COMP3308/3608, Lecture 5

ARTIFICIAL INTELLIGENCE

Introduction to Machine Learning.
K-Nearest Neighbor.
Rule-Based Algorithms: 1R

Reference: Russell and Norvig, p.693-697, 738-741

Witten, Frank, Hall and Pal, ch. 1-2, ch.4: p.91-96, 135-141

Outline

- **Introduction to Machine Learning (ML)**
  - **What is learning and ML?**
  - **Classification of ML methods**
- **K-nearest neighbor**
- **Learning rules – 1R algorithm**

Learning and Machine Learning

- ***Machine Learning (ML)*** **is the area of AI that is concerned with writing computer programs that can learn from**
  - **examples**
  - **domain knowledge**
  - **user feedback**
- ***ML*** **is the core of AI – without an ability to learn, a system cannot be considered intelligent**

- **What does it mean *to learn*? What do you understand by *learning*? When are you sure that you have *learned something*?**

Definitions of learning

- **Definitions of *learning* from dictionary:**
  - **1) To get knowledge of something by study, experience, or being taught**
  - **2) To become aware by information or from observations**
  - **3) To commit to memory**
  - **4) To be informed of, ascertain; to receive instruction**

- ***Learning* is making useful changes in our minds (*Marvin Minsky*)**

- **But when talking about computers (i.e. *ML*) these definitions have shortcomings!**

Learning Definitions – Shortcomings

- **1) and 2) - impossible to test if learning has been achieved or not**
  - **How do you know if a machine has *got knowledge of…*?**
  - **Or if it has *become aware*…? Can computers be aware or conscious – philosophical issue**

- **3) and 4) - committing to memory, receiving instructions**
  - **Sound too passive; trivial tasks for computers**
  - **You can receive instructions and memorize things without being able to benefit from them, e.g. not able to apply new knowledge to new situations**

Learning – Operational Definition

- ***Learning*** **denotes changes in the system that are adaptive in the sense that they enable the system to do the same task or tasks drawn from the same population *more efficiently* and *more effectively* the next time (*Herbert Simon*)**
  - **Computers learn when they change their behavior in a way that makes them perform better in the future**
  - **Ties learning to *performance* rather than *knowledge***

Types of ML

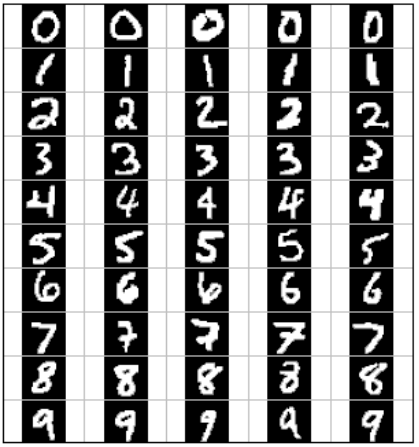
- **Three main types**
  - **Supervised**
  - **Unsupervised**
  - **Reinforcement**
- **Forth type: associations learning – developed within the database community in early 90s**

Supervised ML Tasks - Examples

- ***Classification task*: recognizing post codes (=recognizing digits)**

  <u>Given:</u> Handwritten digits and their corresponding label (class)

  <u>Task:</u> Build a classifier that can recognise new handwritten digits



For the image:

- 5 people wrote the numbers from 0 to 9

- 1 example = 1 handwritten digit (50 in total)

- each example is labelled with the digit it represents (0,1,…or 9) => there are 10 classes

image ref: http://www-inst.eecs.berkeley.edu/~cs188/fa06/projects/classification/4/writeup/img2.gif

- ***Regression task:* predicting the exchange rate of AUD**

  <u>Given:</u> data from previous years (economic indicators, political events), with their corresponding *exchange rate*

  <u>Task:</u> Build a classifier to predict the exchange rate for future days

- **The difference between classification and regression is in the type of the class (target) variable – nominal vs numeric**

Supervised Learning - Definition

- **Given: a set of pre-classified (labelled) examples {x,y}, x – input vector, y - target output**
- **Task: learn a *function (classifier, model)* which encapsulates the information in these examples (i.e. maps x->y) and can be used predictively**
  - **i.e. to predict the value of one variable (y) from the known values of other variables (x)**

Why is it called supervised?

- **2 types of supervised learning**
  - ***Classification*: the variable to be predicted is *categorical* (i.e. its values belong to a pre-specified, finite set of possibilities)**
  - ***Regression*: the variable to be predicted is *numeric***
- **Examples of supervised algorithms: 1R, k-NN, DTs, NB, neural networks (perceptron, backpropagation, deep), SVM**

Classification - Ex.1

input data

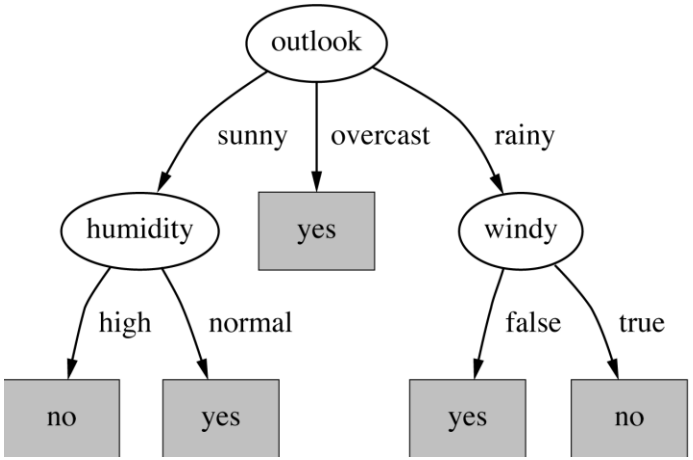
input vector, with 4 features

target class

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

model (classifier)

model 1: decision tree



model 2: rules

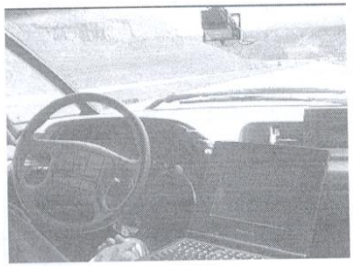
if `outlook=sunny` **then** `play=no`
elseif `outlook=overcast` **then** `play=yes`
elseif `outlook=rainy` **then** `play=yes`

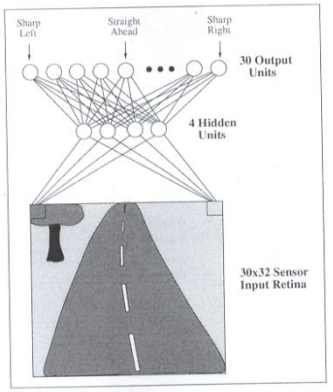
model 3: …

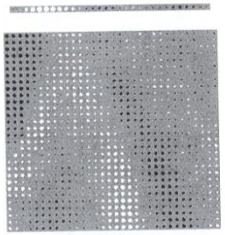
We can learn different types of models

Classification Ex.2 - Driving Motor Vehicles

- **ALVINN, Pomerleau et al., 1993**
- **Driving a van along a highway**
- **Uses a neural network classifier**
- **Data**
  - **Input vectors: derived from the 30x32 image (black and white values)**
  - **Outputs (classes): 32 classes, corresponding to the turning directions - left, straight, right; different degrees**
  - **1 labelled example is: input vector + class label (turning direction)**







