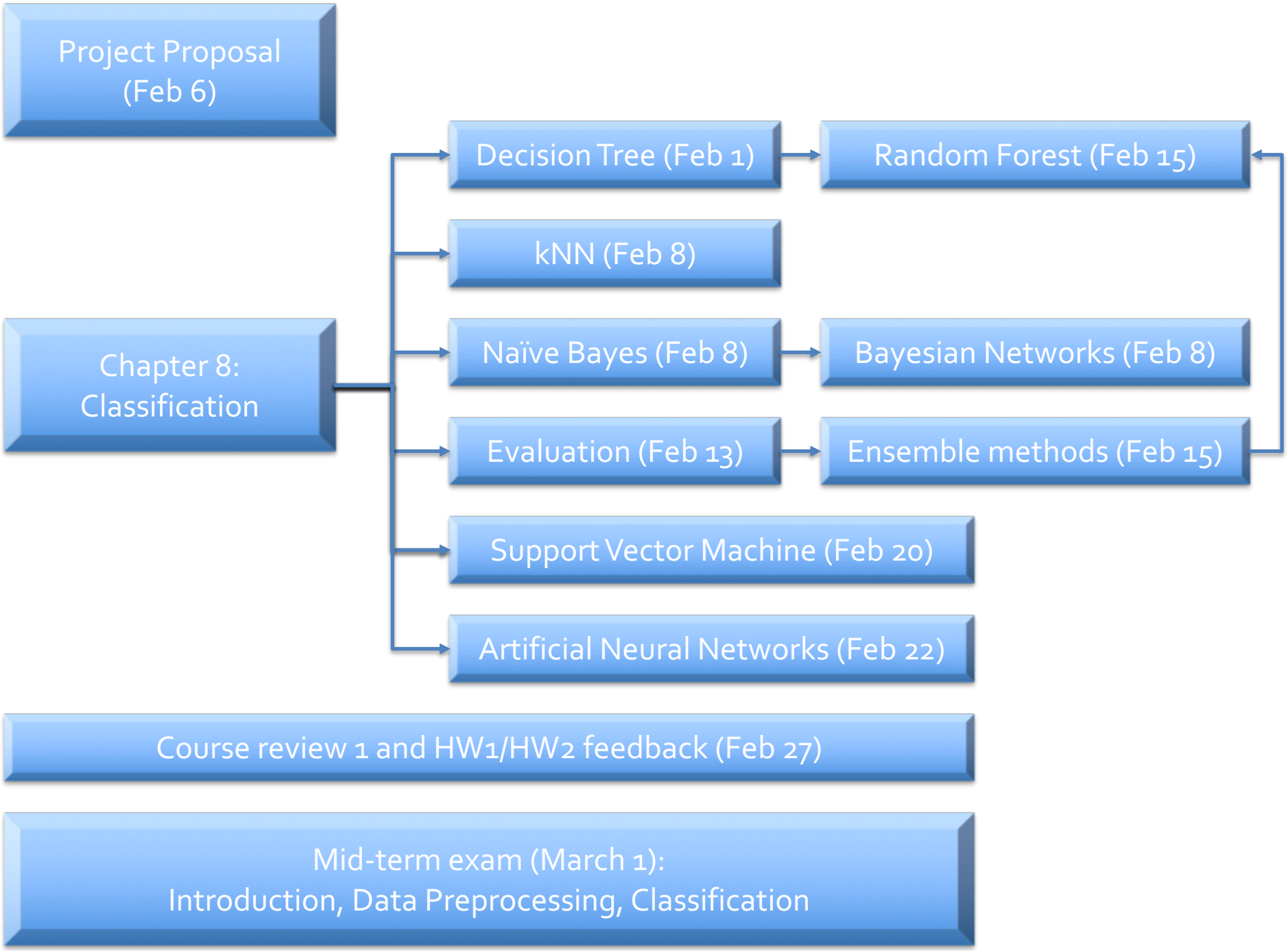
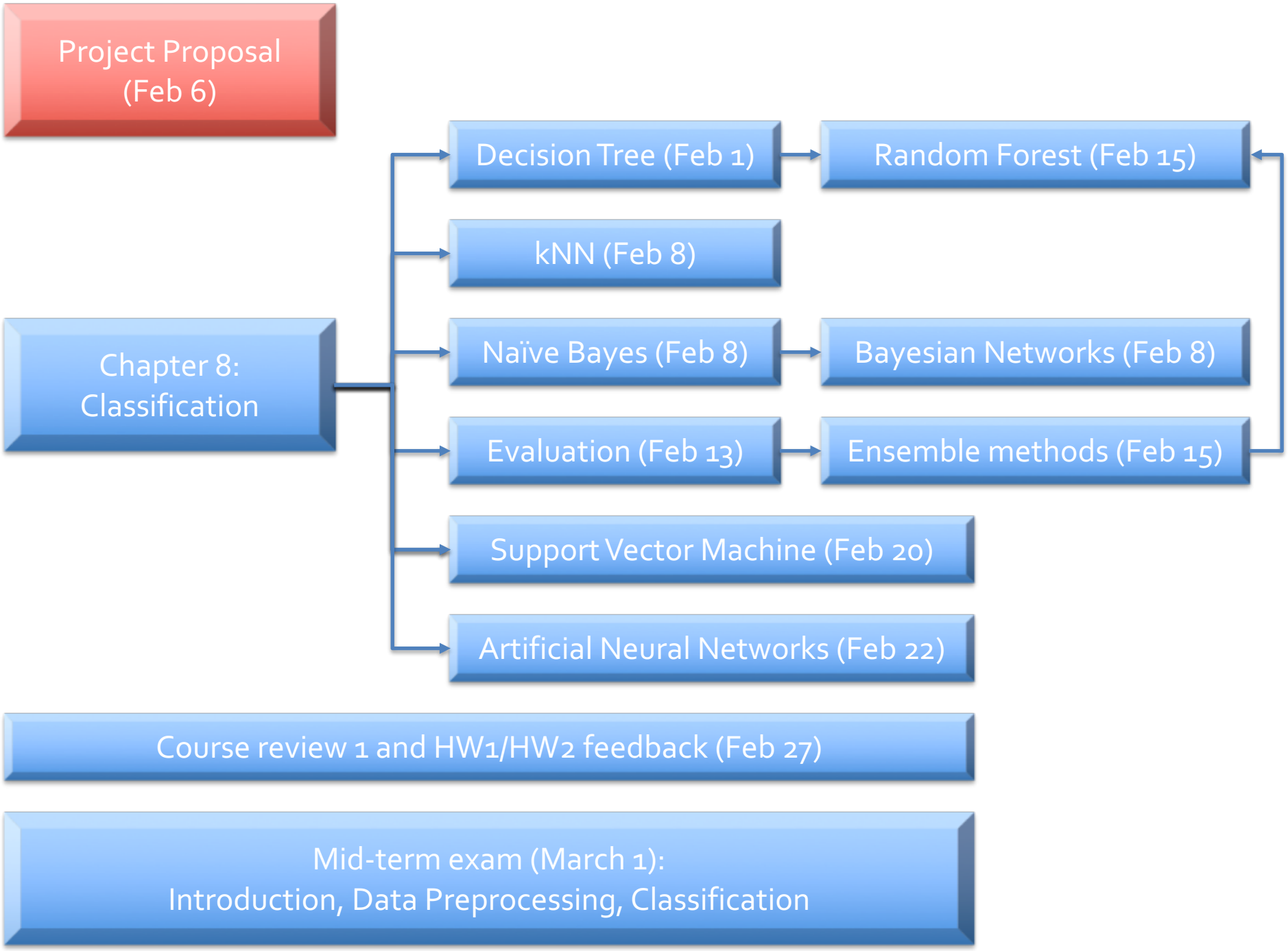
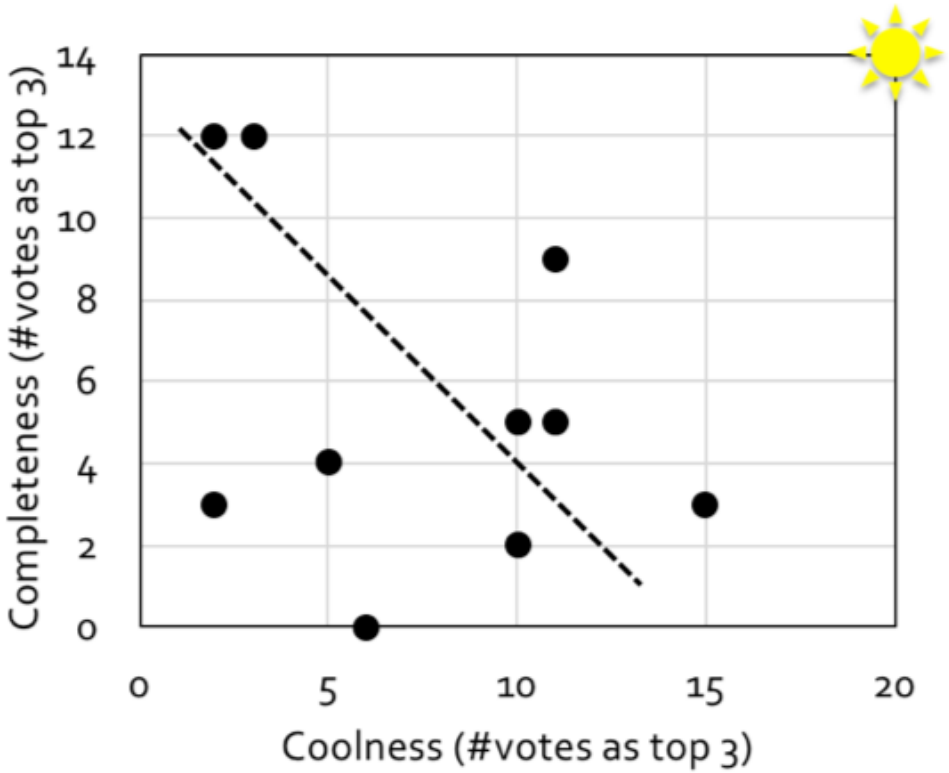

Project Proposal (Feb 6)
Chapter 8: Classification
Decision Tree (Feb 1)
Random Forest (Feb 15)
kNN (Feb 8)
Naïve Bayes (Feb 8)
Bayesian Networks (Feb 8)
Evaluation (Feb 13)
Ensemble methods (Feb 15)
Support Vector Machine (Feb 20)
Artificial Neural Networks (Feb 22)
Course review 1 and HW1/HW2 feedback (Feb 27)
Mid-term exam (March 1):
Introduction, Data Preprocessing, Classification


Project Proposal (Feb 6)
Chapter 8: Classification
Decision Tree (Feb 1)
Random Forest (Feb 15)
kNN (Feb 8)
Naïve Bayes (Feb 8)
Bayesian Networks (Feb 8)
Evaluation (Feb 13)
Ensemble methods (Feb 15)
Support Vector Machine (Feb 20)
Artificial Neural Networks (Feb 22)
Course review 1 and HW1/HW2 feedback (Feb 27)
Mid-term exam (March 1):
Introduction, Data Preprocessing, Classification

Correlation Coefficient ρ =-0.34

- Two important features of a successful project
  - **Completeness**: Problem, Dataset, Method, Evaluation, etc.
  - **Coolness**!



Return me your proposal-commenting handout after class ☺

You will find your proposal grade {10,9,8,7,0} and comments from other teams this weekend on Sakai.

Collaborate in your team

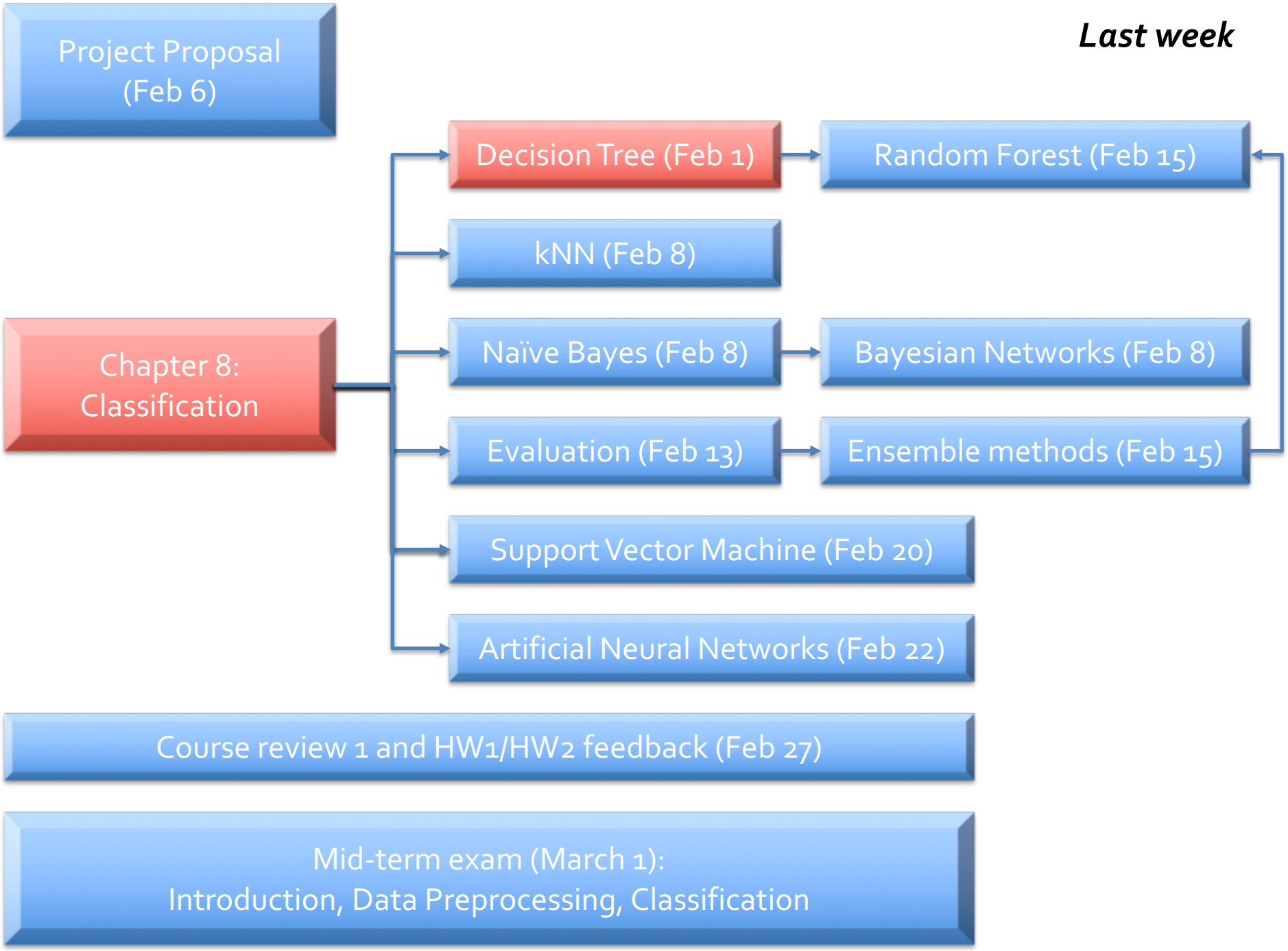
- Leadership!
- Responsibility!
- Loyalty!

