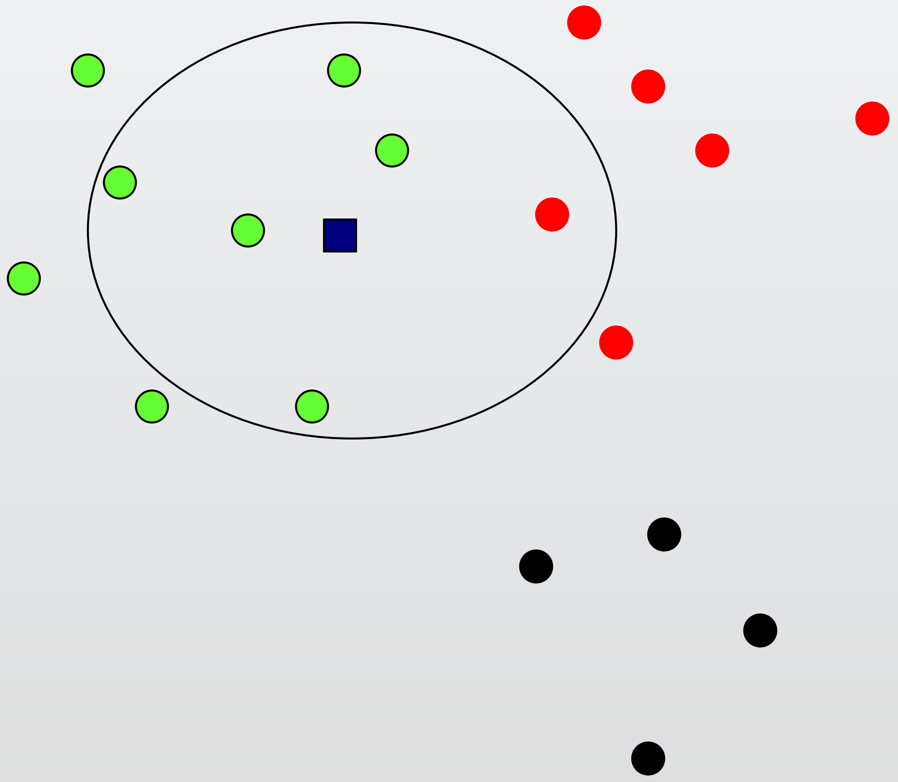
KNN

Cinderella



$$Pr(No|\ Cinderella\ ) \quad = 2/3 = 0.67$$
$$Pr(Yes|\ Cinderella\ ) = 1/3 = 0.33$$

Example: k=6 (6NN)



Government

Science

Arts

A new point ■

$Pr(science|\blacksquare)$?

2. K-Nearest Neighbour

- To classify a test instance d, define K neighbourhood as K nearest neighbours of d.

- Count number n of training instances in neighbourhood that belong to class c_j

- Estimate $\Pr(c_j|d)$ as n/K

- No training is needed. Classification time is linear in training set size for each test case.

2. K-Nearest Neighbour

- **Algorithm**:
  1. Compute the distance between d and every training sample in $\mathcal{D}$;
  2. Choose the K sample in $\mathcal{D}$ that are nearest to d
  3. Assign d the class that is the most frequent class in the neighbourhood (or the majority class)

- *k* is usually chosen empirically via a validation set or cross-validation by trying a range of *k* values.
- Distance function is crucial, but depends on applications.

2. K-Nearest Neighbour

- KNN can deal with complex and arbitrary decision boundaries.
- Despite its simplicity, researchers have shown that the classification accuracy of KNN can be quite strong and in many cases as accurate as those elaborated methods.
- KNN is slow at the classification time
- KNN does not produce an understandable model

3. Decision Tree

- Decision tree
  - A flow-chart-like tree structure
  - Internal node denotes a test on an attribute
  - Branch represents an outcome of the test
  - Leaf nodes represent class labels or class distribution
- Decision tree generation consists of two phases
  - Tree construction
    - At start, all the training examples are at the root
    - Partition examples recursively based on selected attributes
  - Tree pruning
    - Identify and remove branches that reflect noise or outliers
- Use of decision tree: Classifying an unknown sample
  - Test the attribute values of the sample against the decision tree

3. Decision Tree

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Training data (example from Quinlan's ID3)