- ***The machine that changed the world***
  http://www.youtube.com/watch?v=tXMaFhO6dIY
  Early NN: minute 39-41, ALVINN and NetTalk: minute 41-46

Classification - More Examples

- **Banking 1: Is a mortgage application a good or bad credit risk?**
- **Banking 2: Is a credit card transaction fraudulent or not?**
- **Medicine: Is a particular disease present or not?**
- **Law: Was a given will written by the real diseased person or by somebody else?**
- **Security: Is a given behavior a possible terrorist threat?**

Regression - Example

input

model

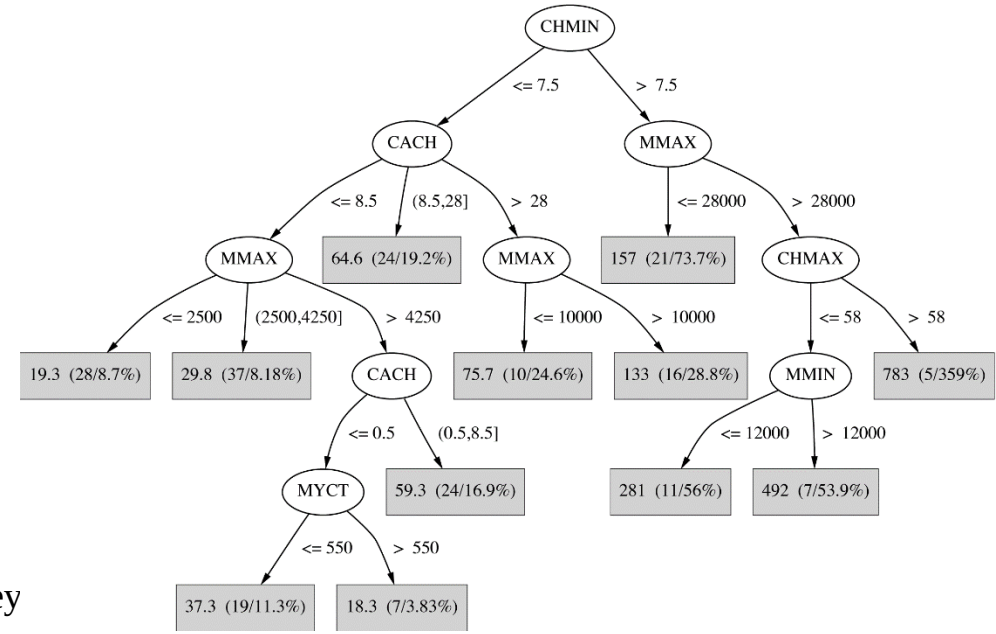
CPU performance data

	cycle time (ns)	main memory (Kb)		cache	channels		perfor-mance
	MYCT	MMIN	MMAX	CACH	CHMIN	CHMAX	PRP
1	125	256	6000	256	16	128	198
2	29	8000	32000	32	8	32	269
3	29	8000	32000	32	8	32	220
4	29	8000	32000	32	8	32	172
5	29	8000	16000	32	8	16	132
...							
207	125	2000	8000	0	2	14	52
208	480	512	8000	32	0	0	67
209	480	1000	4000	0	0	0	45

linear regression

```
PRP =  - 56.1 + 0.049 MYCT  + 0.015 MMIN
       + 0.006 MMAX + 0.630 CACH
       - 0.270 CHMIN + 1.46 CHMAX
```

regression tree



Task: Predict computer performance

Irena Koprinska, irena.koprinska@sydney

More Regression Examples

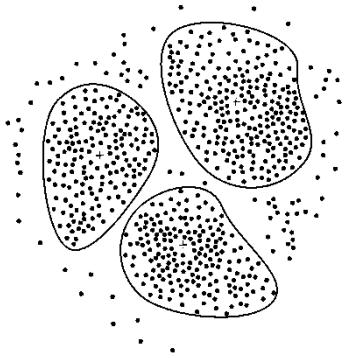
- **Predict electricity demand in the next hour from previous demands**
- **Predict retirement savings from current savings and market indicators**
- **Predict the house prices in Sydney in 2020**
- **Predict the sales of a new product based on advertisement expenditure**
- **Predict wind velocity based on temperature, humidity, pressure**

Reinforcement Learning

- **Each example has a score (grade) instead of correct output**
- **Much less common that supervised learning**
- **Most suited to control systems applications**

Unsupervised Learning (Clustering)

- **Given: a collection of input vectors x**
  - **no target outputs y are given**
- **Task: group (cluster) the input examples into a finite number of clusters so that the examples**
  - **From each cluster are similar to each other**
  - **From different clusters are dissimilar to each other**
- **Examples of clustering algorithms: k-means, nearest neighbor, hierarchical clustering**



Clustering – Example

- **Customer profiling**

  A department store wants to segment its customers into groups and create a special catalog for each group. The attributes for the grouping included customer's income, location and physical characteristics (age, height, weight, etc.).
- **Clustering was used to find clusters of similar customers**
- **A catalogue was created for each cluster based on the cluster characteristics and mailed to each customer**

Associations Learning

- **Find relationships in data**
  - ***market-basket analysis* - find combinations of items that occur typically together**
  - ***sequential analysis* – find frequent sequences in data**

Market-Basket Analysis – Example

- **Uses the information about <u>what</u> customers buy to give us insight into <u>who</u> they are and <u>why</u> they make certain purchases**

  - **Ex.1. A grocery store owner is trying to decide if to put bread on sale. He generates association rules and finds what other products are typically purchased with bread. A particular type of cheese is sold 60% of the time the bread is sold and a jam is sold 70% of the time. Based on these findings, he decides:**
    - **1) to place some cheese and jam at the end of the aisle where the bread is**
    - **2) not to place either of these 3 items on sale at the same time.**

Frequent Sequences – Example

- **Goal: Given a sequence of events, find frequent sub-sequences**
  - **These patterns are similar to market-basket analysis but the relationship is based on time**
- **Ex. 1: The webmaster at a company X periodically analyses the web log data to determine how the users of X browse them. He finds that 70% of time the users of page A follow one of the following patterns:**
  - **A->B->C**
  - **A->D->B->C**
  - **A->E->B->C**

  - **=> A-> C if a frequent pattern**
  - **=> he then decides to add a link from page A to C**

- **Ex.2: Finding sub-sequences in DNA data for particular species**

More ML Applications

- **Fraud detection**
  - **Health care - medical insurance fraud, inappropriate medical treatment**
  - **Credit card services, phone card and retail fraud**
  - **Data: historical transactions and other data**
- **Sport - analyzing game statistics (shots blocked, assists and fouls) to gain competitive advantage**
  - **"When player X is on the floor, player Y's shot accuracy decreases from 75% to 30%"**
- **Astronomy**
  - **JPL and the Palomar Observatory discovered 22 quasars using ML**
- **Web applications**
  - **Mining web logs to discover customer preferences and behavior, analyze effectiveness of web marketing, improve web site organization**

Why is ML Important?

adapted from http://www.site.uottawa.ca/~nat/Courses/csi5387.html

- Some tasks cannot be defined well, except by examples
  - e.g. recognizing people (man vs women), handwritten digits, etc.
- The amount of knowledge available about certain tasks is too big for explicit encoding into rules or difficult to extract from experts)
  - e.g. medical diagnosis - easier to learn from cases (symptoms-> diagnosis)
- Need for adaptation
  - Humans often produce machines that do not work as well as desired in the environments in which they are used
  - Environments change over time - e.g. a spam email filter; the characteristics of spam email change over time
- Relationships and correlations can be hidden within large amounts of data. ML and Data Mining may be able to find these relationships.

