

# Decision Trees

# 3

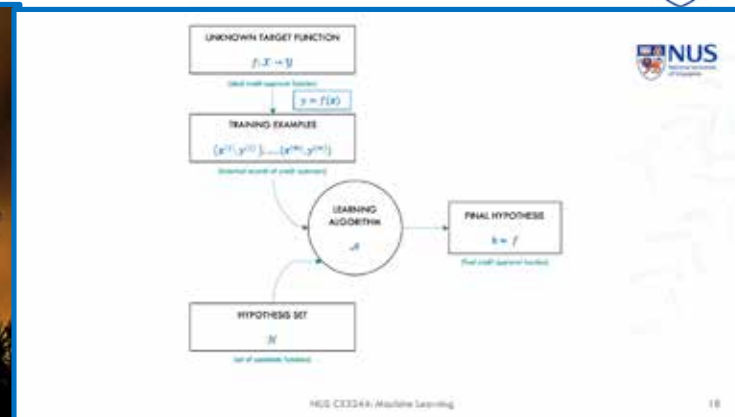
# A

**CS 3244**  
**Machine Learning**



**NUS** | Computing

# Recap from Week 02



## Futility of Bias Free Learning

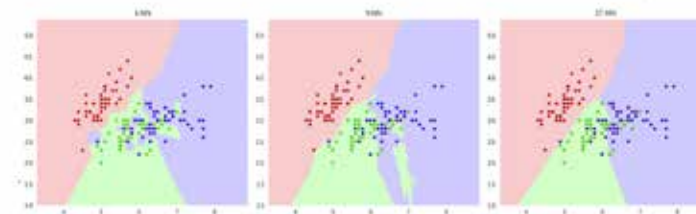
A learner that makes no *a priori* assumptions on  $f$  has *no rational basis* for classifying any unseen instances.

Making prior assumptions – **Inductive Bias** – is the only way to make learning feasible.

Each family (“tribe”) of models  $\mathcal{A}$  has a different take on its inductive bias, which forms of  $h$  it can represent:  $\mathcal{H}$ .

## Effect of $k$

Smaller  $k$ : complex surface.  
Larger  $k$ : smoother surface.



# Forecast for Week 03



- Learning Outcomes for this week:
- Understand and implement Decision Trees;
- Master Information Gain as a means of choosing optimal features for decisions;
- Formalize cost functions as a means of evaluating a supervised learner's performance;
- Describe simple ensembling methods that integrate multiple classifiers;
  - Understand Decision Trees as an instance of an ensemble classifier of conditional decision stamps;
- Reason about inductive bias of the decision tree algorithm

# A few things going forward...



We will be using Zoom breakouts based on your tutorial groups during **in-lecture activities**. These will be stable for the semester.

While Brian and Min are happy to answer your questions, it is best if you first try to ask in your **#tg-xx first**.

(Help our TAs revise their own ML experience 🤔 🤔 )

# Thursday: AMA and Datasets



Your subgroup may want to attend Thursday to get to talk to some of the TAs who know some of the curated datasets well.

We'll have breakout rooms for different curated datasets so you can check with the TAs what might be suitable projects, even before the proposal. 😎





# Decision Trees

**CS3244 Machine Learning**



Department of Computer Science  
School of Computing

# How do people make decisions?

Well, let's examine some recent choices you made:

*What module did you enroll in?*

*What did you eat today for lunch?*

*What did you wear today?*



## Pre-Lecture Activity from last week



1 day ago

Which module I enrolled in:

- Does ModReg allow me to take or not
- Availability of the module in the current semester
- Prerequisite tree of the module (including my past modules and future plan)

Variables:

- How important is the mod to my career
- The grade should I aim to before registering
- Number of friends taking in the same cohort
- Workload and time management

Logic:

- Prioritising on core modules or remaining technical electives if not taken.  
Considering the time management, making sure I can have reasonable work-live balance and meaningful learning outcomes.



3 days ago

Which module you enrolled in?

- **Context and variables:** Overall module plan, Course requirements, prerequisites, modules taken, interest, timetable scheduling, vacancy of slots
- **Logic:** Narrow down to modules from focus areas of interest, map out possibilities of module plans based on restrictions (prerequisites, course availability, modules taken, course requirements) then fill in possible UEs with other miscellaneous modules. Pick the modules of the most ideal plan. If module is highly oversubscribed, consider other plans' modules.

(edited)



1 day ago

Deciding which modules I am enrolled in

Context: Semester and year

Variable:

1. Interest
2. Workload
3. Whether it affect my graduation
4. Whether any friends is taking it with me
5. Other commitments
6. Will it be beneficial to internship and future career ambition

Logic:

If I am very interested in this mod  
or the workload is reasonable  
or it affects my graduation if I do not take it  
or many of my friends are taking it  
or my other commitments are low  
or it is beneficial to my internship and future career endeavour,  
then I will take it,  
else I will not take it



2 days ago

Question: Which module you enrolled in?

Context: I need certain modules to clear my graduation requirements / requirements for specialisation.

Variables: Professor that is going to teach this sem. Whether this module clashes with any of other modules I want to take. Number of students enrolled in this module.

Logic: Firstly, I check whether this module satisfies my degrees requirements, what was this module about and who's going to teach this module (the usefulness of this module). Then I check the feasibility of me taking the module by inspecting whether there is any clash or whether there are too many students enrolled.



## Pre-Lecture Activity from last week



1 day ago

1. Which module to enrol in?
  - a. How many modules the module is a prereq for.
  - b. My interest in the aforementioned module.
  - c. is it a core requirement?
  - d. How long the prereq chain of that module is.



3 days ago

- Which module you enrolled in?
  - Context: Semester and Year of Study
  - Variables:
    - i. Interest in the module
    - ii. Perceived difficulty of the module
    - iii. Utility of the module for my personal career ambitions
    - iv. Friends taking the module in the same semester



6 hours ago

- Which module I enrolled in:
- Context: prerequisites, major requirements, workload & difficulty level, school's module schedule (e.g. certain modules are not offered every sem), potential timetable clash, synergy between modules
  - Variable: module info (timetable, workload, lecturer), my own info (major requirements, preference)
  - Logic: Similar to "Greedy" - try to fit in as many modules that I want to take as possible. If impossible, prioritise those core modules, modules with less flexible schedule or modules that are prerequisite for future modules.



4 days ago

**Decision:**  
Which module you enrolled in?

**Context:**  
 - How easy the module is?  
 - How relevant it might be for future career prospects?  
 - How enjoyable/interesting module may be?

**Variables:**  
 - Module Faculty  
 - Module level (1k/2k/3k/4k etc)  
 - Module popularity (vacancies/demand-to-vacancy-ratio)  
 - Is/Are Friend(s) taking?



3 days ago

**Question:** Which module you enrolled in?

**Context:** Whether it counts to my graduation requirements, Whether it helps to clear my focus area.

**Variables:** Number of friends taking the module.  
 Professor / Lecturer teaching the module.  
 Number of Smart People taking the module (aka bell curve wreckers)  
 Difficulty and Workload of the module

**Logic:** Simple linear comparison of the modules that I plan to take.

If a module is more difficult and has the same professor teaching, I would generally avoid taking the module (edited)



# How do people make decisions?



In Zoom breakout or physical subgroups, do:

(1 min): Introduce yourself! 🙄

(5 mins): *What's the inductive bias of our method?*

Is it like how  $k$  nearest neighbors decides?

Ask one member to write your collective answer to the [#general](#) thread. Upvote others that you like.

# How do people make decisions?



Would we say it's like how  $k$  nearest neighbors make a decision? (i.e., looking at what you did for a similar item last time?)

No, not exactly. It doesn't make much sense in this instance. Instead, people often focus on just a 1 or 2 dimensions of the evidence.

Decision trees embody this paradigm.

# Programming

Imperative programming control flow also embodies this:

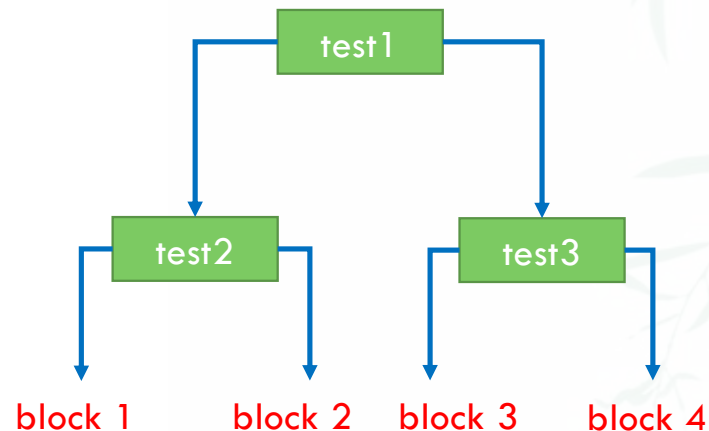
```
if test1:
    if test2:
        # block 1
    else:
        # block 2
else:
    if test3:
        # block 3
    else:
        # block 4
```

# Programming

Imperative programming control flow also embodies this

```
if test1:  
    if test2:  
        # block 1  
    else:  
        # block 2  
else:  
    if test3:  
        # block 3  
    else:  
        # block 4
```

This is a decision tree!





