Machine Learning
CMPSCI 589

Bruno C. da Silva
bsilva@cs.umass.edu

Probabilistic Classifiers
Naive Bayes (1/2)

UMassAmherst
College of Information & Computer Sciences

Review: Selecting an Attribute to Test

- Decision tree to predict whether a person will play tennis

Weather	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

- Which attribute to test first?
- Let's consider testing Weather

Original dataset: 9 instances "Yes"
5 instances "No"

yes yes yes yes yes yes yes yes yes
no no no no no

Weather

Review: Selecting an Attribute to Test

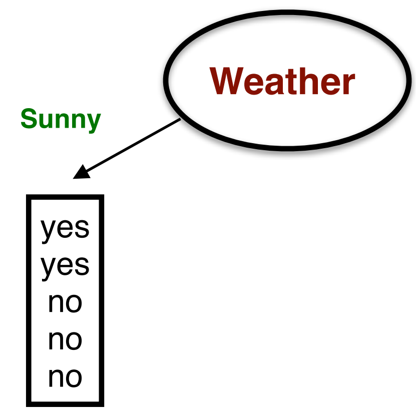
- Decision tree to predict whether a person will play tennis

Weather	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

- Which attribute to test first?
- Let's consider testing Weather

Original dataset: 9 instances "Yes"
5 instances "No"

yes yes yes yes yes yes yes yes yes
no no no no no



Review: Selecting an Attribute to Test

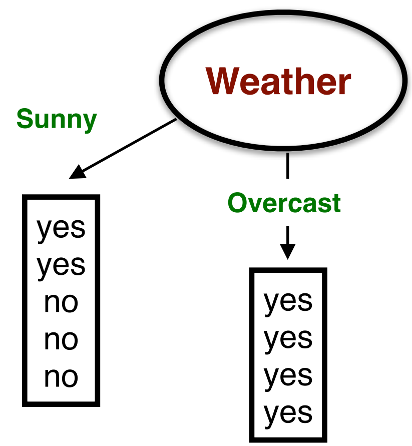
- Decision tree to predict whether a person will play tennis

Weather	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

- Which attribute to test first?
- Let's consider testing Weather

Original dataset: 9 instances "Yes"
5 instances "No"

yes yes yes yes yes yes yes yes yes
no no no no no



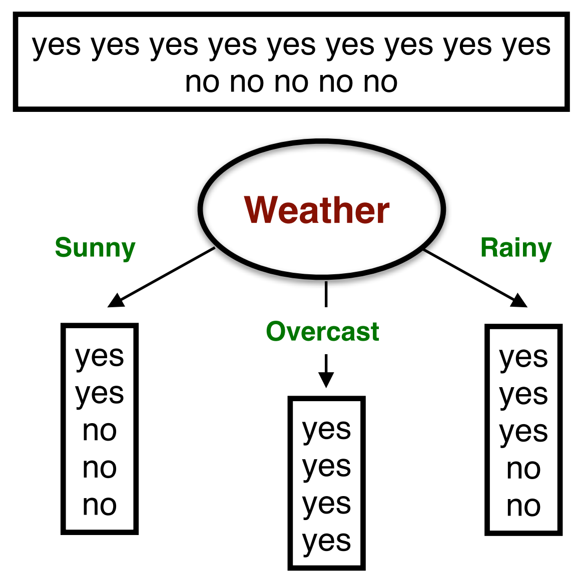
Review: Selecting an Attribute to Test

- Decision tree to predict whether a person will play tennis

Weather	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

- Which attribute to test first?
- Let's consider testing Weather

Original dataset: 9 instances "Yes"
5 instances "No"



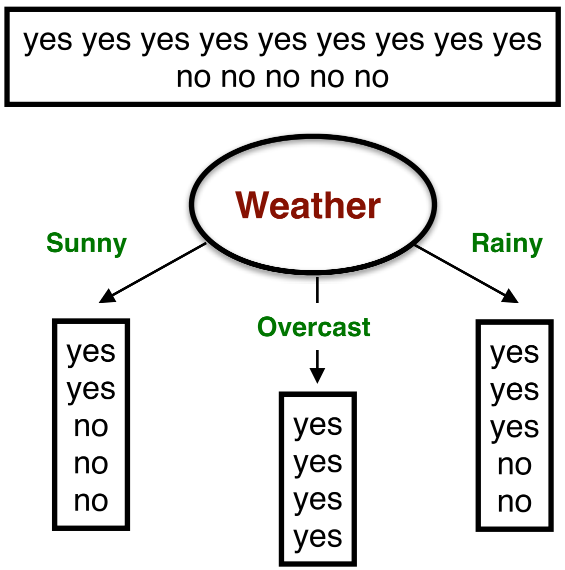
Review: Selecting an Attribute to Test

- Decision tree to predict whether a person will play tennis

- **Entropy of the original dataset:**
  - $\mathbf{I}(9/14, 5/14) = -9/14 \log_2(9/14) - 5/14 \log_2(5/14)$
    $= \mathbf{0.940\ bits}$
- **Entropy of partitions resulting from testing Weather:**
  - **Weather=Sunny**
    - $\mathbf{I}(2/5, 3/5) = -2/5 \log_2(2/5) - 3/5 \log_2(3/5)$
      $= 0.971$ bits
  - **Weather=Overcast**
    - $\mathbf{I}(4/4, 0/4) = -4/4 \log_2(4/4) - 0/4 \log_2(0/4)$
      $= 0$ bits
  - **Weather=Rainy**
    - $\mathbf{I}(3/5, 2/5) = -3/5 \log_2(3/5) - 2/5 \log_2(2/5)$
      $= 0.971$ bits
- **Average entropy of the resulting partitions**
  - $(5/14)\times 0.971 + (4/14)\times 0 + (5/14)\times 0.971 = \mathbf{0.693\ bits}$

- Which attribute to test first?
- Let's consider testing Weather

Original dataset: 9 instances "Yes"
5 instances "No"



 Bruno Castro da Silva

Review: Selecting an Attribute to Test

- Decision tree to predict whether a person will play tennis

- **Entropy of the original dataset:**
  - **I**(9/14, 5/14) = -9/14 $\log_2$(9/14) -5/14 $\log_2$(5/14)
    = **0.940 bits**
- **Entropy of partitions resulting from testing Weather:**
  - **Weather=Sunny**
    - **I**(2/5, 3/5) = -2/5 $\log_2$(2/5) -3/5 $\log_2$(3/5)
      = 0.971 bits
  - **Weather=Overcast**
    - **I**(4/4, 0/4) = -4/4 $\log_2$(4/4) -0/4 $\log_2$(0/4)
      = 0 bits
  - **Weather=Rainy**
    - **I**(3/5, 2/5) = -3/5 $\log_2$(3/5) -2/5 $\log_2$(2/5)
      = 0.971 bits
- **Average entropy of the resulting partitions**
  - (5/14)x0.971 + (4/14)x0 + (5/14)x0.971 = **0.693 bits**

By testing the attribute **Weather**, the entropy of the classes decreased by 0.940 - 0.693 = **0.247 bits**

Information Gain

Review: Criteria for Selecting an Attribute to Test

- We have discussed <u>one</u> possible criterion for selecting which attribute to test
  - **Information Gain**

- Many other criteria have been proposed — each with different properties
- Intuitively:
  - A split that keeps the same proportion of classes in each partition is **useless**
  - A split where the instances in each partition have the same class is **useful**!