3. Decision Tree

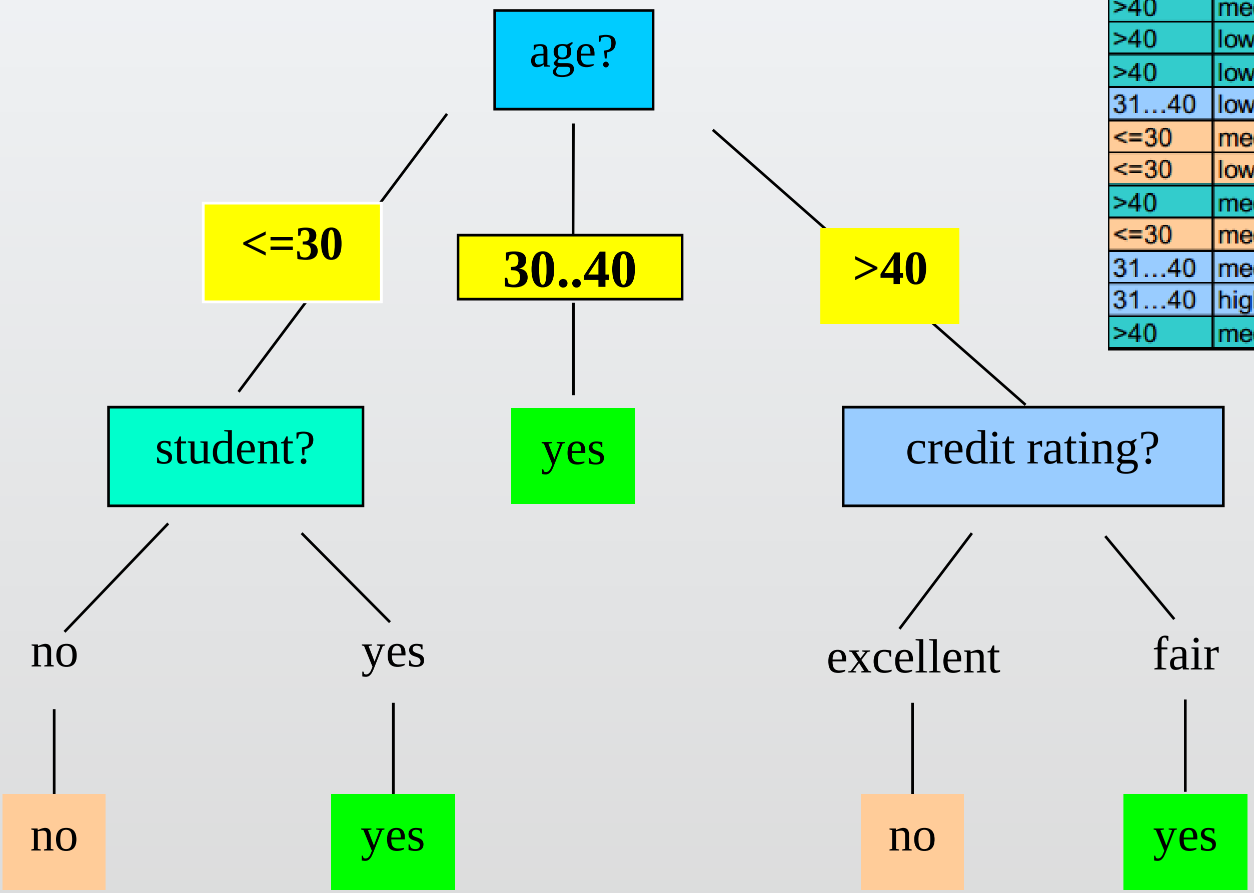
- Three Data Sets formed after division at root node on the basis of “age” attribute.

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

3. Decision Tree

- Output decision tree for “buys_ computer”

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no



On the basis of tree constructed in the manner described, classify a test sample (age, student, creditrating, buys_computer) **(<=30, yes, excellent, ?**)
-Will this student buy computer?

Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left

Information Gain Calculation (ID3/C4.5)

- Select the attribute with the highest information gain
- Assume there are two classes, ***P*** and ***N*** (yes and no from example)
- Let the set of examples $\mathcal{D}$ contain *p* elements of class ***P*** and *n* elements of class ***N***
  - The **amount of information**, needed to decide if an arbitrary example in *S* belongs to ***P*** or ***N*** is defined as

$$I(p,n) = -\left(\frac{p}{p+n}\log_2\frac{p}{p+n}\right) - \left(\frac{n}{p+n}\log_2\frac{n}{p+n}\right)$$

Information Gain Calculation (ID3/C4.5)

- Assume that using attribute A, a set $\mathcal{D}$ will be partitioned into sets $\{S_1, S_2, \ldots, S_v\}$
  - If S_i contains p_i examples of P and n_i examples of N, the **entropy,** or the expected information needed to classify objects in all subtrees S_i is

