

COMP9418: Advanced Topics in Statistical Machine Learning

Propositional Logic

Instructor: Gustavo Batista

University of New South Wales

Introduction

- This lecture provides an introduction to propositional logic
 - Syntax and semantics of propositional logic
 - Monotonicity of propositional logic
- This topic will provide a basis to review probability calculus
 - As an extension of logic concepts
 - Allow to reason in the presence of uncertainty

Syntax of Propositional Sentences

- Consider an alarm for detecting burglaries
 - It can also be triggered by an earthquake
- An event of a burglary or earthquake can be expressed by the following sentence
 - *Burglary* and *Earthquake* are propositional variables
 - \vee is a logical disjunction (or)
- Propositional logic can be used to express more complex statements
 - \Rightarrow is the logical *implication*
 - It means that a burglary or an earthquake is guaranteed to trigger the alarm
- Consider also this sentence
 - \neg is a logical *negation* (not) and \wedge is the logical *conjunction* (and)
 - It means that if there is no burglary and no earthquake, the alarm will not trigger

Syntax of Propositional Sentences

- Consider an alarm for detecting burglaries
 - It can also be triggered by an earthquake
- An event of a burglary or earthquake can be expressed by the following sentence
 - *Burglary* and *Earthquake* are propositional variables
 - \vee is a logical *disjunction* (or)
- Propositional logic can be used to express more complex statements
 - \Rightarrow is the logical *implication*
 - It means that a burglary or an earthquake is guaranteed to trigger the alarm
- Consider also this sentence
 - \neg is a logical *negation* (not) and \wedge is the logical *conjunction* (and)
 - It means that if there is no burglary and no earthquake, the alarm will not trigger

$$Burglary \vee Earthquake$$

$$Burglary \vee Earthquake \Rightarrow Alarm$$

$$\neg Burglary \wedge \neg Earthquake \Rightarrow \neg Alarm$$

Syntax of Propositional Sentences

- Propositional sentences are formed using a set of propositional variables
 - These variables are also called *Boolean variables* or *binary variables*
 - Assume one of two possible values, typically indicated by *true* and *false*
- The simplest sentence has the form P_i
 - It is an *atomic sentence*
 - It means that the variable P_i has the value *true*
- More generally, propositional sentences are formed as follows
 - Every propositional variable P_i is a *sentence*
 - If α and β are sentences, then $\neg\alpha$, $\alpha \wedge \beta$, and $\alpha \vee \beta$ are also *sentences*

P_1, \dots, P_n

Syntax of Propositional Sentences

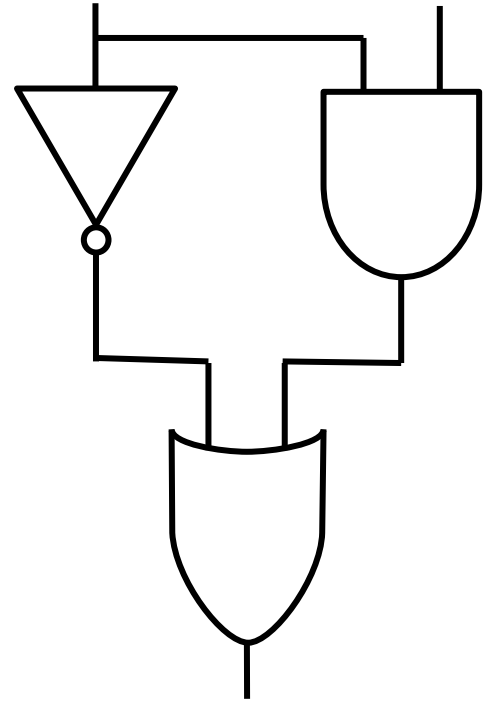
- The symbols \neg , \wedge and \vee are *logical connectives*
 - They stand for negation, conjunction and disjunction
- Other connectives can also be introduced
 - Such as implication (\Rightarrow) and equivalence (\Leftrightarrow)
 - But these can be defined in terms of the three primitive connectives
- A *propositional knowledge base* is a set of propositional sentences $\alpha_1, \alpha_2, \dots, \alpha_n$
 - Interpreted as a conjunction $\alpha_1 \wedge \alpha_2 \wedge \dots \wedge \alpha_n$

$$\alpha \Rightarrow \beta \equiv \neg \alpha \vee \beta$$

$$\alpha \Leftrightarrow \beta \equiv (\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$$

