OFFICE HOURS (POSTED ON PIAZZA)

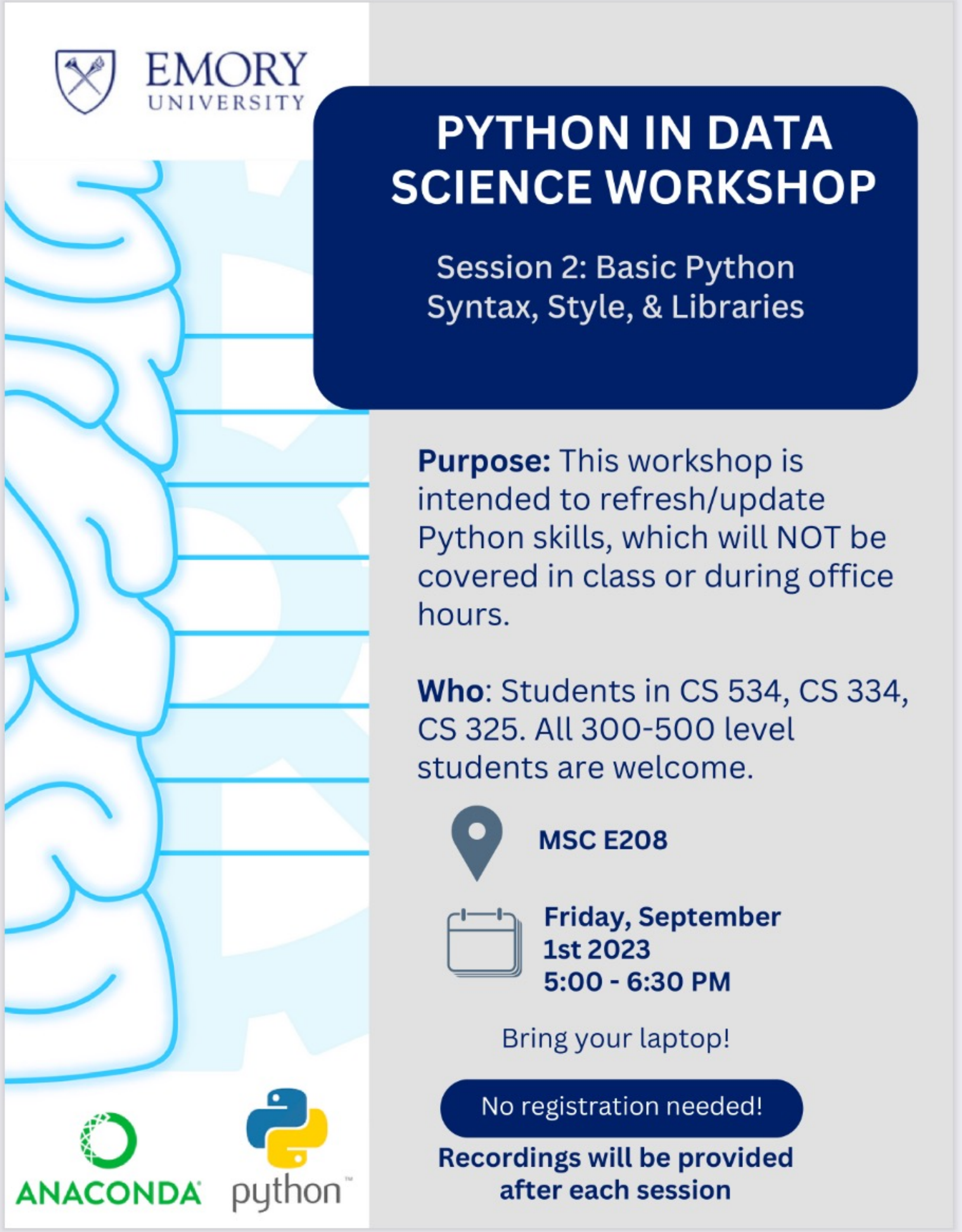
Name	Office Hours	
Li Xiong	When?	MW 2:15-3:15pm
	Where?	Math & Science Center, E412
Joyce Ho	When?	T 1:00-2:00pm; Th 2:00-3:00pm
	Where?	Math & Science Center, W302M
Hong kyu Lee	When?	T 2:30-3:30
	Where?	Math & Science Center, E308
Helen Zeng	When?	T 3:00-5:00pm
	Where?	Math & Science Center, E308
Tiantian Li	When?	W 10:00am-12:00pm
	Where?	Math & Science Center, E308

HOMEWORK #1 ANNOUNCEMENT

- Out 8/28, Due **9/12 @ 11:59 PM ET** on Gradescope
- 4 questions
  - Q1-Q2: Get familiar with Python
    - Numerical programming (Numpy)
    - Dataset loading and visualization (Pandas and other libraries)
  - Q3-Q4: kNN
    - Implement kNN (use Numpy)
    - Evaluate kNN (use sklearn)

PYTHON WORKSHOP 2

- Basics
- Libraries
  - Numpy
  - Pandas
  - Matplotlib



MACHINE LEARNING WITH PYTHON

- Scikit-learn ML library
- Various ML algorithms, preproessing, model evaluation and selection metrics/tools
- Works well with other Python libraries, such as NumPy for linear algebra and array vectorization, Pandas dataframes for data manipulation, Matplotlib and Pandas for plotting

KNN IN PYTHON: SKLEARN

- Import the class containing the classification method

- Create an instance of the class (setting k and distance)

- Fit the training data and predict the values

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=3,
                           metric='minkowski',
                           p=2)
knn.fit(xData)
yHat = knn.predict(xData)
```

FEATURE SCALING: SKLEARN

- Import the class containing the scaling method

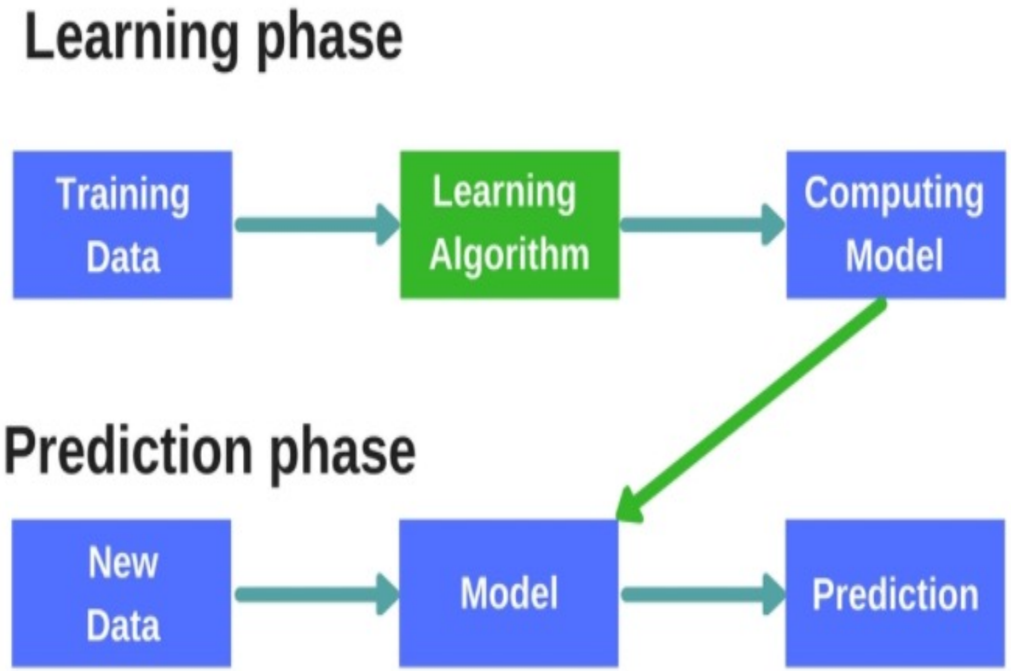
- Create an instance of the class

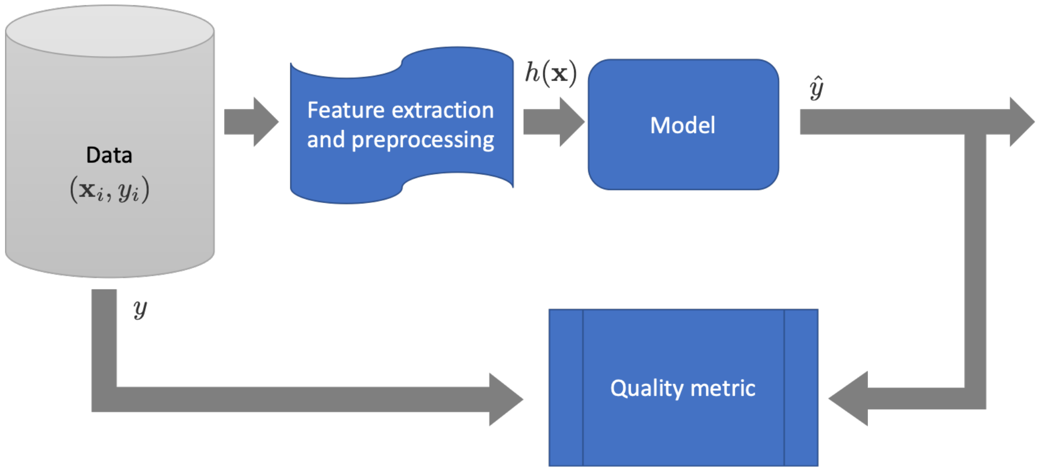
- Fit the scaling parameter and transform data