Machine Learning vs Data Mining

- ***Data Mining (DM):*** **search for hidden patterns in *large* datasets**
  - **these patterns should be meaningful, useful and actionable**
- **Most of the techniques used for DM have been developed in ML**
  - **DM deals with large and multidimensional data, ML not necessary**
  - **DM is applied ML**
- **Motivation for DM**
  - **Data explosion – huge databases**

    - **due to automated data collection tools and mature database technology**
    - **examples: supermarket transaction data, credit card usage data, mobile usage data, government statistics, molecular databases, medical records, Wikipedia and other large test collections, etc.**
  - **We are drowning in data but starving for knowledge!**

What jobs will disappear in the 21 century?

mailcarriers

prison guards

teachers

stockbrokers

orthodontists

insurance and retail estate agents, autodealesrs

CEOs

truckers

fathers

housekeepers

What will be the 10 hottest jobs of the 21 century?

pharmers (vaccine carrying tomato)

tissue engineers

Turing testers

hot-line handyman (remote diagnostics)

gene programmers

data miners

narrowcasters (personalised ads)

Time magazine, June 26, 2000

If you were born in 2012….you would work in Data Mining

- **SMH, 6 April 2012**

http://www.smh.com.au/lifestyle/life/whats-the-future-baby-20120405-1wfez.html

- **"My schooling will become more interesting as I go, as today's digital natives grow up to become teachers. They'll know how to use all the gadgets at their disposal to make learning easier, fun and compatible with my short attention span. I'll always be switched on. I'll crowdsource my big decisions, taking votes among my closest 30 or so net friends. I'll do a university degree of course - just about everyone will. I'll probably work in a knowledge-based service industry which will depend on mining data from customer transactions in unimaginable volumes to determine which services to provide to whom, where and when".**

- **Side question: Technology vs "chalk & talk" teaching – which one is better?**

https://www.openlearning.com/educationist/ChalkAndTalkOrTechnologyDoIHaveAChoicePartOne

The 10 Toughest Jobs To Fill In 2016 – Data Scientist

- **Forbes magazine, 24 September 2015**

https://www.forbes.com/sites/susanadams/2015/09/24/the-10-toughest-jobs-to-fill-in-2016/#d665d4a6fcca

- **"With the explosion of big data and the need to track it, employers keep on hiring data scientists. But qualified candidates are in short supply. The field is so new, the Bureau of Labor Statistics doesn't even track it as a profession. Yet thousands of companies, from startups that analyze credit card data in order to target marketing and advertising campaigns, to giant corporations like Ford Motor F +0.26% and Price WaterhouseCoopers, are bringing on scores of people who can take gigantic data sets and wrestle them into usable information. As an April report from technology market research firm Forrester put it, *"Businesses are drowning in data but starving for insights."***

The 10 Toughest Jobs To Fill In 2017 – Data Scientist (again)

- **Forbes magazine, 8 February 2017**

https://www.forbes.com/sites/karstenstrauss/2017/02/08/the-toughest-jobs-to-fill-in-2017/#44c245ee7f14

- **"One job that made the list this year – as it did last year – is *data scientist*. Says Kensing: "Universities now are just starting to integrate specific majors for that field. It's got a high growth outlook but right now it's still a burgeoning field." According to the numbers, the data scientist occupation has a 16% growth outlook over the next eight years, and right now the median annual salary for that position is more than $128,000."**

Top Emerging Jobs in 2020

- **Forbes magazine, 5 January 2020**

https://www.forbes.com/sites/louiscolumbus/2020/01/05/ai-specialist-is-the-top-emerging-job-in-2020-according-to-linkedin/?sh=1d32f6c37495

- **"Artificial Intelligence Specialist - Artificial Intelligence and Machine Learning have both become synonymous with innovation, and LinkedIn data shows that's more than just buzz. Hiring growth for this role has grown 74% annually in the past 4 years and encompasses a few different titles within the space that all have a very specific set of skills despite being spread across industries, including artificial intelligence and machine learning engineer. According to Indeed, Machine Learning Engineer job openings grew 344% between 2015 to 2018 and have an average base salary of $146,085 ...**
- **Data Scientist – LinkedIn is seeing a 37% annual increase in demand for Data Scientists and related technical positions today. Data Science is another field that has topped the LinkedIn Emerging Jobs list for three years running. It's a specialty that's continuing to grow significantly across all industries. …"**

Classification

K-Nearest Neighbor Algorithm

Classification Setup Again

- **Given: a set of pre-labelled examples**
  - **14 examples**
  - **4 attributes: *outlook, temperature, humidity* and *windy*)**
  - **the class is *play* (values: *yes, no*)**
- **Task: Build a model (classifier) that can be used to predict the class of <u>new</u> <u>(unseen)</u> examples**

  e.g. predict the class (*yes* or *no*) of

  outlook=sunny, temp=hot, humidity=low, windy=true

attributes (features, variables) class

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

- **Examples used to build the model are called *training data***
- **Success is measured empirically on another set called *test data***
  - **Test data hasn't been used to build the classifier; it is also labelled**
  - **Performance measure: *accuracy* – proportion of correctly classified test examples**

Nominal and Numeric Attributes

- **2 types of attributes (features):**
  - ***numeric (continuous)*** **- their values are numbers**
  - ***nominal (categorical)*** **- their values belong to a pre-specified, finite set of possibilities**