Review: Criteria for Selecting an Attribute to Test

- We have discussed one possible criterion for selecting which attribute to test
  - **Information Gain**

- Many other criteria have been proposed — each with different properties
- Intuitively:
  - A split that keeps the same proportion of classes in each partition is **useless**
  - A split where the instances in each partition have the same class is **useful**!

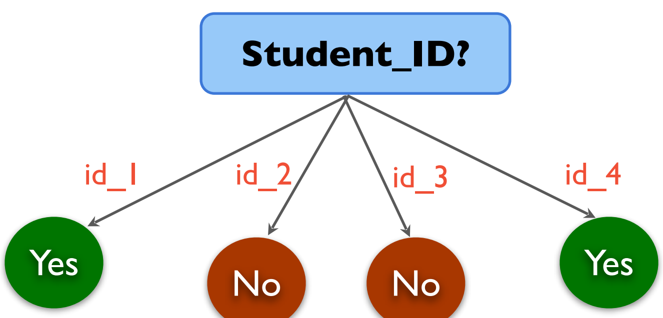
- Main criteria for selecting which attribute to test:
  - **Information Gain** - ID3 Algorithm (Quilan, 1987)
  - **Information Gain Ratio** - C4.5 Algorithm (Quilan, 1988)
  - **Gini Impurity** - CART Algorithm (Breiman, 1984)

Review: Information Gain

- This is the criterion discussed earlier → results in a method known as **ID3**
- Intuitively, it selects the attribute A that maximizes the difference between:
  - The entropy of the original dataset D (before splitting it based on A)
  - The average entropy of the resulting partitions if we split dataset D based on A

- Often results in a decision tree that is not necessarily the "simplest" one
- Intuitively, it often chooses attributes with many possible values (like Student_ID, Name, etc)

Student_ID	Student	Age	Credit_Score	Will_Buy_Computer
id_1	Yes	Young	Regular	Yes
id_2	Yes	Middle Age	Excellent	No
id_3	No	Young	Excellent	No
id_4	No	Older Adult	Regular	Yes

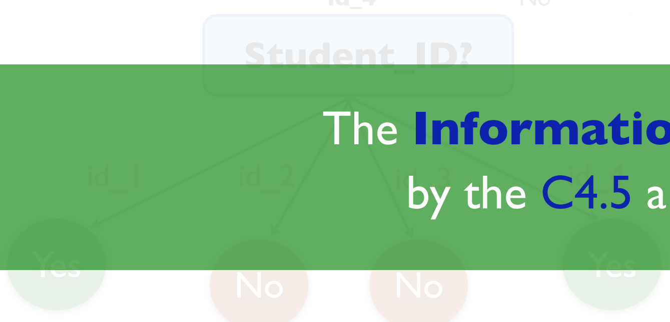


- Perfect split!
- With just one test, can "predict" the class perfectly
- But it is clearly overfitting ("memorizing" the dataset)

Review: Information Gain

- This is the criterion discussed earlier → results in a method known as **ID3**
- Intuitively, it selects the attribute A that maximizes the difference between:
  - The entropy of the original dataset D (before splitting it based on A)
  - The average entropy of the resulting partitions if we split dataset D based on A
- Often results in a decision tree that is not necessarily the "simplest" one
- Intuitively, it often chooses attributes with many possible values (like Student_ID, Name, etc)

Student_ID	Student	Age	Credit_Score	Will_Buy_Computer
id_1	Yes	Young	Regular	Yes
id_2	Yes	Middle Age	Excellent	No
id_3	No	Young	Excellent	No
id_4	No	Older Adult	Regular	Yes



- Perfect split!
- With just one test, can "predict" the class perfectly
- But it is clearly overfitting ("memorizing" the dataset)

The **Information Gain Ratio** criterion, implemented by the C4.5 algorithm, tries to mitigate this issue

Review: Criteria for Selecting an Attribute to Test

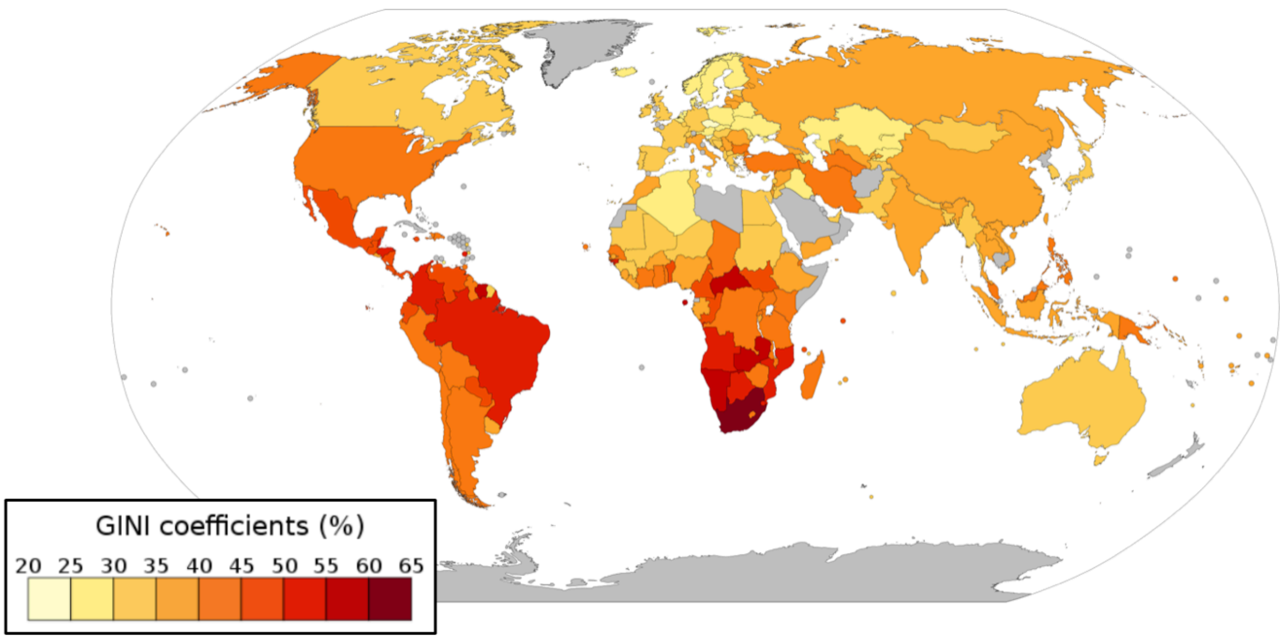
- We have discussed one possible criterion for selecting which attribute to test
  - **Information Gain**

- Many other criteria have been proposed — each with different properties
- Intuitively:
  - A split that keeps the same proportion of classes in each partition is **useless**
  - A split where the instances in each partition have the same class is **useful**!