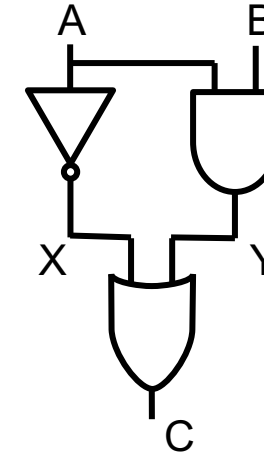
Syntax of Propositional Sentences

- Consider this digital circuit with two inputs and one output
 - Let's write a propositional knowledge base that captures the behavior of this circuit
 - We need to choose a set of propositional variables
 - Common choice is one variable to each wire



Syntax of Propositional Sentences

- Consider this digital circuit with two inputs and one output
 - Let's write a propositional knowledge base that captures the behavior of this circuit
 - We need to choose a set of propositional variables
 - Common choice is one variable to each wire
- The idea is that when the variable is true, the corresponding wire is high
 - Also, a when a variable is false, the wire is low
 - This leads to the following knowledge base



$$\Delta = \left\{ \begin{array}{ll} A & \Rightarrow \neg X \\ \neg A & \Rightarrow X \\ A \wedge B & \Rightarrow Y \\ \neg(A \wedge B) & \Rightarrow \neg Y \\ X \vee Y & \Rightarrow C \\ \neg(X \vee Y) & \Rightarrow \neg C \end{array} \right.$$

Semantics of Propositional Sentences

- Propositional logic provides a framework for defining
 - Properties of sentences such as consistency and validity
 - Relationships among them, such as implication, equivalence and mutual exclusiveness
- For instance, this sentence
 - Logically implies
- These properties and relationships are easy to figure out for simple sentences
 - $A \wedge \neg A$ is inconsistent (will never hold)
 - $A \vee \neg A$ is valid (always hold)
 - A and $(A \Rightarrow B)$ implies B
 - $A \vee B$ is equivalent to $B \vee A$

$$Burglary \vee Earthquake \Rightarrow Alarm$$

$$Burglary \Rightarrow Alarm$$

$$Alarm \wedge \neg Burglary \Rightarrow Earthquake$$

Semantics of Propositional Sentences

- Yet it may not be obvious that $A \Rightarrow B$ and $\neg B \Rightarrow A$ are equivalent
 - Or that $(A \Rightarrow B) \wedge (A \Rightarrow \neg B)$ implies $\neg A$
- For this reason we need formal definitions of logical properties and relationships
 - These notions are relatively easy ones the notion of world is defined

Worlds, Models and Events

- A *world* is a state in which the value of each variable is known
 - In the Burglary example, we have three variables and eight worlds
- A world w is a function that maps each propositional variable P_i into a value $w(P_i) \in \{true, false\}$
 - A world is also called a *truth assignment*, *variable assignment* or *variable instantiation*
- Worlds allow to decide the truth of sentences without ambiguity
 - *Burglary* is true at world w_1
 - $\neg \text{Burglary}$ is true at world w_3
 - $\text{Burglary} \vee \text{Earthquake}$ is true at world w_4

world	Earthquake	Burglary	Alarm
w_1	true	true	true
w_2	true	true	false
w_3	true	false	true
w_4	true	false	false
w_5	false	true	true
w_6	false	true	false
w_7	false	false	true
w_8	false	false	false

Worlds, Models and Events

- We use the notation $w \models \alpha$ to mean that the sentence α is true at world w
 - Also, world w *satisfies* (or *entails*) sentence α
- A set of worlds that satisfy a sentence α is called the *models of α*
 - Every sentence α can be viewed as representing a set of worlds $Mods(\alpha)$, which is called the *event* denoted by α
 - We use the terms “sentence” and “events” interchangeably
- We can prove the following properties
 - $Mods(\alpha \wedge \beta) = Mods(\alpha) \cap Mods(\beta)$
 - $Mods(\alpha \vee \beta) = Mods(\alpha) \cup Mods(\beta)$
 - $Mods(\neg \alpha) = \overline{Mods(\alpha)}$