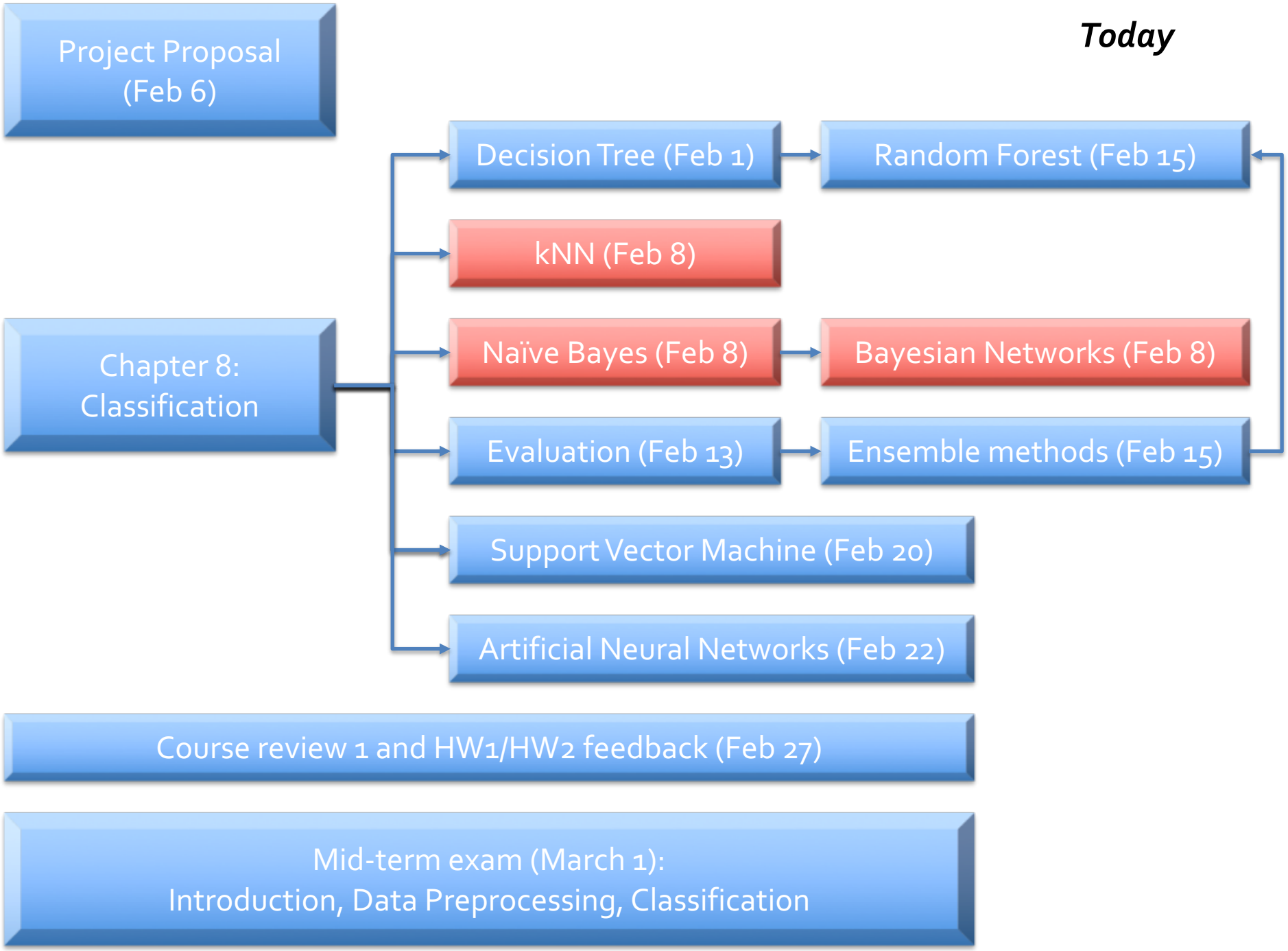
Collaborate across teams

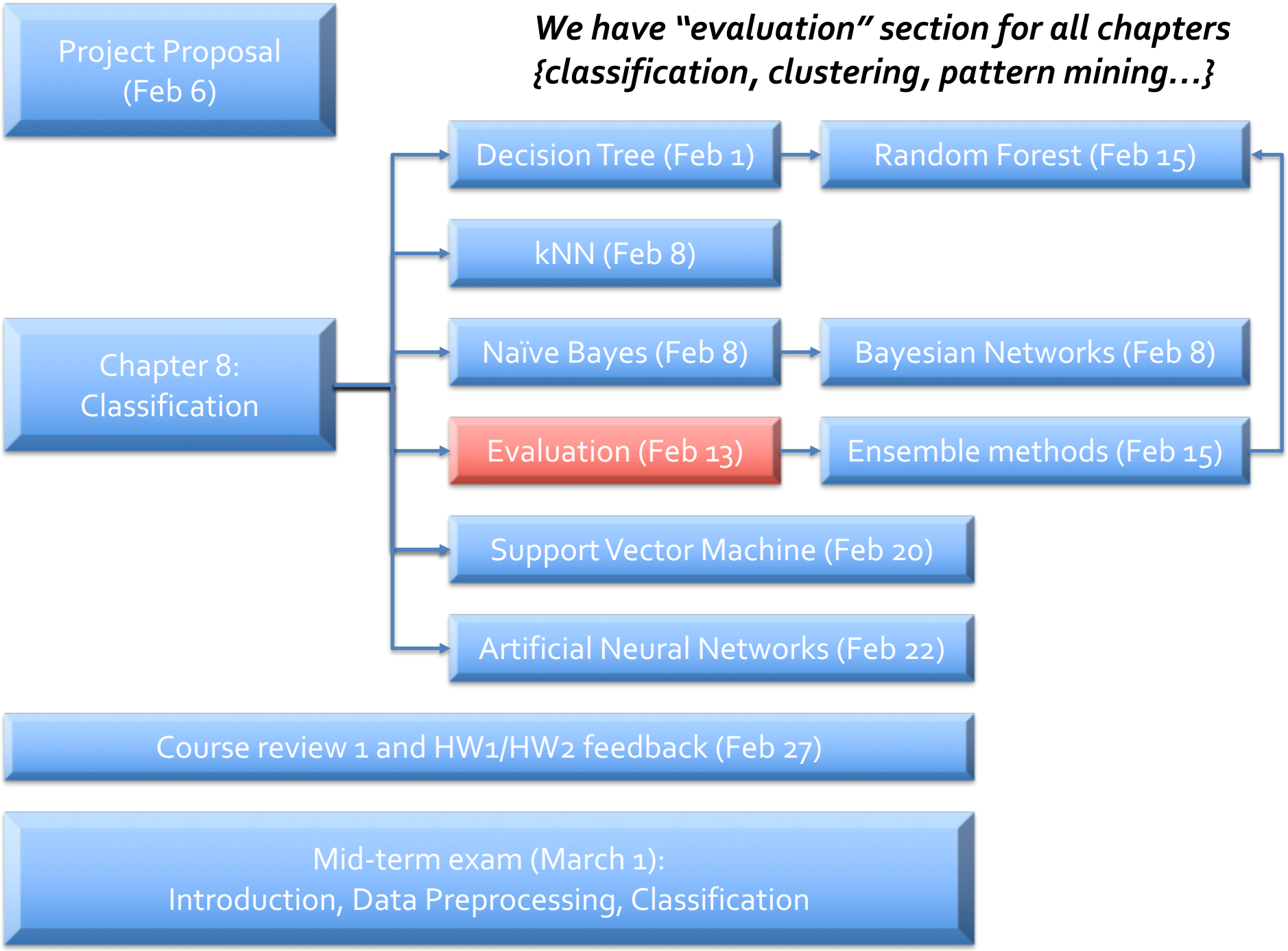
- Similar topics?
- Data sharing?
- Interesting tools?

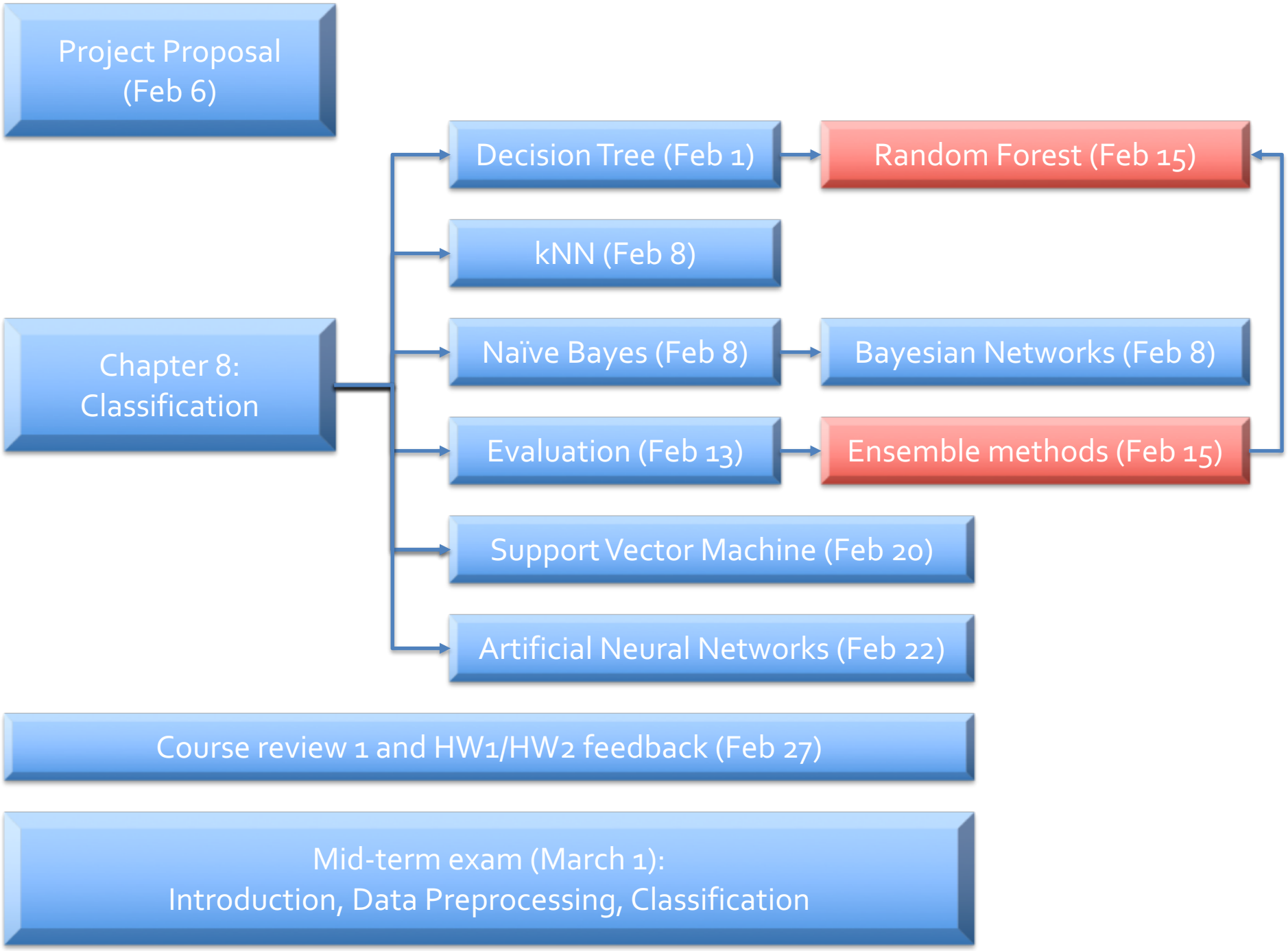
Do it early!

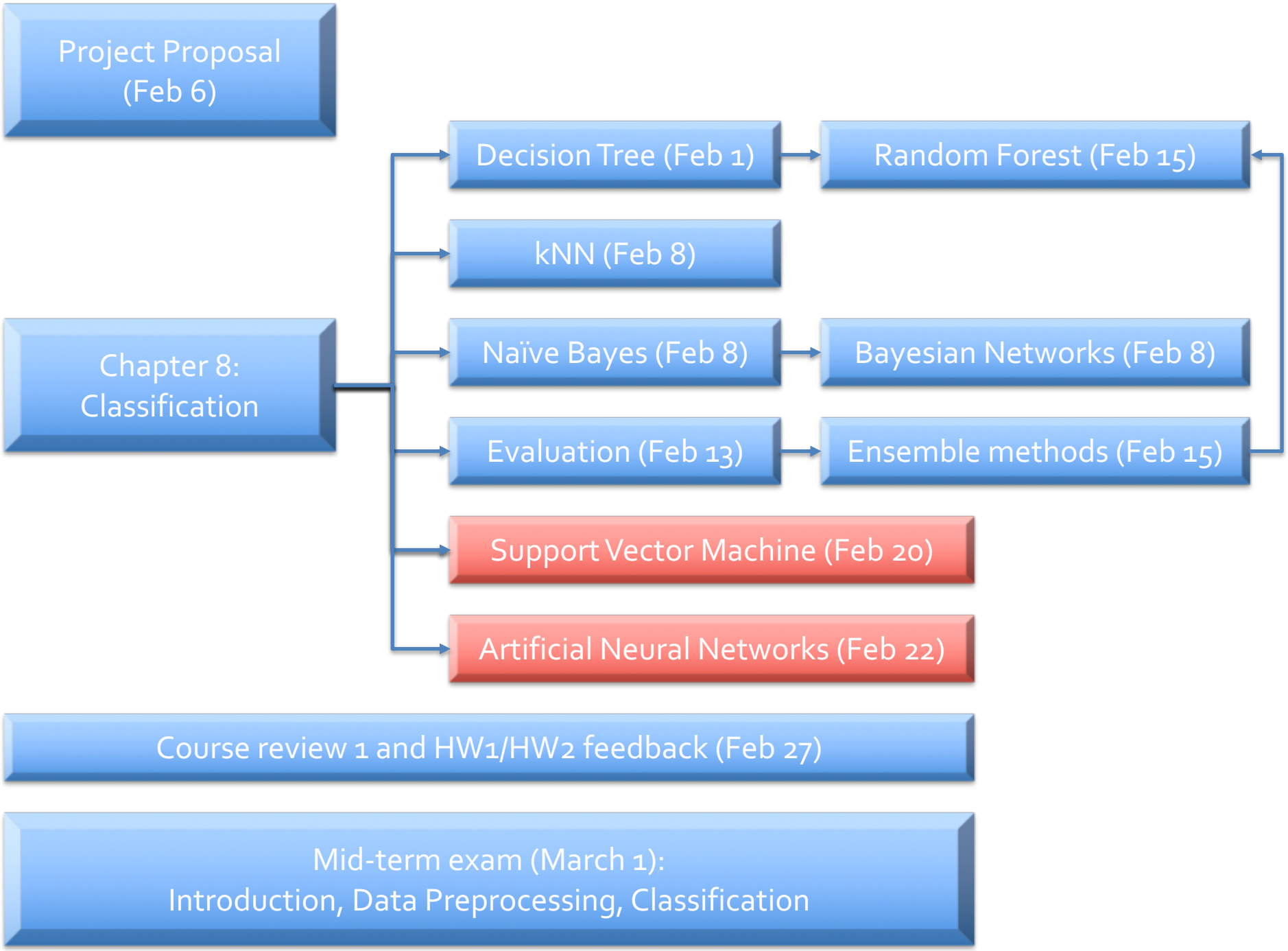
- Project: 30% of the course


Last week
Project Proposal (Feb 6)
Chapter 8: Classification
Decision Tree (Feb 1)
Random Forest (Feb 15)
kNN (Feb 8)
Naïve Bayes (Feb 8)
Bayesian Networks (Feb 8)
Evaluation (Feb 13)
Ensemble methods (Feb 15)
Support Vector Machine (Feb 20)
Artificial Neural Networks (Feb 22)
Course review 1 and HW1/HW2 feedback (Feb 27)
Mid-term exam (March 1):
Introduction, Data Preprocessing, Classification


Today
Project Proposal (Feb 6)
Chapter 8: Classification
Decision Tree (Feb 1)
Random Forest (Feb 15)
kNN (Feb 8)
Naïve Bayes (Feb 8)
Bayesian Networks (Feb 8)
Evaluation (Feb 13)
Ensemble methods (Feb 15)
Support Vector Machine (Feb 20)
Artificial Neural Networks (Feb 22)
Course review 1 and HW1/HW2 feedback (Feb 27)
Mid-term exam (March 1):
Introduction, Data Preprocessing, Classification


Project Proposal (Feb 6)
We have "evaluation" section for all chapters {classification, clustering, pattern mining…}
Chapter 8: Classification
Decision Tree (Feb 1)
Random Forest (Feb 15)
kNN (Feb 8)
Naïve Bayes (Feb 8)
Bayesian Networks (Feb 8)
Evaluation (Feb 13)
Ensemble methods (Feb 15)
Support Vector Machine (Feb 20)
Artificial Neural Networks (Feb 22)
Course review 1 and HW1/HW2 feedback (Feb 27)
Mid-term exam (March 1):
Introduction, Data Preprocessing, Classification


Project Proposal (Feb 6)
Chapter 8: Classification
Decision Tree (Feb 1)
Random Forest (Feb 15)
kNN (Feb 8)
Naïve Bayes (Feb 8)
Bayesian Networks (Feb 8)
Evaluation (Feb 13)
Ensemble methods (Feb 15)
Support Vector Machine (Feb 20)
Artificial Neural Networks (Feb 22)
Course review 1 and HW1/HW2 feedback (Feb 27)
Mid-term exam (March 1):
Introduction, Data Preprocessing, Classification


Project Proposal (Feb 6)
Chapter 8: Classification
Decision Tree (Feb 1)
Random Forest (Feb 15)
kNN (Feb 8)
Naïve Bayes (Feb 8)
Bayesian Networks (Feb 8)
Evaluation (Feb 13)
Ensemble methods (Feb 15)
Support Vector Machine (Feb 20)
Artificial Neural Networks (Feb 22)
Course review 1 and HW1/HW2 feedback (Feb 27)
Mid-term exam (March 1):
Introduction, Data Preprocessing, Classification

You should be able to describe and use …

Project Proposal (Feb 6)