# Parts of a tree



## 1. Internal nodes are tests

In most systems, a node tests exactly one component of  $x$ : (a feature; e.g.,  $x_0$ )

## 2. A branch in the tree corresponds to a test result

A branch corresponds to an attribute value or a range of attribute values

## 3. Each leaf node assigns

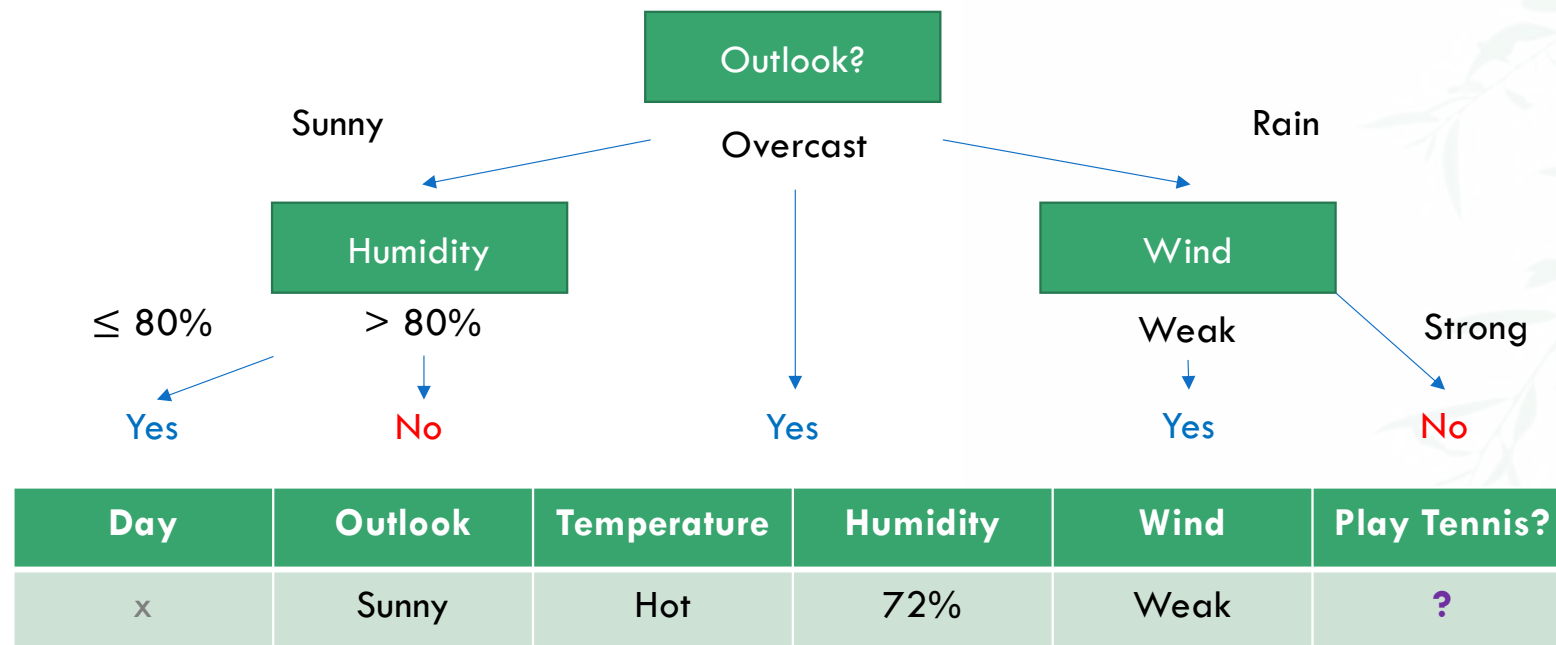
- A class  $y \in \{enum\}$ : Classification tree
- Real value  $y \in \mathbb{R}$ : Regression tree

# Shall we play tennis?

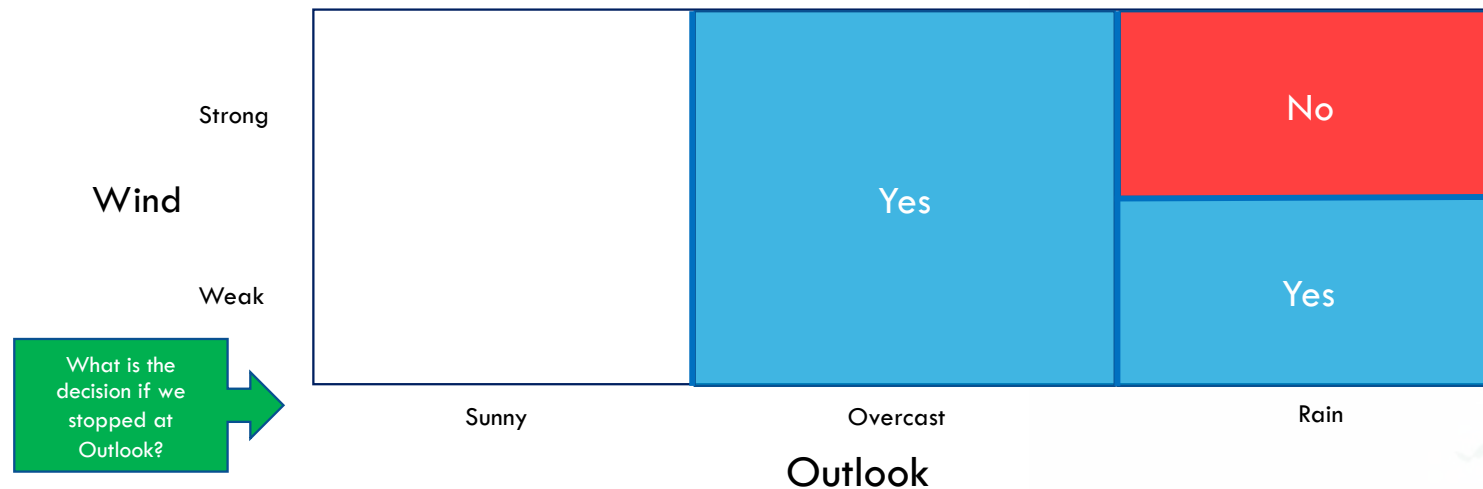


Day	Outlook	Temperature	Humidity	Wind	Play Tennis?
1	Sunny	Hot	95%	Weak	No
2	Sunny	Hot	92%	Strong	No
3	Overcast	Hot	95%	Weak	Yes
4	Rain	Mild	90%	Weak	Yes
5	Rain	Cool	60%	Weak	Yes
6	Rain	Cool	65%	Strong	No
7	Overcast	Cool	70%	Strong	Yes
8	Sunny	Cool	70%	Weak	Yes