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

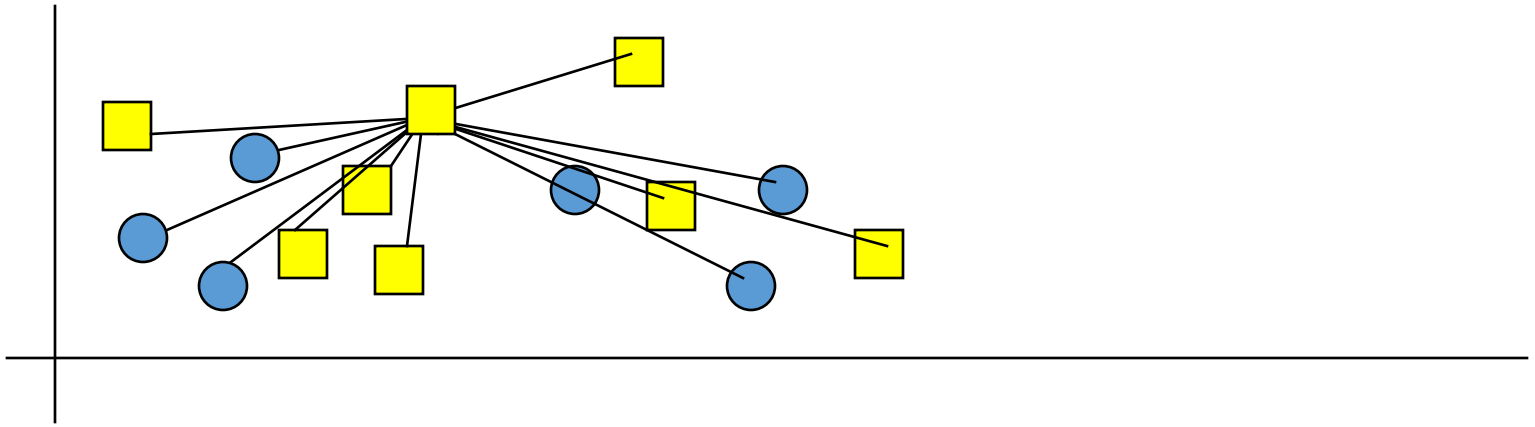
nominal

sepal length	sepal width	petal length	petal width	iris type
5.1	3.5	1.4	0.2	iris setosa
4.9	3.0	1.4	0.2	iris setosa
4.7	3.2	1.3	0.2	iris setosa
...				
6.4	3.2	4.5	1.5	iris versicolor
6.9	3.1	4.9	1.5	iris versicolor
5.5	2.3	4.0	1.3	iris versicolor
6.5	2.8	4.6	1.5	iris versicolor
6.3	3.3	6.0	2.5	iris virginica
5.8	2.7	5.1	1.9	iris virginica
...				

numeric

Nearest Neighbor Algorithm

- **Remember all your training data**
- **To classify a new, unlabelled example:**
  - **Find the nearest training data example**
  - **Return the answer (class) associated with it**
- **"Nearest" is determine based on a distance measure**



Commonly Used Distance Measures

- **A, B - examples with attribute values a_1, a_2,..., a_n & b_1, b_2,..., b_n**

  e.g. A= [1, 3, 5], B=[1, 6, 9]

- **Euclidean distance – most frequently used:**

$$D(A,B) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \ldots + (a_n - b_n)^2}$$

- **Manhattan distance:**

$$D(A,B) = |a_1 - b_1| + |a_2 - b_2| + \ldots + |a_n - b_n|$$

- **Minkowski distance – generalization of Eucledian and Manhattan:**

$$D(A,B) = (|a_1 - b_1|^q + |a_2 - b_2|^q + \ldots + |a_n - b_n|^q)^{1/q}$$

q – positive integer

q=? Minkowski = Euclidean distance?

- **q=? Minkowski =Manhattan distance?**

Nearest Neighbor - Example

- **Given is the following dataset where *years_experince* is the only attribute and salary is the class. What will be the prediction for the new example years_experince=5 using the Nearest Neighbor algorithm with the Manhattan distance?**

	years_experience	salary
1	4	low
2	9	medium
3	8	medium
4	15	high
5	20	high
6	7	medium
7	12	medium
8	23	medium
9	1	low
10	16	medium

Answer

- **Given is the following dataset where *years_experince* is the only attribute and salary is the class. What will be the prediction for the new example years_experince=5 using the Nearest Neighbor algorithm with the Manhattan distance?**

	years_experience	salary	D(new, current)
1	4	low	1
2	9	medium	4
3	8	medium	3
4	15	high	10
5	20	high	15
6	7	medium	2
7	12	medium	7
8	23	medium	18
9	1	low	4
10	16	medium	11

- **The closest neighbor is ex.1 (distance=1) => Nearest Neighbor will predict *salary=low***

Need for Normalization

- **Different attributes are measured on different scales**
- **When calculating the distance between 2 examples, the effect of the attributes with the smaller scale will be less significant than those with the larger**
- **Example: Predict the car petrol consumption based on 2 attributes - car weight [kg] and number of cylinders**

  Ex1: 6000, 4

  Ex2: 1000, 2

  D(ex1,ex2)=|6000-1000|+|4-2|=5002

  The effect of *number of cylinders* will be lost as *car weight* dominates the calculation!

- **Solution: Normalize attribute values between 0 and 1**

Normalization

- **Solution: Normalize (scale) attribute values between 0 and 1**

$$a_i = \frac{v_i - \min v_i}{\max v_i - \min v_i}$$

v_i – the actual value of attribute *i*

min **v_i and** ***max*** **v_i – the minimum and maximum value of v_i taken over all examples in the training set (for attribute *i*)**

Ex.: Consider a dataset with 4 examples
values of the attribute *car weight*:
original: 1000, 6000, 8000, 10000
normalized: 0, 0.55, 0.77, 1

values of the attribute *number of cylinders*:
original: 2, 4, 6, 4
normalised: 0, 0.5, 1, 0.5

	original	**->**	**normalised**
ex1:	**1000, 2**		**0, 0**
ex2:	**6000, 4**		**0.55, 0.5**
ex3:	**8000, 6**		**0.77, 1**
ex4:	**10000, 4**		**1, 0.5**

dist(ex1,ex2)=|0-0.55|+|0-0.5|=1.05
Both attributes are contributing,
car weight **doesn't dominate**

Lazy Learning

- **Nearest neighbor is also called instance-based learning**
- **Nearest neighbor is an example of *lazy learning***
  - **stores all training examples and does not build a classifier until a new (unlabeled) example needs to be classified**
- **Opposite to eager learning**
  - **Construct classifier before receiving new examples to classify**
  - **Examples: 1R, decision trees, Naïve Bayes, neural networks, SVM**
- **Lazy classifiers are faster at training but slower at classification**
  - **training = memorizing**
  - **all computations are delayed to classification time**

Computation Complexity – Time and Space

- **Training is fast – no model is built; just storing training examples**
- **Time complexity**
  - **During the classification phase (prediction for a new example), we need to compare each new example with each training example**
  - **If *m* training examples with dimensionality *n* => lookup for 1 new example takes *m*n* computations, i.e. *O(mn)***
- **Memory requirements: *O(mn)* - we need to remember each training example**
- **=> Impractical for large datasets due to time and memory requirements**
  - **Variations - using more efficient data structures (KD-trees and ball trees), see Witten and Frank p.132-137, Norvig and Russell p.739-741, allowing to forget some examples and also not to compare with some (Hand et al., *Handbook of Data Mining*, p.352)**

K-Nearest Neighbor

- **Using the closest 1 example (to predict the class of the new example) is called 1-Nearest Neighbor**
- **Using the *k* closest examples is called k-Nearest Neighbor**
  - **majority voting is used to take the decision – e.g. 3-Nearest Neighbor: circle, circle, square -> the new example is classified as circle**
  - **increases the robustness of the algorithm to noisy examples**
  - **increases the computational time (slightly)**
- **K-Nearest Neighbor is very sensitive to the value of *k***
  - **rule of thumb: $k \leq \text{sqrt}(\#\text{training\_examples})$**
  - **commercial packages typically use k=10**
- **Can be used also for regression tasks**
  - **The classifier returns the average value of the target variable associated with the *k* nearest neighbors of the new example, i.e.:**

prediction for new example X_{new} =

$(\text{target-value-neighbour}_1 + \ldots + \text{target-value-neighbour}_k)/k$

K-Nearest Neighbor for Our Example

- **What will be the prediction for the new example years_experince=5 using the 1- and 3-Nearest Neighbor with the Manhattan distance?**

	years_experience	salary	D(new, current)
1	4	low	1
2	9	medium	4
3	8	medium	3
4	15	high	10
5	20	high	15
6	7	medium	2
7	12	medium	7
8	23	medium	18
9	1	low	4
10	16	medium	11

- **1-Nearest Neighbor will predict *salary=low* (we saw this before)**
- **3-Nearest Neighbor will predict *salary=medium***
  - **closest 3 neighbors: ex.1 (*low*), ex.6 (*medium*) and ex.3 (*medium*) => the majority prediction is medium**

Distance for Nominal Attributes

ex.1: (red, new)

ex2: (blue, new)

distance (ex1, ex2) =?