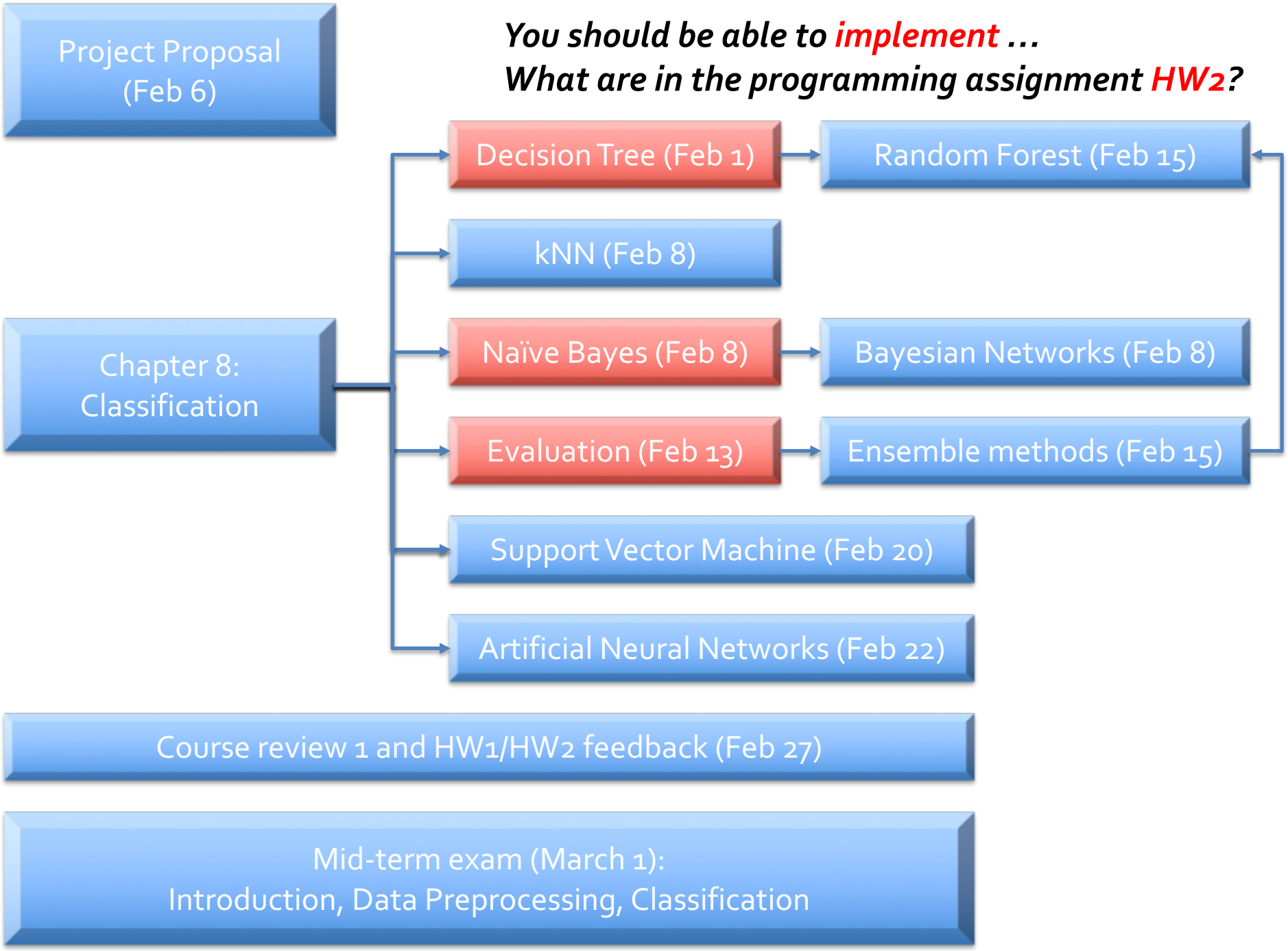
Chapter 8: Classification

- Decision Tree (Feb 1) → Random Forest (Feb 15)
- kNN (Feb 8)
- Naïve Bayes (Feb 8) → Bayesian Networks (Feb 8)
- Evaluation (Feb 13) → Ensemble methods (Feb 15)
- Support Vector Machine (Feb 20)
- Artificial Neural Networks (Feb 22)

Course review 1 and HW1/HW2 feedback (Feb 27)

Mid-term exam (March 1):
Introduction, Data Preprocessing, Classification

You need a calculator!!!


Project Proposal (Feb 6)
You should be able to implement …
What are in the programming assignment HW2?
Chapter 8: Classification
Decision Tree (Feb 1)
Random Forest (Feb 15)
kNN (Feb 8)
Naïve Bayes (Feb 8)
Bayesian Networks (Feb 8)
Evaluation (Feb 13)
Ensemble methods (Feb 15)
Support Vector Machine (Feb 20)
Artificial Neural Networks (Feb 22)
Course review 1 and HW1/HW2 feedback (Feb 27)
Mid-term exam (March 1):
Introduction, Data Preprocessing, Classification

ID	Date	Opponent	Is Home or Away?	Is Opponent in AP Top 25 at Preseason?	Media	Label: Win/Lose
1	9/5/15	Texas	Home	Out	1-NBC	Win
2	9/12/15	Virginia	Away	Out	4-ABC	Win
3	9/19/15	Georgia Tech	Home	In	1-NBC	Win
4	9/26/15	UMass	Home	Out	1-NBC	Win
5	10/3/15	Clemson	Away	In	4-ABC	Lose
6	10/10/15	Navy	Home	Out	1-NBC	Win
7	10/17/15	USC	Home	In	1-NBC	Win
8	10/31/15	Temple	Away	Out	4-ABC	Win
9	11/7/15	PITT	Away	Out	4-ABC	Win
10	11/14/15	Wake Forest	Home	Out	1-NBC	Win
11	11/21/15	Boston College	Away	Out	1-NBC	Win
12	11/28/15	Stanford	Away	In	3-FOX	Lose
13	9/4/16	Texas	Away	Out	4-ABC	Lose
14	9/10/16	Nevada	Home	Out	1-NBC	Win
15	9/17/16	Michigan State	Home	Out	1-NBC	Lose
16	9/24/16	Duke	Home	Out	1-NBC	Lose
17	10/1/16	Syracuse	Home	Out	2-ESPN	Win
18	10/8/16	North Carolina State	Away	Out	4-ABC	Lose
19	10/15/16	Stanford	Home	In	1-NBC	Lose
20	10/29/16	Miami Florida	Home	Out	1-NBC	Win
21	11/5/16	Navy	Home	Out	5-CBS	Lose
22	11/12/16	Army	Home	Out	1-NBC	Win
23	11/19/16	Virginia Tech	Home	In	1-NBC	Lose
24	11/26/16	USC	Away	In	4-ABC	Lose
25	9/2/17	Temple	Home	Out	1-NBC	Win
26	9/9/17	Georgia	Home	In	1-NBC	Lose
27	9/16/17	Boston College	Away	Out	2-ESPN	Win
28	9/23/17	Michigan State	Away	Out	3-FOX	Win
29	9/30/17	Miami Ohio	Home	Out	1-NBC	Win
30	10/7/17	North Carolina	Away	Out	4-ABC	Win
31	10/21/17	USC	Home	In	1-NBC	Win
32	10/28/17	North Carolina State	Home	Out	1-NBC	Win
33	11/4/17	Wake Forest	Home	Out	1-NBC	Win
34	11/11/17	Miami Florida	Away	In	4-ABC	Lose
35	11/18/17	Navy	Home	Out	1-NBC	Win
36	11/25/17	Stanford	Away	In	4-ABC	Lose

Dataset given in HW2

Notre Dame football games:
Three categorical features
→ binary label

Season 15, 16 for **training**
Season 17 for **testing and evaluation**

[25'] Question 1: ID3 model, a decision tree model using "Information Gain"

(1) Programming: Use **ID3** to construct a deci
Use the tree to predict labels of instances i
attributes. Calculate Accuracy, Precision,
(2) Please submit your code as **YourNetid-H**
tree (either hand- or electronically drawn)
testing games as well as evaluation result i

You are not allowed to use decision tree /naïve bayes Python packages.

You can **either** implement a general model – you can use it for your project **or** implement some scratches for this particular dataset (if your coding ability is not strong).

[25'] Question 2: C4.5 model, a decision tree n

(1) Programming: Use **C4.5** to construct a decision tree based on the training set (24 games). Use the tree to predict labels of instances in the testing set (12 games) based on their attributes. Calculate Accuracy, Precision, Recall, and F1 score on the testing result.
(2) Please submit your code as **YourNetid-HW2-Q2-py**. Attach a figure of your decision tree (either hand- or electronically drawn) and write down prediction label of the 12 testing games as well as evaluation result in the PDF.

[35'] Question 3: Naïve Bayes model

(1) Programming: Use **Naïve Bayes** to predict labels of instances in the testing set (12 games) based on the training set (24 games). Calculate Accuracy, Precision, Recall, and F1 score on the testing result.
(2) Please submit your code as **YourNetid-HW2-Q3-py**. Attach a figure of your decision tree (either hand- or electronically drawn) and write down prediction label of the 12 testing games as well as evaluation result in the PDF.

[15'] For this Notre Dame game prediction task, which model is the best, which model performs the worst? Can you explain why? Write down in the PDF.

[25'] Question 1: ID3 model, a decision tree model using "Information Gain"

(1) Programming: Use **ID3** to construct a decision tree based on the training set (24 games). Use the tree to predict labels of instances in the testing set (12 games) based on their attributes. Calculate Accuracy, Precision, Recall, and F1 score on the testing result.
(2) Please submit your code as **YourNetid-HW2-Q1-py**. Attach a figure of your decision tree (either hand- or electronically drawn) and write down prediction label of the 12 testing games as well as evaluation result in the PDF.

Evaluate your model!

[25'] Question 2: C4.5 model, a decision tree model using "Gain Ratio"

(1) Programming: Use **C4.5** to construct a decision tree based on the training set (24 games). Use the tree to predict labels of instances in the testing set (12 games) based on their attributes. Calculate Accuracy, Precision, Recall, and F1 score on the testing result.
(2) Please submit your code as **YourNetid-HW2-Q2-py**. Attach a figure of your decision tree (either hand- or electronically drawn) and write down prediction label of the 12 testing games as well as evaluation result in the PDF.

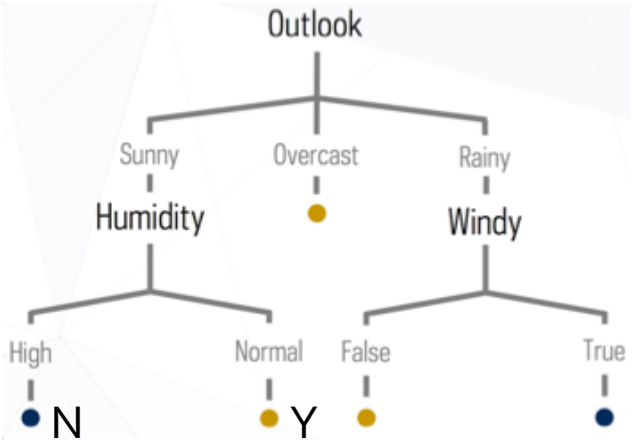
[35'] Question 3: Naïve Bayes model

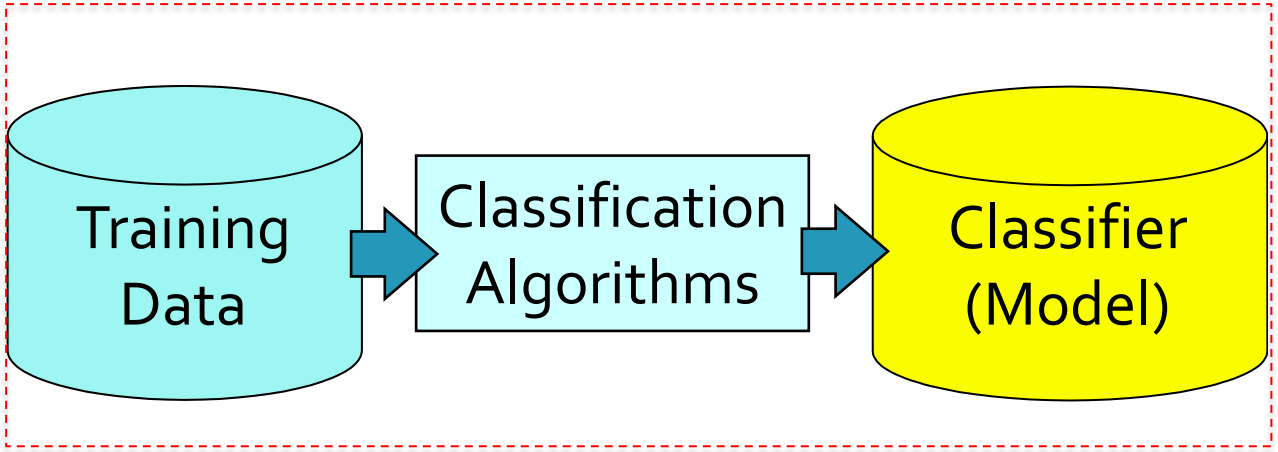
(1) Programming: Use **Naïve Bayes** to predict labels of instances in the testing set (12 games) based on the training set (24 games). Calculate Accuracy, Precision, Recall, and F1 score on the testing result.
(2) Please submit your code as **YourNetid-HW2-Q3-py**. Attach a figure of your decision tree (either hand- or electronically drawn) and write down prediction label of the 12 testing games as well as evaluation result in the PDF.

[15'] For this Notre Dame game prediction task, which model is the best, which model performs the worst? Can you explain why? Write down in the PDF.

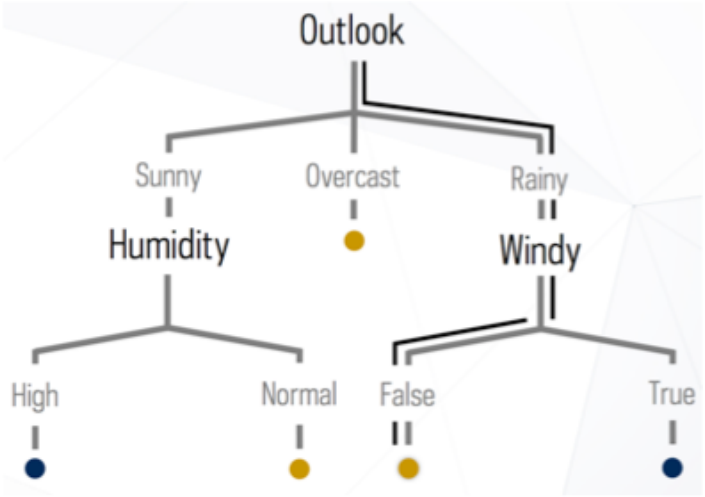
How to Implement ID3/C4.5?

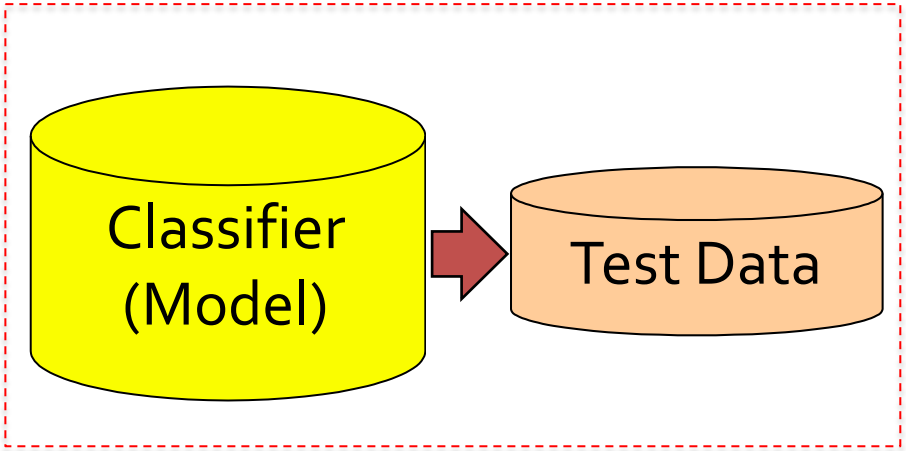
ID	Outlook	Temperature	Humidity	Windy	Label: Play?
1	Sunny	Hot	High	"False"	No
2	Sunny	Hot	High	"True"	No
3	Overcast	Hot	High	"False"	Yes
4	Rainy	Mild	High	"False"	Yes
5	Rainy	Cool	Normal	"False"	Yes
6	Rainy	Cool	Normal	"True"	No
7	Overcast	Cool	Normal	"True"	Yes
8	Sunny	Mild	High	"False"	No
9	Sunny	Cool	Normal	"False"	Yes
10	Rainy	Mild	Normal	"False"	Yes
11	Sunny	Mild	Normal	"True"	Yes
12	Overcast	Mild	High	"True"	Yes
13	Overcast	Hot	Normal	"False"	Yes
14	Rainy	Mild	High	"True"	No





How to Apply ID3/C4.5?