```python
from sklearn.preprocessing import StandardScaler

stdScale = StandardScaler()
stdScale.fit(xData)
xScaled = stdScale.transform(xData)
```

MODEL ASSESSMENT (MORE LATER)

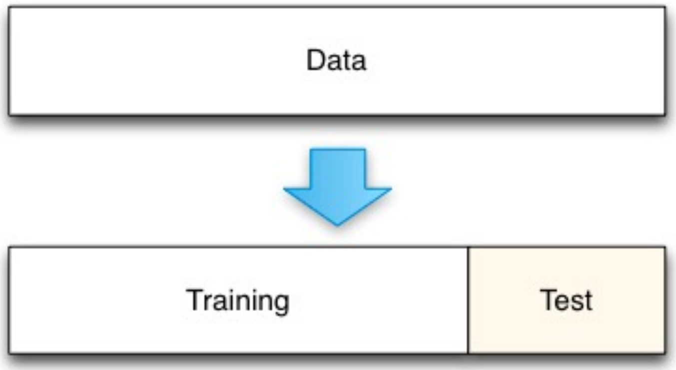




Accuracy is a common metric

HOLDOUT: FORMING A TEST SET

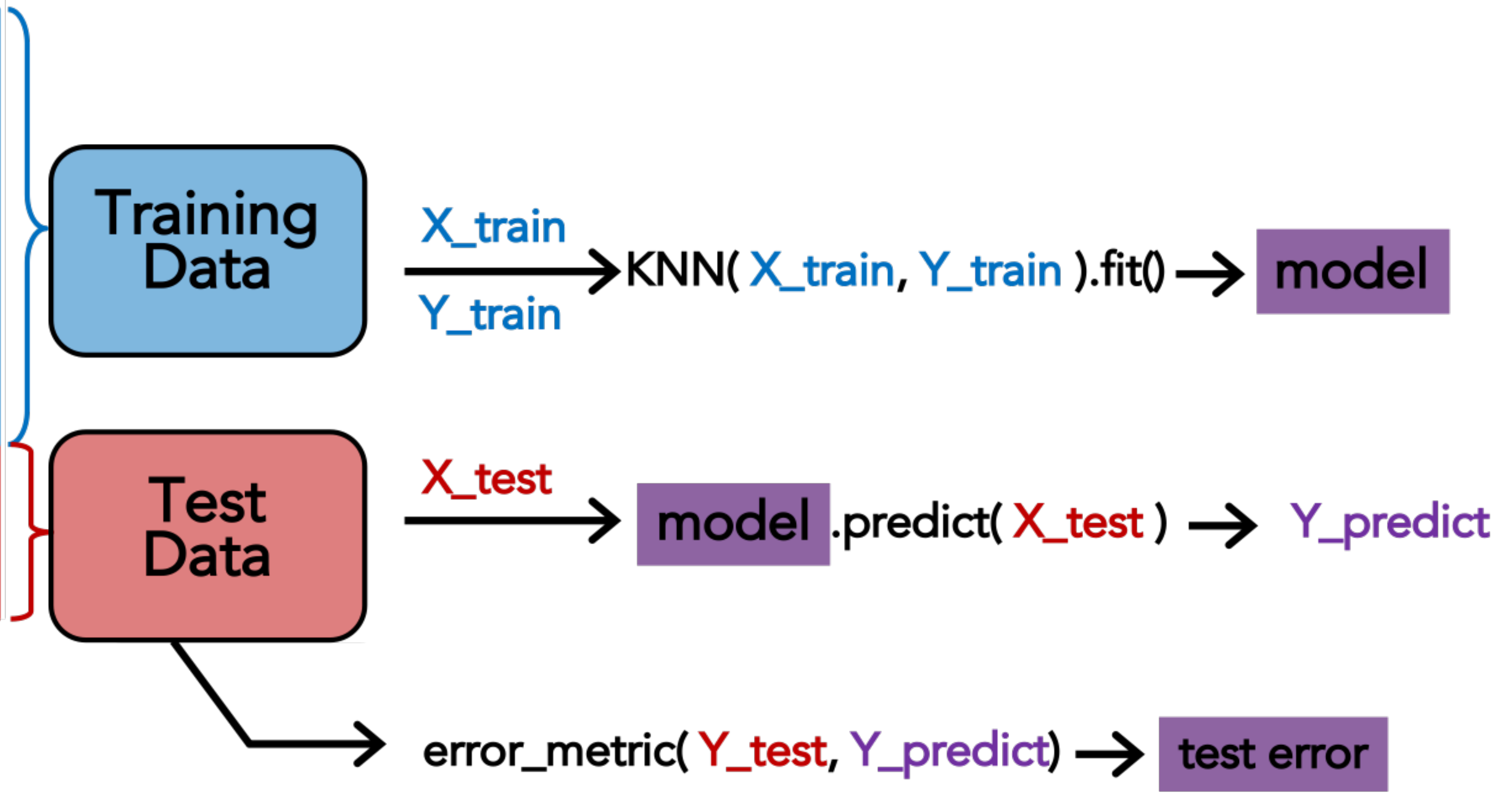
- Hold out some data (i.e., test data) that are not used for training the model
- Proxy for "everything you might see"



http://scott.fortmann-roe.com/docs/MeasuringError.html

TRAINING/TEST SPLIT AND TEST ERROR

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143



DEMO: KNN.IPYNB

https://colab.research.google.com/drive/1ZoDU_oPBiunvhrr_4FlaURBqkKqIVVkm#scrollTo=RUDuGt6y-pwM

COURSE OUTLINE

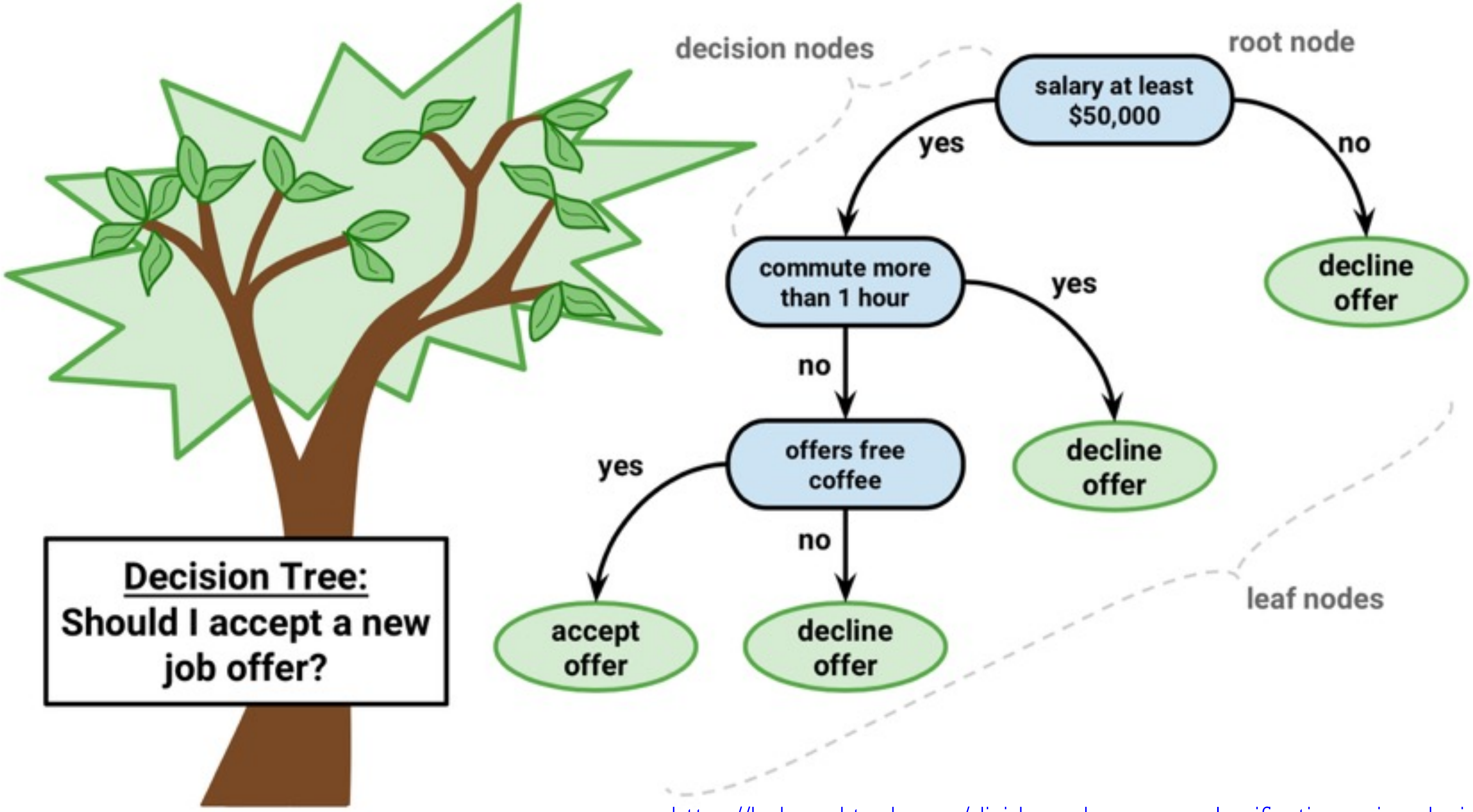
- Algorithms for supervised learning: nearest neighbors, **decision trees**, linear regression, logistic regression, neural networks, naïve bayes, ensembles, boosting, deep learning
- Algorithms for unsupervised learning: principal component analysis
- Model assessment and model selection