$$\boxed{E(A) = \sum_{i=1}^{\nu} \frac{p_i + n_i}{p + n} I(p_i, n_i)}$$

- The encoding information that would be gained by branching on A

$$\boxed{Gain(A) = I(p, n) - E(A)}$$

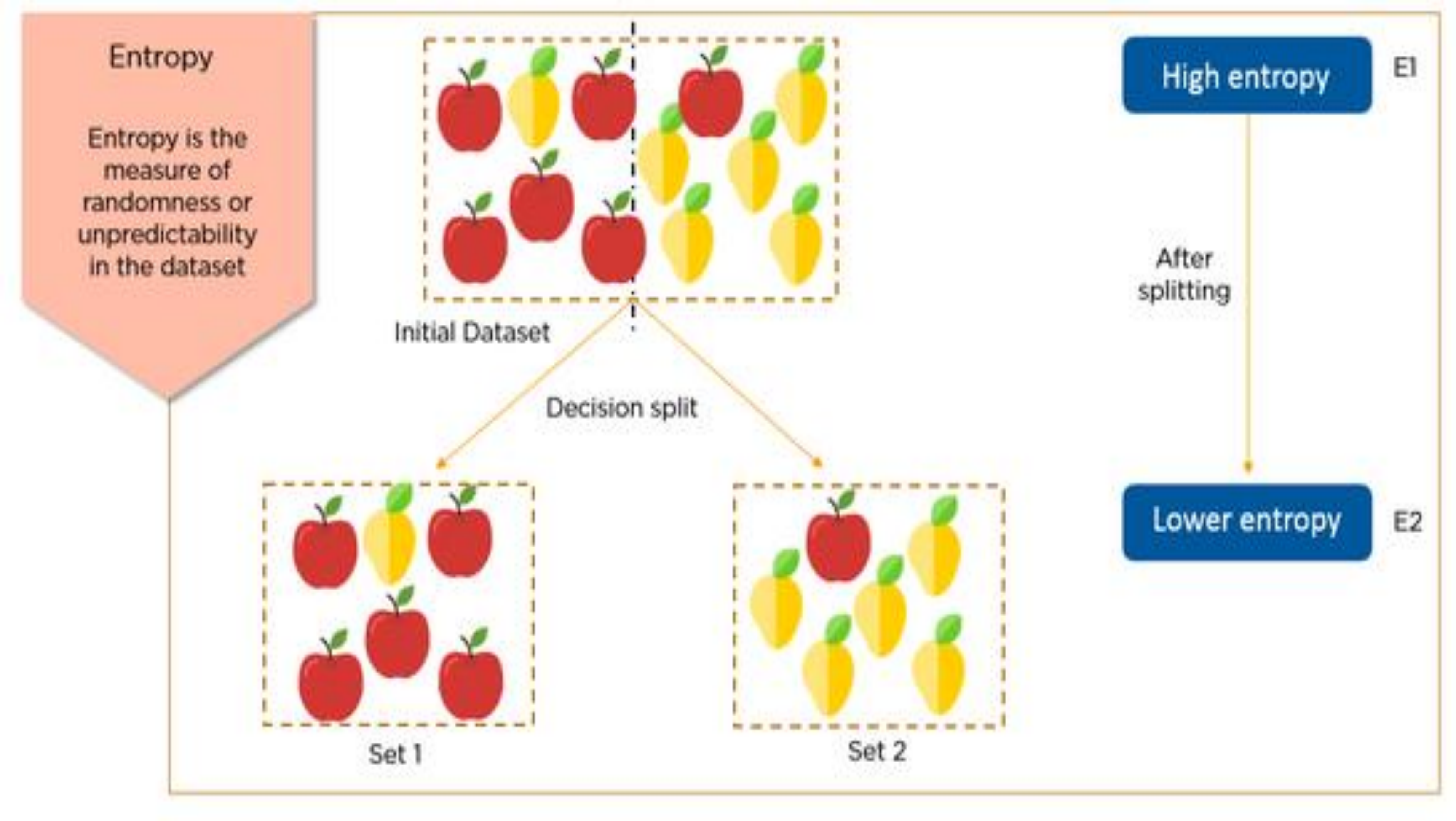
Entropy???

Harder to predict
(Higher entropy)

Easier to predict
(Lower entropy)

Entropy is the measure of randomness in a dataset.

Aim of DT – split the data in a way that the entropy in the data decreases -> easier to make predictions



Initial Dataset – all mixed up

Split – less random -> entropy decreases

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

$$I(p,n) = -(\frac{p}{p+n}\log_2\frac{p}{p+n}) - (\frac{n}{p+n}\log_2\frac{n}{p+n})$$

$$E(A) = \sum_{i=1}^{v}\frac{p_i+n_i}{p+n}I(p_i,n_i)$$

$$Gain(A) = I(p,n) - E(A)$$

3. Decision Tree

Attribute Selection by Information Gain Computation

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"
- $I(p, n) = I(9, 5) = 0.940$
- Compute the entropy for *age*:

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
30…40	4	0	0
>40	3	2	0.971

$$I(p,n) = -\left(\frac{p}{p+n}\log_2\frac{p}{p+n}\right) - \left(\frac{n}{p+n}\log_2\frac{n}{p+n}\right)$$

$$E(age) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2) = 0.69$$

$$E(A) = \sum_{i=1}^{v}\frac{p_i+n_i}{p+n}I(p_i,n_i)$$

Hence,

$$Gain(age) = I(p,n) - E(age)$$
$$= 0.940 - 0.69 = 0.25$$

$$Gain(A) = I(p,n) - E(A)$$

Similarly,

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit\_rating) = 0.048$$

3. Decision Tree

Attribute Selection by Information Gain Computation

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"
- $I(p, n) = I(9, 5) = 0.940$
- Compute the entropy for *age*:

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
30…40	4	0	0
>40	3	2	0.971

$$E(age) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0)$$

Better Information gain then the others, so it is selected for next branch

Hence,

$$Gain(age) = I(p,n) - E(age)$$

$$= 0.940 - 0.69 = 0.25$$

Similarly,

$$Gain(income) = 0.029$$

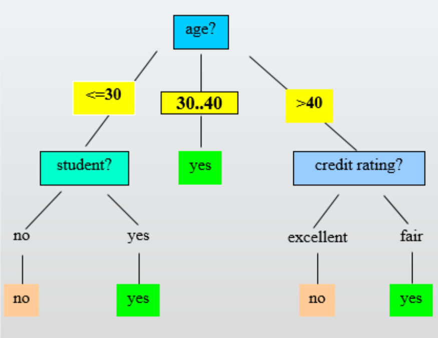
$$Gain(student) = 0.151$$

$$Gain(credit\_rating) = 0.048$$

3. Decision Tree

Extracting Classification Rules from Trees

- Represent the knowledge in the form of **IF-THEN** rules
- One rule is created for each path from the root to a leaf
- Each attribute-value pair along a path forms a conjunction
- The leaf node holds the class prediction
- Rules are easier for humans to understand
- Example



IF *age* = "<=30" AND *student* = "*no*" THEN *buys_computer* = "*no*"

IF *age* = "<=30" AND *student* = "*yes*" THEN *buys_computer* = "*yes*"

IF *age* = "31…40" THEN *buys_computer* = "*yes*"

IF *age* = ">40" AND *credit_rating* = "*excellent*" THEN *buys_computer* = "*yes*"

IF *age* = ">40" AND *credit_rating* = "*fair*" THEN *buys_computer* = "*no*"