- **Simple solution:**
  - **1 - the difference between two attribute values that are not the same**
  - **0 – the difference between two attribute values that are the same**
  - **no need for normalization**

- **=>manhattan_distance(ex1,ex2)=1+0=1**

Distance for Nominal Attributes with Missing Values

- **Principle: a missing value is maximally different from any other value**

  - **Case 1: Nominal attributes**

  ex.1: (red, ?) // "?" means missing value

  ex2: (blue, new)

  d(ex1, ex2)=?

  - **1 – one or both of the attribute values are missing**
  - **0 - the two attribute values are the same and are non-missing**

  - **=>mantattan_distance(ex1,ex2)=1+1=2**

Distance for Numeric Attributes with Missing Values?

- **Principle: a missing value is maximally different from any other value**
  - **Case 2: numeric attributes**

ex.1: (0.2, ?) // "?" means missing value

ex2: (0.7, 0.1)

d(ex1, ex2)=?

- **Firstly, normalise the data**
- **If the two values are missing, the difference between them is 1**
- **If only one of them is missing, the difference is either the other value or 1 minus that value, whichever is larger**
  - **i.e. we are applying the principle that the difference should be as large as possible**
- **=>Manhattan_distance(ex1,ex2)=|0.2-0.7|+0.9=1.4**

Variations: Weighted Nearest Neighbor

- **Idea: Closer neighbors count more than distant neighbors**
- **Distance-weighted nearest-neighbor algorithm**
  - **Find the *k* nearest neighbors**
  - **Weight their contribution based on their distance to the new example**
    - **bigger weight if they are closer**
    - **smaller weight if they are further**

$$w_i = \frac{1}{D(X_{new}, X_i)}$$

- **More often used for regression tasks**
  - **e.g. prediction for new example X_{new} =**

    w_1*target-value-neighbour$_1$+…+ w_k*target-value-neighbour$_k$
- **Another variation: instead of using only the *k* nearest neighbors, use all training examples**
  - **the very distant will have very little effect (weight will be too small)**
  - **disadvantage – slower algorithm**

Curse of Dimensionality

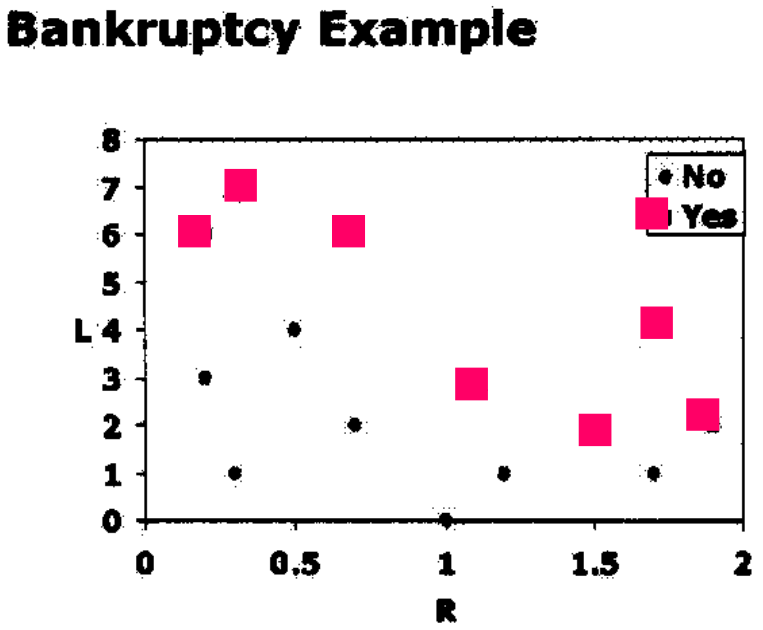
adapted from 6.034 AI, MIT

- **High dimensional data causes problems for all classifiers - *overfitting***
- **Nearest neighbors algorithms are great in low dimensions (up to 6) but as the dimensionality increases they become ineffective**
- **In high dimensions most examples are**
  - **far from each other**
  - **close to the boundaries**
    - **imagine sprinkling data points uniformly within a 10-dim unit hypercube (cube whose sides are of length 1). 90% of the data are outside a cube with sides >0.63**
- **=> the notion of *nearness* that is very effective in low dimensional space becomes *ineffective in a high dimensional space***
- **=> nearest neighbor classification cannot be trusted for high-dimensional data**
- **Solution: feature selection to reduce dimensionality**

1-Nearest Neighbor Decision Boundary

Example taken from 6.034 AI, MIT

- What is the decision boundary of 1-NN algorithm?
- Think of each example as a point in the feature space