- New learning paradigms and emerging topics

DECISION TREES

CS 334: Machine Learning

Slides adapted from Joyce Ho, Intel ML Academy, David Sontag, Luke Zettlermoyer, Carlos Guestrin, Andrew Moore, Mengye Ren, Matthew MacKay, and Yubin Park

REAL-WORLD INSPIRATION



https://hub.packtpub.com/divide-and-conquer-classification-using-decision-trees-and-rules/

GROUP ACTIVITY

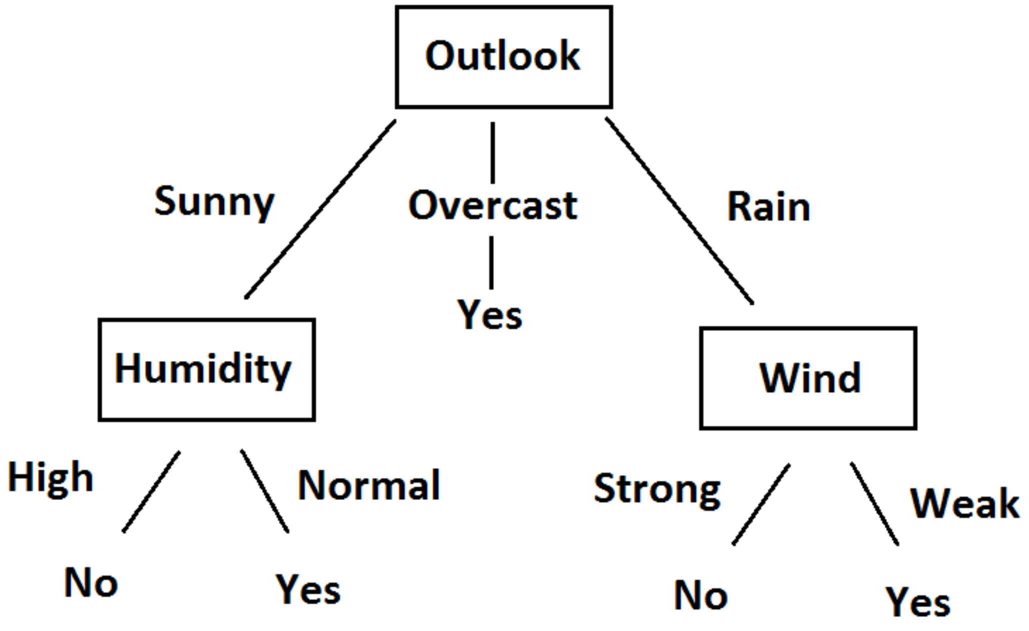
DO WE PLAY TENNIS TODAY?

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

today **Sunny** **Mild** **High** **Weak** **?**

DO WE PLAY TENNIS TODAY?

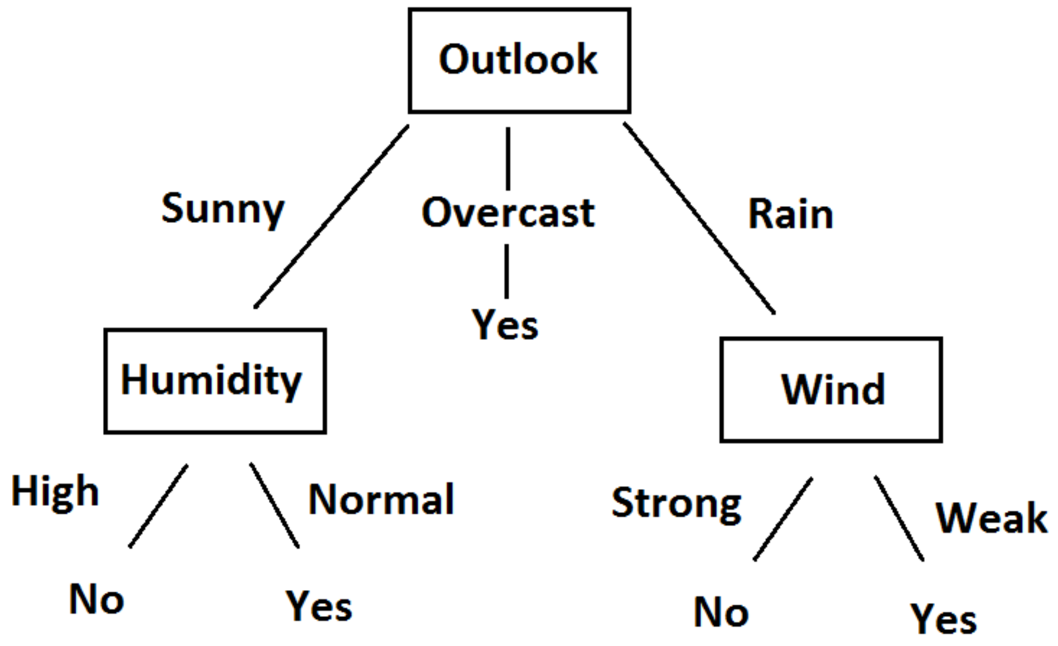
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No
today	**Sunny**	**Mild**	**High**	**Weak**	**?**



DECISION TREE

A tree structure

- Each internal (decision) node represents a test on a feature, each branch represents a value
- Each leaf node represents a class label
- Each path represents a classification/decision rule following successive choices



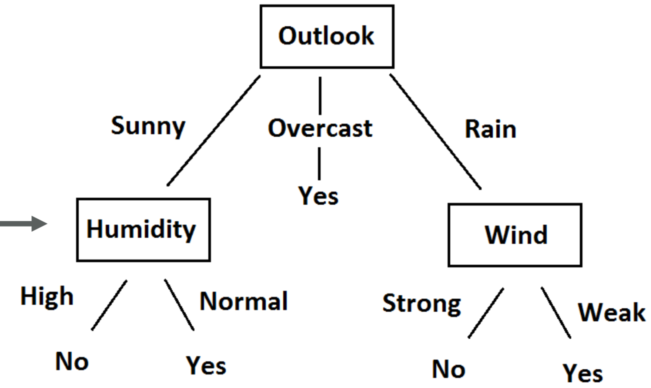