$$Mods(\alpha) \stackrel{\text{def}}{=} \{w : w \models \alpha\}$$

Worlds, Models and Events

- Some example sentences and their truth at worlds
 - $Earthquake$ is true at worlds w_1, \dots, w_4
 $Mods(Earthquake) = \{w_1, \dots, w_4\}$
 - $\neg Earthquake$ is true at worlds w_5, \dots, w_8
 $Mods(\neg Earthquake) = \overline{Mods(Earthquake)}$
 - $\neg Burglary$ is true at worlds w_3, w_4, w_7, w_8
 - $Alarm$ is true at worlds w_1, w_3, w_5, w_7
 - $\neg(Earthquake \vee Burglary)$ is true at worlds w_7, w_8
 $Mods(\neg(Earthquake \vee Burglary)) = \overline{Mods(Earthquake) \cup Mods(Burglary)}$
 - $\neg(Earthquake \vee Burglary) \vee Alarm$ is true at worlds w_1, w_3, w_5, w_7, w_8
 - $(Earthquake \vee Burglary) \Rightarrow Alarm$ is true at worlds w_1, w_3, w_5, w_7, w_8
 - $\neg Burglary \wedge Burglary$ is not true at any world

world	Earthquake	Burglary	Alarm
w_1	true	true	true
w_2	true	true	false
w_3	true	false	true
w_4	true	false	false
w_5	false	true	true
w_6	false	true	false
w_7	false	false	true
w_8	false	false	false

Logical Properties

- We say that a sentence α is *consistent* if and only if there is at least one world w at which α is true
 - Otherwise, the sentence α is *inconsistent*
 - We also use the terms *satisfiable/unsatisfiable*
 - The symbol *false* is used to denote a sentence that is unsatisfiable
- A sentence α is *valid* if and only if it is true at every world
 - If a sentence α is not valid, we can identify a world w at which α is false
 - The symbol *true* is used to denote a sentence that is valid. We also write $\models \alpha$

$$Mods(\alpha) \neq \emptyset$$

$$Mods(\alpha) = \emptyset$$

$$Mods(\alpha) = \Omega$$

$$Mods(\alpha) \neq \Omega$$

Logical Relationships

- Logical properties apply to single sentences. Logical relationships apply to two or more sentences
 - Sentences α and β are *equivalent* iff they are true at the same set of worlds
 - Sentences α and β are *mutually exclusive* iff they are never true at the same world
 - Sentences α and β are *exhaustive* iff each world satisfies at least one of the sentences
 - Sentence α *implies* sentence β iff β is true whenever α is true
- Symbol \models is also used to indicate implication between sentences
 - α implies or *entails* β

$$Mods(\alpha) = Mods(\beta)$$

$$Mods(\alpha) \cap Mods(\beta) = \emptyset$$

$$Mods(\alpha) \cup Mods(\beta) = \Omega$$

$$Mods(\alpha) \subseteq Mods(\beta)$$

$$\alpha \models \beta$$

Equivalences

- These equivalences are useful when working with propositional logic
 - They are defined for *schemas*, which are templates
 - For instance $\alpha \Rightarrow \beta$ is a schema for $\neg A \Rightarrow (B \vee \neg C)$

Schema	Equivalent Schema	Name
$\neg true$	$false$	
$\neg false$	$true$	
$false \wedge \beta$	$false$	
$\alpha \wedge true$	α	
$false \vee \beta$	β	
$\alpha \vee true$	$true$	
$\neg \neg \alpha$	α	Double negation
$\neg(\alpha \wedge \beta)$	$\neg \alpha \vee \neg \beta$	de Morgan
$\neg(\alpha \vee \beta)$	$\neg \alpha \wedge \neg \beta$	de Morgan
$\alpha \vee (\beta \wedge \gamma)$	$(\alpha \vee \beta) \wedge (\alpha \vee \gamma)$	distribution
$\alpha \wedge (\beta \vee \gamma)$	$(\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$	distribution
$\alpha \Rightarrow \beta$	$\neg \beta \Rightarrow \neg \alpha$	contraposition
$\alpha \Rightarrow \beta$	$\neg \alpha \vee \beta$	definition of \Rightarrow
$\alpha \Leftrightarrow \beta$	$(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$	definition of \Leftrightarrow

Reductions

- We can also state several reductions between logical properties and relationships
 - This table shows how the relationships of implication, equivalence, mutual exclusiveness and exhaustiveness can be defined in terms of satisfiability and validity

Relationship	Property
α implies β	$\alpha \wedge \neg\beta$ is unsatisfiable
α implies β	$\alpha \Rightarrow \beta$ is valid
α and β are equivalent	$\alpha \Leftrightarrow \beta$ is valid
α and β are mutually exclusive	$\alpha \wedge \beta$ is unsatisfiable
α and β are exhaustive	$\alpha \vee \beta$ is valid