ID	Outlook	Temperature	Humidity	Windy	Label: Play?
15	Rainy	Hot	High	"False"	?

Questions from Lecture 6 (Decision Tree)

- Q1: Why do we have "log" in InformationGain(.) and SplitInfo(.)?

Questions from Lecture 6 (Decision Tree)

- Q1: Why do we have "log" in InformationGain(.) and SplitInfo(.)?

max_X IG(Y|X) = H(Y) – H(Y|X)

$$H(Y) = -\sum_{i=1}^{m} p_i \log(p_i) \quad where\ p_i = P(Y = y_i)$$

Unconditional entropy of class Y

Conditional entropy of class Y given feature X

SplitInfo: Entropy of instances distributed into branches

$$SplitInfo(S, F) = -\sum \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

S: "samples"
F: feature

Questions from Lecture 6 (Decision Tree)

- Q2: How to implement Decision Trees in Python? What are the statements?

Questions from Lecture 6 (Decision Tree)

- Q2: How to implement Decision Trees in Python? What are the statements?

Sorry… this is HW2 programming assignment.

But I can put Wiki's ID3 pseudo code here:

```
ID3 (Examples, Target_Attribute, Attributes)
    Create a root node for the tree
    If all examples are positive, Return the single-node tree Root, with label = +.
    If all examples are negative, Return the single-node tree Root, with label = -.
    If number of predicting attributes is empty, then Return the single node tree Root,
    with label = most common value of the target attribute in the examples.
    Otherwise Begin
        A ← The Attribute that best classifies examples.
        Decision Tree attribute for Root = A.
        For each possible value, v_i, of A,
            Add a new tree branch below Root, corresponding to the test A = v_i.
            Let Examples(v_i) be the subset of examples that have the value v_i for A
            If Examples(v_i) is empty
                Then below this new branch add a leaf node with label = most common target value in the examples
            Else below this new branch add the subtree ID3 (Examples(v_i), Target_Attribute, Attributes – {A})
    End
    Return Root
```

https://en.wikipedia.org/wiki/ID3_algorithm

Questions from Lecture 6 (Decision Tree)

- Q3: How to use Decision Trees if the label is not categorical, or if the feature is not categorical?

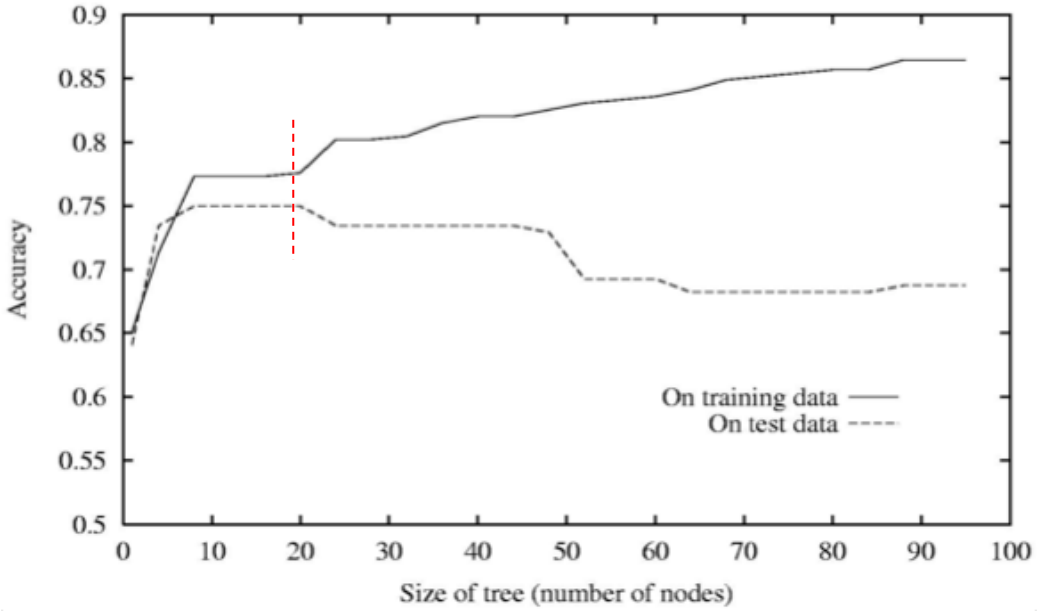
Questions from Lecture 6 (Decision Tree)

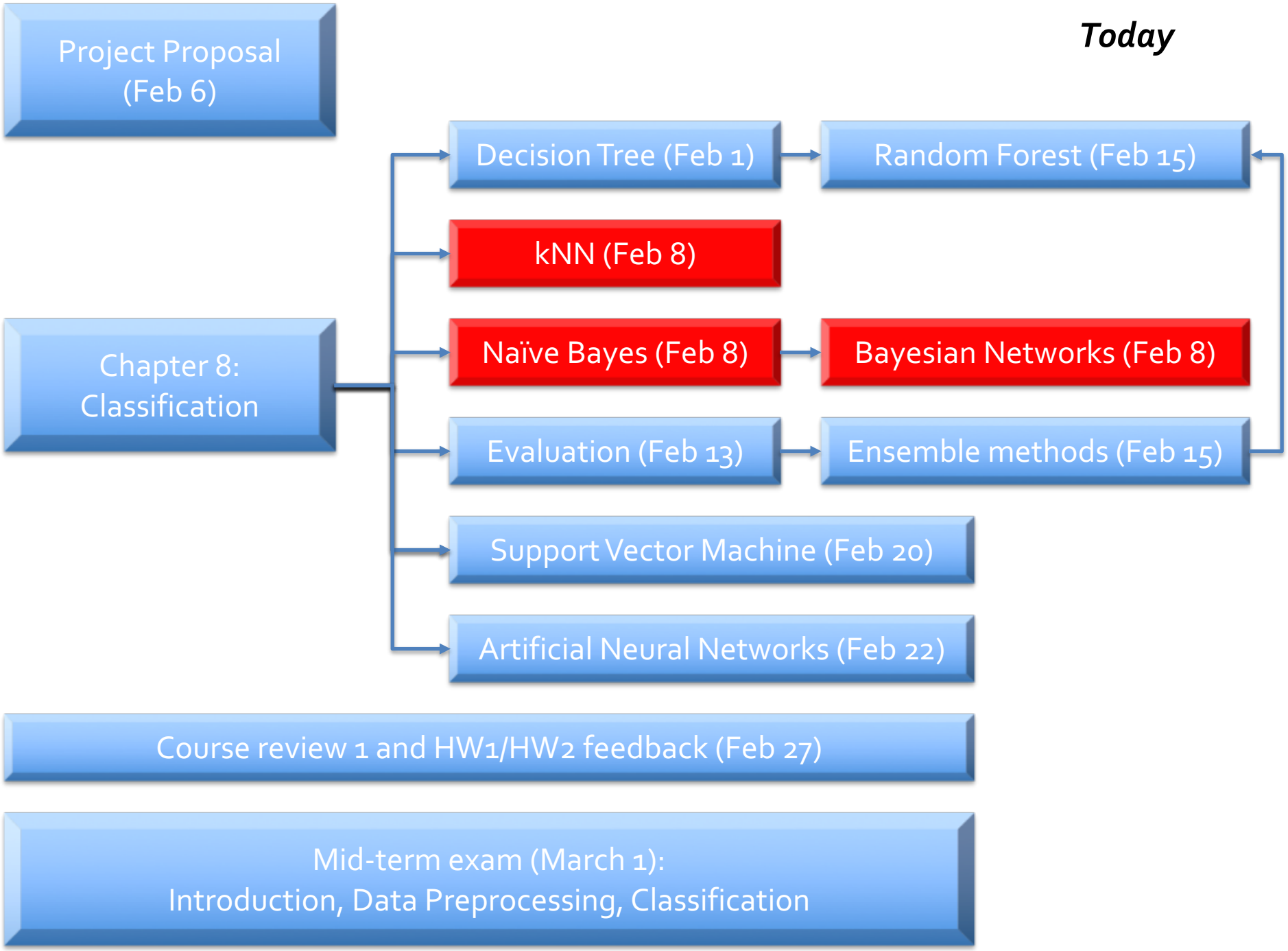
- Q3: How to use Decision Trees if the label is not categorical, or if the feature is not categorical?

		Label	
		Categorical (Classification)	**Numerical (Regression)**
Feature	**Categorical**	• ID3 • C4.5 • CART • KNN • Naïve Bayes • Bayesian Network • …	Feature transformation (to below) Label transformation (to left)
	Numerical	• **CART** • KNN • SVM • Neural Networks • …	• Linear/Nonlinear Regression • Support Vector Regression • …

Questions from Lecture 6 (Decision Tree)

- Q4: Overfitting?



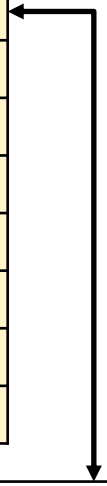

Today
Project Proposal (Feb 6)
Chapter 8: Classification
Decision Tree (Feb 1)
Random Forest (Feb 15)
kNN (Feb 8)
Naïve Bayes (Feb 8)
Bayesian Networks (Feb 8)
Evaluation (Feb 13)
Ensemble methods (Feb 15)
Support Vector Machine (Feb 20)
Artificial Neural Networks (Feb 22)
Course review 1 and HW1/HW2 feedback (Feb 27)
Mid-term exam (March 1):
Introduction, Data Preprocessing, Classification

Today: KNN, Naïve Bayes, and Bayesian Networks

- Describe Nearest Neighbor Classifier
- Implement kNN algorithm
- Describe Bayesian learning
- Implement Naïve Bayes algorithm
- Describe Bayesian network models

An intuitive idea: What's the label of the most similar training instance?

ID	Outlook	Temperature	Humidity	Windy	Label: Play?
1	Sunny	Hot	High	"False"	No
2	Sunny	Hot	High	"True"	No
3	Overcast	Hot	High	"False"	Yes
4	Rainy	Mild	High	"False"	Yes
5	Rainy	Cool	Normal	"False"	Yes
6	Rainy	Cool	Normal	"True"	No
7	Overcast	Cool	Normal	"True"	Yes
8	Sunny	Mild	High	"False"	No
9	Sunny	Cool	Normal	"False"	Yes
10	Rainy	Mild	Normal	"False"	Yes
11	Sunny	Mild	Normal	"True"	Yes
12	Overcast	Mild	High	"True"	Yes
13	Overcast	Hot	Normal	"False"	Yes
14	Rainy	Mild	High	"True"	No

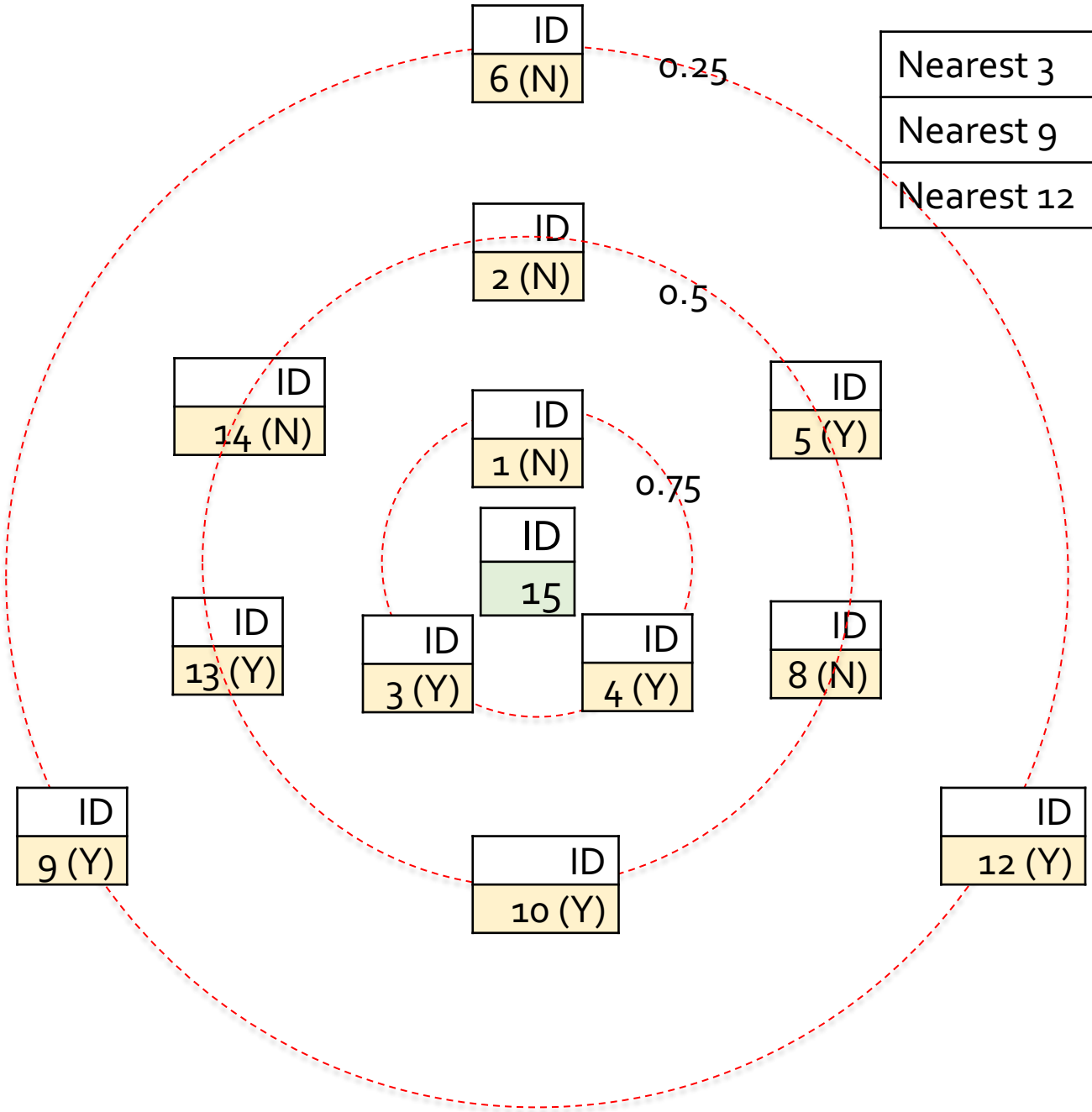


ID	Outlook	Temperature	Humidity	Windy	Label: Play?
15	Rainy	Hot	High	"False"	?

An intuitive idea: What's the label of the most similar training instance?

ID	Outlook	Temperature	Humidity	Windy	Label: Play?	Similarity
1	Sunny	Hot	High	"False"	No	0.75
2	Sunny	Hot	High	"True"	No	0.5
3	Overcast	Hot	High	"False"	Yes	0.75
4	Rainy	Mild	High	"False"	Yes	0.75
5	Rainy	Cool	Normal	"False"	Yes	0.5
6	Rainy	Cool	Normal	"True"	No	0.25
7	Overcast	Cool	Normal	"True"	Yes	0
8	Sunny	Mild	High	"False"	No	0.5
9	Sunny	Cool	Normal	"False"	Yes	0.25
10	Rainy	Mild	Normal	"False"	Yes	0.5
11	Sunny	Mild	Normal	"True"	Yes	0
12	Overcast	Mild	High	"True"	Yes	0.25
13	Overcast	Hot	Normal	"False"	Yes	0.5
14	Rainy	Mild	High	"True"	No	0.5

ID	Outlook	Temperature	Humidity	Windy	Label: Play?
15	Rainy	Hot	High	"False"	?