https://towardsdatascience.com/decision-tree-in-machine-learning-e380942a4c96

https://sefiks.com/2017/11/20/a-step-by-step-id3-decision-tree-example/

DECISION TREE

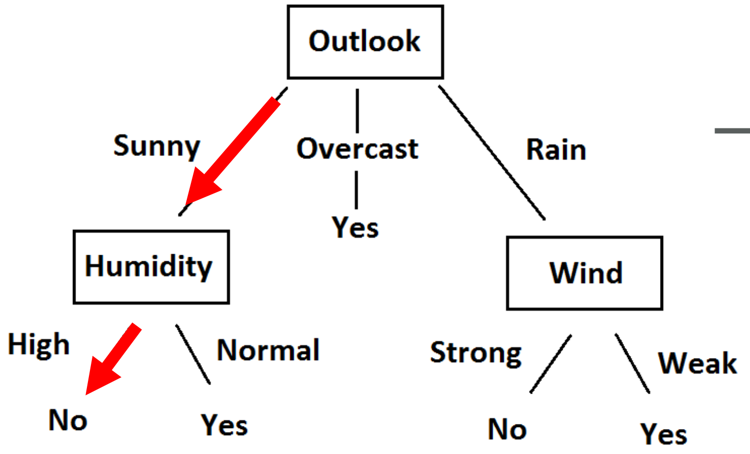
- Training: Build a decision tree from training data

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



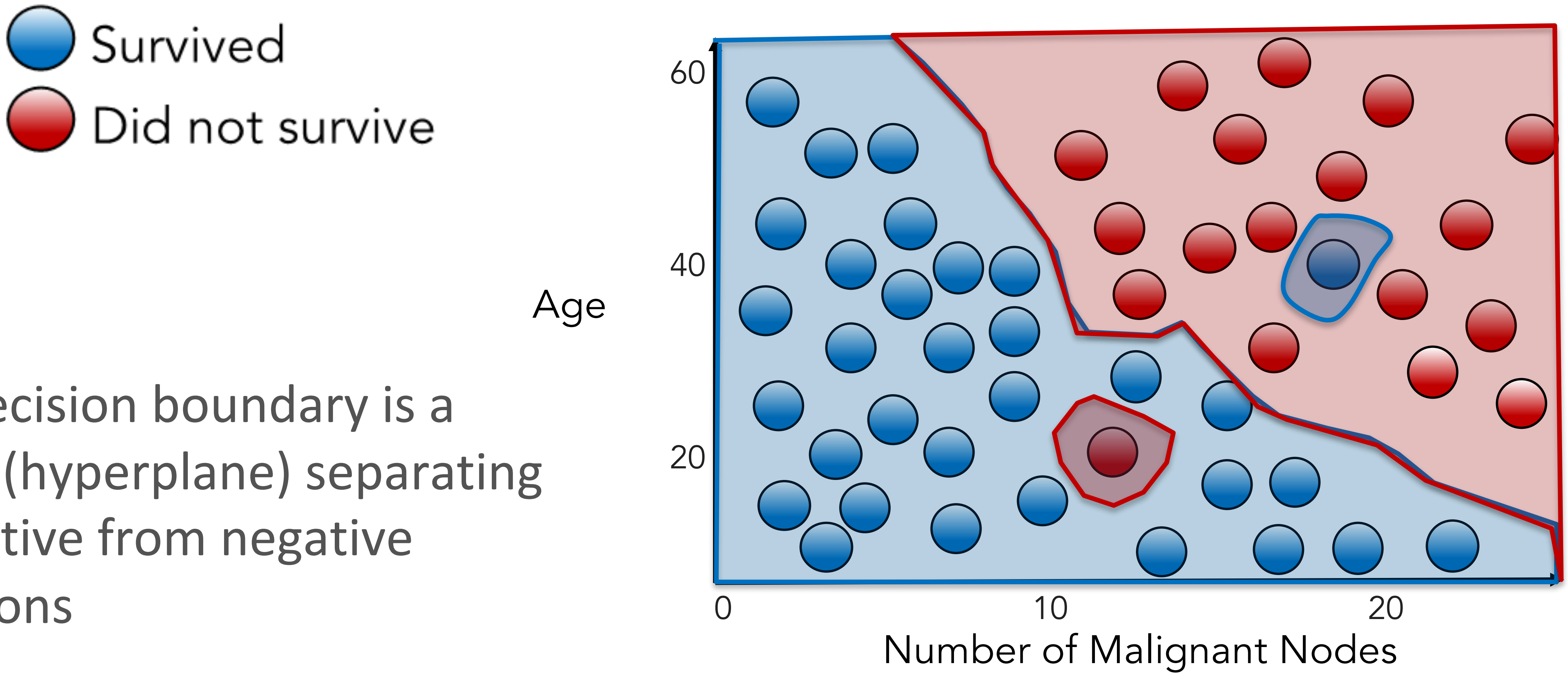
- Prediction: Given a new data point, find the path using its features and predict the label at the leaf node

(Sunny, Mild, High, Weak)



no

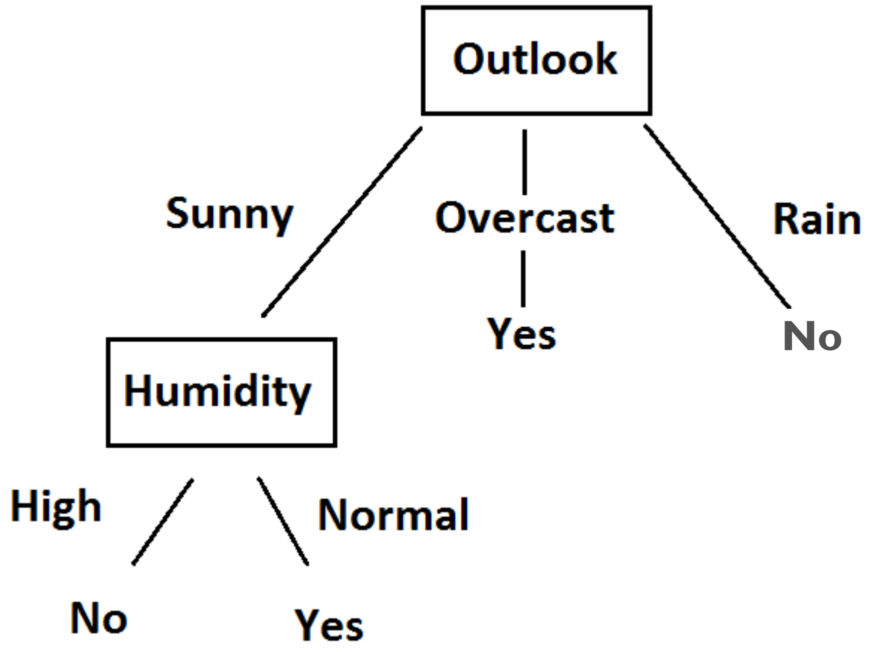
RECALL: KNN DECISION BOUNDARIES



A decision boundary is a line (hyperplane) separating positive from negative regions

DECISION TREE: DECISION BOUNDARY

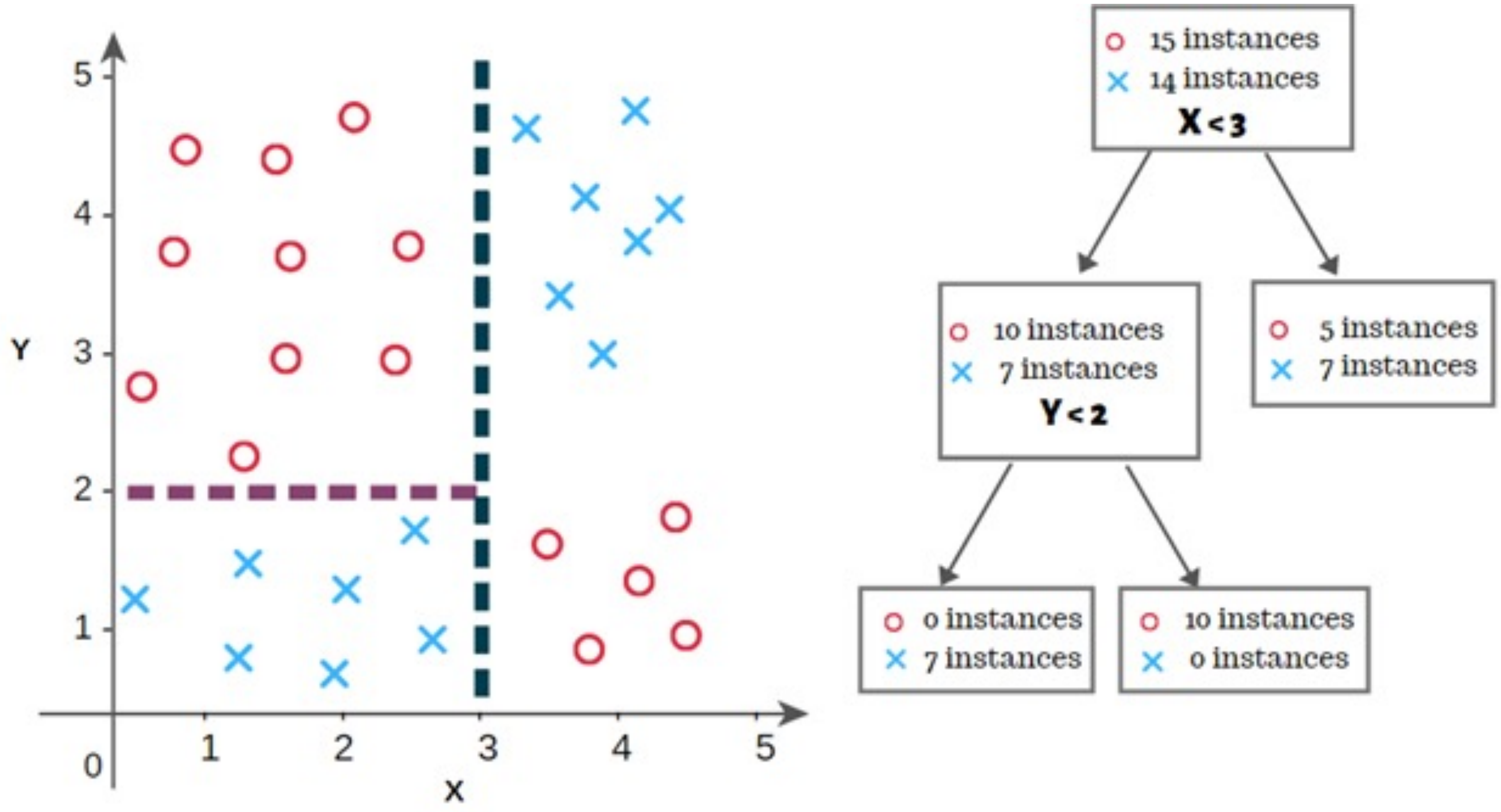
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



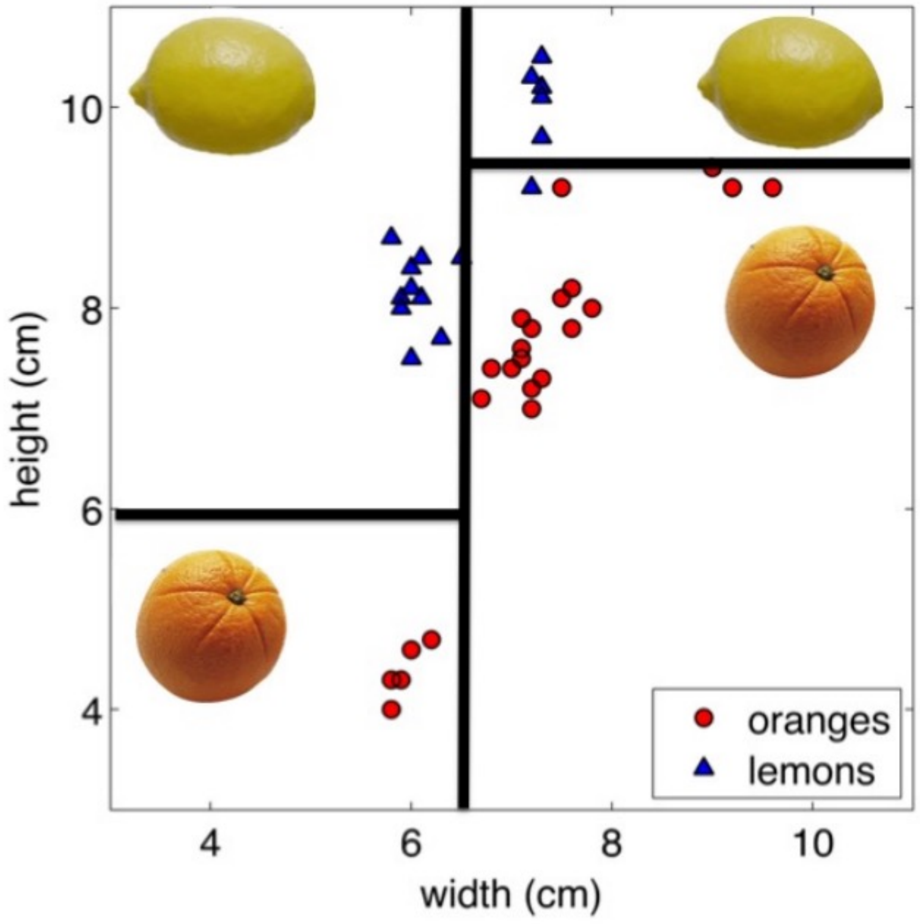
What does the decision boundary look like for a decision tree? (Think about only outlook and humidity)

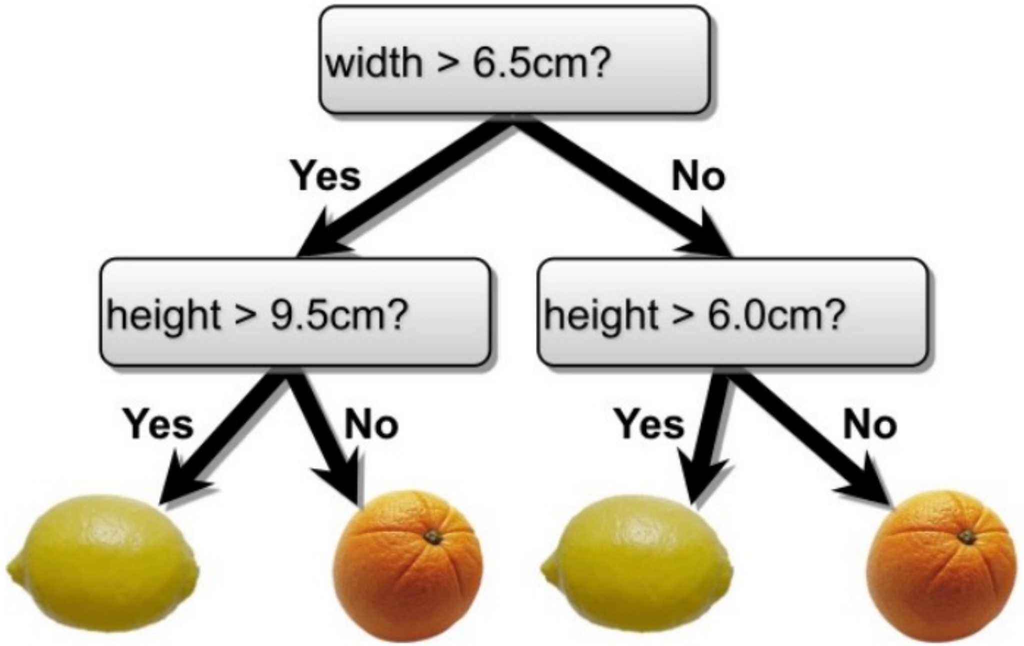
DECISION TREE: DECISION BOUNDARY

- A decision tree gives axes-aligned decision boundaries that divide the feature space into rectangles (hypercubes)