- Main criteria for selecting which attribute to test:
  - **Information Gain** - ID3 Algorithm (Quilan, 1987)
  - **Information Gain Ratio** - C4.5 Algorithm (Quilan, 1988)
  - **Gini Impurity** - CART Algorithm (Breiman, 1984)

Gini Criterion

- Originally proposed to quantify how uneven income is across a population



- Gini coefficient → how uneven income/wealth distribution across a population is
  - Gini = 1 → very uneven income/wealth distribution across a population
  - Gini = 0 → very even income/wealth distribution across a population

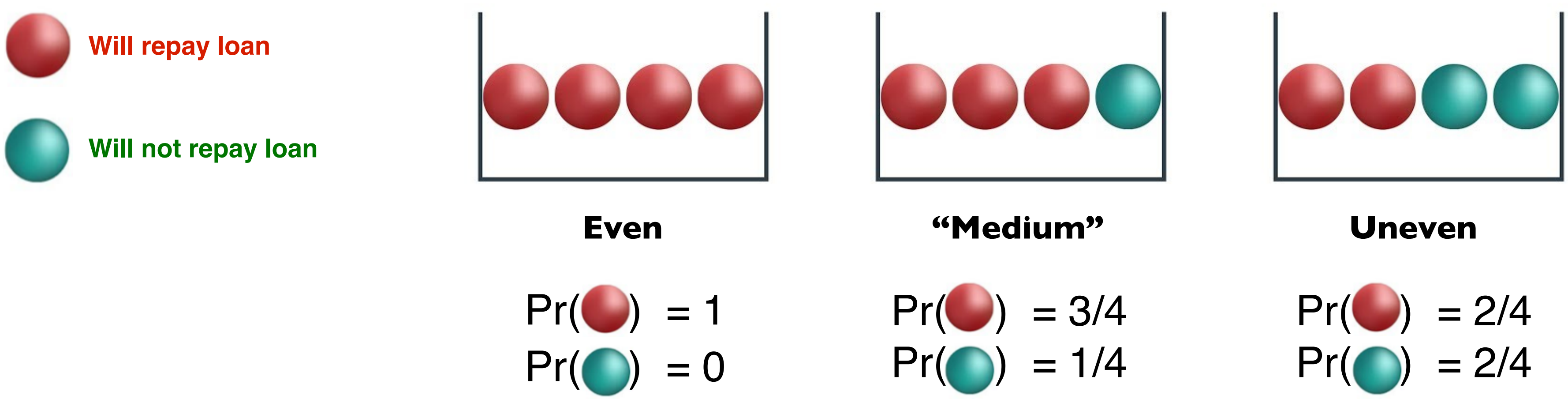
Gini Criterion

- Gini coefficient → how uneven income/wealth distribution across a population is
- In the context of decision trees
  - how uneven (or non-homogeneous) are the classes after a split

Gini Criterion

- Gini coefficient → how uneven income/wealth distribution across a population is
- In the context of decision trees
  - how uneven (or non-homogeneous) are the classes after a split

Let's suppose we test Age, and the instances associated with Age=Young look like this



Gini Criterion

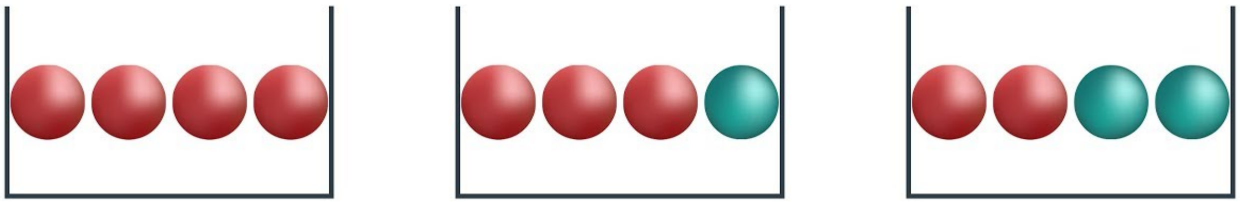
- Gini coefficient → how uneven income/wealth distribution across a population is
- In the context of decision trees
  - how uneven (or non-homogeneous) are the classes after a split

Gini(D) = 1 - (Pr( )² + Pr( )²)

Let's suppose we test Age, and the instances associated with Age=Young look like this

Will repay loan

Will not repay loan



Even	**"Medium"**	**Uneven**
Pr( ) = 1	Pr( ) = 3/4	Pr( ) = 2/4
Pr( ) = 0	Pr( ) = 1/4	Pr( ) = 2/4

Gini $\Rightarrow$

$$1-(1^2+0^2) \\ =0$$

$$1-\left((3/4)^2+(1/4)^2\right) \\ =0.375$$

$$1-\left((2/4)^2+(2/4)^2\right) \\ =0.5$$

Gini Criterion

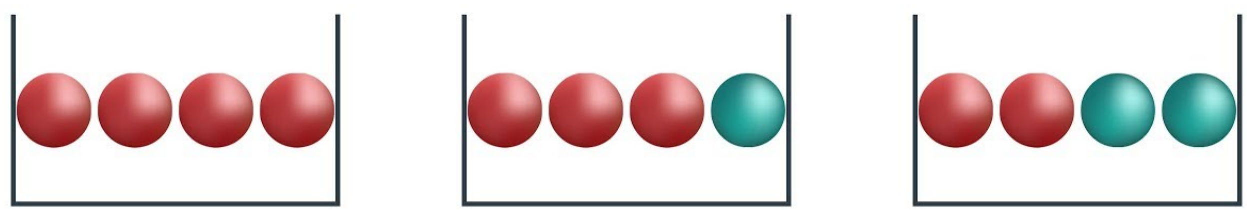
- Gini coefficient → how uneven income/wealth distribution across a population is
- In the context of decision trees
  - how uneven (or non-homogeneous) are the classes after a split

$$\text{Gini}(D) = 1 - (\ \Pr(\bullet)^2 + \Pr(\bullet)^2)$$

Let's suppose we test Age, and the instances associated with Age=Young look like this

Will repay loan

Will not repay loan

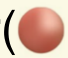
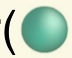


	Even	**"Medium"**	**Uneven**
Gini ⇒	$1-(1^2+0^2)$ $=0$	$1-\left((3/4)^2+(1/4)^2\right)$ $=0.375$	$1-\left((2/4)^2+(2/4)^2\right)$ $=0.5$

more homogenous partition → ideal result of a split (smaller value of the Gini coefficient)

Gini Criterion

- In the context of decision trees
  - how uneven (or non-homogeneous) are the classes after a split

Gini(D) = 1 - (Pr()^2 + Pr()^2)

- More generally, if there are m classes in a dataset D

$$\text{Gini}(D) \;=\; 1 - \left(\sum_{i=1}^{m} (p_i)^2 \right)$$

where p_i be the probability that the label/class i occurs in instances in a dataset D