Nearest 3	(>=0.75)	2Y 1N	**Y**
Nearest 9	(>=0.5)	5Y 4N	**Y**
Nearest 12	(>=0.25)	7Y 5N	**Y**

Nearest Neighbor Classifier

- The Idea:

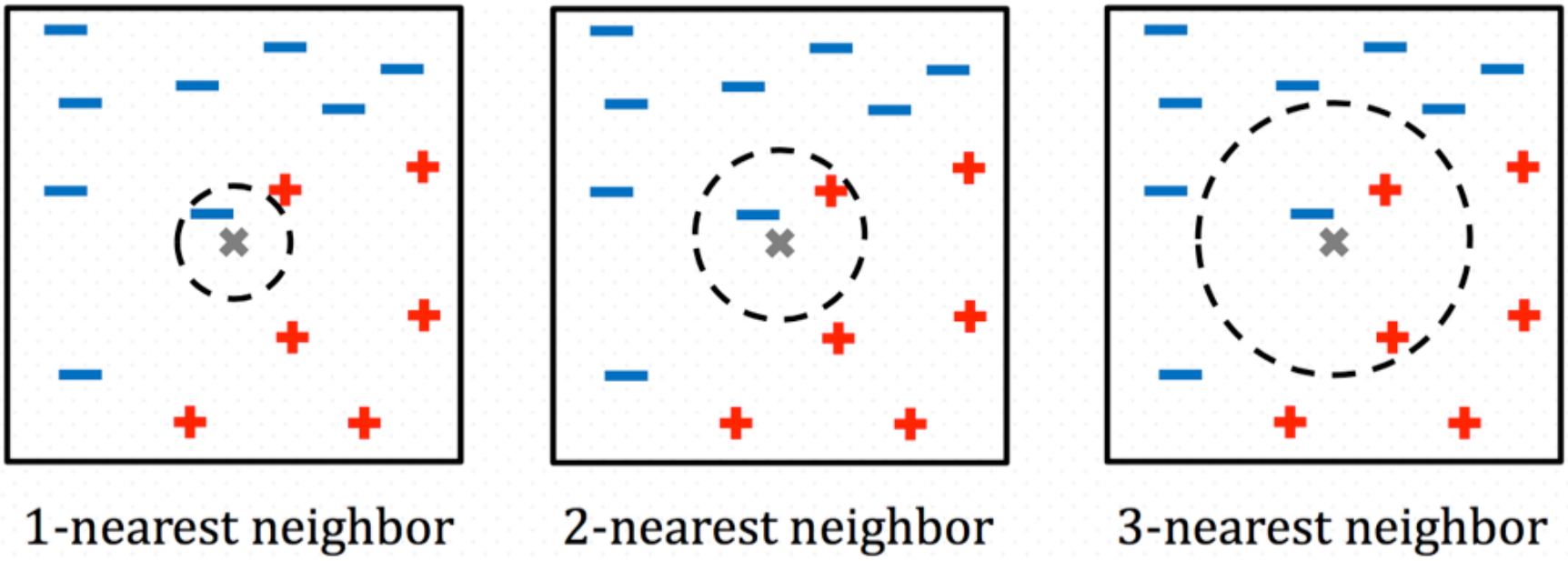
Find which training data is closest to the test instance and classify the test instance as that class.

k-Nearest Neighbor (kNN) Classifier

- The Idea:

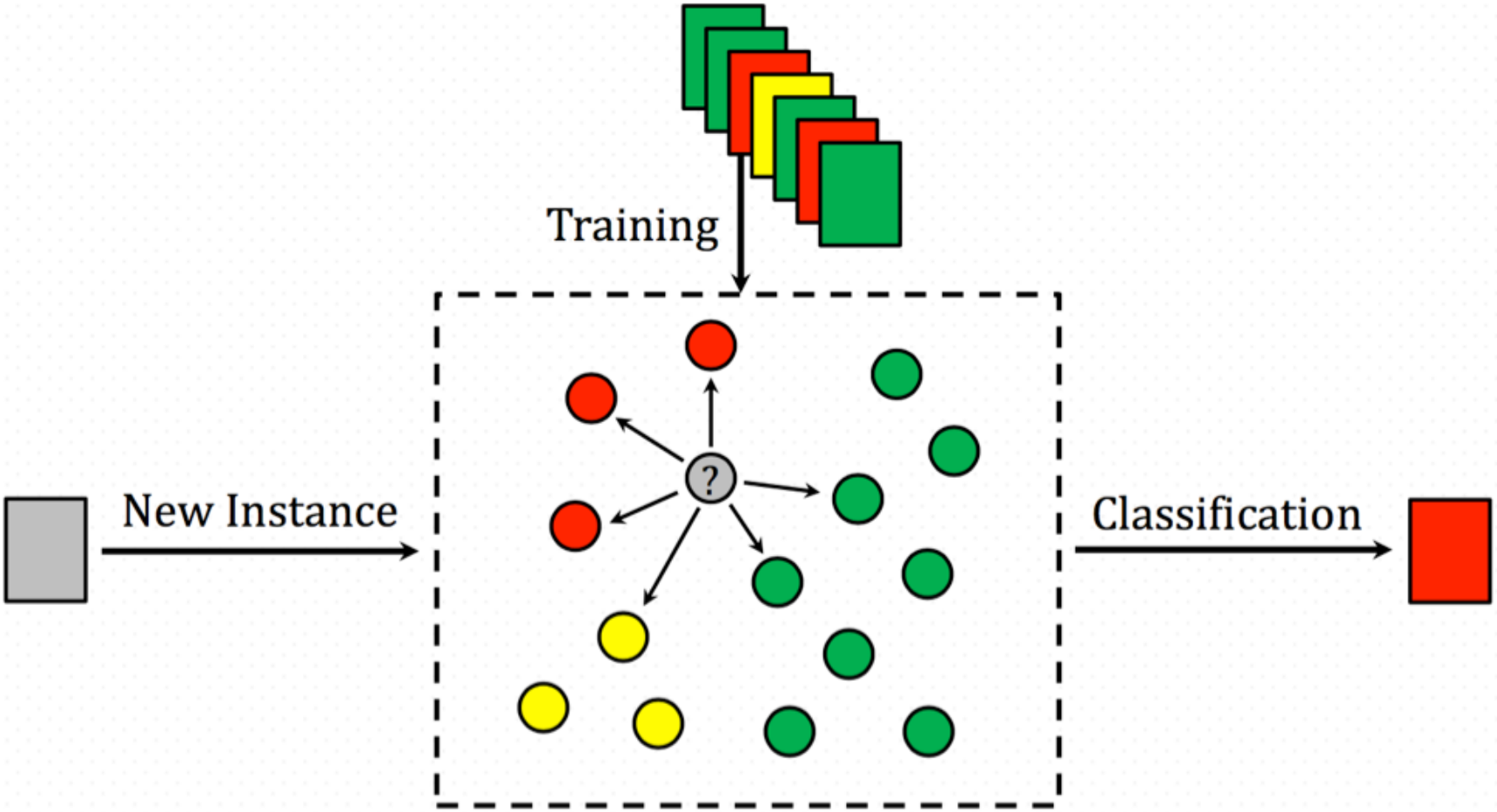
Find the k training instances that are closest to the test instance, and classify the test instance as the majority class of the k nearest training instances.

Nearest Neighbors



The k nearest neighbors of an example x are the data points that have the k smallest distances to x.

k-Nearest Neighbor (kNN) Classifier



The kNN Algorithm

The k-nearest neighbor classification algorithm.

1: Let k be the number of nearest neighbors and D be the set of training examples.

2: **for** each test example $z = (\mathbf{x}', y')$ **do**

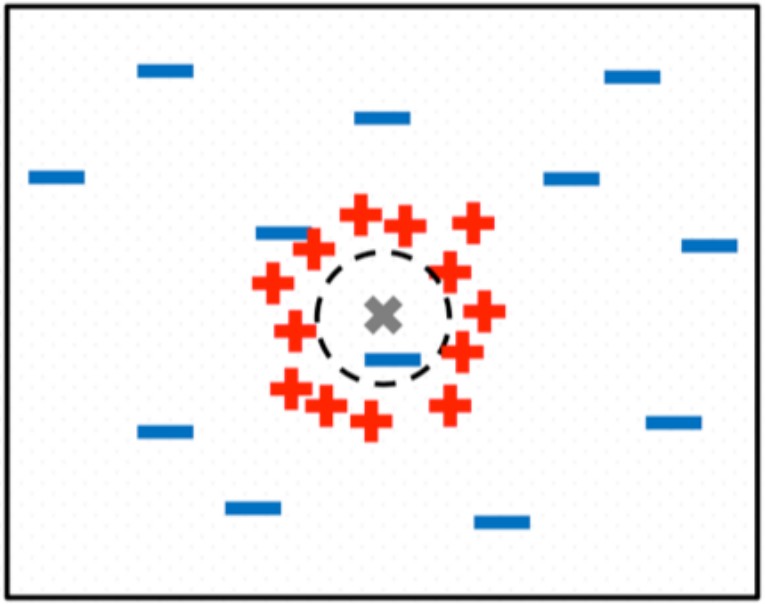
3: Compute $d(\mathbf{x}', \mathbf{x})$, the distance between z and every example, $(\mathbf{x}, y) \in D$.

4: Select $D_z \subseteq D$, the set of k closest training examples to z.

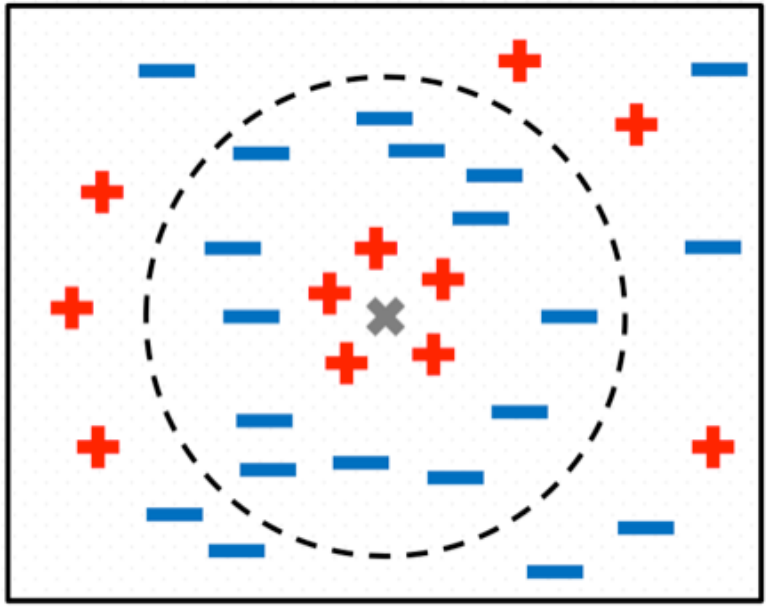
5: $y' = \underset{v}{\operatorname{argmax}} \sum_{(x_i, y_i) \in D_z} I(v = y_i)$

6: **end for**

Choosing the k for kNN



k-nearest neighbor classification with small k.



k-nearest neighbor classification with big k.

Choosing the k for kNN

- If k is too small, then the nearest-neighbor classifier may be susceptible to overfitting because of noise in the training data.
- If k is too large, the nearest-neighbor classifier may misclassify the test instance because its list of nearest neighbors may include data points that are located far away from its neighborhood.

kNN Classification

- Take the majority vote of class labels among the k-nearest neighbors:

$$\text{Majority voting:} \quad y = \underset{v}{\operatorname{argmax}} \sum_{(\mathbf{x}_i, y_i) \in D_z} I(v = y_i)$$

where v is a class label, y_i is the class label for one of the nearest neighbors, and $I(\cdot)$ is an indicator function that returns the value 1 if its argument is true and 0 otherwise.

kNN Regression

- Predict the average value of the class value of the k-nearest neighbors.

$$\text{Average value:} \qquad y = \frac{1}{(\mathbf{x}_i, y_i) \in D_z} \sum_{(\mathbf{x}_i, y_i) \in D_z} y_i$$

where v is a class label and y_i is the class label for one of the nearest neighbors.

kNN Distance-Weighted Classification

- In the majority voting approach, every neighbor has the same impact on the classification.
- The influence of each nearest neighbor x_i can be weighted according to distance: $w_i = 1 / d(\mathbf{x}', \mathbf{x}_i)^2$.
- Then the class label can be determined as follows:

$$y = \underset{v}{\operatorname{argmax}} \sum_{(\mathbf{x}_i, y_i) \in D_z} w_i \times I(v = y_i)$$

kNN Classifier

- Advantages:
  - Generate their predictions based on local information, so it can produce **arbitrarily shaped decision boundaries**.
  - **Very efficient in model induction** (training)
    - Only store the training data.
    - Note that 1NN and kNN are equally efficient
      - Retrieving the k nearest neighbors is (almost) no more expensive than retrieving a single nearest neighbor
      - k nearest neighbors can be maintained in a queue.
- Disadvantages:
  - Can easily produce **wrong predictions** without appropriate data preprocessing.
  - **Not particularly efficient** in testing
    - Computation of distance measure to every training instance

An NLP Task: Entity Name Typing

- s_method = 'method algorithm model approach framework process scheme implementation procedure strategy architecture'
- s_problem = 'problem technique process system application task evaluation tool paradigm benchmark software'
- s_dataset = 'data dataset database'
- s_metric = 'value score measure metric function parameter'
- types = ['METHOD','PROBLEM','DATASET','METRIC']

Entity name	Count	1NN	2NN	3NN
association rules	2391	METHOD MD:9	DATASET MD:25 PM:8 DT:29 MC:8	METHOD MD:88 PM:49 DT:56 MC:6
active learning	2327	METHOD MD:284 PM:108 MC:1	METHOD MD:55 PM:35 DT:13 MC:14	METHOD MD:59 PM:27 DT:28 MC:8
decision tree	2252	METHOD MD:163 PM:3	METHOD MD:98 PM:22 DT:8 MC:2	METHOD MD:62 PM:23 DT:25 MC:10
logistic regression	2091	METHOD MD:292 PM:16 DT:2 MC:13	METHOD MD:47 PM:9 DT:3 MC:5	METHOD MD:69 PM:17 DT:15 MC:10
recommender systems	2085	DATASET MD:1 PM:1 DT:2	PROBLEM MD:16 PM:28 DT:9 MC:5	PROBLEM MD:12 PM:16 DT:16 MC:2
link prediction	1901	PROBLEM MD:78 PM:330 MC:32	PROBLEM MD:41 PM:54 DT:2 MC:6	PROBLEM MD:25 PM:40 DT:22 MC:9
frequent itemsets	1612	METHOD MD:2 PM:1	DATASET MD:10 PM:3 DT:18 MC:1	METHOD MD:43 PM:28 DT:39 MC:1
topic models	1589	METHOD MD:2 PM:1	PROBLEM MD:8 PM:16 DT:12 MC:5	METHOD MD:51 PM:24 DT:10 MC:4
standard deviation	1567	METRIC MC:2	METRIC MD:1 DT:1 MC:12	METRIC MD:4 PM:6 DT:16 MC:24
f measure	1415	METRIC MD:3 PM:7 MC:46	METRIC MD:6 PM:3 DT:4 MC:16	METRIC MD:23 PM:18 DT:7 MC:26
support vector machines	1385	METHOD MD:3	METHOD MD:23 PM:11 DT:7 MC:1	METHOD MD:29 PM:17 DT:14 MC:1
decision trees	1358	METHOD MD:5	DATASET MD:12 PM:13 DT:20 MC:1	METHOD MD:36 PM:17 DT:33 MC:3
anomaly detection	1348	METHOD MD:95 PM:56 DT:2 MC:2	METHOD MD:36 PM:16 DT:17 MC:9	DATASET MD:32 PM:18 DT:48 MC:3
matrix factorization	1332	METHOD MD:170 PM:41 MC:1	METHOD MD:42 PM:6 DT:2 MC:5	METHOD MD:32 PM:7 DT:9
information extraction	1332	PROBLEM MD:32 PM:69 DT:3	METHOD MD:23 PM:23 DT:12	METHOD MD:29 PM:11 DT:13
nearest neighbor	1275	METHOD MD:97 PM:17 DT:8 MC:8	METHOD MD:23 PM:5 DT:4 MC:3	METHOD MD:17 PM:3 DT:11 MC:14
classification accuracy	1239	METRIC PM:1 MC:3	METRIC MD:8 PM:9 DT:3 MC:11	METHOD MD:25 PM:8 DT:22 MC:10
text classification	1205	PROBLEM MD:23 PM:59 DT:5 MC:1	METHOD MD:37 PM:27 DT:10	PROBLEM MD:16 PM:28 DT:16 MC:3

Today: KNN, Naïve Bayes, and Bayesian Networks

- Describe Nearest Neighbor Classifier
- Implement KNN algorithm
- **Describe Bayesian learning**
- **Implement Naïve Bayes algorithm**
- Describe Bayesian network models

Bayes' Theorem: Basics

Proof of Bayes Theorem

The probability of two events A and B happening, P(A∩B), is the probability of A, P(A), times the probability of B given that A has occurred, P(B|A).

$$P(A \cap B) = P(A)P(B|A) \tag{1}$$

On the other hand, the probability of A and B is also equal to the probability of B times the probability of A given B.

$$P(A \cap B) = P(B)P(A|B) \tag{2}$$

Equating the two yields:

$$P(B)P(A|B) = P(A)P(B|A) \tag{3}$$

and thus

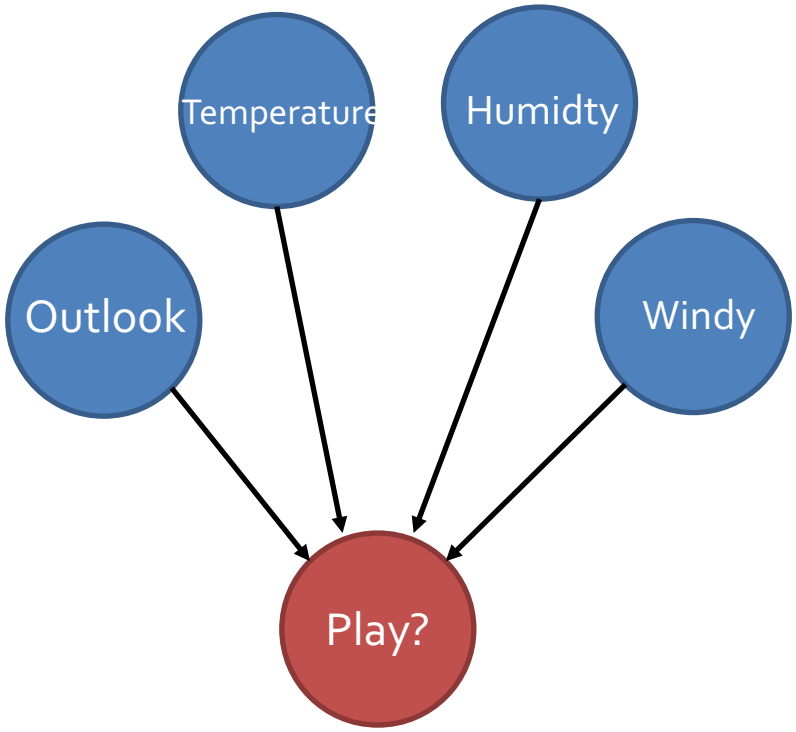
$$P(A|B) = P(A)\frac{P(B|A)}{P(B)} \tag{4}$$

This equation, known as Bayes Theorem is the basis of statistical inference.