- Each internal node (split) represents a decision line splitting the current region into subregions (one for each branch)
- Each leaf node represents a rectangle (hypercube)



EXAMPLE: LEMONS OR ORANGES

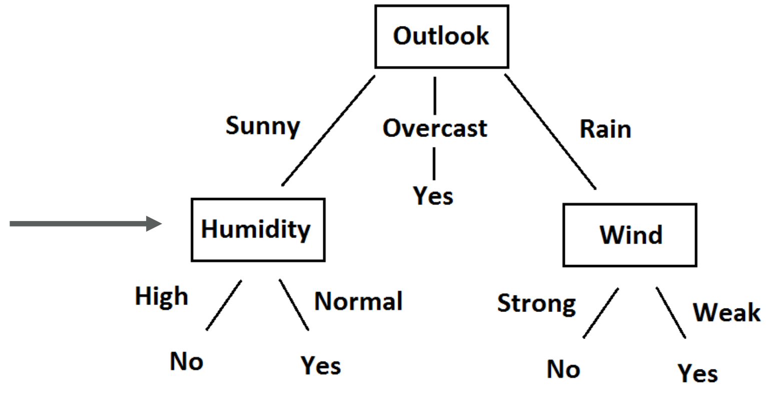




DECISION TREE

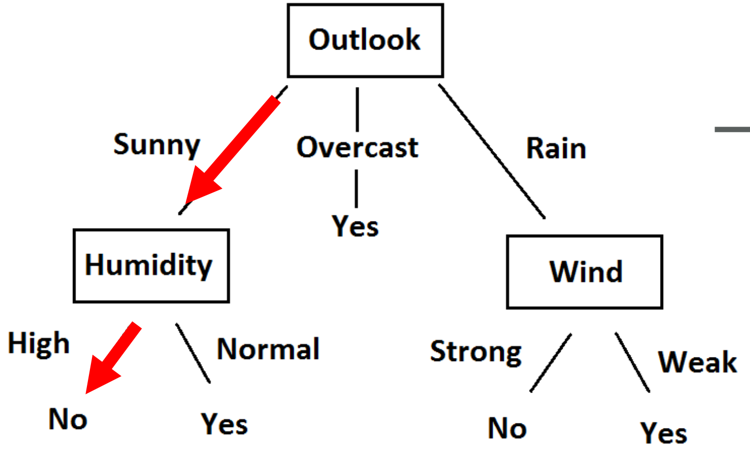
- Training: Build a decision tree from training data

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



- Prediction: Given a new data point, find the path using its features and predict the label at the leaf node

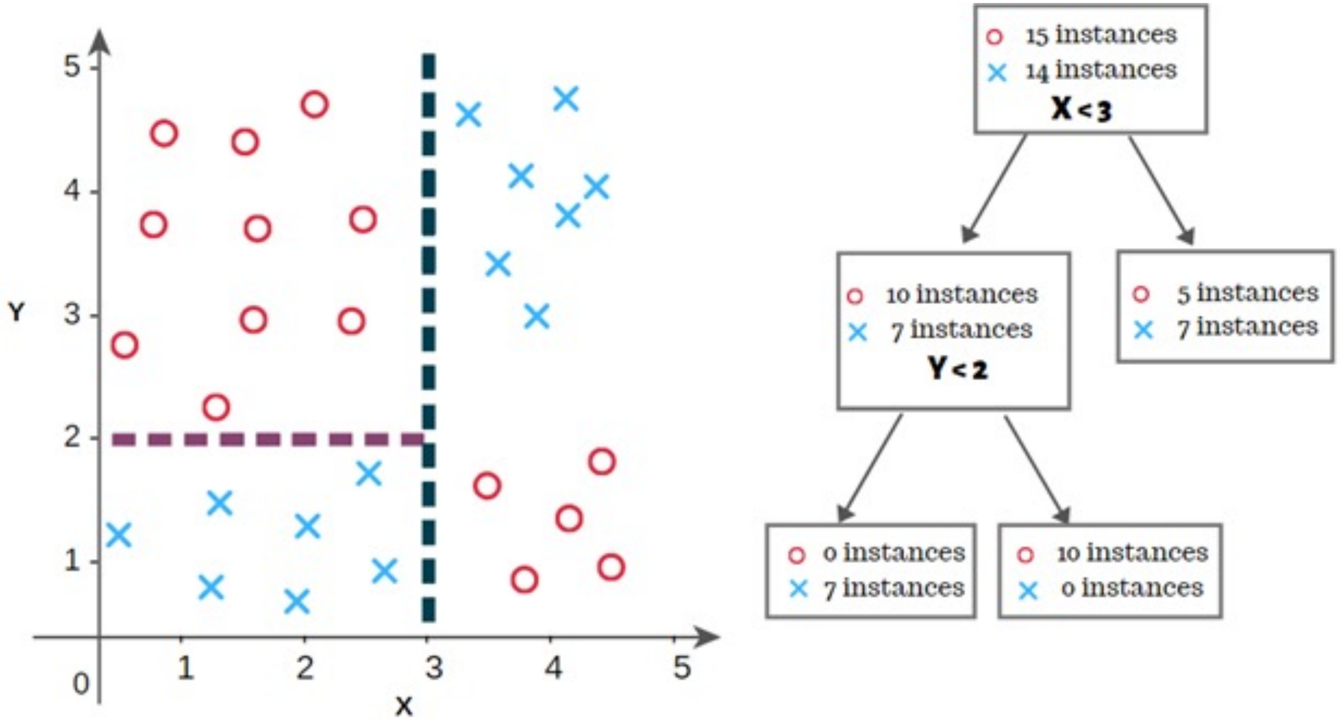
(Sunny, Mild, High, Weak)



→ no

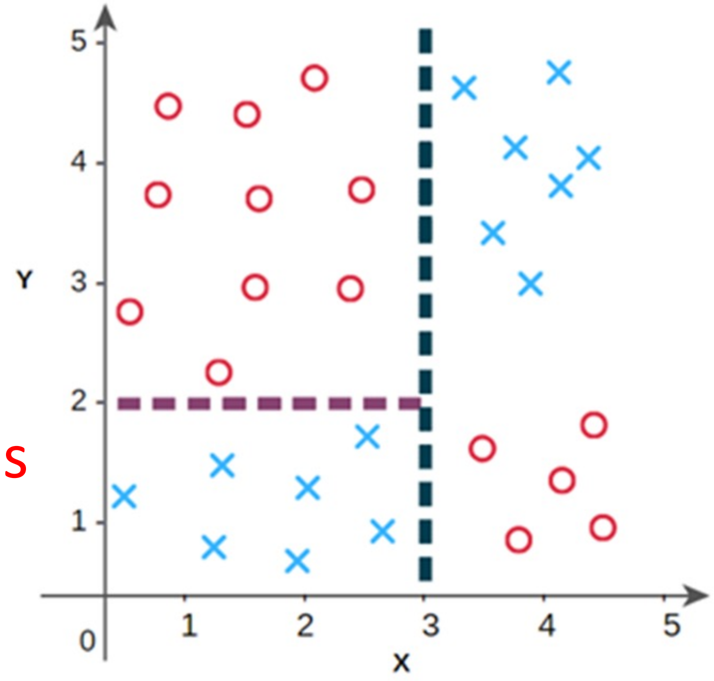
HOW TO LEARN THE TREE?

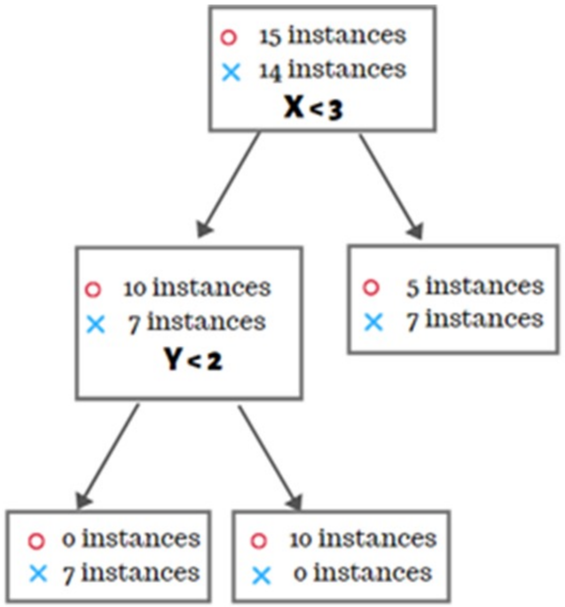
- How to learn the tree?
  - How to choose the node (splits)?
  - When to stop the tree (how big to grow)?



DECISION TREE: TRAINING

- Constructing the optimal (simplest) binary decision tree is an NP-complete problem [Hyafil & Rivest '76]
- Top-down, greedy strategy
  - Start from empty tree