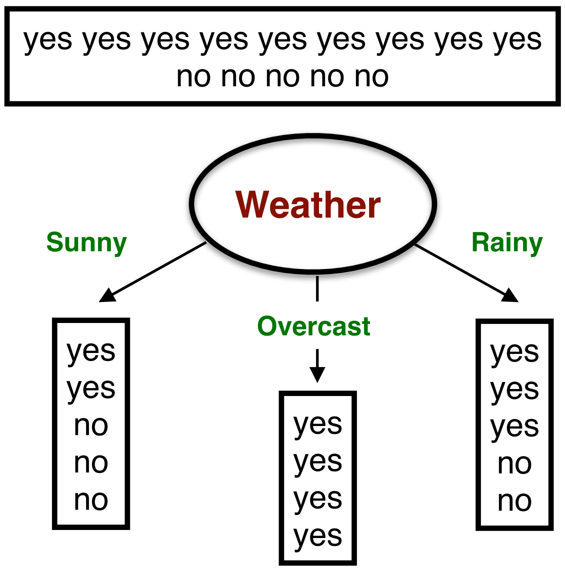
Gini Criterion

- Decision tree to predict whether a person will play tennis

Weather	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

- Let's consider testing Weather

Original dataset: 9 instances "Yes"
5 instances "No"



Gini Criterion

- Decision tree to predict whether a person will play tennis

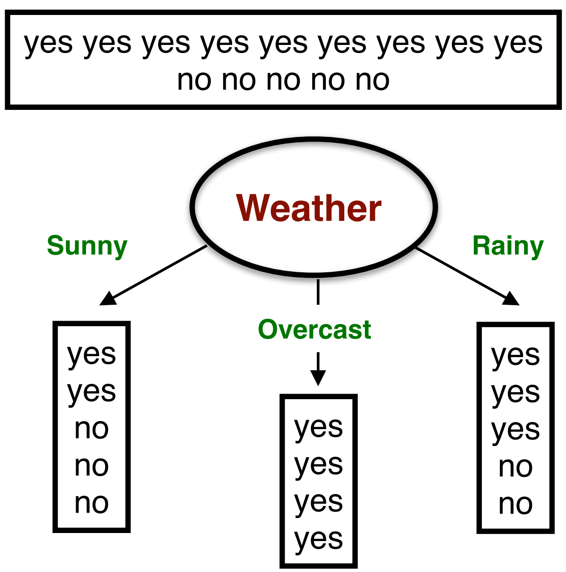
- **Gini coefficient of the original dataset:**
  - **Gini**(9/14, 5/14) = 1 - ($(9/14)^2 + (5/14)^2$)
    = **0.459**

- **Gini coeff. of partitions resulting from testing Weather:**
  - **Weather=Sunny**
    - $\mathbf{Gini}_{Sunny}(2/5, 3/5) = 1 - ((2/5)^2 + (3/5)^2)$
      = 0.48
  - **Weather=Overcast**
    - $\mathbf{Gini}_{Overcast}(4/4, 0/4) = 1 - ((4/4)^2 + (0/4)^2)$
      = 0
  - **Weather=Rainy**
    - $\mathbf{Gini}_{Rainy}(3/5, 2/5) = 1 - ((3/5)^2 + (2/5)^2)$
      = 0.48

- **Average Gini coefficient of the resulting partitions**
  - (5/14)x0.48 + (4/14)x0 + (5/14)x0.48 = **0.3428**

- Let's consider testing Weather

Original dataset: 9 instances "Yes"
5 instances "No"



 Bruno Castro da Silva

Gini Criterion

- Decision tree to predict whether a person will play tennis

- **Gini coefficient of the original dataset:**
  - **Gini**(9/14, 5/14) = 1 - ($(9/14)^2 + (5/14)^2$)
    = **0.459**

- **Gini coeff. of partitions resulting from testing Weather:**
  - **Weather=Sunny**
    - $\textbf{Gini}_{Sunny}$(2/5, 3/5) = 1 - ($(2/5)^2 + (3/5)^2$)
      = 0.48
  - **Weather=Overcast**
    - $\textbf{Gini}_{Overcast}$(4/4, 0/4) = 1 - ($(4/4)^2 + (0/4)^2$)
      = 0
  - **Weather=Rainy**
    - $\textbf{Gini}_{Rainy}$(3/5, 2/5) = 1 - ($(3/5)^2 + (2/5)^2$)
      = 0.48

- **Average Gini coefficient of the resulting partitions**
  - (5/14)x0.48 + (4/14)x0 + (5/14)x0.48 = **0.3428**

Testing the attribute **Weather**:
Gini(Weather) = **0.3428**

- Now proceed similarly as when selecting attributes via Information Gain…

- Compute Gini coefficient of each candidate attribute

- Split dataset using the attribute with the **lowest Gini coefficient**

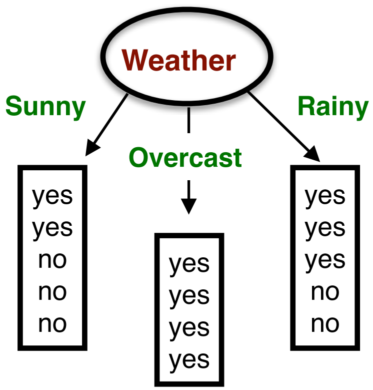
Criteria for Selecting an Attribute to Test

- Main criteria for selecting which attribute to test:
  - **Information Gain** - ID3 Algorithm (Quilan, 1987)
  - **Information Gain Ratio** - C4.5 Algorithm (Quilan, 1988)
  - **Gini Impurity** - CART Algorithm (Breiman, 1984)

- **Empirically:**
  - Information Gain Ratio is almost always better than Information Gain
    - in terms of predictive power and complexity of the resulting decision trees
  - However, in practice
    - which criterion will work best depends heavily on the application
    - should test them all and compare the resulting performances

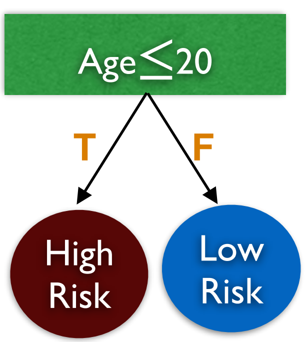
Dealing with Numerical Attributes

- So far we have studied how to select which **categorical attribute** to split



→ One branch per possible value of the attribute

- How do we decide a splitting point/value in case of **numerical attributes**?



→ Consider deciding how to split the attribute **Age**

Pick a threshold value, V
Generate two branches/disjoint partitions:

- one partition with instances s.t. **Age** $\leq V$
- one partition with instances s.t. **Age** $> V$

Dealing with Numerical Attributes

- How do we decide a splitting point/value in case of **numerical attributes**?

  - one partition with instances s.t. **Age** $\leq V$
  - one partition with instances s.t. **Age** $> V$

1) Sort the instances according to the value of the attribute

Name	*Age*	*Gender*	*TrafficTicket*	Class: High-Risk Driver
John	43	M	Yes	**High Risk**
Peter	18	M	No	**High Risk**
Anna	35	F	No	**Low Risk**
Paula	19	F	No	**High Risk**
Mark	90	M	Yes	**High Risk**
Marisa	21	F	Yes	**High Risk**
Bob	30	M	No	**Low Risk**

Dealing with Numerical Attributes

- How do we decide a splitting point/value in case of **numerical attributes**?