# Shall we play tennis?

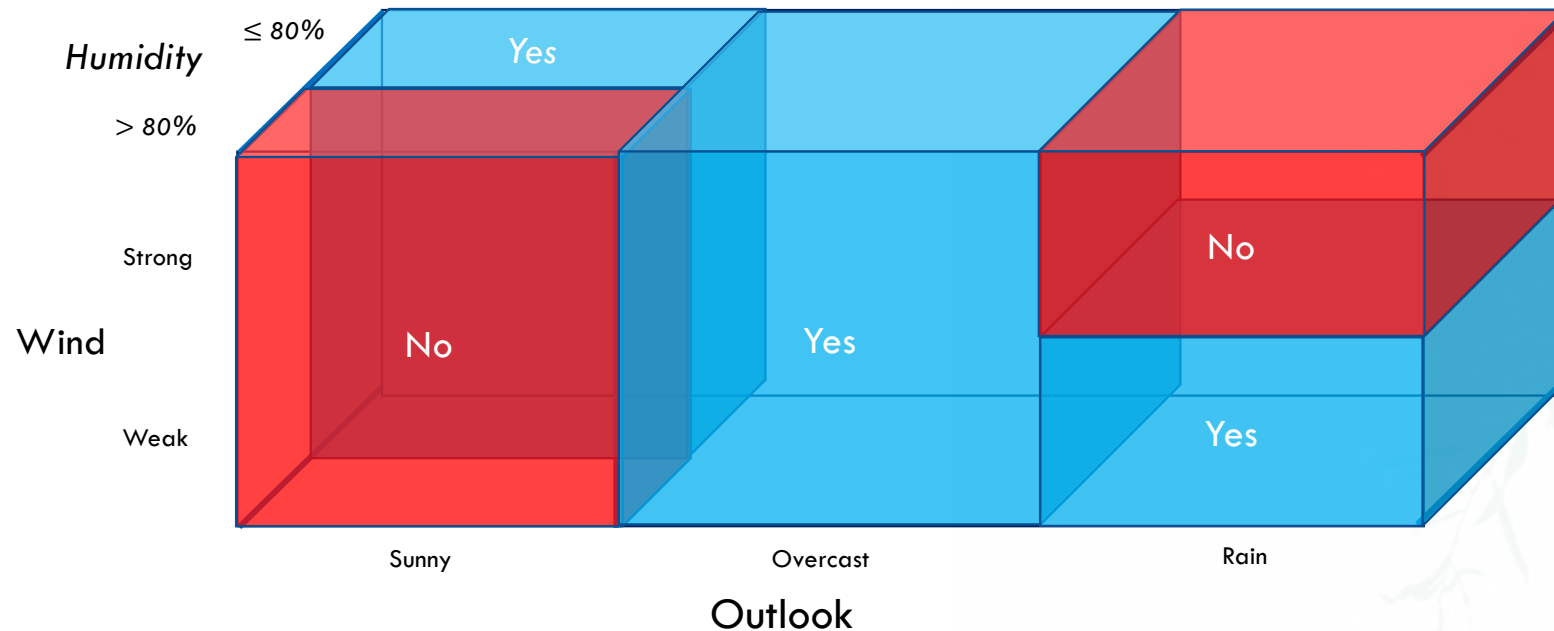


# Inductive Bias – Decision boundary




Axis perpendicular (recursive) decision boundaries

# Inductive Bias – Decision boundary



Axis perpendicular (recursive) decision boundaries





```
function DTL(examples, attributes, default) returns a decision tree
  if examples is empty then return default
  else if all examples have the same classification then return the classification
  else if attributes is empty then return MODE(examples)
  else
    best ← CHOOSE-ATTRIBUTE(attributes, examples)
    tree ← a new decision tree with root test best
    for each value  $v_i$  of best do
      examplesi ← {elements of examples with best =  $v_i$ }
      subtree ← DTL(examplesi, attributes − best, MODE(examples))
      add a branch to tree with label  $v_i$  and subtree subtree
  return tree
```

# 3 Important Questions

Prefer important decisions at the top

1. How do we pick a feature to split on?
2. How do we discretize continuous features?

Prune back complete trees<sup>\*</sup>

3. How do we decide where to stop pruning?

<sup>\*</sup> we could decide when to stop growing the tree, but in practice this doesn't work well.

# How do you choose which feature is important?

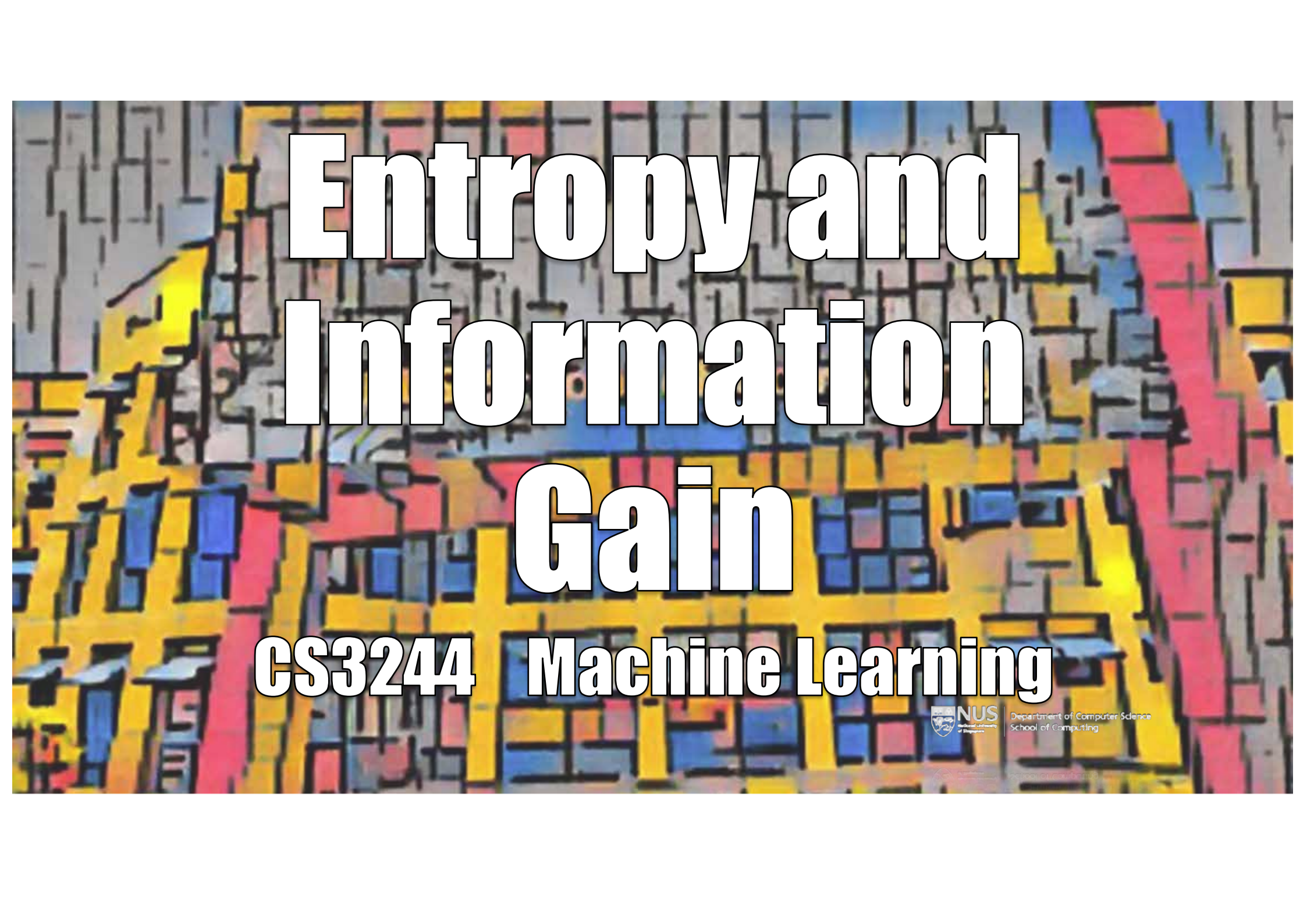


In Zoom breakout or physical subgroups, do:

**(5 mins):** Discuss any **one** of your three decisions as a team.

Answer: *What criteria do you use to decide which feature of your decision is the most important one to use first?*

Ask one member to write your collective answer to the **#general** thread. Upvote others that you like.



# Entropy and Information Gain

**CS3244 Machine Learning**

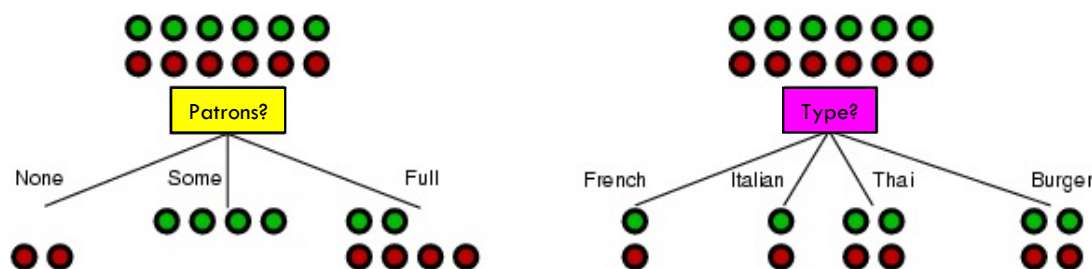


Department of Computer Science  
School of Computing

# Choosing an attribute

Idea: a good attribute (feature) splits the training set into subsets that are (ideally) “all positive” or “all negative”.

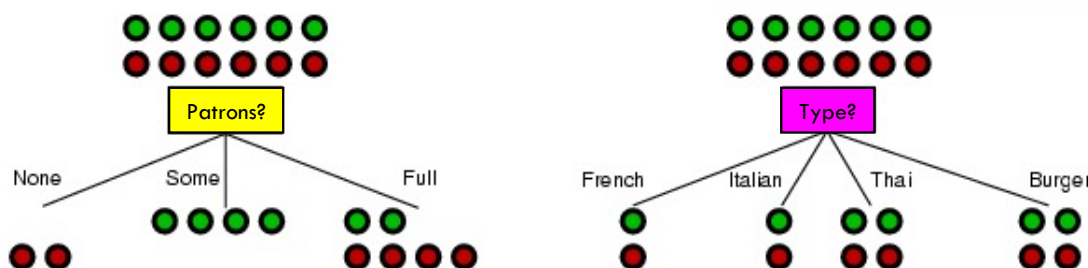
Let's look at a case of *What did you eat for lunch?*





# Choosing an attribute

Idea: a good attribute (feature) splits the training set into subsets that are (ideally) “all positive” or “all negative”.



**Patrons?** is a better choice. Why?

Purity of the subproblems.

# Q1. How do we pick a feature to split on?



To implement **Choose Attribute** in DTL for enumerated feature sets with  $C$  possible values, we first need a concept of purity. We'll use **entropy**:

$$H(X) = - \sum_{i=0}^C P_i \log_2(P_i)$$

⚠  $P \equiv$  probability,  
 $p \equiv$  # of positive examples.