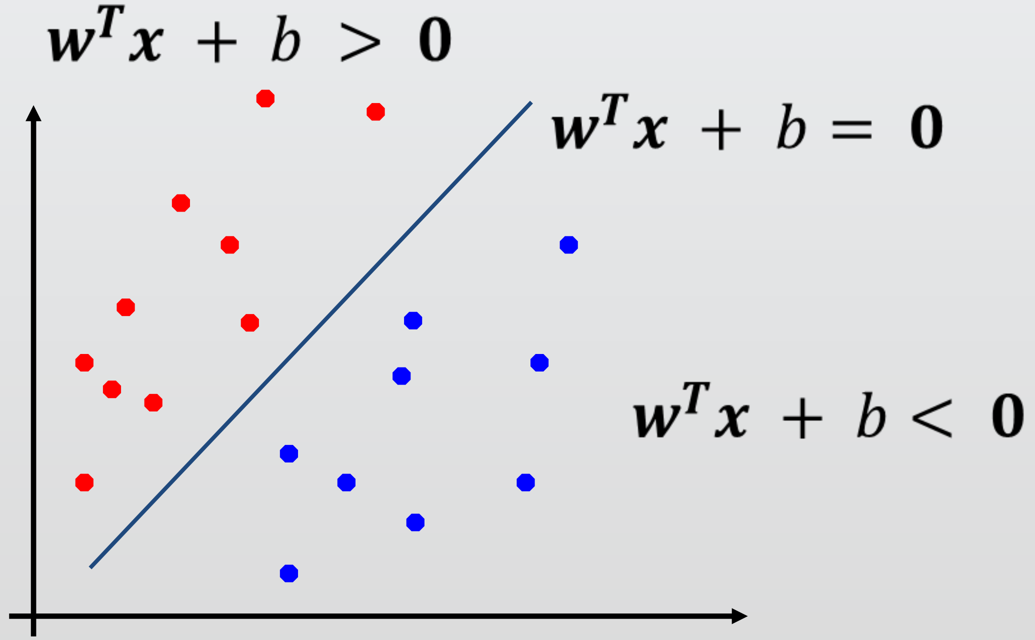
Avoid Overfitting in Classification

- The generated tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Result is in poor accuracy for unseen samples
- Two approaches to avoid overfitting
  - Prepruning: Halt tree construction early—do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Postpruning: Remove branches from a “fully grown” tree—get a sequence of progressively pruned trees
    - Use a set of data different from the training data to decide which is the “best pruned tree”

4. Support Vector Machine

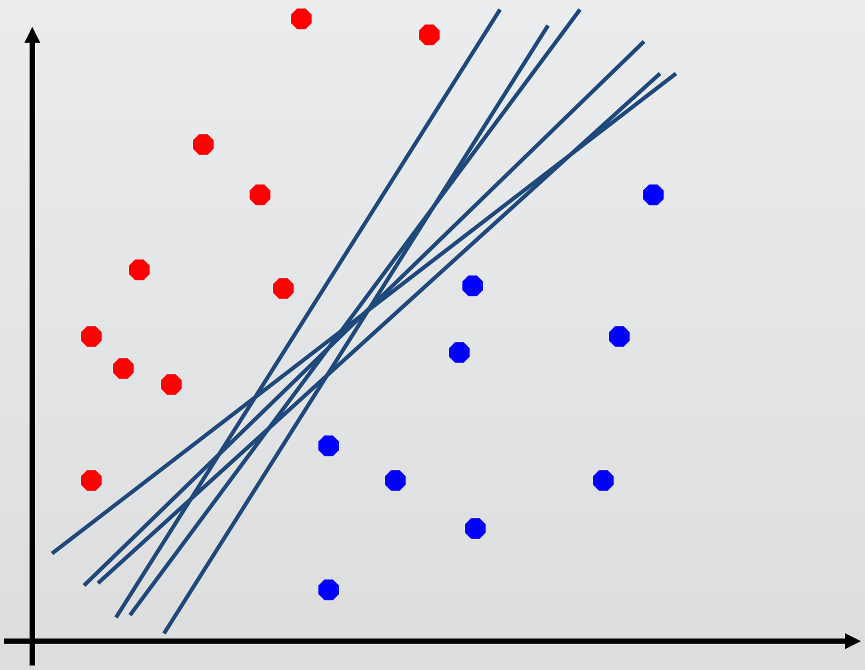
- Support Vector Machines find the “best” hyperplane that separates the two sets of points.

$$y(\boldsymbol{x}) = sign(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x} + b)$$