  - one partition with instances s.t. **Age** $\leq V$
  - one partition with instances s.t. **Age** $> V$

1) Sort the instances according to the value of the attribute

Name	***Age***	***Gender***	***TrafficTicket***	**Class: High-Risk Driver**
Peter	18	M	No	**High Risk**
Paula	19	F	No	**High Risk**
Marisa	21	F	Yes	**High Risk**
Bob	30	M	No	**Low Risk**
Anna	35	F	No	**Low Risk**
John	43	M	Yes	**High Risk**
Mark	90	M	Yes	**High Risk**

Dealing with Numerical Attributes

- How do we decide a splitting point/value in case of **numerical attributes**?

  - one partition with instances s.t. **Age** $\leq V$
  - one partition with instances s.t. **Age** $> V$

1) Sort the instances according to the value of the attribute

2) Evaluate splits done using as threshold the mean values between consecutive Ages

Name	*Age*	*Gender*	*TrafficTicket*	Class: High-Risk Driver
Peter	18	M	No	High Risk
Paula	19	F	No	High Risk
Marisa	21	F	Yes	High Risk
Bob	30	M	No	Low Risk
Anna	35	F	No	Low Risk
John	43	M	Yes	High Risk
Mark	90	M	Yes	High Risk

Dealing with Numerical Attributes

- How do we decide a splitting point/value in case of **numerical attributes**?

  - one partition with instances s.t. **Age** $\leq V$
  - one partition with instances s.t. **Age** $> V$

1) Sort the instances according to the value of the attribute

2) Evaluate splits done using as threshold the mean values between consecutive Ages

Name	*Age*	*Gender*	*TrafficTicket*	Class: High-Risk Driver
Peter	18	M	No	High Risk
Paula	19	F	No	High Risk
Marisa	21	F	Yes	High Risk
Bob	30	M	No	Low Risk
Anna	35	F	No	Low Risk
John	43	M	Yes	High Risk
Mark	90	M	Yes	High Risk

Age≤18.5

Dealing with Numerical Attributes

- How do we decide a splitting point/value in case of **numerical attributes**?

  - one partition with instances s.t. **Age** $\leq V$
  - one partition with instances s.t. **Age** $> V$

1) Sort the instances according to the value of the attribute

2) Evaluate splits done using as threshold the mean values between consecutive Ages

Name	*Age*	*Gender*	*TrafficTicket*	Class: High-Risk Driver
Peter	18	M	No	High Risk
Paula	19	F	No	High Risk
Marisa	21	F	Yes	High Risk
Bob	30	M	No	Low Risk
Anna	35	F	No	Low Risk
John	43	M	Yes	High Risk
Mark	90	M	Yes	High Risk

Age≤18.5

Age≤20

Dealing with Numerical Attributes

- How do we decide a splitting point/value in case of **numerical attributes**?

  - one partition with instances s.t. **Age** $\leq V$
  - one partition with instances s.t. **Age** $> V$

1) Sort the instances according to the value of the attribute

2) Evaluate splits done using as threshold the mean values between consecutive Ages

Name	*Age*	*Gender*	*TrafficTicket*	Class: High-Risk Driver
Peter	18	M	No	High Risk
Paula	19	F	No	High Risk
Marisa	21	F	Yes	High Risk
Bob	30	M	No	Low Risk
Anna	35	F	No	Low Risk
John	43	M	Yes	High Risk
Mark	90	M	Yes	High Risk

- Age≤18.5
- Age≤20
- Age≤25.5
- Age≤32.5
- Age≤39
- Age≤66.5

Dealing with Numerical Attributes

- How do we decide a splitting point/value in case of **numerical attributes**?

  - one partition with instances s.t. **Age** $\leq V$
  - one partition with instances s.t. **Age** $> V$

1) Sort the instances according to the value of the attribute

2) Evaluate splits done using as threshold the mean values between consecutive Ages

Age≤18.5	Age≤20	Age≤25.5	Age≤32.5	Age≤39	Age≤66.5

3) Pick the split threshold that maximizes the criterion of interest ***(Info. Gain, Gini, etc.)***

- It has been shown that, for most commonly-used splitting criteria
  - testing only thresholds that correspond to such mean values is sufficient