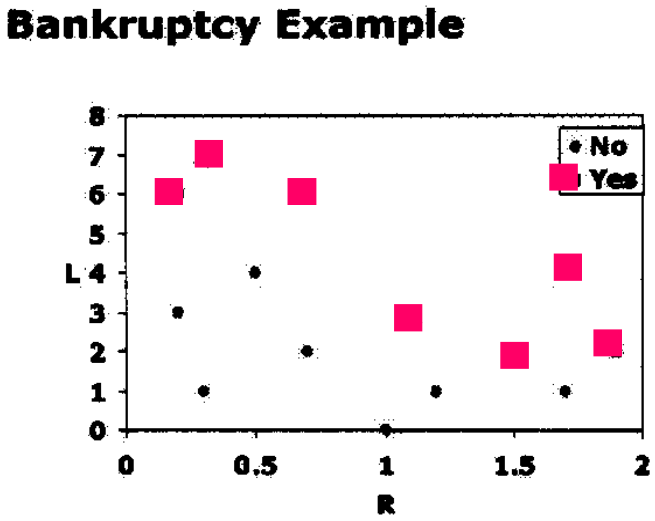


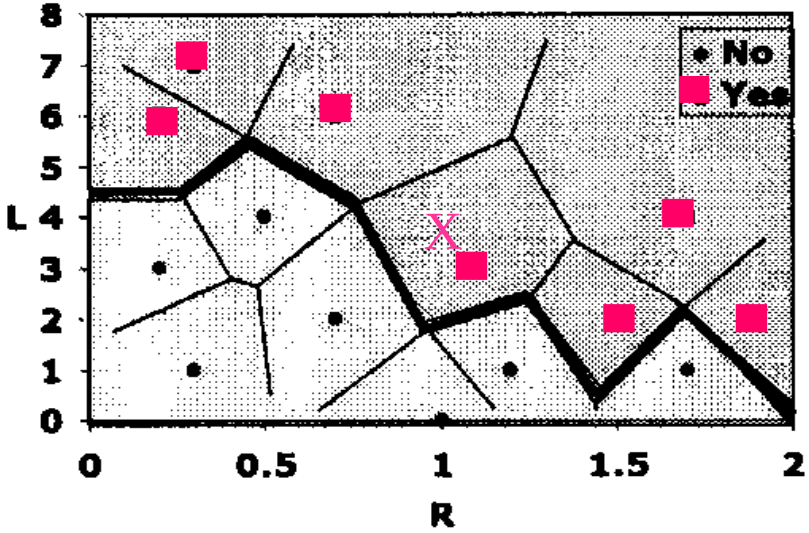
- Predicting bankruptcy
- 2 attributes:
  - on x: R – ratio of earnings to expenses
  - on y: L – number of late payments on credit cards over the past year
- 2 classes:
  - yes – high risk for bankruptcy
  - no - low risk for bankruptcy

1-Nearest Neighbor Decision Boundary

Example taken from 6.034 AI, MIT

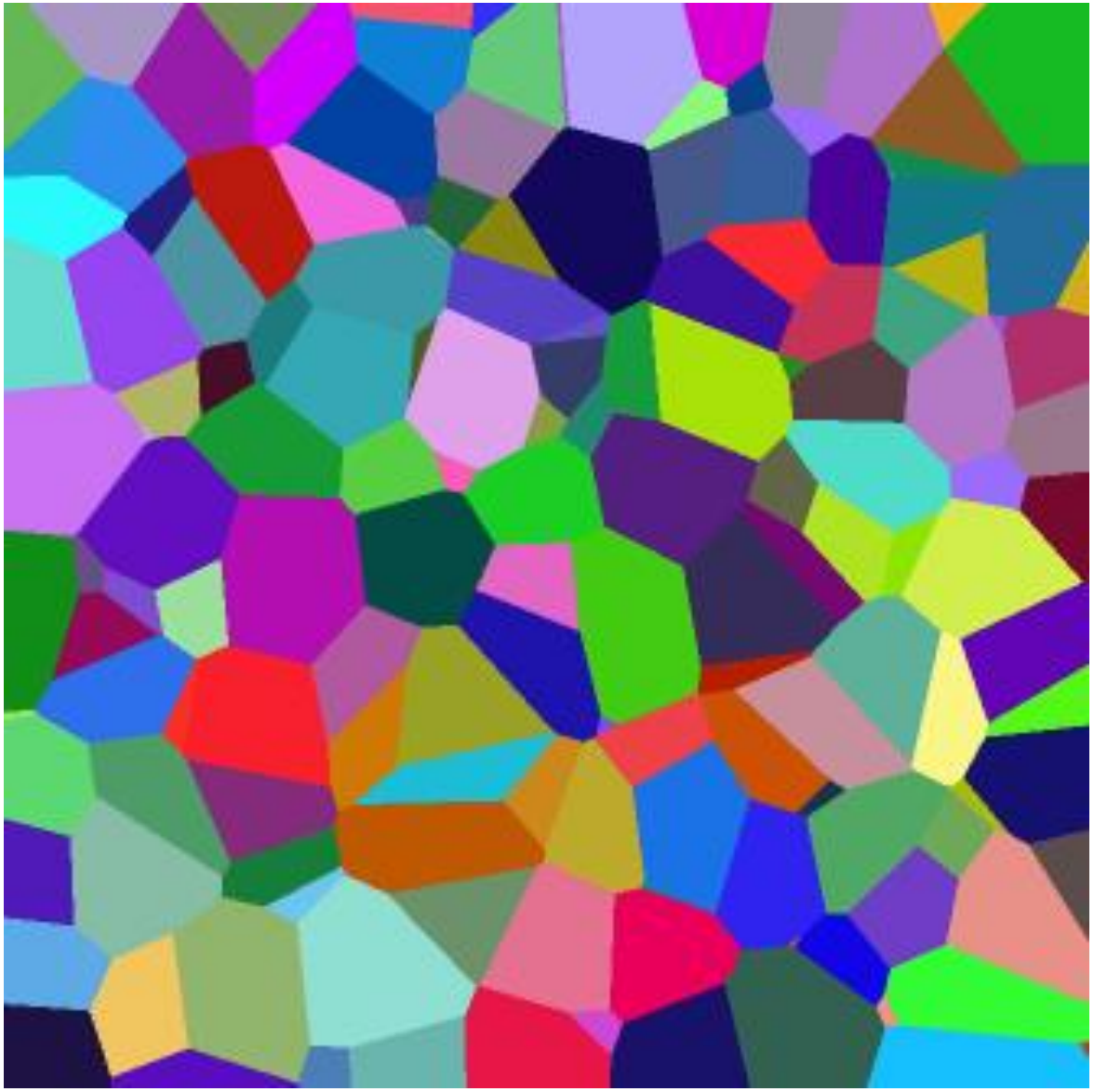
- The training data defines a partitioning in the feature space called Voronoi partition (tessellation)
  - Each training example has an associated Voronoi region
  - Voronoi region for a point (training example) = its nearby area
  - More precisely: if a new example Y falls in the Voronoi region of the training example X, then X is the closest training example to Y
- In 1-NN a decision boundary is represented by the edges in the Voronoi space that separate the points of the two classes





Voronoi diagram

www.math.unimaas.nl/personal/ronaldw/DAM/voordracht-DAM_2.ppt



K-Nearest-Neighbor – Discussion

- **Simple method; works well in practice (like 1R and Naive Bayes)**
- **Slow for big databases as the entire database have to be compared with each testing example**
  - **requires efficient indexing + there are more efficient variations**
- **Curse of dimensionality – "nearness" is ineffective in high dim**
  - **solution: feature selection to reduce dimensionality**
- **Very sensitive to irrelevant attributes, solutions:**
  - **weight each attribute when calculating the distance**
  - **feature selection to select informative attributes**
- **Produces arbitrarily shaped decision boundary defined by a subset of the Voronoi edges**
- **High variability of the decision boundary depending on the composition of training data and number of nearest neighbors**
- **Sensitive to noise**

K-Nearest-Neighbor – Discussion (2)

- **Local vs global information**
- **The standard k-nearest neighbor algorithm (which considers only a few neighbors) makes predictions based on local information**
- **Other ML algorithms, e.g. 1R, decision trees and neural networks try to find a global model that fits the training set**

Rule Learning: 1R Algorithm

1R Algorithm

- **1R stands for "1-rule"**
- **Generates *1 rule* that tests the value of *a single attribute***
  - **e.g. a rule that tests the value of `outlook`**