Example: for a training set containing  $p$  positive examples and  $n$  negative examples:

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

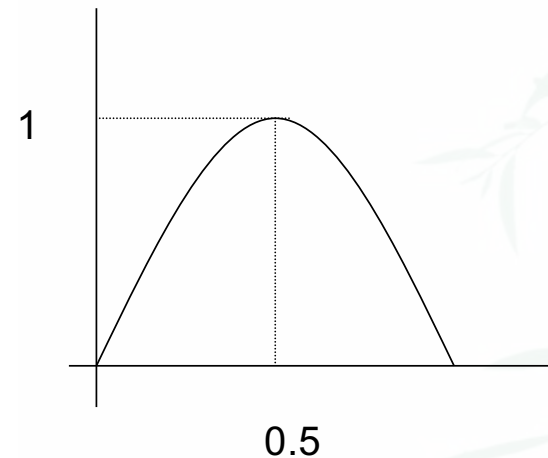
# Entropy curve

For  $\frac{p}{p+n}$ , the 2-class entropy is:

0 when  $\frac{p}{p+n}$  is 0

1 when  $\frac{p}{p+n}$  is 0.5

0 when  $\frac{p}{p+n}$  is 1



- monotonically increasing between 0 and 0.5
- monotonically decreasing between 0.5 and 1

# Information gain

A chosen feature  $x_i$  divides the example set  $S$  into subsets  $S_1, S_2, \dots, S_c$  according to the  $C_i$  distinct values for  $x_i$ .

The entropy then reduces to the entropy of the subsets  $S_1, S_2, \dots, S_c$ :

$$\text{remainder}(S, x_i) = \sum_{j=1}^{C_i} \frac{|S_j|}{|S|} H(S_j)$$

**Information Gain** (IG; “reduction in entropy”) from knowing the value of  $x_i$ .  
Choose the attribute with the largest IG:

$$\text{IG}(S, x_i) = H(S) - \text{remainder}(S, x_i)$$

# Information gain

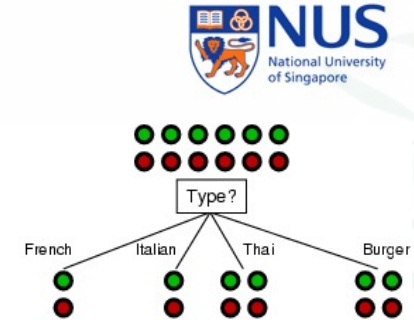
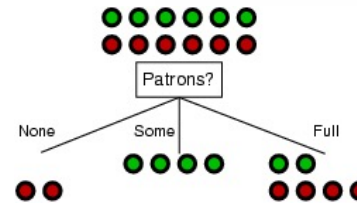
For the training set at the root,  
 $p = n = 6$ ,  $H\left(\frac{6}{12}, \frac{6}{12}\right) = 1$  bit.

Consider the attributes **Patrons** and **Type**:

$$IG(\text{Patrons}) = 1 - \left[ \frac{2}{12} H(0,1) + \frac{4}{12} H(1,0) + \frac{6}{12} H\left(\frac{2}{6}, \frac{4}{6}\right) \right] = 0.541 \text{ bits}$$

$$IG(\text{Type}) = 1 - \left[ \frac{2}{12} H\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{2}{12} H\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{4}{12} H\left(\frac{2}{4}, \frac{2}{4}\right) + \frac{4}{12} H\left(\frac{2}{4}, \frac{2}{4}\right) \right] = 0 \text{ bits}$$

*Patrons* has the highest IG, and so is chosen by DTL as the root.







# Cost Functions

**CS3244 Machine Learning**



NUS School of Computing



Department of Computer Science  
School of Computing

# Cost functions

What does  $h_\theta \approx f$  mean?

Need a **cost** function  $L(h_\theta, f)$



“Cost” is same “loss”. Also, you’ll see  $J(h_\theta, f)$  instead of  $L(h_\theta, f)$

This is almost always a pointwise definition:  $l(h_\theta(x), f(x))$

Simple examples:

Squared error

$$(h_\theta(x) - f(x))^2$$

Binary error

$$[[h_\theta(x) \neq f(x)]]$$



Which is for  
classification?

# From pointwise to overall



Overall cost  $L(h, f) =$

average of pointwise cost  $l(h(x), f(x))$

Training cost:

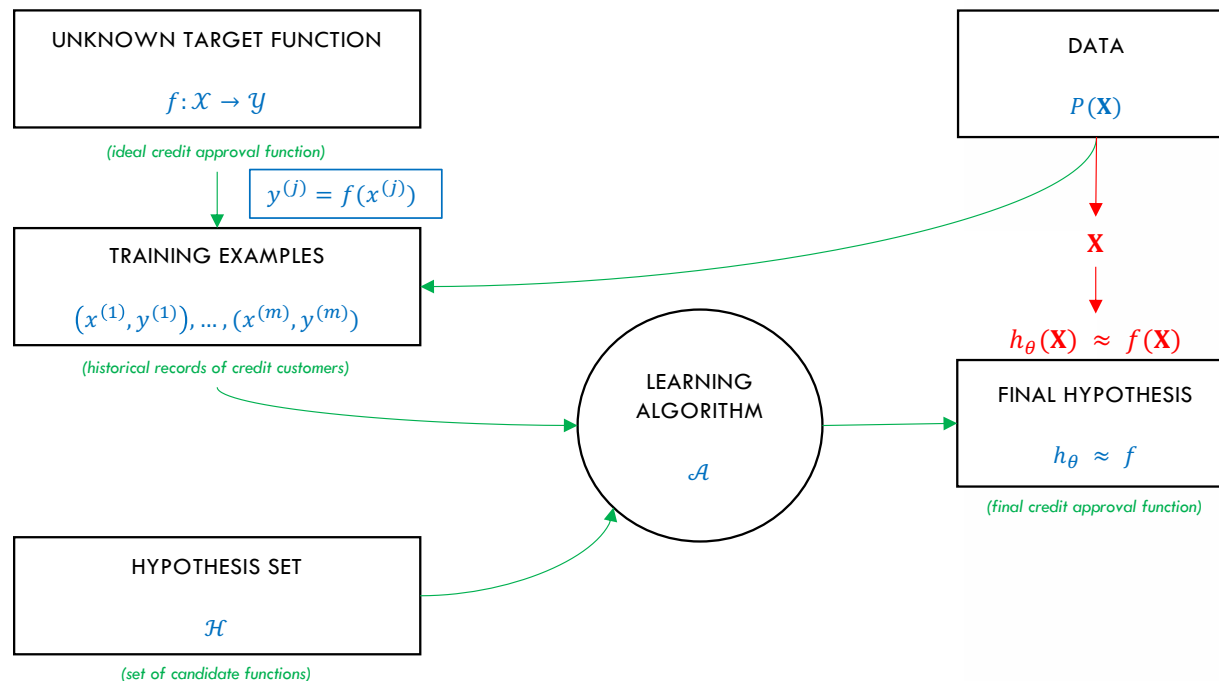
$$L_{train} = \frac{1}{m} \sum_{j=1}^m l(h_{\theta}(x^{(j)}), f(x^{(j)}))$$

Why not a sum  
instead of an  
average?

Test cost:

$$L_{test} = \mathbb{E}_x[l(h_{\theta}(x), f(x))]$$

# The learning diagram with testing data, pointwise error



# Choosing your cost function

Are you sick?

Two types of costs:

*false positive* (accept) or

*false negative* (reject)

How should we penalize  
for each type?

$$f \longrightarrow \begin{cases} +1 & \text{sick} \\ -1 & \text{well} \end{cases}$$

$f$

$h_\theta$

	+1	-1
+1	True Positive	False Positive
-1	False Negative	True Negative

# During your last final exam before graduation

Are you sick?

*False negative* – get better on your own, or come back to the clinic later. At least you graduate on time.

*False positive* – Take the exam next year. Possibly pay tuition fees. \$\$\$



$$f \longrightarrow \begin{cases} +1 & \text{sick} \\ -1 & \text{well} \end{cases}$$

$f$

$h_\theta$

	+1	-1
+1	0 True Positive	100 Positive
-1	1 False Negative	0 Negative



# During COVID-19 Pandemic



Are you sick?

*False negative* highly costly!  
People and the economy dies.

*False positive* requires the  
inconvenience of quarantine.

$$f \longrightarrow \begin{cases} +1 & \text{sick} \\ -1 & \text{well} \end{cases}$$

$f$

$h_\theta$

	+1	-1
+1	0 True Positive	False Positive
-1	False Negative 10,000	0 True Negative

# Your measure matters

Where possible, we should use cost functions that fit the task, specified by the user.