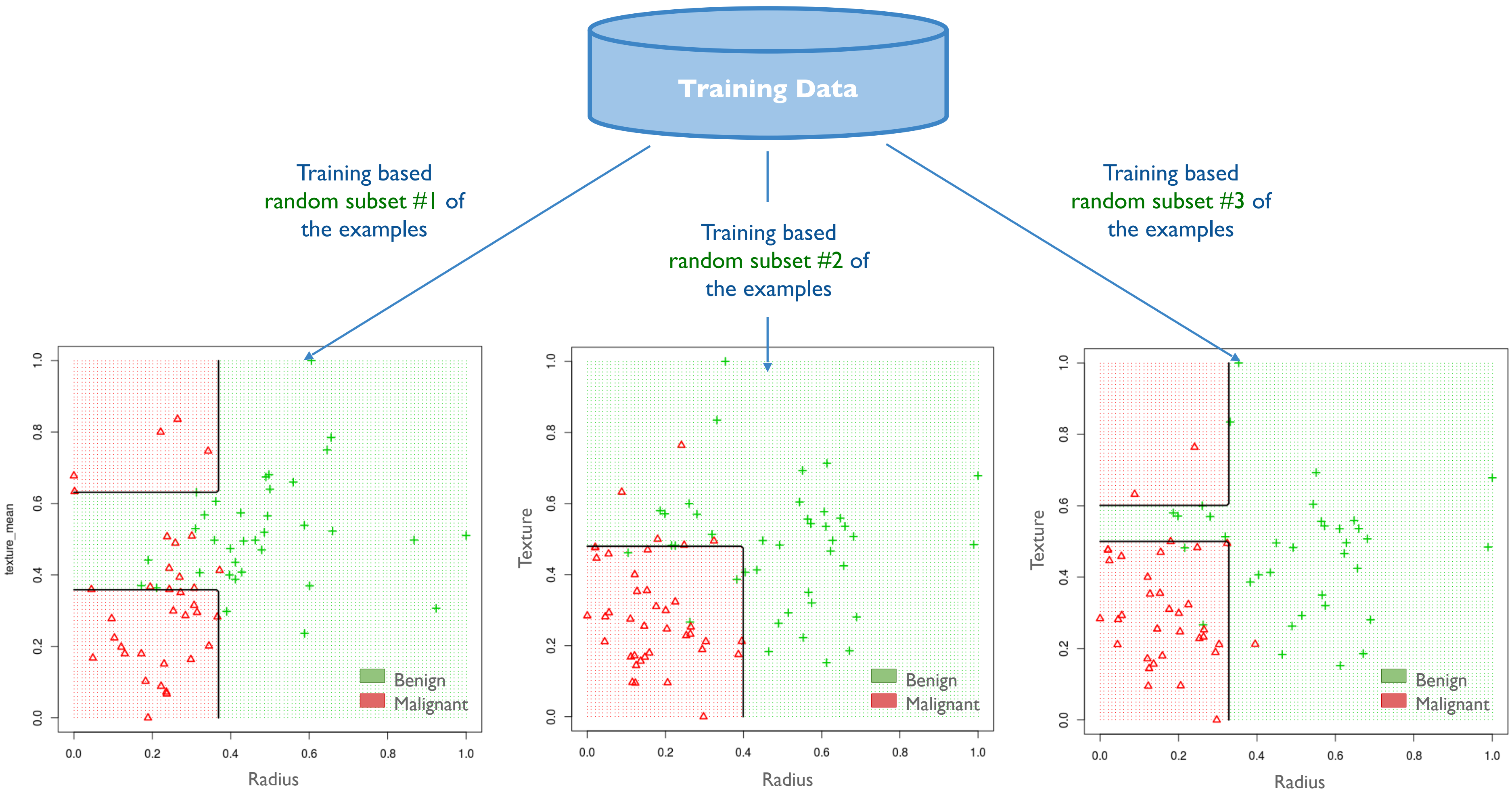
Decision Trees: Pros and Cons

- **Pros:**
  - Simple for humans to understand and interpret
  - Handles both numerical and categorical attributes
  - Requires little data preparation *(e.g., no need to normalize attributes)*
  - Performs well with large datasets
  - “Automatically" ignores irrelevant attributes not useful to predict the class/label

- **Cons:**
  - Non-robust: small variations in the dataset can generate completely different trees

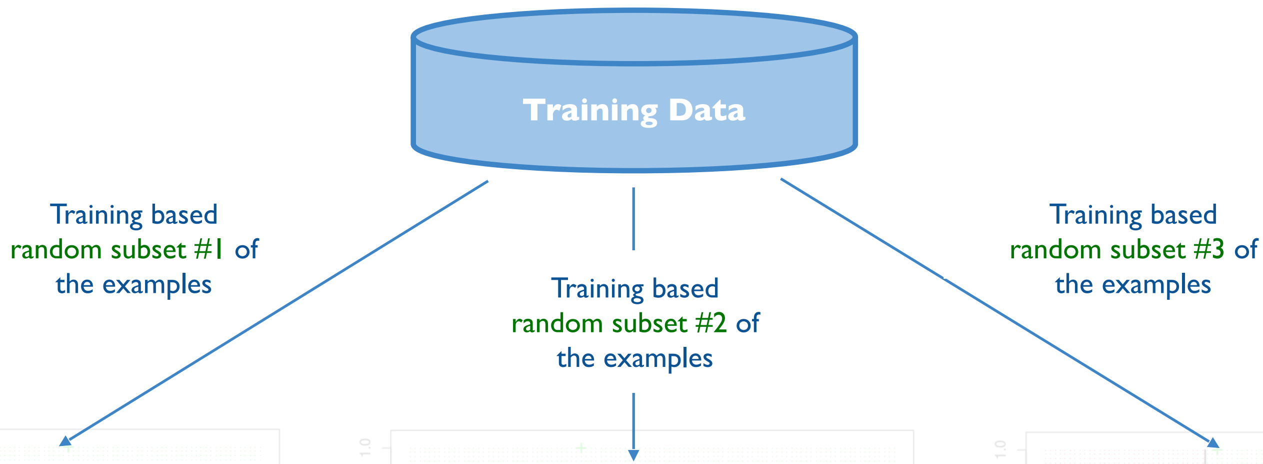
Decision Trees Can be Non-Robust

Decision boundary can be **heavily influenced** by **small changes** to the training data



Decision Trees Can be Non-Robust

Decision boundary can be **heavily influenced** by **small changes** to the training data



Tackling this issue is the motivation to develop the Random Forests algorithm (will be studied later)

Decision Trees: Pros and Cons

- **Pros:**
  - Simple for humans to understand and interpret
  - Handles both numerical and categorical attributes
  - Requires little data preparation *(e.g., no need to normalize attributes)*
  - Performs well with large datasets
  - “Automatically" ignores irrelevant attributes not useful to predict the class/label

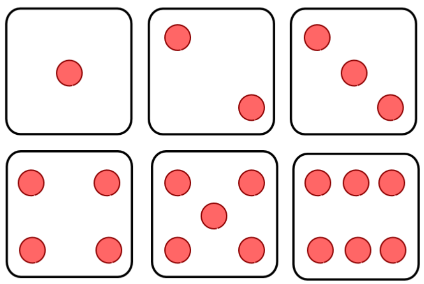
- **Cons:**
  - Non-robust: small variations in the dataset can generate completely different trees
  - Often generate overly-complicated trees that overfit to training data
    - i.e., that do not generalize well (make correct predictions) to new instances
  - Although it is possible to deal with numerical attributes, it is time-consuming
    - estimates suggest that processing them takes ~70% of execution time (Catlett, 1991)

Naive Bayes Algorithm

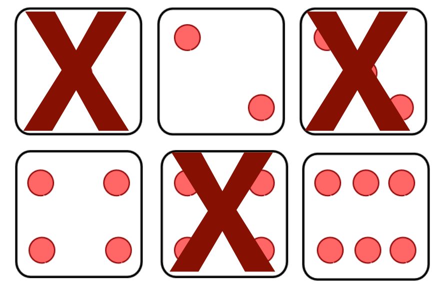
but before that…

Probability Theory - Review

Review: Probability Theory

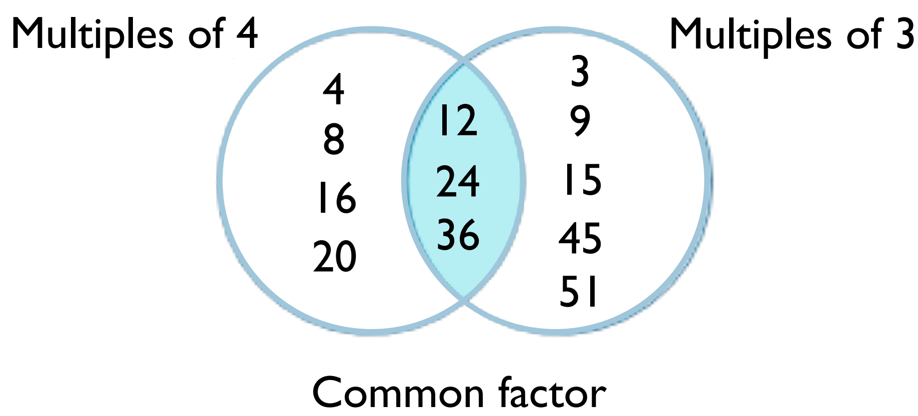


Die: probability of rolling a 2 **= 1/6**



Die: probability of rolling a 2
given that it’s a special die
that only produces even numbers **= 1/3**