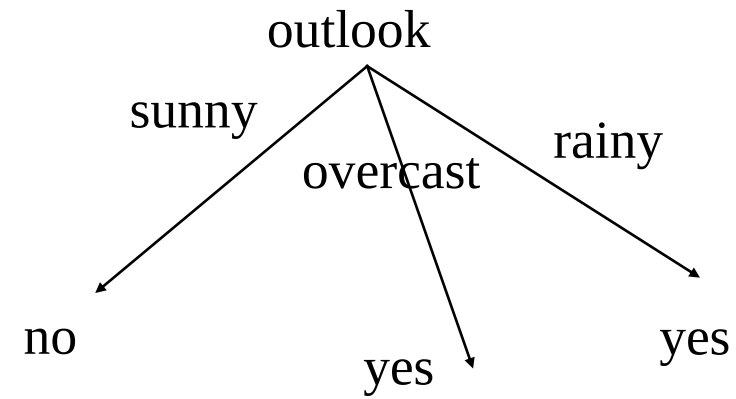
    if outlook=sunny then class=no

    elseif outlook=overcast then class=yes

    elseif outlook=rainy then class=yes

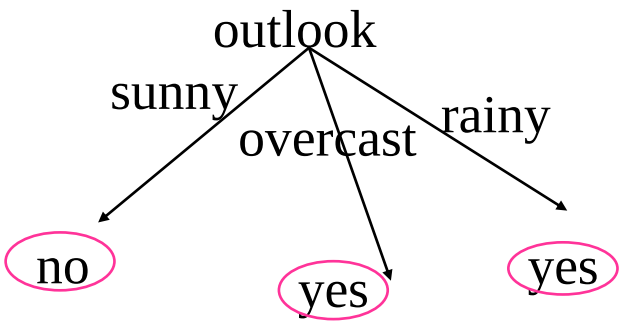
outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

- **The rule can be represented as a 1-level decision tree (decision stump)**
  - **At the root: test the attribute value**
  - **Each branch corresponds to a value**
  - **Each leaf corresponds to a class**



Two Questions

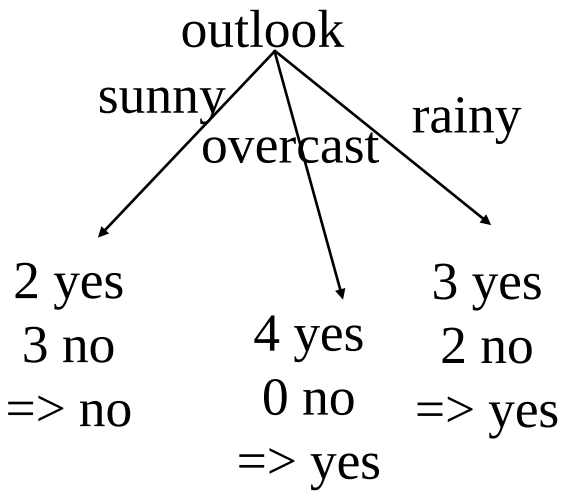
1. **How to determine the class for the leaves?**



2. **There are several rules, one for each attribute. How to select the best one?**

How to Determine the Class for the Leaves?

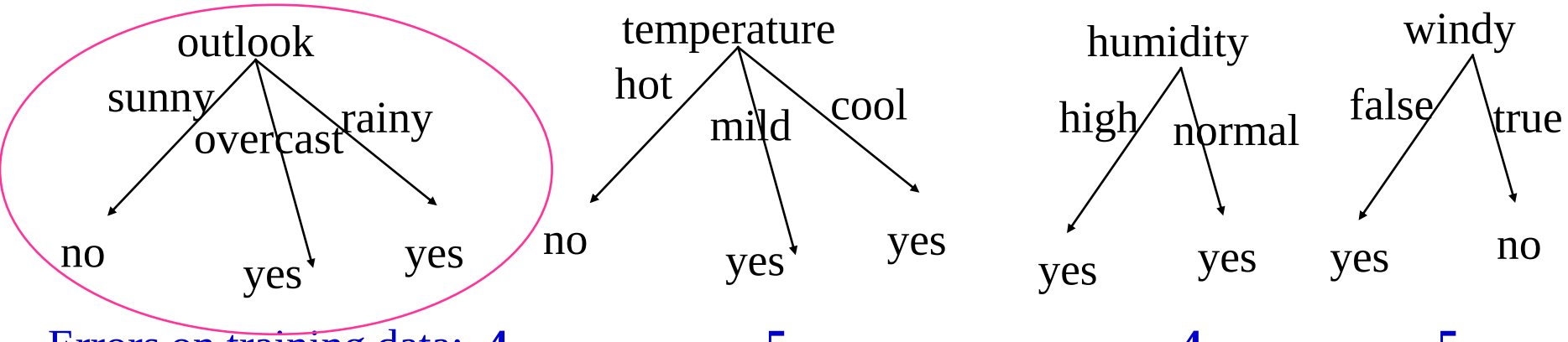
- **Take the majority class of the leaf**
- **This means minimizing the number of examples from the training data that will be misclassified**



outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

How to Select the Best Rule?

- **A rule is generated for each attribute**
- **Which is the *best rule* (i.e. the *best attribute*)?**
  - **The one with the smallest number of misclassifications (errors) on the training data (i.e. the one with the *highest accuracy*)**
- **1R algorithm**
  - **Generate a rule (decision stump) for each attribute**
  - **Evaluate each rule on the training data and calculate the number of errors**
  - **Choose the one with the smallest number of errors**



Errors on training data: **4** **5** **4** **5**

1R on the Weather Data

rule No	attri-bute	attribute values & counts	rules	errors	total errors
1	outlook	sunny: 2 yes, 3 no	sunny -> no	2/5	4/14
		overcast: 4 yes, 0 no	overcast -> yes	0/4	
		rainy: 3 yes, 2 no	rainy -> yes	2/5	
2	temp.	hot: 2 yes, 2 no*	hot -> no	2/4	5/14
		mild: 4 yes, 2 no	mild -> yes	2/6	
		cool: 2 yes, 1 no	cool -> yes	1/4	
3	humidity	high: 4 yes, 3 no	high -> yes	3/7	4/14
		normal: 6 yes, 1 no	normal -> yes	1/7	
5	windy	true: 3 yes, 3 no*	false -> yes	2/8	5/14
		false: 6 yes, 2 no	true -> no	3/6	

*** - random choice in case of ties**

Final rule - rule 1 (tie with rule 3):
if outlook=sunny then play=no
elseif outlook=overcast then play=yes
elseif outlook=rainy then play=yes

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

1R: Pseudo Code and Components

- **Pseudo code**

For each attribute,
For each value of that attribute, make a rule as follows:
- count how often each class appears
- find the most frequent class
- make the rule assign that class to this attribute value
Calculate the error rate of the rules
Choose the rule with the smallest error rate

- **Components of 1R**
  - **Model – a rule testing the value of 1 attribute**
  - **Preference function – number of misclassifications on training data**
  - **Search method – evaluate all attributes**

Handling Examples with Missing Values

- **How to use 1R or other classification algorithms if there are missing values? (generic topic, not only about 1R)**

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
?	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
?	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	?	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
?	mild	high	true	no

- **Method 1: Ignore all instances with missing attribute value - tempting solution! But:**
  - **often these instances are useful**
  - **sometimes the attributes whose values are missing play no part in the decision, in which case these instances are as good as any other**
- **Method 2 (used in 1R) : - treat the missing value as another possible value of the attribute; this assumes that the absence of a value is significant**
- **Ex.: if there are missing values for `outlook`, a 1R rule for `outlook` will consider 4 splits (on `sunny, overcast, rainy` and `missing`)**