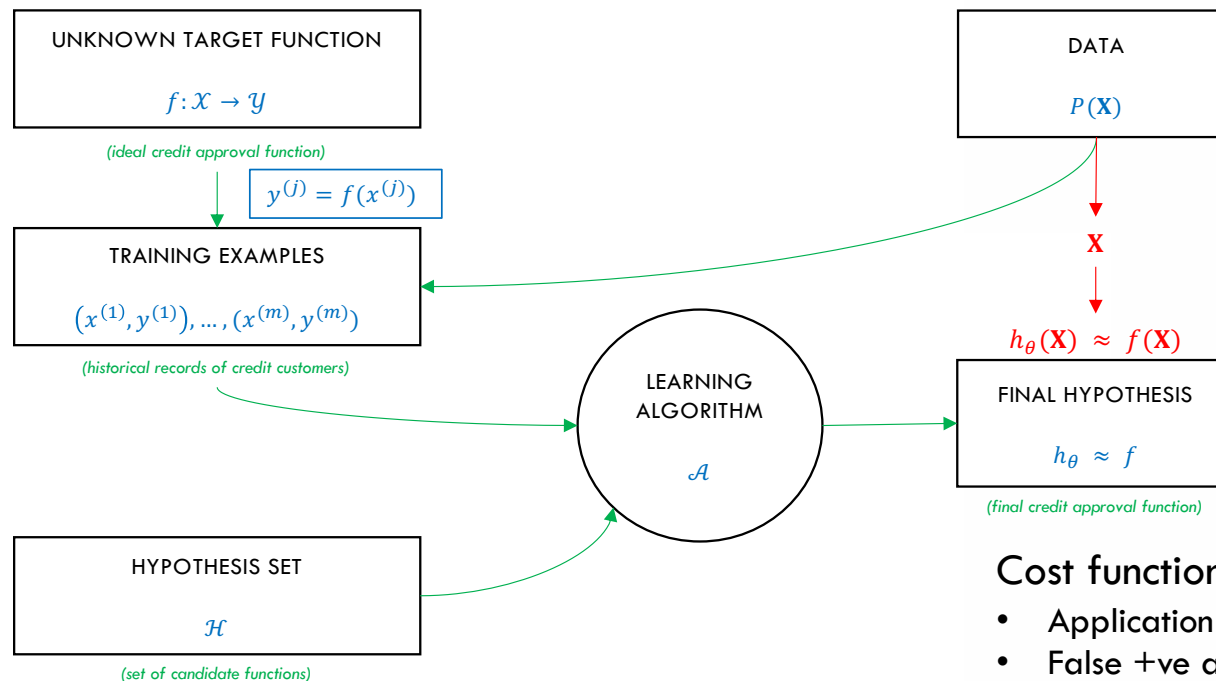
However, this isn't always possible. Then use:

**Plausible** measures:      squared error  $\equiv$  Gaussian noise

**Convenient** measures:      closed form solution,  
convex optimization

*The sweet spot:  
Plausible AND  
Convenient*

# Summary – the learning diagram with testing data, pointwise error



**Quick Question:**  
where does the  
cost function go?

## Cost functions


- Application specific, user should specify
- False +ve and -ve may differ in cost

# How you measure matters



**(5 mins):** Go through the three decisions from your pre-lecture activity as a team. Answer the question *What do you think the appropriate false positive and negative costs should be for each decision?*

Choose one random answer and write your collective answer to the **#general** thread. Upvote others that you like.



NUS School of Computing

# Ensembles

**CS3244 Machine Learning**

NUS CS3244: Machine Learning



Department of Computer Science  
School of Computing

39



# Ensembles:

## Revisiting the Netflix Prize

Just three weeks after it began, at least 40 teams had bested the Netflix classifier.

Top teams showed about 5% improvement.



### Leaderboard

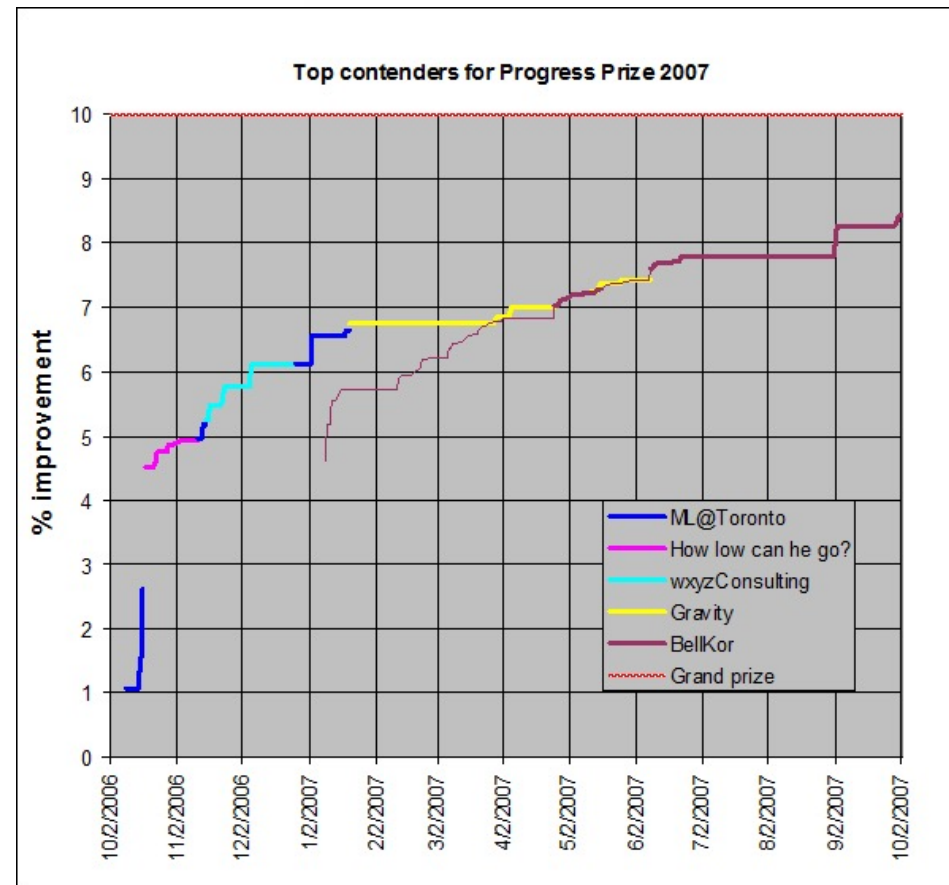
Team Name	Best Score	% Improvement
No Grand Prize candidates yet	--	--
Grand Prize - RMSE <= 0.8563		
How low can he go?	0.9046	4.92
<a href="#">ML@UToronto A</a>	0.9046	4.92
<a href="#">ssorkin</a>	0.9089	4.47
<a href="#">wxyzconsulting.com</a>	0.9103	4.32
The Thought Gang	0.9113	4.21
<a href="#">NIPS Reject</a>	0.9118	4.16
<a href="#">simonfunk</a>	0.9145	3.88
Bozo_The_Clown	0.9177	3.54
Elliptic Chaos	0.9179	3.52
datcracker	0.9183	3.48
<a href="#">Foreseer</a>	0.9214	3.15
bsd fish	0.9229	3.00
Three Blind Mice	0.9234	2.94
<a href="#">Bocsimacko</a>	0.9238	2.90
Remco	0.9252	2.75
<a href="#">karmatics</a>	0.9301	2.24
Chapelator	0.9314	2.10
<a href="#">Flmod</a>	0.9325	1.99
mthrox	0.9328	1.96



However,  
improvement  
slowed...

Source:

<http://www.research.att.com/~volinsky/netflix/>



# Rookies

“Thanks to Paul Harrison's collaboration, a simple mix of our solutions improved our result from 6.31 to 6.75”

--	No Progress Prize candidates yet	--	--
Progress Prize - RMSE <= 0.8625			
1	<a href="#">BellKor</a>	0.8705	8.50
Progress Prize 2007 - RMSE = 0.8712 - Winning Team: KorBell			
2	<a href="#">KorBell</a>	0.8712	8.43
3	<a href="#">When Gravity and Dinosaurs Unite</a>	0.8717	8.38
4	<a href="#">Gravity</a>	0.8743	8.10
5	<a href="#">basho</a>	0.8746	8.07
6	<a href="#">Dinosaur Planet</a>	0.8753	8.00
7	<a href="#">ML@UToronto A</a>	0.8787	7.64
8	<a href="#">Arek Paterek</a>	0.8789	7.62
9	<a href="#">NIPS Reject</a>	0.8808	7.42
10	Just a guy in a garage	0.8834	7.15
11	<a href="#">Ensemble Experts</a>	0.8841	7.07
12	<a href="#">mathematical capital</a>	0.8844	7.04
13	<a href="#">HowLowCanHeGo2</a>	0.8847	7.01
14	<a href="#">The Thought Gang</a>	0.8849	6.99
15	<a href="#">Reel Ingenuity</a>	0.8855	6.93
16	<a href="#">strudeltamale</a>	0.8859	6.88
17	NIPS Submission	0.8861	6.86
18	Three Blind Mice	0.8869	6.78
19	TrainOnTest	0.8869	6.78
20	<a href="#">Geoff Dean</a>	0.8869	6.78
21	<a href="#">Rookies</a>	0.8872	6.75
22	<a href="#">Paul Harrison</a>	0.8872	6.75
23	ATTEAM	0.8873	6.74
24	<a href="#">wxyzconsulting.com</a>	0.8874	6.73
25	ICMLsubmission	0.8875	6.72
26	Efratko	0.8877	6.70
27	<a href="#">Kitty</a>	0.8881	6.65
28	SecondaryResults	0.8884	6.62
29	<a href="#">Birgit Kraft</a>	0.8885	6.61