4. Support Vector Machine

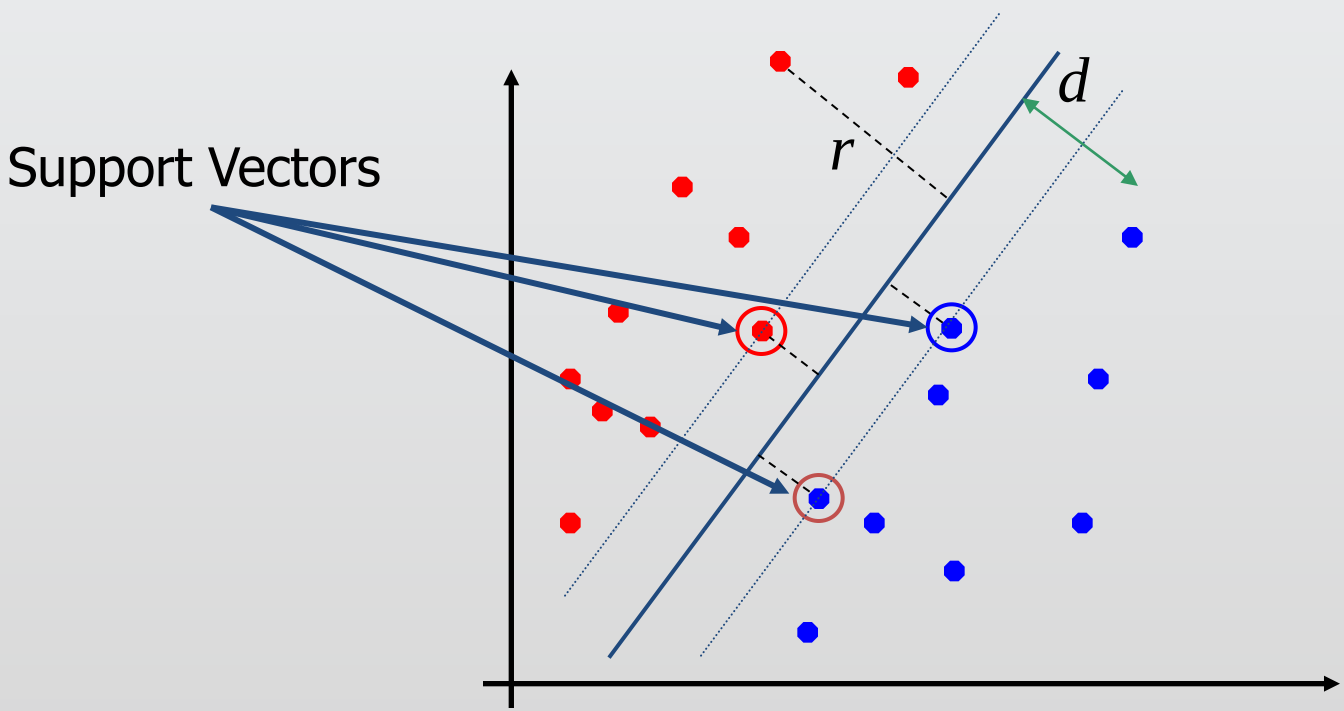
Which one is the best?



4. Support Vector Machine

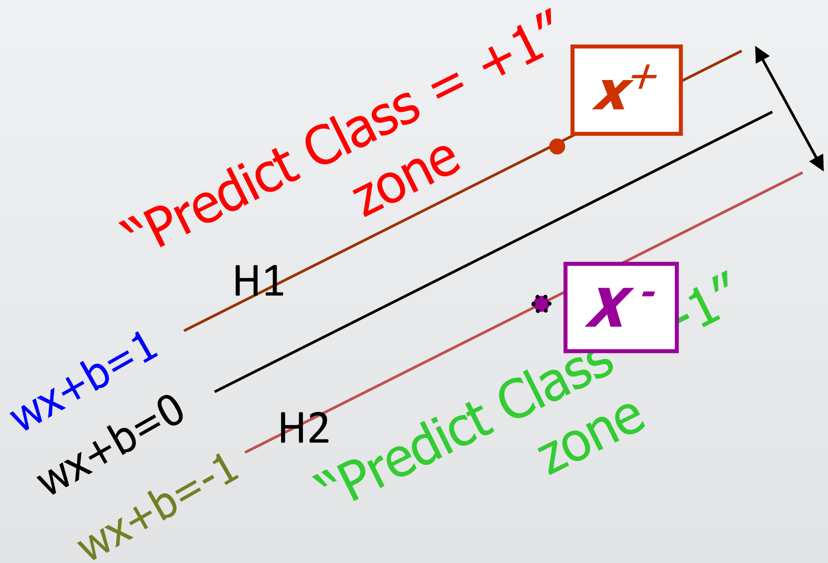
Classifier margin

- Distance from example $\mathbf{x}_i$ to the separator is $r = \dfrac{\mathbf{w}^T\mathbf{x}_i + b}{\|\mathbf{w}\|}$
- Examples closest to the hyperplane are ***support vectors***.
- ***Margin*** ρ of the separator is the distance between support vectors.



4. Support Vector Machine

Linear SVM Mathematically



ρ=Margin Width

$$\rho = \frac{(x^+ - x^-) \cdot w^T}{||w||} = \frac{2}{||w||}$$

In order to maximize the margin, we need to minimize ||w||. With the condition that there are no datapoints between H1 and H2:

w•x$_i$+b $\geq$ +1 when y_i =+1

w•x$_i$+b $\leq$ -1 when y_i =-1

Can be combined into

$$\boldsymbol{y_i(w \cdot x_i + b) \geq 1}$$

What we know:

- $\boldsymbol{w^T} \cdot \boldsymbol{x^+} + b \geq +1$
- $\boldsymbol{w^T} \cdot \boldsymbol{x^-} + b \leq -1$
- $\boldsymbol{w^T} \cdot (\boldsymbol{x^+} - \boldsymbol{x^-}) = \boldsymbol{2}$

4. Support Vector Machine

- Maximising the distance is the same as minimising $\frac{1}{2}\boldsymbol{w} \cdot \boldsymbol{w}$
- Subject to $\boldsymbol{y_i}(\boldsymbol{w} \cdot \boldsymbol{x_i} + \boldsymbol{b}) \geq \mathbf{1}$
- If we introduce Lagrange multipliers the problem becomes

$$\frac{\mathbf{1}}{\mathbf{2}}\boldsymbol{w} \cdot \boldsymbol{w} - \sum_{\mathbf{1}}^{\boldsymbol{N}} \boldsymbol{\alpha_i}(\boldsymbol{y_i}(\boldsymbol{w} \cdot \boldsymbol{x_i} + \boldsymbol{b}) - \mathbf{1})$$