  - Select the "best" remaining feature and split current region to subregions
  - Recurse on each subregion
  - Stop when a node has unambiguous outcome or no more features





DECISION TREE: TRAINING (C4.5 ALGORITHM)

Algorithm 1.1 C4.5(D)

Input: an attribute-valued dataset D

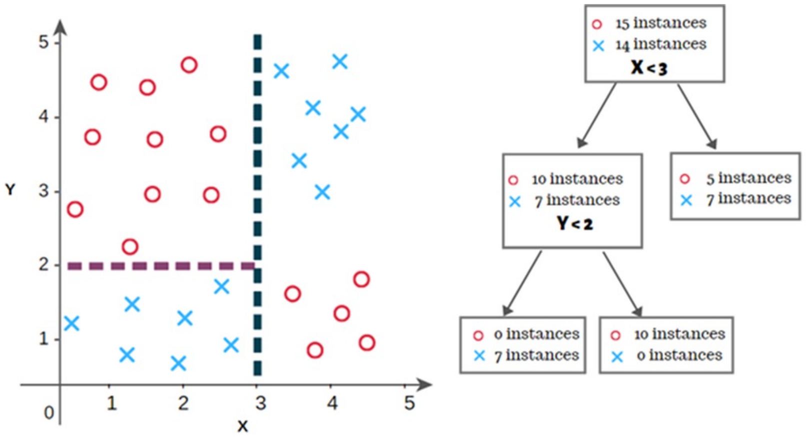
Tree = {}
if D is "pure" OR other stopping criteria met **then**
  terminate
end if
for all attribute $a \in D$ **do**
  Compute information-theoretic criteria if we split on a
end for
a_{best} = Best attribute according to above computed criteria
Tree = Create a decision node that tests a_{best} in the root
D_v = Induced sub-datasets from D based on a_{best}
for all D_v **do**
  Tree$_v$ = C4.5(D_v)
  Attach Tree$_v$ to the corresponding branch of Tree
end for
return Tree

Stop criteria

Scoring criteria

Select best split attribute

Recurse on subregions

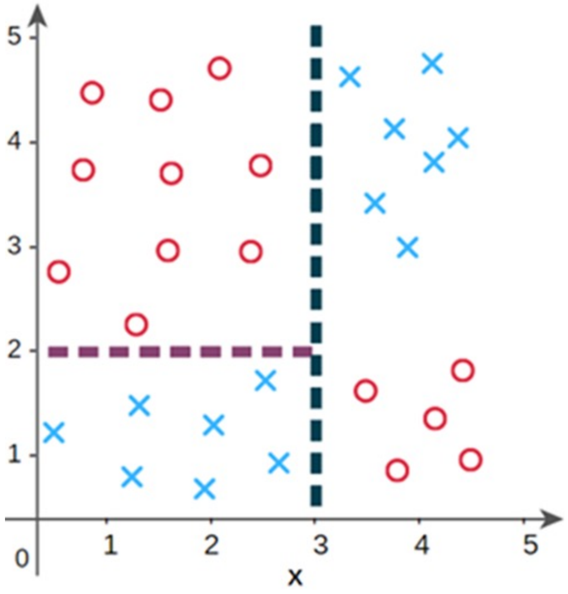


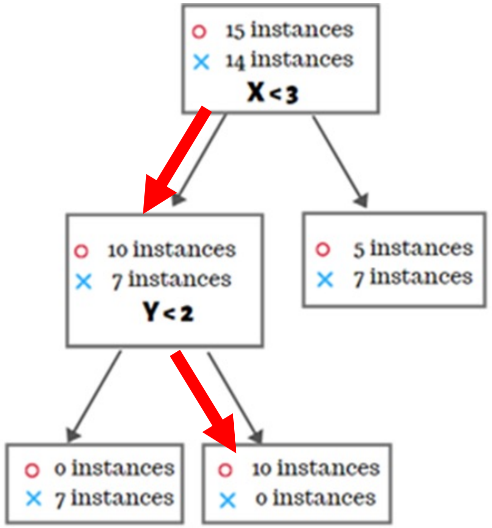
DECISION TREE: PREDICTION

test point = (2, 5)

Algorithm 2 DecisionTreeTest(*tree*, *test point*)