# Arek Paterek

“My approach is to **combine the results of many methods** (also two-way interactions between them) using linear regression on the test set. The best method in my ensemble is regularized SVD with biases, post processed with kernel ridge regression”

[http://rainbow.mimuw.edu.pl/~ap/ap\\_kdd.pdf](http://rainbow.mimuw.edu.pl/~ap/ap_kdd.pdf)

NUS CS3244: Machine Learning

--	No Progress Prize candidates yet	--	--
Progress Prize - RMSE <= 0.8625			
1	<a href="#">BellKor</a>	0.8705	8.50
Progress Prize 2007 - RMSE = 0.8712 - Winning Team: KorBell			
2	<a href="#">KorBell</a>	0.8712	8.43
3	<a href="#">When Gravity and Dinosaurs Unite</a>	0.8717	8.38
4	<a href="#">Gravity</a>	0.8743	8.10
5	<a href="#">basho</a>	0.8746	8.07
6	<a href="#">Dinosaur Planet</a>	0.8753	8.00
7	<a href="#">ML@UToronto A</a>	0.8787	7.64
8	<a href="#">Arek Paterek</a>	0.8789	7.62
9	<a href="#">NIPS Reject</a>	0.8808	7.42
10	Just a guy in a garage	0.8834	7.15
11	<a href="#">Ensemble Experts</a>	0.8841	7.07
12	<a href="#">mathematical capital</a>	0.8844	7.04
13	<a href="#">HowLowCanHeGo2</a>	0.8847	7.01
14	<a href="#">The Thought Gang</a>	0.8849	6.99
15	<a href="#">Reel Ingenuity</a>	0.8855	6.93
16	<a href="#">strudelamale</a>	0.8859	6.88
17	NIPS Submission	0.8861	6.86
18	Three Blind Mice	0.8869	6.78
19	TrainOnTest	0.8869	6.78
20	Geoff Dean	0.8869	6.78
21	<a href="#">Rookies</a>	0.8872	6.75
22	<a href="#">Paul Harrison</a>	0.8872	6.75
23	ATTEAM	0.8873	6.74
24	<a href="#">xyzconsulting.com</a>	0.8874	6.73
25	ICMLsubmission	0.8875	6.72
26	Efratko	0.8877	6.70
27	<a href="#">Kitty</a>	0.8881	6.65
28	SecondaryResults	0.8884	6.62
29	<a href="#">Birgit Kraft</a>	0.8885	6.61

# When Gravity and Dinosaurs Unite

“Our common team blends the result of team Gravity and team Dinosaur Planet.”

Might have guessed from the name...

--	No Progress Prize candidates yet	--	--
Progress Prize - RMSE <= 0.8625			
1	<a href="#">BellKor</a>	0.8705	8.50
Progress Prize 2007 - RMSE = 0.8712 - Winning Team: KorBell			
2	<a href="#">KorBell</a>	0.8712	8.43
3	<a href="#">When Gravity and Dinosaurs Unite</a>	0.8717	8.38
4	<a href="#">Gravity</a>	0.8743	8.10
5	<a href="#">basho</a>	0.8746	8.07
6	<a href="#">Dinosaur Planet</a>	0.8753	8.00
7	<a href="#">ML@UToronto A</a>	0.8787	7.64
8	<a href="#">Arek Paterek</a>	0.8789	7.62
9	<a href="#">NIPS Reject</a>	0.8808	7.42
10	Just a guy in a garage	0.8834	7.15
11	<a href="#">Ensemble Experts</a>	0.8841	7.07
12	<a href="#">mathematical capital</a>	0.8844	7.04
13	<a href="#">HowLowCanHeGo2</a>	0.8847	7.01
14	<a href="#">The Thought Gang</a>	0.8849	6.99
15	<a href="#">Reel Ingenuity</a>	0.8855	6.93
16	<a href="#">strudelamale</a>	0.8859	6.88
17	NIPS Submission	0.8861	6.86
18	Three Blind Mice	0.8869	6.78
19	TrainOnTest	0.8869	6.78
20	Geoff Dean	0.8869	6.78
21	<a href="#">Rookies</a>	0.8872	6.75
22	<a href="#">Paul Harrison</a>	0.8872	6.75
23	ATTEAM	0.8873	6.74
24	<a href="#">xyzconsulting.com</a>	0.8874	6.73
25	ICMLsubmission	0.8875	6.72
26	Efratko	0.8877	6.70
27	<a href="#">Kitty</a>	0.8881	6.65
28	SecondaryResults	0.8884	6.62
29	<a href="#">Birgit Kraft</a>	0.8885	6.61

# Ensembles, for the win



The team BellKor's Pragmatic Chaos is a combined team of BellKor, Pragmatic Theory and BigChaos. [BellKor](#) consists of Robert Bell, Yehuda Koren and Chris Volinsky. The members of [Pragmatic Theory](#) are Martin Piotte and Martin Chabbert. Andreas Töschler and Michael Jahrer form the team [BigChaos](#).

“...There were two main factors which improved the overall accuracy: The quality of the individual algorithms and [the ensemble idea](#) ... Best blending results are achieved when the whole ensemble has the right tradeoff between diversity and accuracy. ”

## Leaderboard

Display top  leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
1	<a href="#">BellKor's Pragmatic Chaos</a>	0.8558	10.05	2009-06-26 18:42:37
Grand Prize - RMSE <= 0.8563				
2	<a href="#">PragmaticTheory</a>	0.8582	9.80	2009-06-25 22:15:51
3	<a href="#">BellKor in BigChaos</a>	0.8590	9.71	2009-05-13 08:14:09
4	<a href="#">Grand Prize Team</a>	0.8593	9.68	2009-06-12 08:20:24
5	<a href="#">Dace</a>	0.8604	9.56	2009-04-22 05:57:03
6	<a href="#">BigChaos</a>	0.8613	9.47	2009-06-23 23:06:52

[http://www.netflixprize.com/assets/GrandPrize2009\\_BPC\\_BigChaos.pdf](http://www.netflixprize.com/assets/GrandPrize2009_BPC_BigChaos.pdf)



## Feature Film, Rating Count at least 25,000, Sci-Fi (Sorted by IMDb Rating Descending)

1-50 of 797 titles. | [Next »](#)

View Mode: [Compact](#) | [Detailed](#)

Sort by: [Popularity](#) | [Release Date](#) | [Date](#)

# Meta Learning: Ensembles



### 1. **Inception** (2010)

PG-13 | 148 min | Action, Adventure, Sci-Fi

★ 8.8

★ Rate this

74 Metascore

A thief who steals corporate secrets through the use of dream-sharing technology is given the inverse task of planting an idea into the mind of a C.E.O.

Director: [Christopher Nolan](#) | Stars: [Leonardo DiCaprio](#), [Joseph Gordon-Levitt](#), [Elliot Page](#), [Ken Watanabe](#)

Votes: 2,156,755 | Gross: \$292.58M



### 2. **The Matrix** (1999)

R | 136 min | Action, Sci-Fi



Photo by [Wan San Yip](#) on [Unsplash](#)



Photo by [David Travis](#) on [Unsplash](#)

NUS CS3244: Machine Learning



Photo by [CDC](#) on [Unsplash](#)

46



# Ensembles



We have  $T$  reports  $h_1, h_2, \dots, h_T$  predicting whether a stock will go up as  $h(x)$ .

We can:

1. Select the most trustworthy of them based on their usual performance

Training performance:  $h(x) = h_{t^*}(x)$  with  $t^* = \operatorname{argmin}_{t \in \{1, 2, \dots, T\}} L_{\text{train}}(h_t)$

2. Let each report have a vote


Uniform Vote:  $h(x) = \operatorname{sign}(\sum_{t=1}^T 1 \cdot h_t(x))$

3. Weight the reports non-uniformly

Weighted Vote:  $h(x) = \operatorname{sign}(\sum_{t=1}^T \alpha_t \cdot h_t(x))$  where  $\alpha_t \geq 0$ .

4. Combine the predictions conditionally

Given the outcome of a test, decide which learner to use.



Which is the  
decision tree?

# Stock Market – Ensemble



We have  $T$  reports  $h_1, h_2, \dots, h_T$  predicting whether a stock will go up as  $h(x)$ .