Handling Examples with Missing Values (2)

Method 3: Replace the missing values - 2 common ways:

A(missing)=most common value for attribute A in all training examples

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
?	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	?	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	?	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

E.g. attribute A=temp., replace ? for this attribute with *mild* (most common for A)

A(missing)=most common value for attribute A in all training examples from the same class as the class of the example with the missing value

E.g. A=temp., replace ? for this attribute with *mild* (most common among all ex. from class *yes*)

Dealing With Numeric Attributes

outlook	temp.	humidity	windy	play
sunny	85	85	false	no
sunny	80	90	true	no
overcast	83	86	false	yes
rainy	70	96	false	yes
rainy	68	80	false	yes
rainy	65	70	true	no
overcast	64	65	true	yes
sunny	72	95	false	no
sunny	69	70	false	yes
rainy	75	80	false	yes
sunny	75	70	true	yes
overcast	73	90	true	yes
overcast	81	75	false	yes
rainy	71	91	true	no

- **`temperature` and `humidity` are numeric**
- **Need to *discretize* numeric attributes, i.e. convert them to nominal**
- **Here is a simple procedure:**

1. Sort the training examples in increasing order according to the value of the numeric attribute values of temperature:

64	65	68	69	70	71	72	73	75	75	80	81	83	85
yes	no	yes	yes	yes	no	no	yes	yes	yes	no	yes	yes	no

2. Place breakpoints whenever the class changes, halfway between the values

3. Take the class for each interval and form the rule

if `temperature` <64.5 than `play=yes`
elseif `temperature` ∈ [64.5, 65.5) than `play=no`
elseif `temperature` ∈ [66.5, 70.5) than `play=yes`
elseif `temperature` ∈ [70.5, 72.5) than `play=no`
elseif `temperature` ∈ [72.5, 77.5) than `play=yes`
elseif `temperature` ∈ [77.5, 80.5) than `play=no`
elseif `temperature` ∈ [80.5, 84) than `play=yes`
elseif `temperature` >=84 than `play=no`

Problem with this Discretization Procedure

- **It tends to generate a large number of intervals, i.e. generates nominal attributes with many values (= highly branching attributes)**
- **This is a problem – may lead to overfitting**
- ***Overfitting:*** **the error on the training set is very small but when a new data is presented to the classifier, the error is high**
  - **=> the classifier has memorized the training examples but has not learned to generalize to new examples!**
- **Example – overfitting in 1R due to noise in data**
  - **1 training example with incorrect class will most likely generate a separate interval (condition in the rule)**
  - **but this new interval is misleading – it was generated because of noise**
  - **=> 1R rule may not classify well new examples falling into this interval**

A Better Discretization Procedure

- **For 1R overfitting is likely to occur when an attribute has a large number of possible values**
- **Simple solution - aggregation:**
  - **Introduce a threshold - requirement for a minimum number of examples of the majority class in each partition, e.g. minimum number=3**

64	65	68	69	70	71	72	72	75	75	80	81	83	85
yes	no	yes	yes	yes	no	no	yes	yes	yes	no	yes	yes	no
		yes				yes						no	

  - **When adjacent partitions have the same majority class, merge them**
  - **Final discretization**

64	65	68	69	70	71	72	72	75	75	80	81	83	85
yes	no	yes	yes	yes	no	no	yes	yes	yes	no	yes	yes	no
					yes								no

if `temperature` ≤ `77.5` then `play=yes`

elseif `temperature > 77.5` then `play=no*`

*** Arbitrary choice `no` for the 2d interval; if `yes` has been chosen => no need for any breakpoint at all**

Data with Numeric Attributes – Final Rule Set

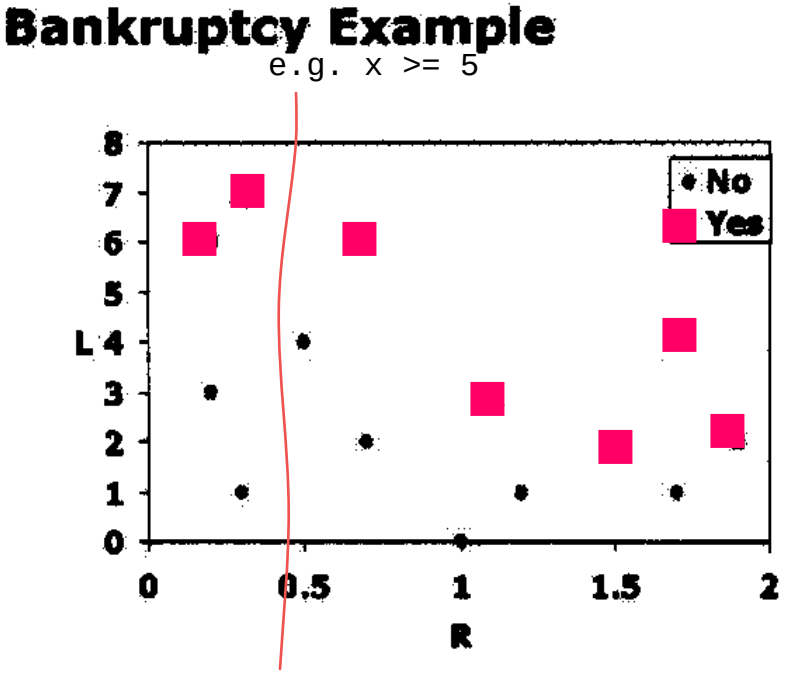
rule No	attri-bute	Rules	errors	total errors
1	outlook	sunny -> no overcast -> yes rainy -> yes	2/5 0/4 2/5	4/14
2	temp.	≤ 77.5 -> yes > 77.5 -> no*	3/10 2/4	5/14
3	humidity	≤ 82.5 -> yes > 82.5 & ≤ 95.5 -> no >95.5 -> yes	1/7 2/6 0/1	3/14
5	windy	false -> yes true -> no	2/8 3/6	5/14

- **Which rule will be selected?**

Decision Boundary

Example taken from 6.034 AI, MIT

- 1R defines a decision boundary in the feature space
- Suppose that the 1R split is binary. What will be the shape of the decision boundary?



- Predicting bankruptcy
- 2 attributes:
  R – ratio of earnings to expenses
  L – number of late payments on credit cards over the past year
- 2 classes:
  yes – high risk for bankruptcy
  no - low risk for bankruptcy

1R – Discussion

- **1R was described in a paper by Holte (1993)**
  - **1R's simple rules were just slightly less accurate than the much more complex decision trees and other state-of- the-art classification algorithms**
- **Simple and computationally cheap algorithm**
- **Simplicity first pays off**
  - **Always try the simple algorithms first!**
  - **Simple algorithms give a useful first impression of the dataset**

Why do Simple Algorithms Work Well?

- **The structure underlying many real world problems may be quite *rudimentary*, e.g.**
  - **just one attribute is sufficient to determine the class of the example**
  - **several attributes contribute independently with equal importance**
  - **a linear combination of attributes may be sufficient**