4. Support Vector Machine

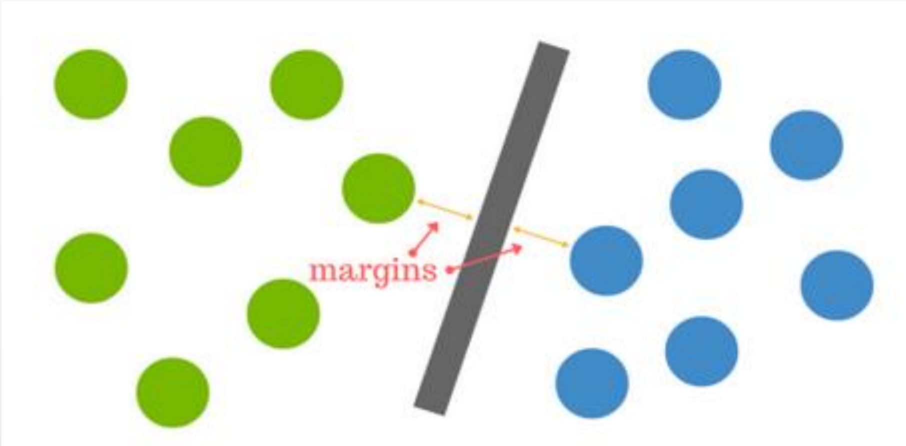
- Minimise wrt w and b
  Maximise wrt α_i
  Some math gymnastics gives

$$\sum_{1}^{N} \boldsymbol{\alpha_i y_i x_i} = \boldsymbol{w} \qquad\qquad \sum_{1}^{N} \boldsymbol{\alpha_i y_i} = \boldsymbol{0}$$

- The hyperplane is determined by very few data points
  i.e. Most of the α_i are zero
- To classify a new data point:
  - Where the α_i are non-zero
  - Only have to calculate the support vectors

$$y(\boldsymbol{x}) = sign(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x} + b)$$

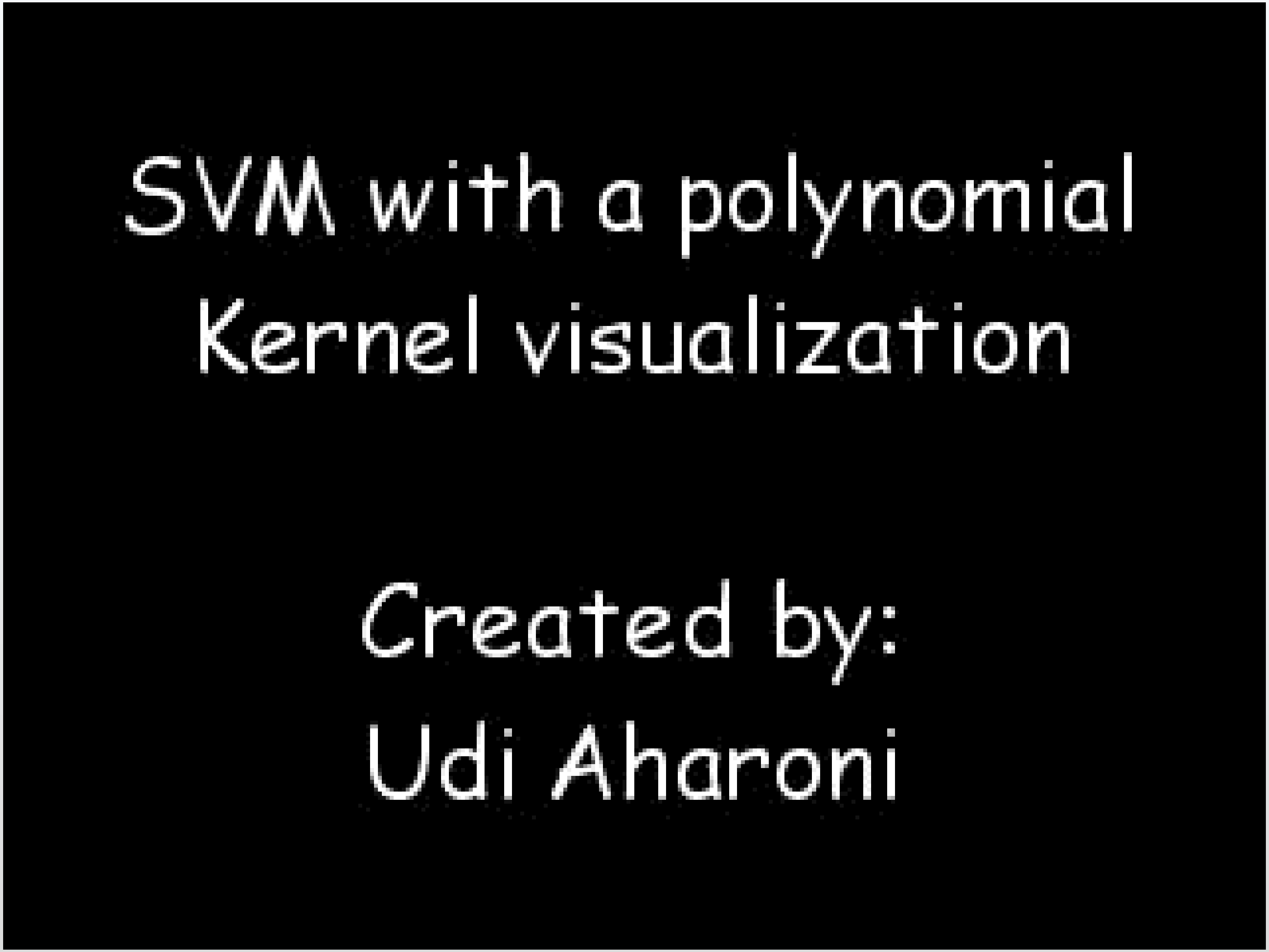
$$y(\boldsymbol{x}) = sign(\sum_{1}^{N}(\boldsymbol{\alpha_i y_i x . x_i} + b))$$



Goal

But what happens when there is no clear hyperplane?

move away from a 2d view of the
data to a 3d view.