Review: Probability Theory

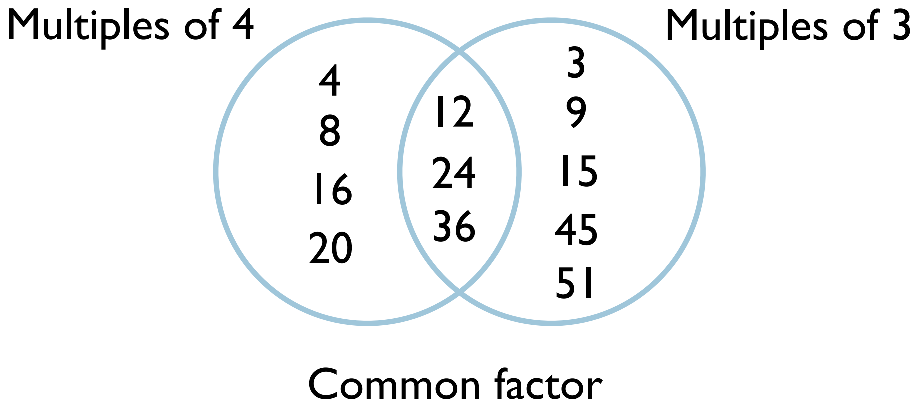


$\Pr(\text{Multiple of 4}) = 7/12$

$\Pr(\text{Multiple of 3}) = 8/12$

$\Pr(\text{Multiple of 4 and Multiple of 3}) = 3/12$

Review: Probability Theory



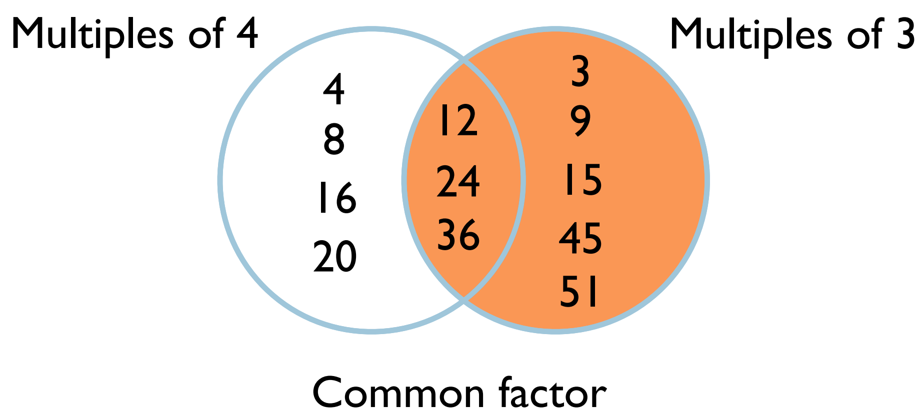
Pr(Multiple of 4) = 7/12

Pr(Multiple of 3) = 8/12

Pr(Multiple of 4 and Multiple of 3) = 3/12

Pr(Multiple of 4 | Multiple of 3)

Review: Probability Theory



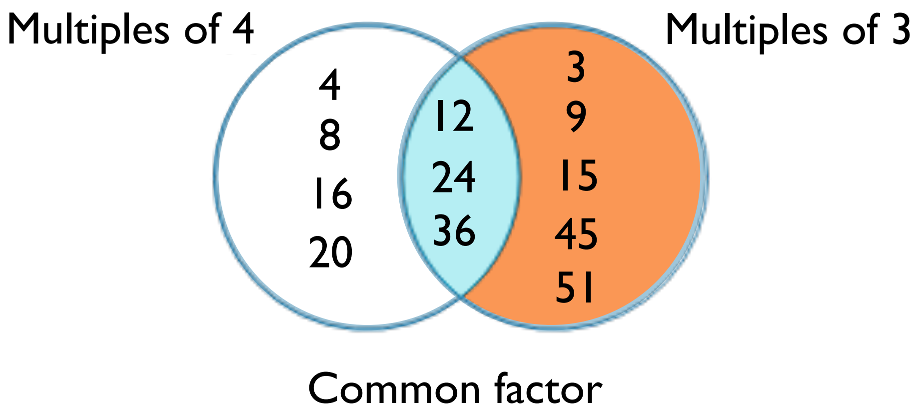
Pr(Multiple of 4) = 7/12

Pr(Multiple of 3) = 8/12

Pr(Multiple of 4 and Multiple of 3) = 3/12

Pr(Multiple of 4 | Multiple of 3)

Review: Probability Theory



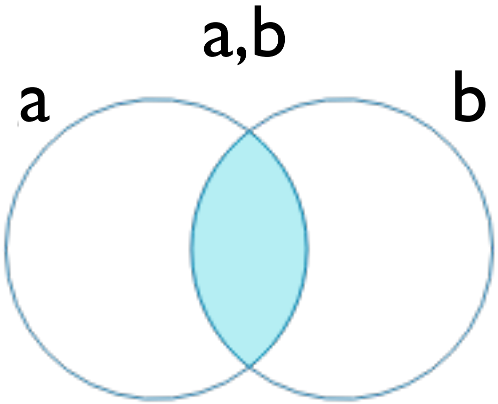
$\Pr(\text{Multiple of 4}) = 7/12$

$\Pr(\text{Multiple of 3}) = 8/12$

$\Pr(\text{Multiple of 4 and Multiple of 3}) = 3/12$

$$\Pr(\text{Multiple of 4} \mid \text{Multiple of 3}) = \frac{\Pr(\text{Multiple of 4 and Multiple of 3})}{\Pr(\text{Multiple of 3})}$$

Review: Probability Theory

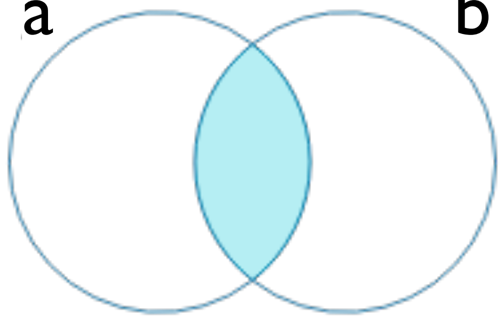


$$\Pr(\text{Multiple of 4} \mid \text{Multiple of 3}) = \frac{\Pr(\text{Multiple of 4 and Multiple of 3})}{\Pr(\text{Multiple of 3})}$$

$$\Pr(a \mid b) = \frac{\Pr(a, b)}{\Pr(b)}$$

 Bruno Castro da Silva

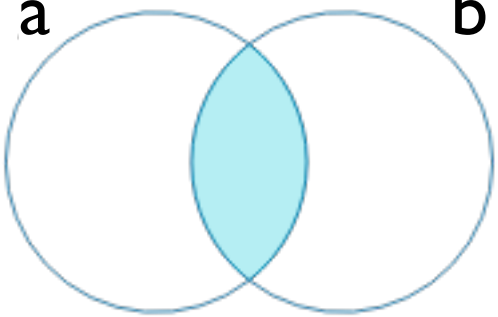
Review: Probability Theory



$$\Pr(a \mid b) = \frac{\Pr(a, b)}{\Pr(b)}$$

$$\Pr(a \mid b)\ \Pr(b) = \Pr(a, b)$$

Review: Probability Theory



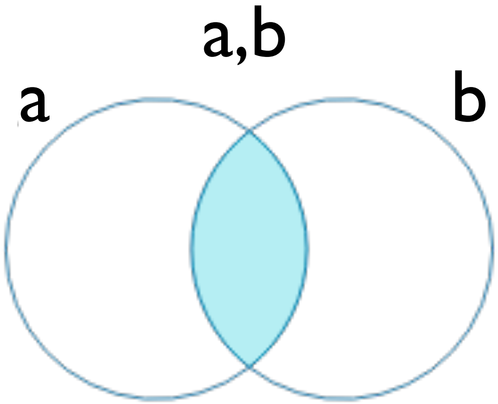
$$\Pr(a \mid b) = \frac{\Pr(a, b)}{\Pr(b)}$$