We can:

1. Select the most trustworthy of them based on their usual performance

Training performance:  $h(x) = h_{t^*}(x)$  with  $t^* = \operatorname{argmin}_{t \in \{1, 2, \dots, T\}} L_{\text{train}}(h_t)$

2. Let each report have a vote

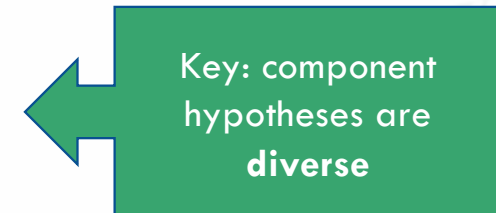
Uniform Vote:  $h(x) = \operatorname{sign}(\sum_{t=1}^T 1 \cdot h_t(x))$

3. Weight the reports non-uniformly

Weighted Vote:  $h(x) = \operatorname{sign}(\sum_{t=1}^T \alpha_t \cdot h_t(x))$  where  $\alpha_t \geq 0$ .

4. Combine the predictions conditionally

This is decision trees, let's finish it up now!



# How is a decision tree an ensemble?



Conditional Ensemble: decide on a learner based on a test.

Divide and Conquer - make smaller problems (like Quicksort)



What type of learner do we use in a decision tree?

Answer: A *decision stump*

# The Decision Stump

a.k.a. 1-level decision tree:

$$h_{s,i,\theta}(x) = s \cdot \text{sign}(x_i - \theta)$$

Positive and negative rays on some feature, three parameters (feature  $i$ , threshold  $\theta$ , direction  $s$ )

Visualization: axis-perpendicular planes in  $\mathbb{R}^n$ .

Efficient to optimize:  $O(n \cdot \log m)$  time

Allows efficient minimization of  $L_{train}$ , but can be too weak to work by itself (hence called a weak learner).

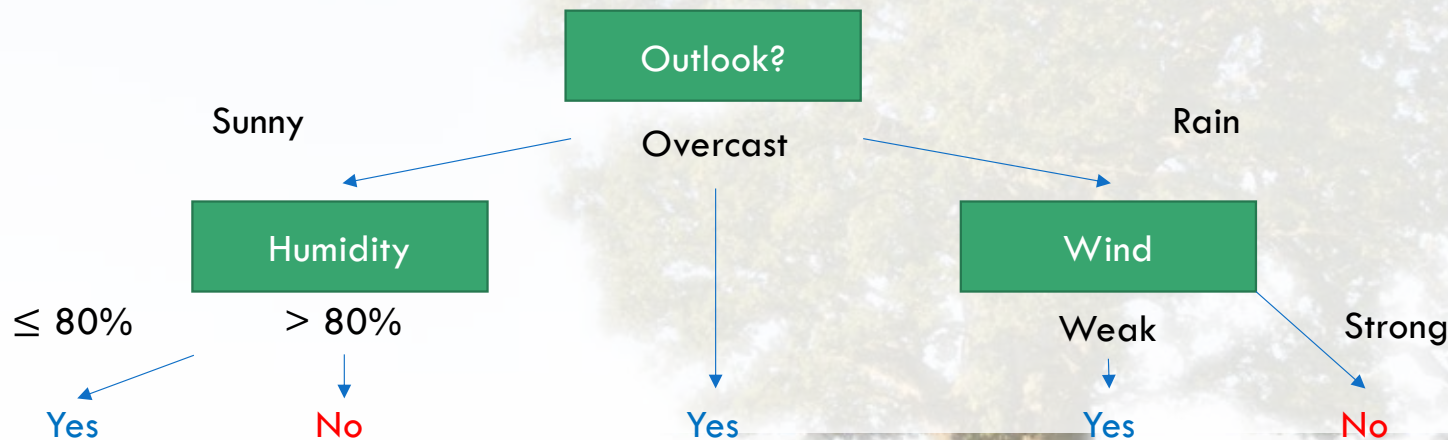


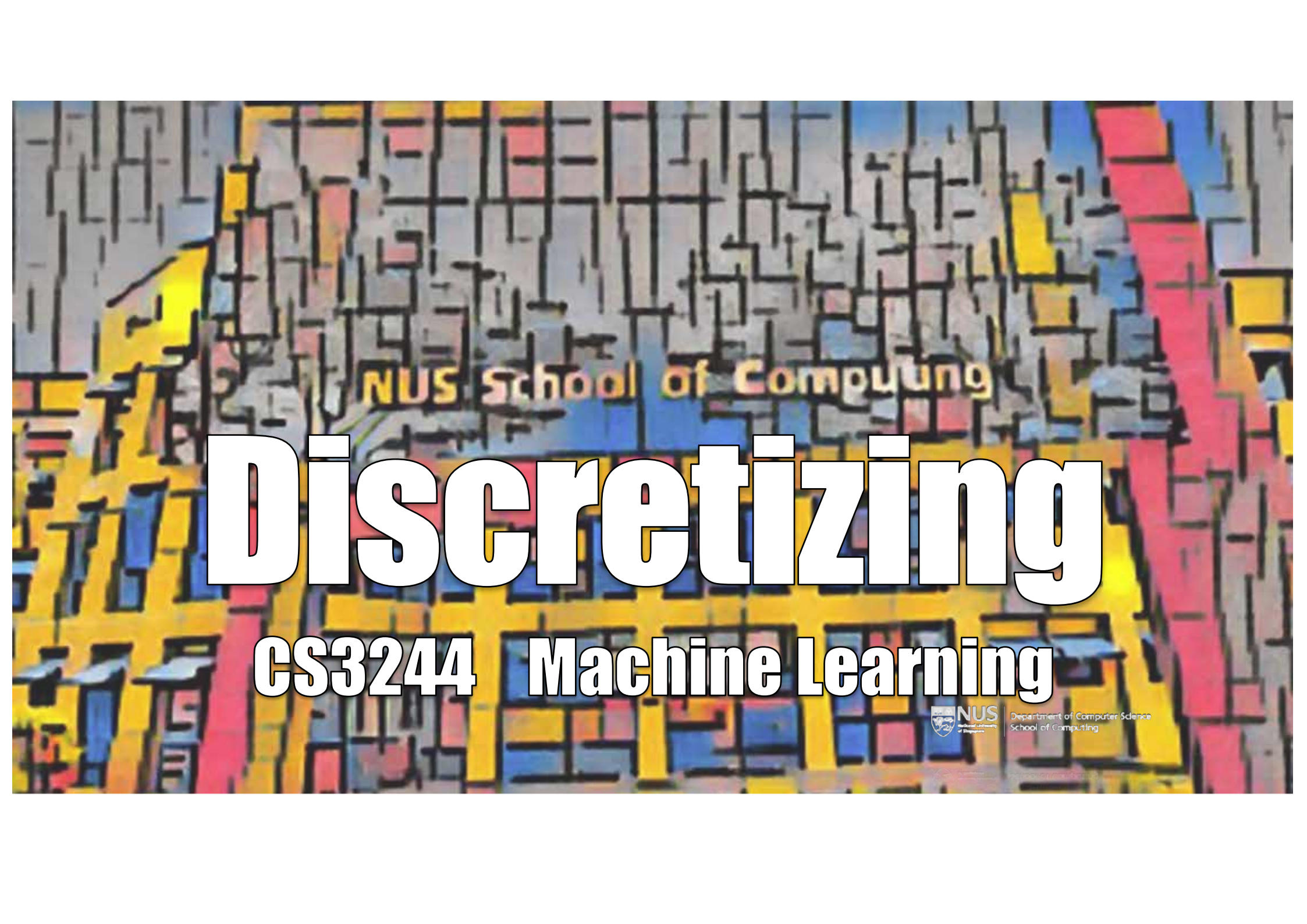


# Building a tree

Conditionally combine each decision stump to build the tree.

Entropy / Information Gain used to make the subproblems smaller and simpler than the root problem.





NUS School of Computing


# Discretizing

CS3244 Machine Learning



Department of Computer Science  
School of Computing





```

function DTL(examples, attributes, default) returns a decision tree
  if examples is empty then return default
  else if all examples have the same classification then return the classification
  else if attributes is empty then return MODE(examples)
  else
    best ← CHOOSE-ATTRIBUTE(attributes, examples)
    tree ← a new decision tree with root test best
    for each value  $v_i$  of best do
      examplesi ← {elements of examples with best =  $v_i$ }
      subtree ← DTL(examplesi, attributes − best, MODE(examples))
      add a branch to tree with label  $v_i$  and subtree subtree
  return tree

```



# 3 Important Questions

Prefer important decisions at the top

1. How do we pick a feature to split on? (*already covered*)
2. How do we discretize continuous features?

Prune back complete trees<sup>\*</sup>

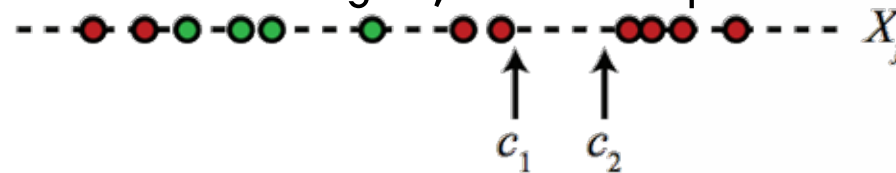
3. How do we decide where to stop pruning?

<sup>\*</sup> we could decide when to stop growing the tree, but in practice this doesn't work well.