SVM with a polynomial
Kernel visualization
Created by:
Udi Aharoni

Bayesian Classification: Why?

- Probabilistic learning: Calculate explicit probabilities for hypothesis, among the most practical approaches to certain types of learning problems.
- Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct. Prior knowledge can be combined with observed data.
- Probabilistic prediction: Predict multiple hypotheses, weighted by their probabilities.
- Standard: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured.

Bayesian Theorem

- Given training data $\mathcal{D}$, *posteriori probability of a hypothesis h*, $P(h|\mathcal{D})$ follows the Bayes theorem

$$P(h|\mathcal{D}) = \frac{P(\mathcal{D}|h)P(h)}{P(\mathcal{D})}$$

- MAP (maximum posteriori) hypothesis

$$h_{MAP} \equiv \arg\max_{h \in H} P(h|\mathcal{D}) = \arg\max_{h \in H} P(\mathcal{D}|h)P(h)$$

- Practical difficulty: require initial knowledge of many probabilities, significant computational cost

5. Naïve Bayesian Classification

- Example of training data: Play tennis? (P-positive or N-negative)

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

Bayesian classification

- The classification problem may be formalized using a-posteriori probabilities:
- $P(C|X)$ = prob. that the sample tuple $X =< A_1, \ldots, A_k >$ is of class C.
- E.g. $P(class = N \mid outlook = sunny, windy = true, \ldots)$
- Idea: assign to sample X the class label C such that $P(C|X)$ is maximal

5. Naïve Bayesian Classification

Estimating a-posteriori probabilities

- Bayes theorem:

$$P(C|X) = P(X|C) \cdot P(C) \,/\, P(X)$$

- $P(X)$ is constant for all classes
- $P(C)$ = relative freqency of class C samples
- C such that $P(C|X)$ is maximum =

  C such that $P(X|C) \cdot P(C)$ is maximum

Naïve Bayesian Classification

- Naïve assumption: attribute independence

$$P(A_1, \ldots, A_k|C) = P(A_1|C) \cdot \cdots \cdot P(A_k|C)$$

5. Naïve Bayesian Classification

Play-tennis example: estimating $P(x_i|C)$

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

P(p) = 9/14	
P(n) = 5/14	

outlook	
P(sunny|p) = 2/9	P(sunny|n) = 3/5
P(overcast|p) = 4/9	P(overcast|n) = 0
P(rain|p) = 3/9	P(rain|n) = 2/5
temperature	
P(hot|p) = 2/9	P(hot|n) = 2/5
P(mild|p) = 4/9	P(mild|n) = 2/5
P(cool|p) = 3/9	P(cool|n) = 1/5
humidity	
P(high|p) = 3/9	P(high|n) = 4/5
P(normal|p) = 6/9	P(normal|n) = 2/5
windy	
P(true|p) = 3/9	P(true|n) = 3/5
P(false|p) = 6/9	P(false|n) = 2/5

5. Naïve Bayesian Classification

Play-tennis example: classifying X

- An unseen sample X **= <rain, hot, high, false>.** Predict $P(C|X)$

$$P(X|p) \cdot P(p) = \\ P(rain|p) \cdot P(hot|p) \cdot P(high|p) \cdot P(false|p) \cdot P(p) \\ = 3/9 \cdot 2/9 \cdot 3/9 \cdot 6/9 \cdot 9/14 = 0.010582$$

$$P(X|n) \cdot P(n) = \\ P(rain|n) \cdot P(hot|n) \cdot P(high|n) \cdot P(false|n) \cdot P(n) \\ = 2/5 \cdot 2/5 \cdot 4/5 \cdot 2/5 \cdot 5/14 = \mathbf{0.018286}$$

- Sample X is classified in class $\boldsymbol{n}$ **(don't play)**

Assessing Classifier

- Contingency table or Confusion Matrix
- Accuracy, Precision and Recall
- ROC curves and Area Under the Curve
- Cross Validation

Contingency table or Confusion Matrix

- A confusion matrix is a table that is often used to **describe the performance of a classification model** (or "classifier") on a set of test data for which the true values are known.

n=165	Predicted: NO	Predicted: YES
Actual: NO	50	10
Actual: YES	5	100

Contingency table or Confusion Matrix

- Another example of Confusion Matrix

N=27

		Predicted		
		Cat	Dog	Rabbit
Actual class	Cat	5	3	0
	Dog	2	3	1
	Rabbit	0	2	11

Accuracy, Precision, and Recall

- Suppose a computer program for recognizing dogs in scenes from a video identifies 7 dogs in a scene containing 9 dogs and some cats.

- If 4 of the identifications are correct, but 3 are actually cats, the program's precision is 4/7 while its recall is 4/9.

Accuracy, Precision, and Recall

- A search engine returns 30 pages with only 20 of which were relevant while failing to return 40 additional relevant pages.

- Its precision is 20/30 = 2/3 while its recall is 20/60 = 1/3. So, in this case, precision is "how useful the search results are", and recall is "how complete the results are".