Goal: Predict Conditional Probabilities

ID	Outlook	Temperature	Humidity	Windy	Label: Play?
1	Sunny	Hot	High	"False"	No
2	Sunny	Hot	High	"True"	No
3	Overcast	Hot	High	"False"	Yes
4	Rainy	Mild	High	"False"	Yes
5	Rainy	Cool	Normal	"False"	Yes
6	Rainy	Cool	Normal	"True"	No
7	Overcast	Cool	Normal	"True"	Yes
8	Sunny	Mild	High	"False"	No
9	Sunny	Cool	Normal	"False"	Yes
10	Rainy	Mild	Normal	"False"	Yes
11	Sunny	Mild	Normal	"True"	Yes
12	Overcast	Mild	High	"True"	Yes
13	Overcast	Hot	Normal	"False"	Yes
14	Rainy	Mild	High	"True"	No

ID	Outlook	Temperature	Humidity	Windy	Label: Play?
15	Rainy	Hot	High	"False"	?



P(Play? = Yes | Rainy, Hot, High, False)
P(Play? = No | Rainy, Hot, High, False)

Bayesian Learning/Inference

- Bayesian inference is a method of statistical inference in which **Bayes' theorem** is used to update the probability for a hypothesis as more evidence or information becomes available.
  - Let **X** be a data sample: class label is unknown
  - Let H be a *hypothesis* that **X** belongs to class C
  - Classification is to determine P(H|**X**), (i.e., *posteriori probability):* the probability that the hypothesis holds given the observed data sample **X**

$$P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})} = P(\mathbf{X}|H) \times P(H) / P(\mathbf{X})$$

Bayesian Learning/Inference

- Bayes' Theorem:
  - P(H) (*prior probability*): the initial probability
  - P(**X**) (*evidence*): probability that sample data is observed
  - P(**X**|H) (*likelihood*): the probability of observing the sample **X**, given that the hypothesis holds

$$P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})} = P(\mathbf{X}|H) \times P(H) / P(\mathbf{X})$$

Written as Posteriori and Hypothesis

- Given training data **X**, *posteriori probability of a hypothesis* H, P(H|**X**), follows the Bayes' theorem

$$P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})} = P(\mathbf{X}|H) \times P(H) / P(\mathbf{X})$$

- Informally, this can be viewed as

  posteriori = likelihood x prior/evidence

- Predicts **X** belongs to C_i iff the probability $P(C_i|\mathbf{X})$ is the highest among all the $P(C_k|\mathbf{X})$ for all the *k* classes

Derive the Maximum Posteriori

- Let D be a training set of tuples and their associated class labels, and each tuple is represented by an n-D attribute vector **X** = (x_1, x_2, …, x_n)
- Suppose there are *m* classes C_1, C_2, …, C_m.
- Classification is to derive the maximum posteriori, i.e., the maximal $P(C_i|\mathbf{X})$
- This can be derived from Bayes' theorem

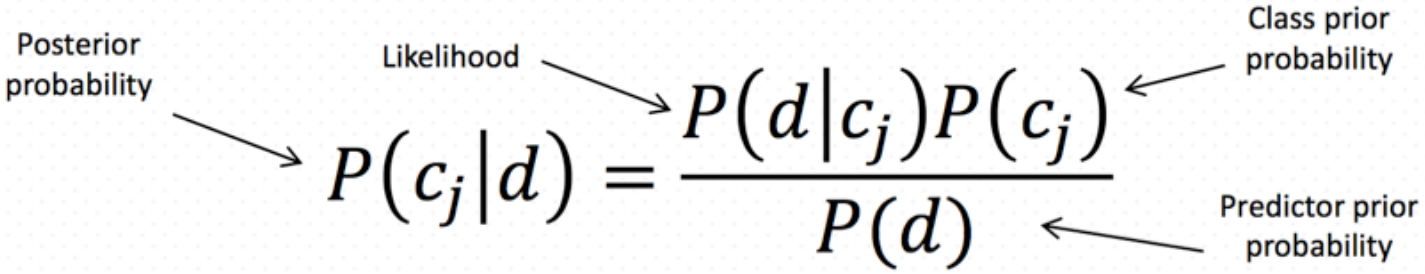
$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

- Since P(**X**) is constant for all classes, only

$$P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$$

needs to be maximized

Written as Instance and Class



$\boldsymbol{P(c_j|d)}$ = probability of instance d being in class c_j

$\boldsymbol{P(d|c_j)}$ = probability of generating instance d given class c_j

$\boldsymbol{P(c_j)}$ = probability of occurrence of class c_j

$\boldsymbol{P(d)}$ = probability of instance d occurring

Naïve Bayes Classifier: Likelihood

- **A simplified (Naïve) assumption:** attributes are conditionally independent (i.e., no dependence relation between attributes):

$$P(\mathbf{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times \ldots \times P(x_n|C_i)$$

- This greatly reduces the computation cost: Only counts the class distribution

Naïve Bayes Classifier: Likelihood

- If A_k is categorical, $P(x_k|C_i)$ is the # of tuples in C_i having value x_k for A_k divided by $|C_{i,D}|$ (# of tuples of C_i in data D).
- If A_k is continuous-valued, $P(x_k|C_i)$ is usually computed based on Gaussian distribution with a mean μ and standard deviation σ

$$g(x,\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

and $P(x_k|C_i)$ is

$$P(\mathbf{X}|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})$$

Quinlan's Example (1986): Playing Tennis

	Outlook	Temperature	Humidity	Windy	Label: Play?
1	Sunny	Hot	High	"False"	No
2	Sunny	Hot	High	"True"	No
3	Overcast	Hot	High	"False"	Yes
4	Rainy	Mild	High	"False"	Yes
5	Rainy	Cool	Normal	"False"	Yes
6	Rainy	Cool	Normal	"True"	No
7	Overcast	Cool	Normal	"True"	Yes
8	Sunny	Mild	High	"False"	No
9	Sunny	Cool	Normal	"False"	Yes
10	Rainy	Mild	Normal	"False"	Yes
11	Sunny	Mild	Normal	"True"	Yes
12	Overcast	Mild	High	"True"	Yes
13	Overcast	Hot	Normal	"False"	Yes
14	Rainy	Mild	High	"True"	No
15	**Rainy**	**Hot**	**High**	**"False"**	**?**

P(H): Prior Probability

$P(C_i)$

- P(Play? = "yes") = 9/14 = 0.643
- P(Play? = "no") = 5/14= 0.357

No
No
Yes
Yes
Yes
No
Yes
No
Yes
Yes
Yes
Yes
Yes
No

P(**X**|H): Likelihood

15	Rainy	Hot	High	"False"	?

- Compute $P(X|C_i)$ for each class

P(Outlook = Rainy | Play? = "yes") = 3/9 = 0.333

P(Outlook = Rainy | Play? = "no") = 2/5 = 0.4

	Outlook	Temperature	Humidity	Windy	Label: Play?
1	Sunny	Hot	High	"False"	No
2	Sunny	Hot	High	"True"	No
3	Overcast	Hot	High	"False"	Yes
4	Rainy	Mild	High	"False"	Yes
5	Rainy	Cool	Normal	"False"	Yes
6	Rainy	Cool	Normal	"True"	No
7	Overcast	Cool	Normal	"True"	Yes
8	Sunny	Mild	High	"False"	No
9	Sunny	Cool	Normal	"False"	Yes
10	Rainy	Mild	Normal	"False"	Yes
11	Sunny	Mild	Normal	"True"	Yes
12	Overcast	Mild	High	"True"	Yes
13	Overcast	Hot	Normal	"False"	Yes
14	Rainy	Mild	High	"True"	No

P(**X**|H): Likelihood

15	Rainy	Hot	High	"False"	?

- Compute $P(X|C_i)$ for each class

P(Temperature = Hot | Play? = "yes) = 2/9 = 0.222

P(Temperature = Hot | Play? = "no") = 2/5 = 0.4

	Outlook	Temperature	Humidity	Windy	Label: Play?
1	Sunny	Hot	High	"False"	No
2	Sunny	Hot	High	"True"	No
3	Overcast	Hot	High	"False"	Yes
4	Rainy	Mild	High	"False"	Yes
5	Rainy	Cool	Normal	"False"	Yes
6	Rainy	Cool	Normal	"True"	No
7	Overcast	Cool	Normal	"True"	Yes
8	Sunny	Mild	High	"False"	No
9	Sunny	Cool	Normal	"False"	Yes
10	Rainy	Mild	Normal	"False"	Yes
11	Sunny	Mild	Normal	"True"	Yes
12	Overcast	Mild	High	"True"	Yes
13	Overcast	Hot	Normal	"False"	Yes
14	Rainy	Mild	High	"True"	No

P(**X**|H): Likelihood

15	**Rainy**	**Hot**	**High**	**"False"**	**?**

- Compute $P(X|C_i)$ for each class

P(Humidity = High | Play? = "yes") = 3/9 = 0.333

P(Humidity = High | Play? = "no") = 4/5 = 0.8

	Outlook	Temperature	Humidity	Windy	Label: Play?
1	Sunny	**Hot**	**High**	**"False"**	No
2	Sunny	**Hot**	**High**	"True"	No
3	Overcast	**Hot**	**High**	**"False"**	Yes
4	**Rainy**	Mild	**High**	**"False"**	Yes
5	**Rainy**	Cool	Normal	**"False"**	Yes
6	**Rainy**	Cool	Normal	"True"	No
7	Overcast	Cool	Normal	"True"	Yes
8	Sunny	Mild	**High**	**"False"**	No
9	Sunny	Cool	Normal	**"False"**	Yes
10	**Rainy**	Mild	Normal	**"False"**	Yes
11	Sunny	Mild	Normal	"True"	Yes
12	Overcast	Mild	**High**	"True"	Yes
13	Overcast	**Hot**	Normal	**"False"**	Yes
14	**Rainy**	Mild	**High**	"True"	No

P(**X**|H): Likelihood

15	Rainy	Hot	High	"False"	?

- Compute $P(X|C_i)$ for each class

P(Windy = "False" | Play? = "yes") = 6/9 = 0.667

P(Windy= "False" | Play? = "no") = 2/5 = 0.4

	Outlook	Temperature	Humidity	Windy	Label: Play?
1	Sunny	Hot	High	"False"	No
2	Sunny	Hot	High	"True"	No
3	Overcast	Hot	High	"False"	Yes
4	Rainy	Mild	High	"False"	Yes
5	Rainy	Cool	Normal	"False"	Yes
6	Rainy	Cool	Normal	"True"	No
7	Overcast	Cool	Normal	"True"	Yes
8	Sunny	Mild	High	"False"	No
9	Sunny	Cool	Normal	"False"	Yes
10	Rainy	Mild	Normal	"False"	Yes
11	Sunny	Mild	Normal	"True"	Yes
12	Overcast	Mild	High	"True"	Yes
13	Overcast	Hot	Normal	"False"	Yes
14	Rainy	Mild	High	"True"	No

P(**X**|H): Likelihood

- Compute $P(X|C_i)$ for each class

P(Outlook = Rainy | Play? = "yes") = 3/9 = 0.333
P(Outlook = Rainy | Play? = "no") = 2/5 = 0.4

P(Temperature = Hot | Play? = "yes) = 2/9 = 0.222
P(Temperature = Hot | Play? = "no") = 2/5 = 0.4

P(Humidity = High | Play? = "yes") = 3/9 = 0.333
P(Humidity = High | Play? = "no") = 4/5 = 0.8

P(Windy = "False" | Play? = "yes") = 6/9 = 0.667
P(Windy= "False" | Play? = "no") = 2/5 = 0.4

P(H|**X**): Posteriori Probability

P(**X**|C_i):

P(**X** | Play? = "yes") = 0.333 x 0.222 x 0.333 x 0.667 = 0.01642

because

P(Outlook = Rainy | Play? = "yes") = 3/9 = 0.333

P(Outlook = Rainy | Play? = "no") = 2/5 = 0.4

P(Temperature = Hot | Play? = "yes) = 2/9 = 0.222

P(Temperature = Hot | Play? = "no") = 2/5 = 0.4

P(Humidity = High | Play? = "yes") = 3/9 = 0.333

P(Humidity = High | Play? = "no") = 4/5 = 0.8

P(Windy = "False" | Play? = "yes") = 6/9 = 0.667

P(Windy= "False" | Play? = "no") = 2/5 = 0.4

P(H|**X**): Posteriori Probability

P(**X**|C_i):

P(**X** | Play? = "no") = 0.4 x 0.4 x 0.8 x 0.4 = 0.0512

because

P(Outlook = Rainy | Play? = "yes") = 3/9 = 0.333

P(Outlook = Rainy | Play? = "no") = 2/5 = 0.4

P(Temperature = Hot | Play? = "yes) = 2/9 = 0.222

P(Temperature = Hot | Play? = "no") = 2/5 = 0.4

P(Humidity = High | Play? = "yes") = 3/9 = 0.333

P(Humidity = High | Play? = "no") = 4/5 = 0.8

P(Windy = "False" | Play? = "yes") = 6/9 = 0.667

P(Windy= "False" | Play? = "no") = 2/5 = 0.4

P(H|**X**): Posteriori Probability

$P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)*P(C_i)/P(\mathbf{X})$:

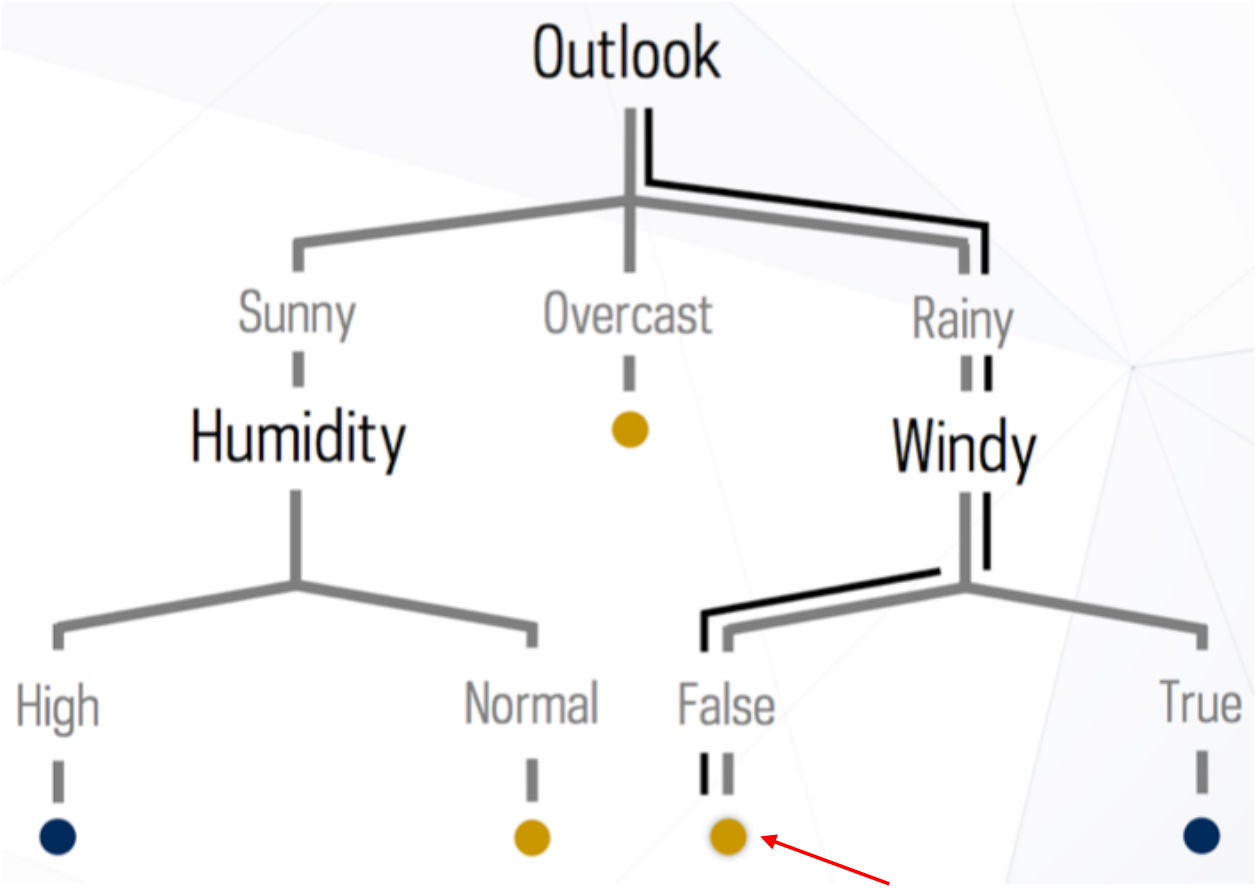
P(Play? = "yes" | **X**) = P(**X** | Play? = "yes") * P(Play? = "yes") / P(**X**)
= 0.01642 x 0.643 / P(**X**) = 0.010 / P(**X**)