```
if tree is of the form Leaf(guess) then
    return guess
else if tree is of the form Node(f, left, right) then
    if f = yes in test point then
        return DecisionTreeTest(left, test point)
    else
        return DecisionTreeTest(right, test point)
    end if
end if
```



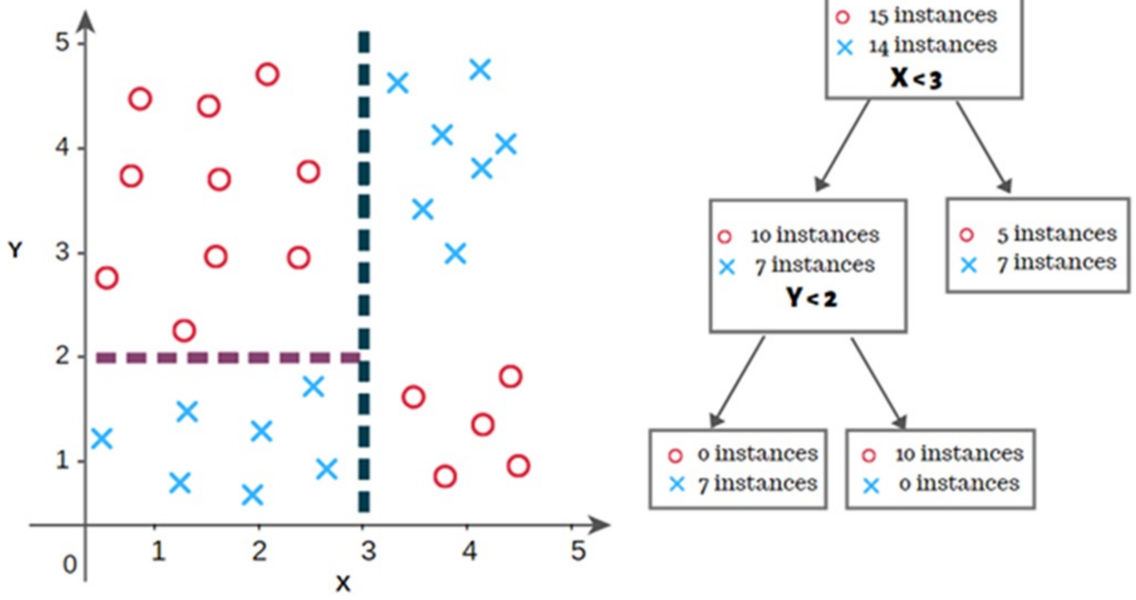


DECISION TREE: CHOOSING BEST ATTRIBUTE

Algorithm 1.1 C4.5(D)

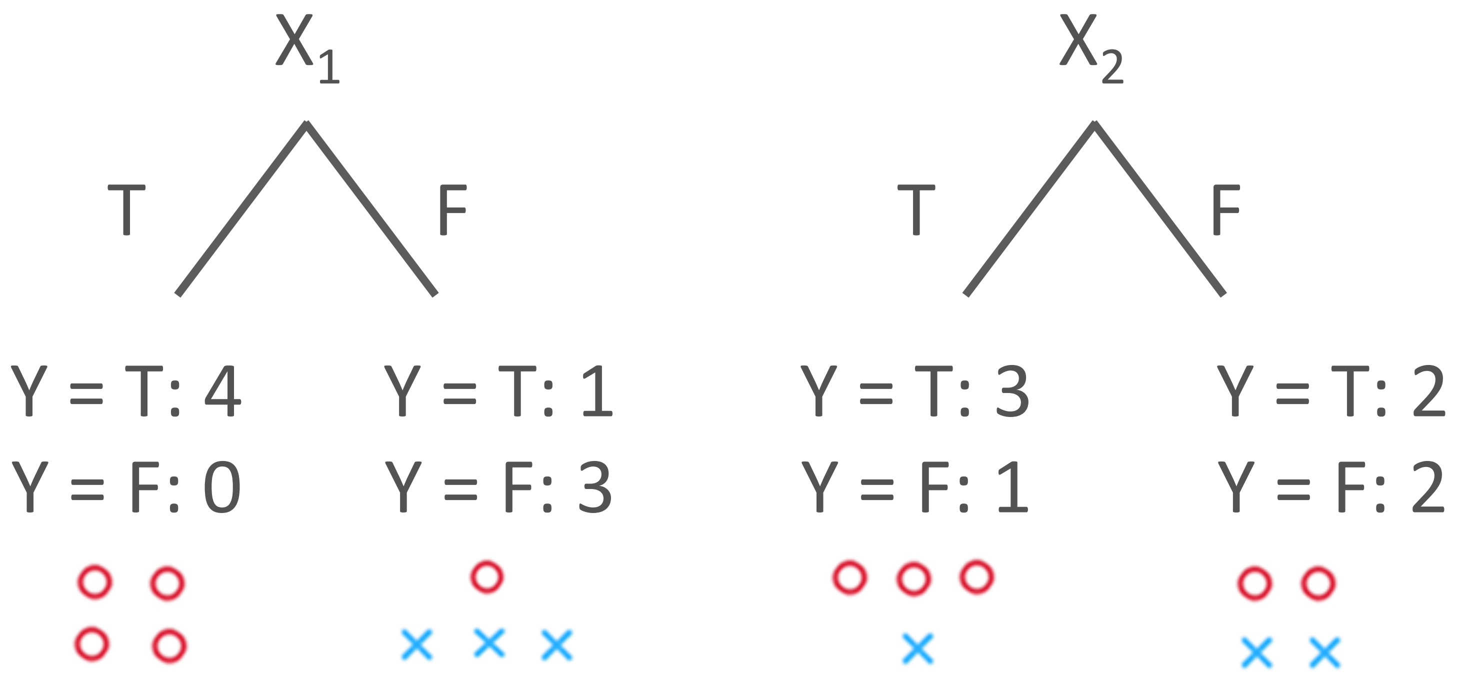
Input: an attribute-valued dataset D