Accuracy, Precision, and Recall

- Let's now define the most basic terms, which are whole numbers (not rates):

- **true positives (TP):** These are cases in which we predicted yes (they have the disease), and they do have the disease.
- **true negatives (TN):** We predicted no, and they don't have the disease.
- **false positives (FP):** We predicted yes, but they don't actually have the disease. (Also known as a "Type I error.")
- **false negatives (FN):** We predicted no, but they actually do have the disease. (Also known as a "Type II error.")

n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

- **Accuracy: Overall, how often is the classifier correct?**
- **(TP+TN)/total = (100+50)/165 = 0.91**

- **Precision: When it predicts yes, how often is it correct?**
- **TP/predicted yes = 100/110 = 0.91**

- **Recall: When it's actually yes, how often does it predict yes?**
- **TP/actual yes = 100/105 = 0.95**

ROC curves and Area Under the Curve

- ROC is the most commonly used way to **visualize the performance of a binary classifier**.

- Area Under the Curve (AUC) is (arguably) the best way to summarize its performance in a single number.

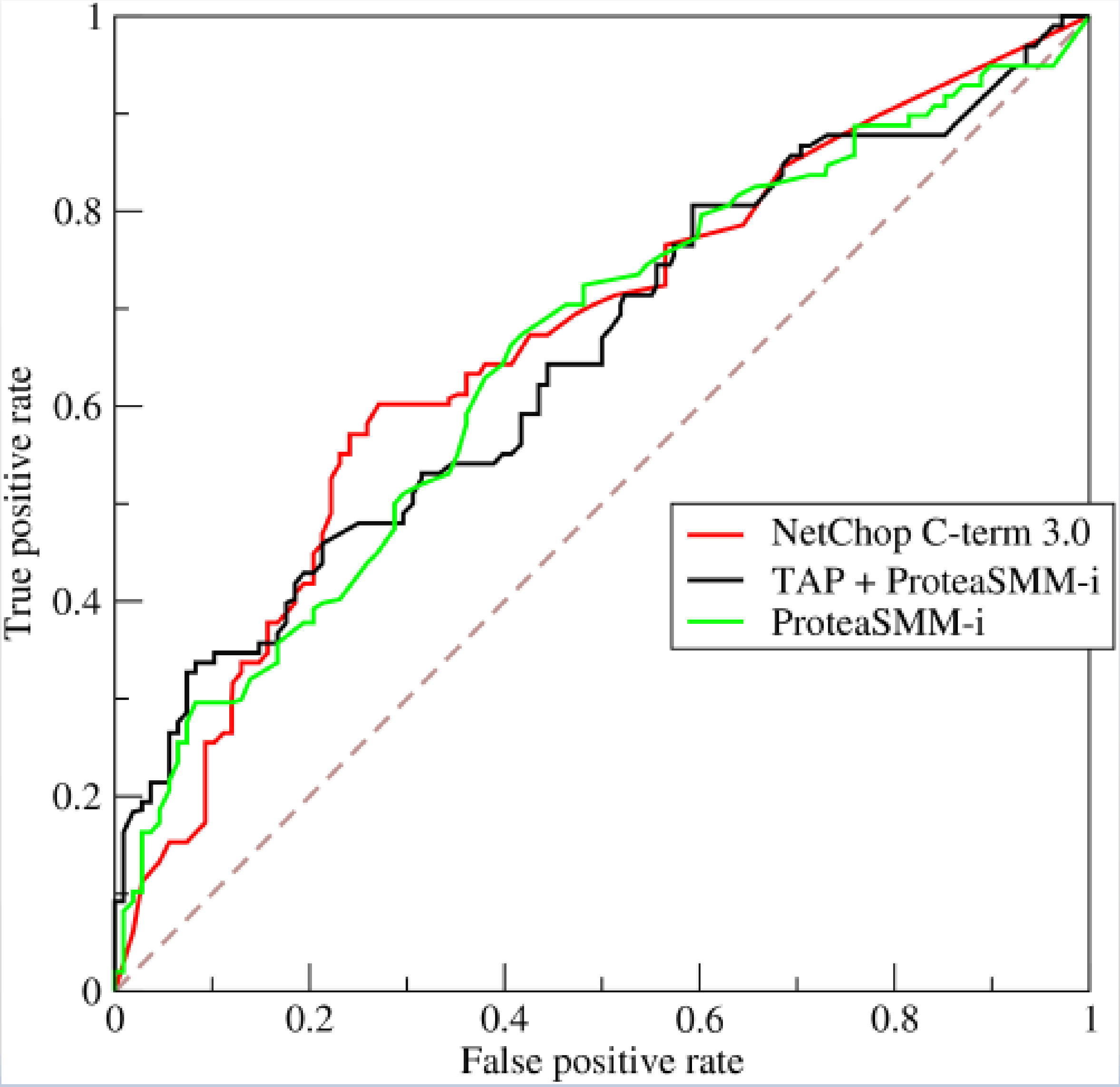
ROC curves and Area Under the Curve

- ROC is the most commonly used way to **visualize the performance of a binary classifier**.

- Area Under the Curve (AUC) is (arguably) the best way to summarize its performance in a single number.

ROC curves and Area Under the Curve

- Example of ROC Curve



Cross Validation

- Labelled data sets are difficult to get
- Leave one out cross validation
  - Leave one example out and test the classification error on that one
  - Iterate through the data set
  - Compute the average classification error
- K-fold cross validation
  - Split the data set in to K sub-sets, leave one out
  - 10 fold cross validation common