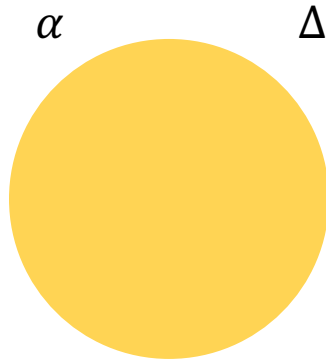
Monotonicity

- Consider the earthquake-burglary-alarm example
 - Suppose we introduce the sentence
 $\alpha: (Earthquake \vee Burglary) \Rightarrow Alarm$
- α makes some of the worlds impossible
 - Changing our state of belief
 - $Mods(\alpha) = \{w_1, w_3, w_5, w_7, w_8\}$

world	Earthquake	Burglary	Alarm	α
w_1	true	true	true	true
w_2	true	true	false	false
w_3	true	false	true	true
w_4	true	false	false	false
w_5	false	true	true	true
w_6	false	true	false	false
w_7	false	false	true	true
w_8	false	false	false	true

Monotonicity

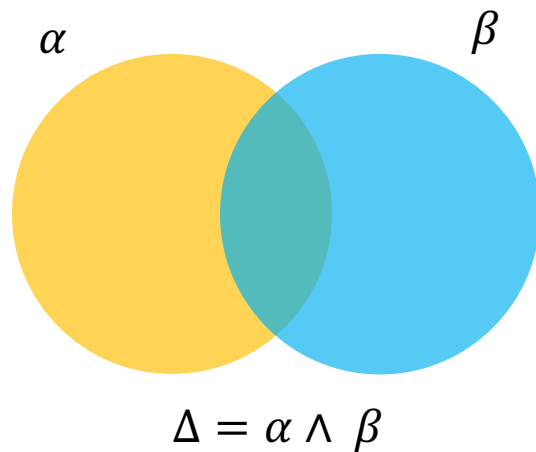
- Suppose we learn
 - $\beta: \text{Earthquake} \Rightarrow \text{Burglary}$
 - $\text{Mods}(\beta) = \{w_1, w_2, w_5, w_6, w_7, w_8\}$
- Our state of belief rules out w_3
 - $\text{Mods}(\alpha \wedge \beta) = \text{Mods}(\alpha) \cap \text{Mods}(\beta)$



world	Earthquake	Burglary	Alarm	α	β
w_1	true	true	true	true	true
w_2	true	true	false	false	true
w_3	true	false	true	true	false
w_4	true	false	false	false	false
w_5	false	true	true	true	true
w_6	false	true	false	false	true
w_7	false	false	true	true	true
w_8	false	false	false	true	true

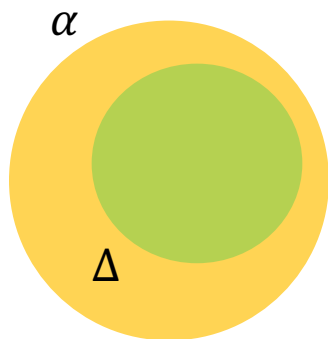
Monotonicity

- Suppose we learn
 - $\beta: \text{Earthquake} \Rightarrow \text{Burglary}$
 - $\text{Mods}(\beta) = \{w_1, w_2, w_5, w_6, w_7, w_8\}$
- Our state of belief rules out w_3
 - $\text{Mods}(\alpha \wedge \beta) = \text{Mods}(\alpha) \cap \text{Mods}(\beta)$

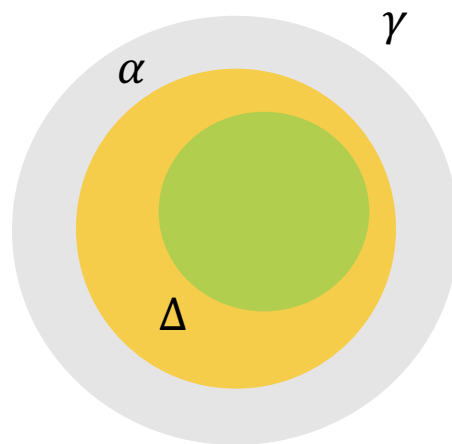


world	Earthquake	Burglary	Alarm	α	β
w_1	true	true	true	true	true
w_2	true	true	false	false	true
w_3	true	false	true	true	false
w_4	true	false	false	false	false
w_5	false	true	true	true	true
w_6	false	true	false	false	true
w_7	false	false	true	true	true
w_8	false	false	false	true	true