$$\Pr(b \mid a) = \frac{\Pr(b, a)}{\Pr(a)}$$

$$\Pr(a \mid b)\ \Pr(b) = \Pr(a, b)$$

$$\Pr(b \mid a)\ \Pr(a) = \Pr(b, a)$$

Review: Probability Theory

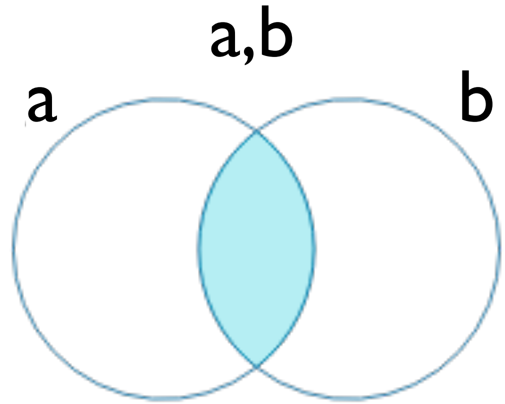


$$\Pr(a \mid b) = \frac{\Pr(a, b)}{\Pr(b)} \qquad \Pr(b \mid a) = \frac{\Pr(b, a)}{\Pr(a)}$$

$$\Pr(a \mid b)\ \Pr(b) = \Pr(a, b) \qquad \Pr(b \mid a)\ \Pr(a) = \Pr(b, a)$$

$$\Pr(a, b) = \Pr(b, a)$$

Review: Probability Theory

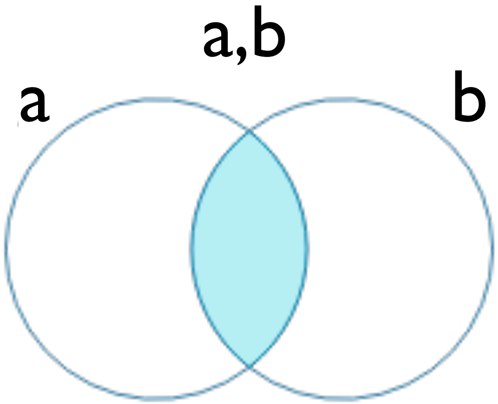


$$\Pr(a \mid b) = \frac{\Pr(a,b)}{\Pr(b)} \qquad \Pr(b \mid a) = \frac{\Pr(b,a)}{\Pr(a)}$$

$$\Pr(a \mid b)\ \Pr(b) = \Pr(a,b) \qquad \Pr(b \mid a)\ \Pr(a) = \Pr(a,b)$$

$$\Pr(a,b) = \Pr(b,a)$$

Review: Probability Theory

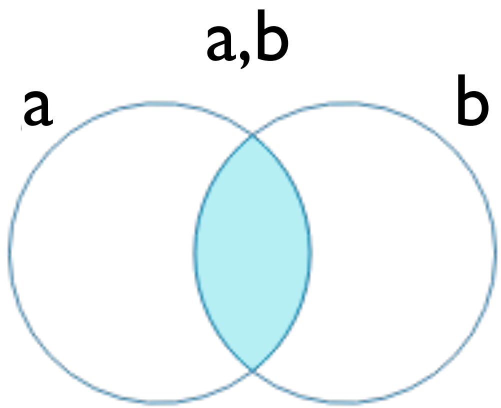


$$\Pr(a \mid b) = \frac{\Pr(a, b)}{\Pr(b)} \qquad \Pr(b \mid a) = \frac{\Pr(b, a)}{\Pr(a)}$$

$$\Pr(a \mid b)\ \Pr(b) = \Pr(a, b)$$

$$\Pr(b \mid a)\ \Pr(a) = \Pr(a, b)$$

Review: Probability Theory



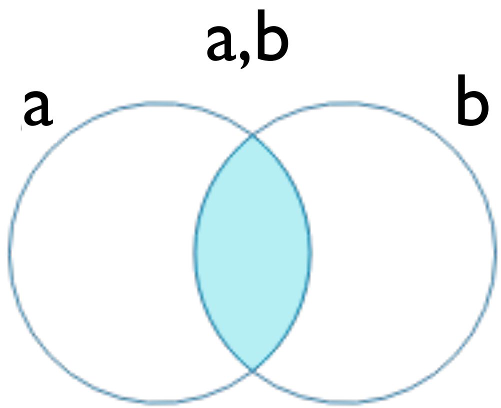
$$\Pr(a \mid b) = \frac{\Pr(a, b)}{\Pr(b)} \qquad \Pr(b \mid a) = \frac{\Pr(b, a)}{\Pr(a)}$$

$$\Pr(a \mid b)\ \Pr(b) = \Pr(a, b)$$

$$\Pr(b \mid a)\ \Pr(a) = \Pr(a, b)$$

$$\Pr(a \mid b)\ \Pr(b) = \Pr(b \mid a)\ \Pr(a)$$

Review: Probability Theory

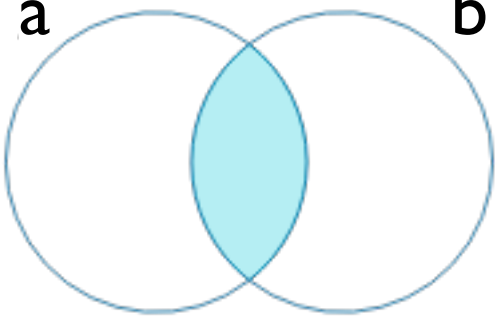


$$\Pr(a \mid b) = \frac{\Pr(a, b)}{\Pr(b)} \qquad \Pr(b \mid a) = \frac{\Pr(b, a)}{\Pr(a)}$$

$$\Pr(a \mid b)\ \Pr(b) = \Pr(b \mid a)\ \Pr(a)$$

$$\Pr(a \mid b) = \frac{\Pr(b \mid a)\ \Pr(a)}{\Pr(b)}$$

Review: Probability Theory



$$\Pr(a \mid b) = \frac{\Pr(a, b)}{\Pr(b)}$$

$$\Pr(b \mid a) = \frac{\Pr(b, a)}{\Pr(a)}$$

$$\Pr(a \mid b) = \frac{\Pr(b \mid a)\ \Pr(a)}{\Pr(b)}$$

Bayes Theorem

Machine Learning
CMPSCI 589

Bruno C. da Silva
bsilva@cs.umass.edu

Probabilistic Classifiers
Naive Bayes (1/2)

UMassAmherst
College of Information
& Computer Sciences