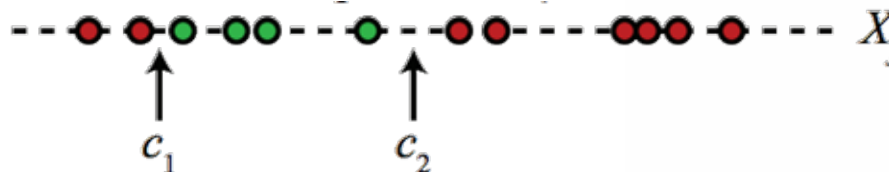
## Q2. How do we discretize continuous features?

How can we choose among an infinite number of split points for a continuous feature  $x_j$ ?

Observation 1. Moving split points between two observed values with the same label has no effect on information gain; use the midpoint.



Thus, 2. Only splitting between examples from different labels  $y$  could improve information gain.





# Pruning

**CS3244 Machine Learning**



Department of Computer Science  
School of Computing

### Q3. How do we decide where to stop pruning?



In Zoom breakout or physical subgroups, do:

(5 mins): Answer: *Why bother pruning? Why not build the decision tree and just use it?*

Ask one member to write your collective answer to the [#general](#) thread.  
Upvote others that you like.

```
function DTL(examples, attributes, default) returns a decision tree
  if examples is empty then return default
  else if all examples have the same classification then return the classification
  else if attributes is empty then return MODE(examples)
  else
    best ← CHOOSE-ATTRIBUTE(attributes, examples)
    tree ← a new decision tree with root test best
    for each value  $v_i$  of best do
      examplesi ← {elements of examples with best =  $v_i$ }
      subtree ← DTL(examplesi, attributes – best, MODE(examples))
      add a branch to tree with label  $v_i$  and subtree subtree
    return tree
```

# The need for pruning

Day	Outlook	Temperature	Humidity	Wind	Tennis?
1	Sunny	Hot	95%	Weak	No
2	Sunny	Hot	92%	Strong	No
3	Overcast	Hot	95%	Weak	Yes
4	Rain	Mild	90%	Weak	Yes
5	Rain	Cool	60%	Weak	Yes
6	Rain	Cool	65%	Strong	No
7	Overcast	Cool	70%	Strong	Yes
8	Sunny	Cool	70%	Weak	Yes



# Analyzing Decision Trees



How expressive is the  $\mathcal{H}$  of decision trees?

They can express any row  $(x_1, x_2, \dots, x_n, y)$  in the data set, trivially (how?).

You can always have  $L_{train} = 0$  for a decision tree, if there is no **noise** in  $y$ .

What's the problem with an unpruned learned decision tree, in terms of learning theory?

Little inductive bias, memorises training data. Won't generalize well.



### Q3. How do we decide where to prune?

Answer:

1. Don't let leaves get too small. Require leaves have at least  $i$  training data.
2. Don't let tree grow too deep. Don't allow more than  $j$  levels of tests.
3. Use part of the training data to check performance (called **validation**)!

Remove node and replace by the **MODE** of labels  $y$  if the pruned tree performs better than original.



# Extending for Regression



*(Regression  $\equiv$  Continuous  $y$  output aka predictor variable)*

Both Decision Trees and  $k$  NN predict categorical features.

How can we extend these two algorithms to handle regression?

- Decision Trees: A test in a leaf predicts the **MEAN** of its training instances.
- $k$  NN: a test instance predicts the **MEAN** of its  $k$  NN.

*Many other options for extending to regression.  
What could you propose?*

# Summary:

## Analyzing Decision Tree Learning



### The good

Interpretable: Intuitive for data exploration.

Efficient: both in training (greedy search) and testing

Discards irrelevant feature through use of Information Gain.

### The bad

Instability: Susceptible to small fluctuations (high variance)

Hard decision boundaries: by default, no probabilistic interpretation of boundaries

Axis perpendicular decisions: doesn't capture interaction between features (feature AND bug)



# Wrapping up Week 03

**CS3244 Machine Learning**



Department of Computer Science  
School of Computing

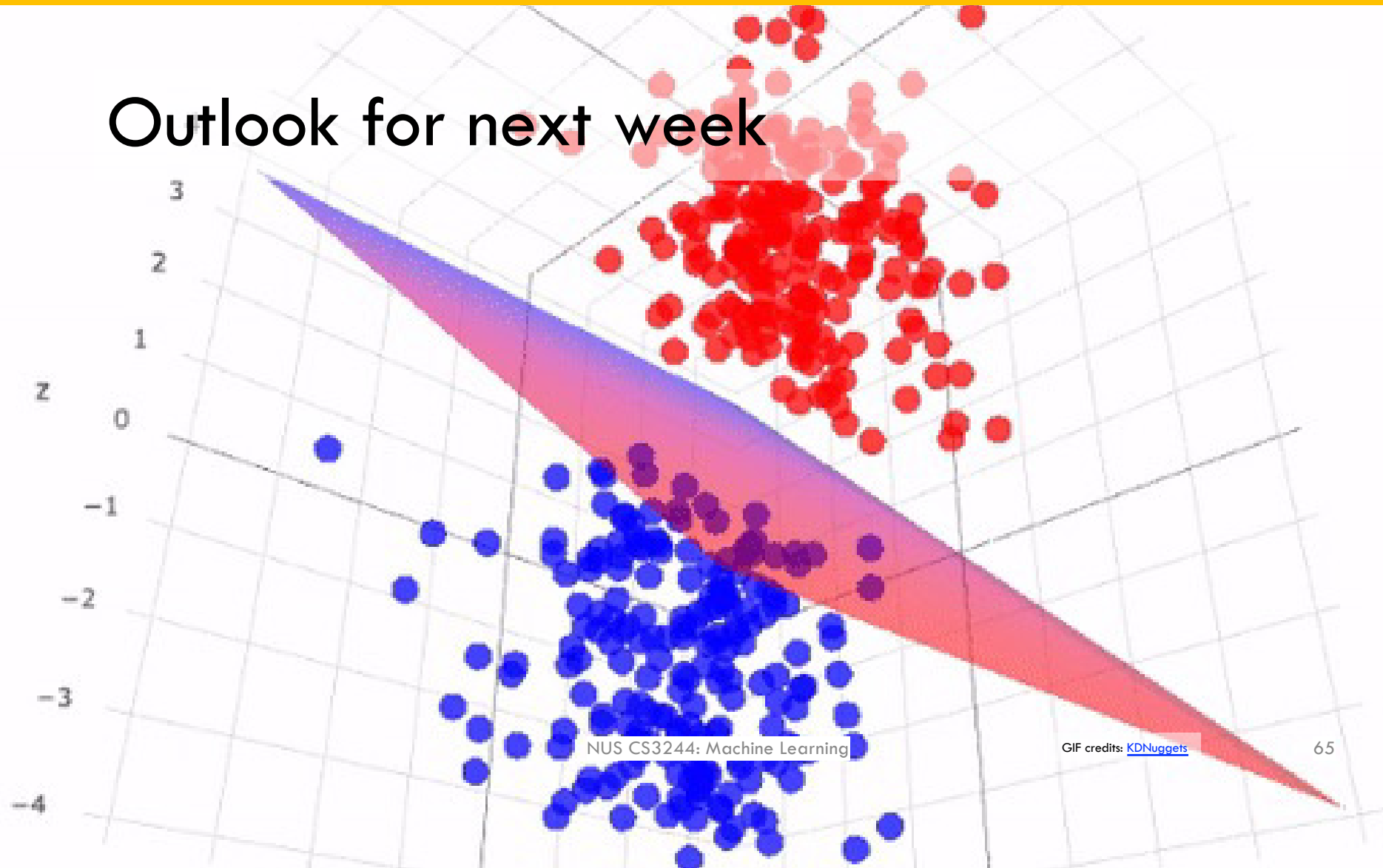


# What did we learn this week?



- Learning Outcomes for this week:
- Understand and implement Decision Trees;
- Master Information Gain as a means of choosing optimal features for decisions;
- Formalize cost functions as a means of evaluating a supervised learner's performance;
- Describe simple ensembling methods that integrate multiple classifiers;
  - Understand Decision Trees as an instance of an ensemble classifier of conditional decision stamps;
- Reason about inductive bias of the decision tree algorithm

# Outlook for next week



NUS CS3244: Machine Learning

GIF credits: [KDNuggets](#)

65



# Assignment #1 released



Min, go over the *k* NN versus Decision Tree assignment:

[http://www.comp.nus.edu.sg/~cs3244/AY2122S1/  
01.assignment.html](http://www.comp.nus.edu.sg/~cs3244/AY2122S1/01.assignment.html)

Due in two weeks.

## Assigned Task (due before next Mon)



Read the Quora post [Whats-the-difference-between-linear-and-non-linear-machine-learning-model](#), especially Robby's answer (5 mins)  
(You may want to use guest / incognito mode, if asked to sign up)

Post a 1-2 sentence answer to the question in tutorial group: [#tg-xx](#)

Can we model non-linearity with a linear classifier?

[Don't worry if you're not sure, we'll cover this again in Week 04.]