P(Play? = "no" | **X**) = P(**X** | Play? = "no") * P(Play? = "no") / P(**X**)
= 0.0512 x 0.357 / P(**X**) = 0.018 / P(**X**)

So, the conclusion is *Play? = "no".*

Call Back: Decision Tree-Prediction

1	Rainy	Hot	High	"False"	?



Quinlan's Example (1986): Playing Tennis

	Outlook	Temperature	Humidity	Windy	Label: Play?
1	Sunny	Hot	High	"False"	No
2	Sunny	Hot	High	"True"	No
3	Overcast	Hot	High	"False"	Yes
4	Rainy	Mild	High	"False"	Yes
5	Rainy	Cool	Normal	"False"	Yes
6	Rainy	Cool	Normal	"True"	No
7	Overcast	Cool	Normal	"True"	Yes
8	Sunny	Mild	High	"False"	No
9	Sunny	Cool	Normal	"False"	Yes
10	Rainy	Mild	Normal	"False"	Yes
11	Sunny	Mild	Normal	"True"	Yes
12	Overcast	Mild	High	"True"	Yes
13	Overcast	Hot	Normal	"False"	Yes
14	Rainy	Mild	High	"True"	No
15	**Rainy**	**Hot**	**High**	**"False"**	**?**

Avoiding the Zero-Probability Problem

- Naïve Bayesian prediction requires each conditional prob. be non-zero. Otherwise, the predicted prob. will be zero.

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)$$

- Ex. Suppose a dataset with 1000 tuples, income=low (0), income= medium (990), and income = high (10)
- Use Laplacian correction (or Laplacian estimator)
  - Adding 1 to each case
    - Prob(income = low) = 1/1003
    - Prob(income = medium) = 991/1003
    - Prob(income = high) = 11/1003

Naïve Bayes Classifier

- Advantages
  - Easy to implement
  - Good results obtained in most of the cases
- Disadvantages
  - Assume conditional independence: Loss of accuracy. Practically, dependencies exist among variables/features
    - E.g., Hospital-patient data
      - Patient profile: age, family history, etc.
      - Symptoms: fever, cough, etc.
      - Disease: lung cancer, diabetes, etc.
    - Dependencies among these cannot be modeled by Naïve Bayes Classifier

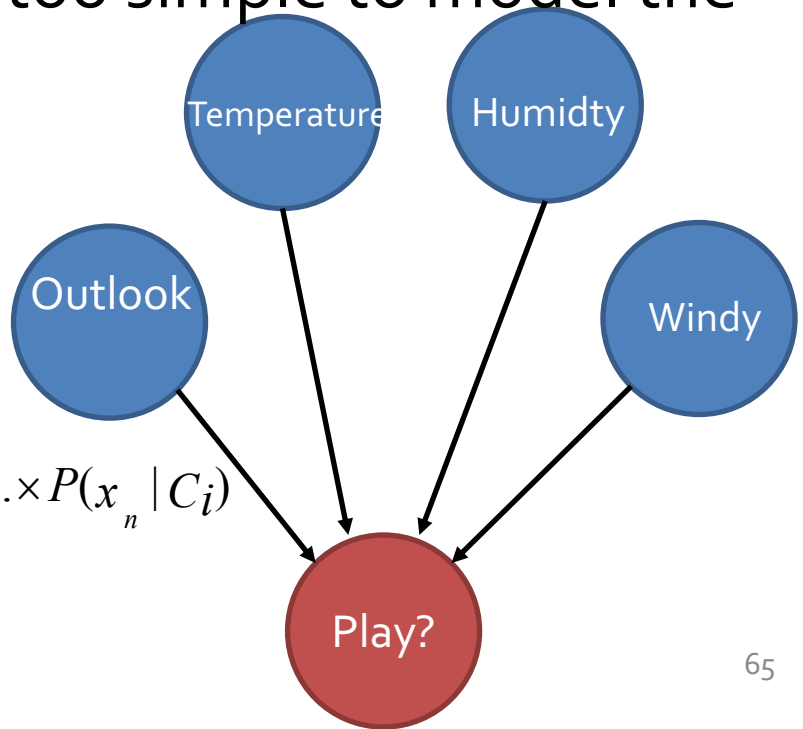
Today: KNN, Naïve Bayes, and Bayesian Networks

- Describe Nearest Neighbor Classifier
- Implement KNN algorithm
- Describe Bayesian learning
- Implement Naïve Bayes algorithm
- **Describe Bayesian network models**

From Naïve Bayes to Bayesian Networks

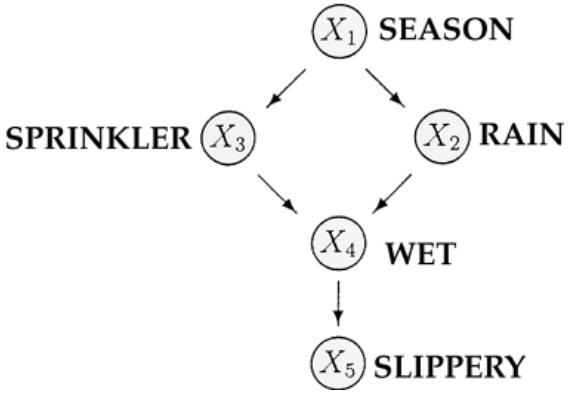
- Naïve Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable
  - This assumption is often too simple to model the real world well

$$P(\mathbf{X} \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i) = P(x_1 \mid C_i) \times P(x_2 \mid C_i) \times \ldots \times P(x_n \mid C_i)$$



From Naïve Bayes to Bayesian Networks

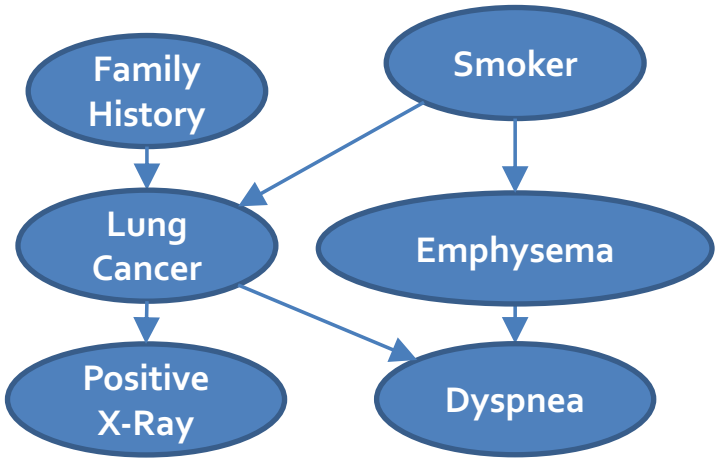
- Bayesian network (or Bayes network, belief network, Bayesian model or probabilistic directed acyclic graphical model) is a probabilistic graphical model
  - Represented by a set of random variables and their conditional dependencies via a directed acyclic graph (DAG)
  - Ex. Given symptoms, the network can be used to compute the probabilities of the presence of various diseases



Bayesian Belief Networks

- **Bayesian belief network** (or **Bayesian network, probabilistic network**):
  - Allows *class conditional independencies* between *subsets* of variables
- Two components:
  - A *directed acyclic graph* (called a structure)
  - A set of *conditional probability tables* (CPTs)

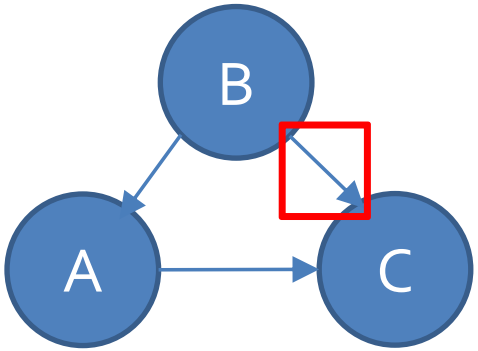
Nodes: random variables Links: dependency



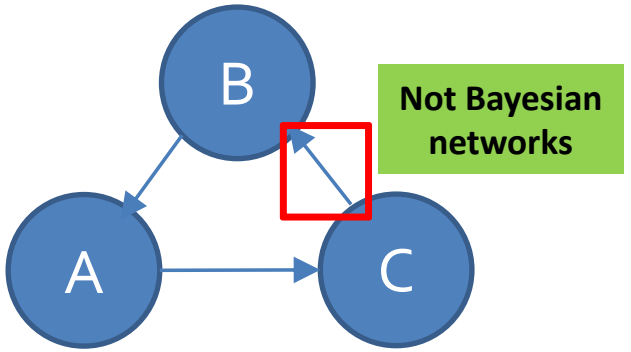
Bayesian Belief Networks

- **Directed Acyclic Graph (DAG):**
  - Represents dependency among the variables (*causal influence* relationship)
  - Gives a specification of joint probability distribution

$$p(A, B, C) = p(B) \cdot p(A|B) \cdot p(C|A, B)$$



directed **acyclic** graphical model



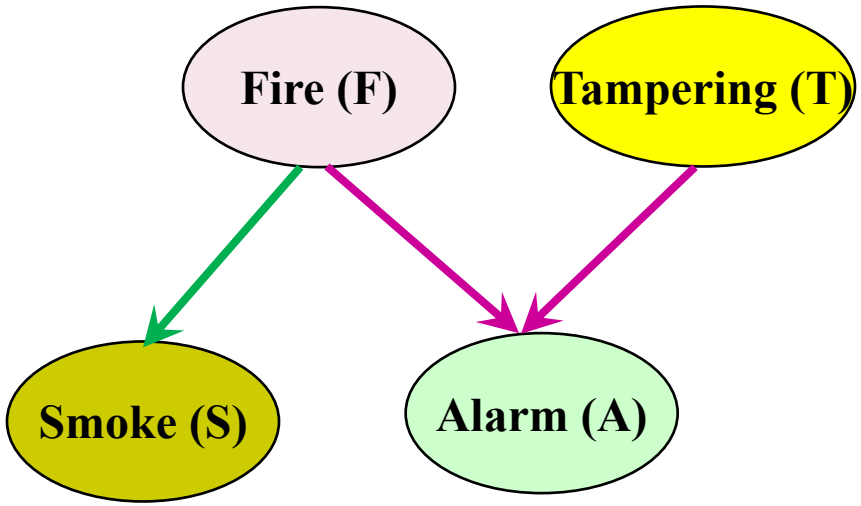
directed **cyclic** graphical model

Example: A Bayesian Network and Its CPTs

Conditional **P**robability **T**ables (CPT)

Fire	Smoke	$\Theta_{s\|f}$
True	True	.90
False	True	.01

Fire	Tampering	Alarm	$\Theta_{a\|f,t}$
True	True	True	.5
True	False	True	.99
False	True	True	.85
False	False	True	.0001

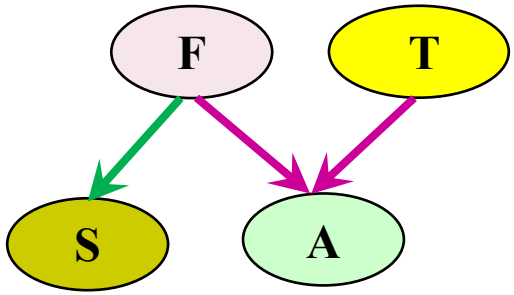


CPT shows the conditional probability for each possible combination of its parents:

$$p(F, S, A, T) = p(F) \cdot p(T) \cdot p(S|F) \cdot p(A|F, T)$$

How Are Bayesian Networks Constructed?

- **Subjective construction**: Identification of (direct) causal structure
  - People are quite good at identifying direct causes from a given set of variables & whether the set contains all relevant direct causes
  - Markovian assumption: Each variable becomes independent of its non-effects once its direct causes are known
    - E.g., S ← F → A ← T, path S → A is blocked once we know F → A



  - HMM (Hidden Markov Model): often used to model dynamic systems whose states are not observable, yet their outputs are.

How Are Bayesian Networks Constructed?

- **Synthesis from other specifications**
  - E.g., from a formal system design: block diagrams & info flow
- **Learning from data** (e.g., from medical records or student admission record)
  - Learn parameters give its structure or learn both structure and parms
  - Maximum likelihood principle: favors Bayesian networks that maximize the probability of observing the given data set

Training Bayesian Networks: Several Scenarios

- Scenario 1: Given both the network structure and all variables observable: *compute only the CPT entries*
- Scenario 2: Network structure known, some variables hidden: *gradient descent* (greedy hill-climbing) method, i.e., search for a solution along the steepest descent of a criterion function
  - Weights are initialized to random probability values
  - At each iteration, it moves towards what appears to be the best solution at the moment, without backtracking
  - Weights are updated at each iteration & converge to local optimum

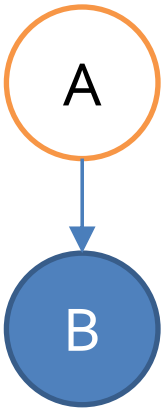
D. Heckerman. A Tutorial on Learning with Bayesian Networks. In *Learning in Graphical Models*, M. Jordan, ed. MIT Press, 1999

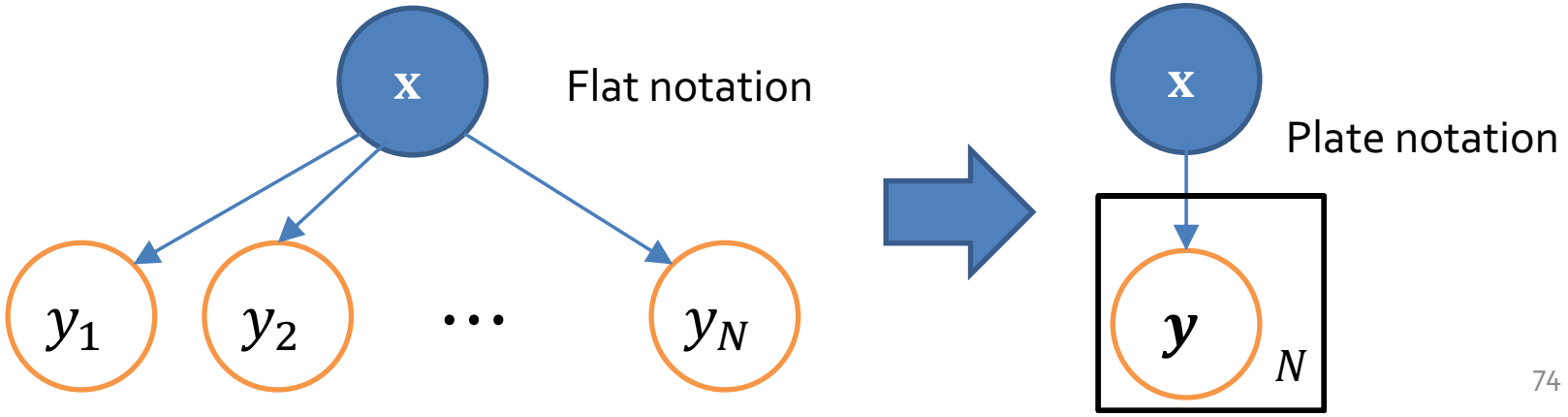
Training Bayesian Networks: Several Scenarios