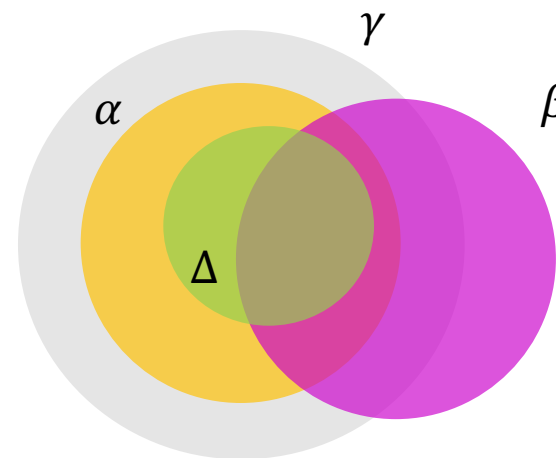
Monotonicity



$$\Delta \models \alpha$$



$$\begin{aligned}\alpha &\models \gamma \\ \Delta &\models \gamma\end{aligned}$$



$$\begin{aligned}\Delta &\models \alpha \wedge \beta \\ \Delta &\models \gamma\end{aligned}$$

Multivalued Variables

- Propositional variables are binary
 - As they assume the values *true* or *false*
 - These values are implicit in the syntax as we use X to mean $X = \text{true}$ and $\neg X$ to mean $X = \text{false}$
- We can generalise propositional logic to allow multivalued variables
 - For instance, an alarm that triggers high or low
 - We will need to explicate the values assigned
$$\text{Burglary} \Rightarrow \text{Alarm} = \text{high}$$
$$\text{Burglary} = \text{true} \Rightarrow \text{Alarm} = \text{high}$$

world	Earthquake	Burglary	Alarm
w_1	true	true	high
w_2	true	true	low
w_3	true	true	off
w_4	true	false	high
w_5	true	false	low
w_6	true	false	off
w_7	false	true	high
w_8	false	true	low
w_9	false	true	off
w_{10}	false	false	high
w_{11}	false	false	low
w_{12}	false	false	off

Multivalued Variables

- Sentences in the generalized logic respect the following rules
 - Every propositional variable is a sentence
 - $V = v$ is a sentence, where V is a variable and v is one of its values
 - If α and β are sentences, then $\neg\alpha$, $\alpha \wedge \beta$, and $\alpha \vee \beta$ are also sentences
- The semantics of generalized logic is similar to the standard logic
 - For example, the sentence
$$Earthquake \wedge Alarm = off$$
rules out all worlds but w_3 and w_6

world	Earthquake	Burglary	Alarm
w_1	true	true	high
w_2	true	true	low
w_3	true	true	off
w_4	true	false	high
w_5	true	false	low
w_6	true	false	off
w_7	false	true	high
w_8	false	true	low
w_9	false	true	off
w_{10}	false	false	high
w_{11}	false	false	low
w_{12}	false	false	off

Variable Instantiation

- An *instantiation* of variables is a propositional sentence if the form
 - For variables A, B and C and values a, b and c
- We simplify the notation by
 - Use simply a instead of $A = a$
 - Replace the operator \wedge by a comma
- A *trivial instantiation* is an instantiation to an empty set of variables
- We will denote variables by upper-case letters (A)
 - Values by lower-case letter (a)
 - Cardinalities (number of values) by $|A|$

$$(A = a) \wedge (B = b) \wedge (C = c)$$

$$a, b, c$$

$$\top$$

Variable Instantiation

- Sets of variables will be denoted by bold-face upper-case letters (**A**)
 - Their instantiations by bold-face lower-case (**a**)
 - Number of instantiations by $A^\#$
- For a Boolean variable A
 - a denotes $A = \text{true}$
 - \bar{a} denotes $A = \text{false}$
- We also use $x \sim y$ to denote x and y are compatible instantiations
 - They agree on the value of all common variables

$$A = \{a_1, a_2, a_3\}$$

$$B = \{b_1, b_2\}$$

$$C = \{c_1, c_2\}$$

$$\mathbf{D} = \{A, B, C\}$$

$$\mathbf{D}^\# = 12$$

$$\{a, b, \bar{c}\} \sim \{b, \bar{c}, \bar{d}\}$$

$\{a, b, \bar{c}\}$ and $\{b, c, \bar{d}\}$ are not compatible since they disagree on C

Conclusion

- Logic is a reasoning framework widely used in AI
 - Logic limitations include the monotonicity property and lack of support to uncertain events
 - This framework has been extended overcome this limitations, such as fuzzy logic and other ad-hoc methods (certainty factors and pseudoprobabilities)
 - This the next lecture, we will extend this framework to handle uncertainly through probabilistic reasoning
- Tasks
 - Read Chapter 2, but Section 2.7 from the textbook (Darwiche)