```
Tree = {}
if D is "pure" OR other stopping criteria met then
    terminate
end if
for all attribute a ∈ D do
    Compute information-theoretic criteria if we split on a
end for
a_best = Best attribute according to above computed criteria
Tree = Create a decision node that tests a_best in the root
D_v = Induced sub-datasets from D based on a_best
for all D_v do
    Tree_v = C4.5(D_v)
    Attach Tree_v to the corresponding branch of Tree
end for
return Tree
```



CHOOSING A GOOD SPLIT

- Would we prefer X_1 or X_2?



X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F

CHOOSING A GOOD SPLIT

- Idea: Use counts at the node to define probability distributions to measure uncertainty
  - Deterministic — good (all positive or all negative) 
  - Uniform — bad (all classes have equal probability) 
  - What about in-between? 
- Common metrics
  - Information entropy and information gain (ID3/C4.5)
  - Gini index (CART)

A LITTLE BIT OF INFORMATION THEORY

Flip three different coins (4 sides):

- Sequence 1: A A A A A A A A A A A A … (deterministic)
- Sequence 2: A B C D B A D C B C D A … (uniform: P_A=¼ P_B=¼ P_C=¼ P_D= ¼)
- Sequence 3: A A B C B C A A B A C A … (biased: P_A=½ P_B=¼ P_C=¼ P_D= 0)

What is the minimum number of bits per letter needed to encode each sequence?

A LITTLE BIT OF INFORMATION THEORY

Flip three different coins (4 sides):

- Sequence 1: A A A A A A A A A A A A … (deterministic) 0 bit/letter
- Sequence 2: A B C D B A D C B C D A … (uniform: P_A=¼ P_B=¼ P_C=¼ P_D= ¼)
  000110110100111001101101 A:00 B:01 C:10 D:11 2 bits/letter
- Sequence 3: A A B C B C A A B A C A … (biased: P_A=½ P_B=¼ P_C=¼ P_D= 0)

Can we use less than 2 bits for sequence 3?