- Scenario 3: Network structure unknown, all variables observable: search through the model space to *reconstruct network topology*
- Scenario 4: Unknown structure, all hidden variables: No good algorithms known for this purpose

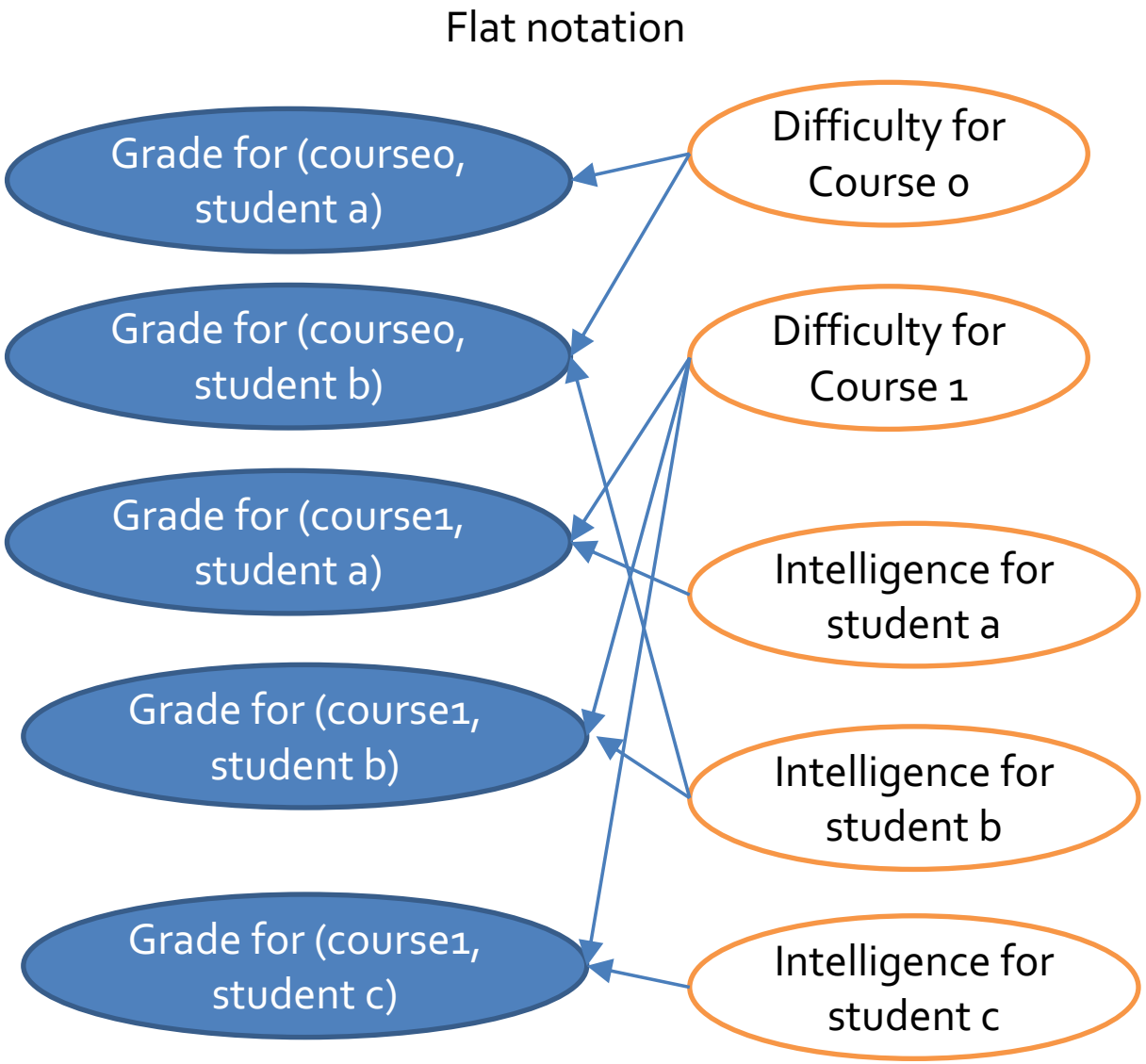
Probabilistic Graphic Model: Plate Notations

- Represent variables that repeat in a graphical model
- Variables
  - A solid (or shaded) circle means the corresponding variable is *observed*; otherwise it is *hidden*
- Dependency among variables:
  - A Directed Acyclic Graphical (DAG) model
- Using plate notation instead of flat notation

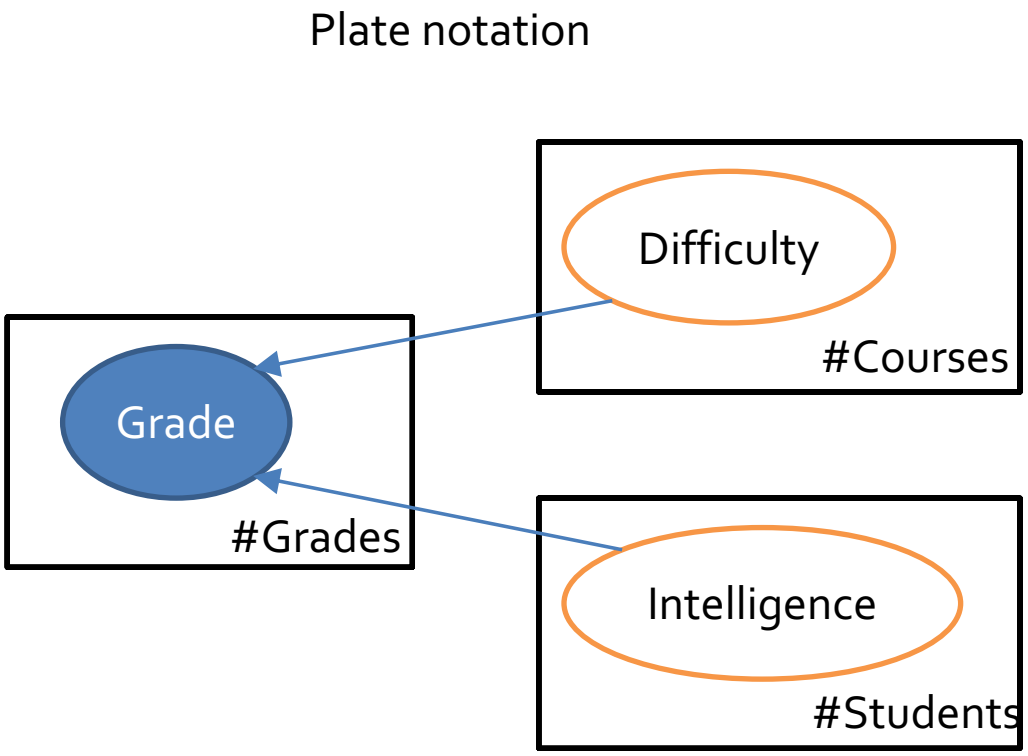




An Example of Plate Notation



An Example of Plate Notation

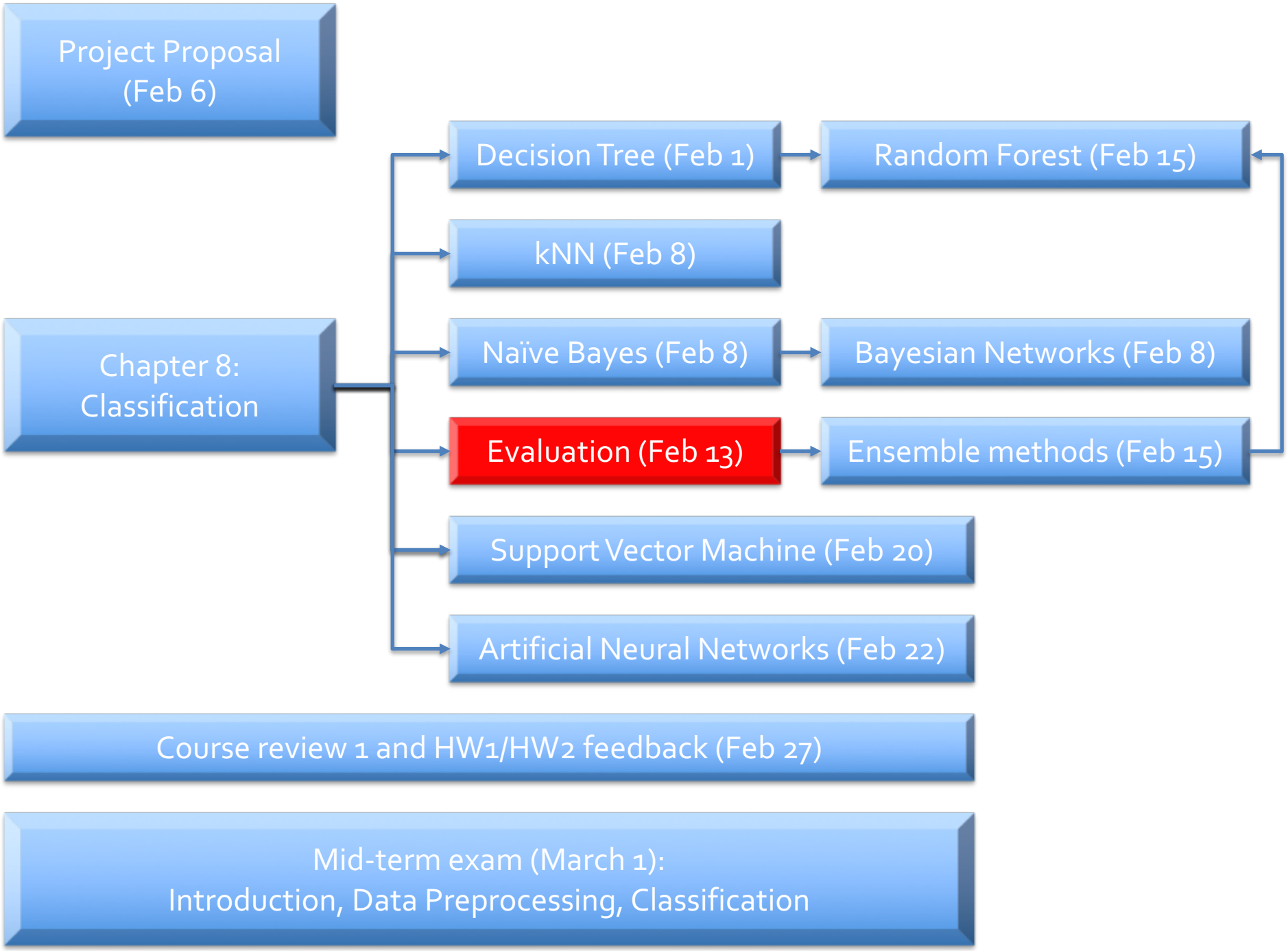


Summary: KNN, Naïve Bayes, and Bayesian Networks

- Describe Nearest Neighbor Classifier
- Implement KNN algorithm
- Describe Bayesian learning
- Implement Naïve Bayes algorithm
- Describe Bayesian network models

ID	Date	Opponent	Is Home or Away?	Is Opponent in AP Top 25 at Preseason?	Media	Label: Win/Lose
1	9/5/15	Texas	Home	Out	1-NBC	Win
2	9/12/15	Virginia	Away	Out	4-ABC	Win
3	9/19/15	Georgia Tech	Home	In	1-NBC	Win
4	9/26/15	UMass	Home	Out	1-NBC	Win
5	10/3/15	Clemson	Away	In	4-ABC	Lose
6	10/10/15	Navy	Home	Out	1-NBC	Win
7	10/17/15	USC	Home	In	1-NBC	Win
8	10/31/15	Temple	Away	Out	4-ABC	Win
9	11/7/15	PITT	Away	Out	4-ABC	Win
10	11/14/15	Wake Forest	Home	Out	1-NBC	Win
11	11/21/15	Boston College	Away	Out	1-NBC	Win
12	11/28/15	Stanford	Away	In	3-FOX	Lose
13	9/4/16	Texas	Away	Out	4-ABC	Lose
14	9/10/16	Nevada	Home	Out	1-NBC	Win
15	9/17/16	Michigan State	Home	Out	1-NBC	Lose
16	9/24/16	Duke	Home	Out	1-NBC	Lose
17	10/1/16	Syracuse	Home	Out	2-ESPN	Win
18	10/8/16	North Carolina State	Away	Out	4-ABC	Lose
19	10/15/16	Stanford	Home	In	1-NBC	Lose
20	10/29/16	Miami Florida	Home	Out	1-NBC	Win
21	11/5/16	Navy	Home	Out	5-CBS	Lose
22	11/12/16	Army	Home	Out	1-NBC	Win
23	11/19/16	Virginia Tech	Home	In	1-NBC	Lose
24	11/26/16	USC	Away	In	4-ABC	Lose
25	9/2/17	Temple	Home	Out	1-NBC	Win
26	9/9/17	Georgia	Home	In	1-NBC	Lose
27	9/16/17	Boston College	Away	Out	2-ESPN	Win
28	9/23/17	Michigan State	Away	Out	3-FOX	Win
29	9/30/17	Miami Ohio	Home	Out	1-NBC	Win
30	10/7/17	North Carolina	Away	Out	4-ABC	Win
31	10/21/17	USC	Home	In	1-NBC	Win
32	10/28/17	North Carolina State	Home	Out	1-NBC	Win
33	11/4/17	Wake Forest	Home	Out	1-NBC	Win
34	11/11/17	Miami Florida	Away	In	4-ABC	Lose
35	11/18/17	Navy	Home	Out	1-NBC	Win
36	11/25/17	Stanford	Away	In	4-ABC	Lose

HW2 due: Feb 20!!!


Project Proposal (Feb 6)
Chapter 8: Classification
Decision Tree (Feb 1)
Random Forest (Feb 15)
kNN (Feb 8)
Naïve Bayes (Feb 8)
Bayesian Networks (Feb 8)
Evaluation (Feb 13)
Ensemble methods (Feb 15)
Support Vector Machine (Feb 20)
Artificial Neural Networks (Feb 22)
Course review 1 and HW1/HW2 feedback (Feb 27)
Mid-term exam (March 1):
Introduction, Data Preprocessing, Classification

References

- C. Apte and S. Weiss. Data mining with decision trees and decision rules. Future Generation Computer Systems, 13, 1997
- P. K. Chan and S. J. Stolfo. Learning arbiter and combiner trees from partitioned data for scaling machine learning. KDD'95
- A. J. Dobson. An Introduction to Generalized Linear Models. Chapman & Hall, 1990.
- R. O. Duda, P. E. Hart, and D. G. Stork. Pattern Classification, 2ed. John Wiley, 2001
- U. M. Fayyad. Branching on attribute values in decision tree generation. AAAI'94.
- Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. J. Computer and System Sciences, 1997.
- J. Gehrke, R. Ramakrishnan, and V. Ganti. Rainforest: A framework for fast decision tree construction of large datasets. VLDB'98.
- J. Gehrke, V. Gant, R. Ramakrishnan, and W.-Y. Loh, BOAT -- Optimistic Decision Tree Construction. SIGMOD'99.
- T. Hastie, R. Tibshirani, and J. Friedman. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer-Verlag, 2001.
- T.-S. Lim, W.-Y. Loh, and Y.-S. Shih. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. Machine Learning, 2000

References (cont.)

- J. Magidson. The Chaid approach to segmentation modeling: Chi-squared automatic interaction detection. In R. P. Bagozzi, editor, Advanced Methods of Marketing Research, Blackwell Business, 1994
- M. Mehta, R. Agrawal, and J. Rissanen. SLIQ : A fast scalable classifier for data mining. EDBT'96
- T. M. Mitchell. Machine Learning. McGraw Hill, 1997
- S. K. Murthy, Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey, Data Mining and Knowledge Discovery 2(4): 345-389, 1998
- J. R. Quinlan. Induction of decision trees. Machine Learning, 1:81-106, 1986.
- J. R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.
- J. R. Quinlan. Bagging, boosting, and c4.5. AAAI'96.
- R. Rastogi and K. Shim. Public: A decision tree classifier that integrates building and pruning. VLDB'98
- J. Shafer, R. Agrawal, and M. Mehta. SPRINT : A scalable parallel classifier for data mining. VLDB'96
- J. W. Shavlik and T. G. Dietterich. Readings in Machine Learning. Morgan Kaufmann, 1990
- P. Tan, M. Steinbach, and V. Kumar. Introduction to Data Mining. Addison Wesley, 2005
- S. M. Weiss and C. A. Kulikowski. Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems. Morgan Kaufman, 1991
- S. M. Weiss and N. Indurkhya. Predictive Data Mining. Morgan Kaufmann, 1997
- I. H. Witten and E. Frank. Data Mining: Practical Machine Learning Tools and Techniques, 2ed. Morgan Kaufmann, 2005