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
  - $\lor$ 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 $\land$ 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)

$$Burglary \vee Earthquake$$

- 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

$$Burglary \vee Earthquake \Rightarrow Alarm$$

- 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

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

Syntax of Propositional Sentences

- Propositional sentences are formed using a set of propositional variables $P_1, \dots, P_n$
  - 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*

Syntax of Propositional Sentences

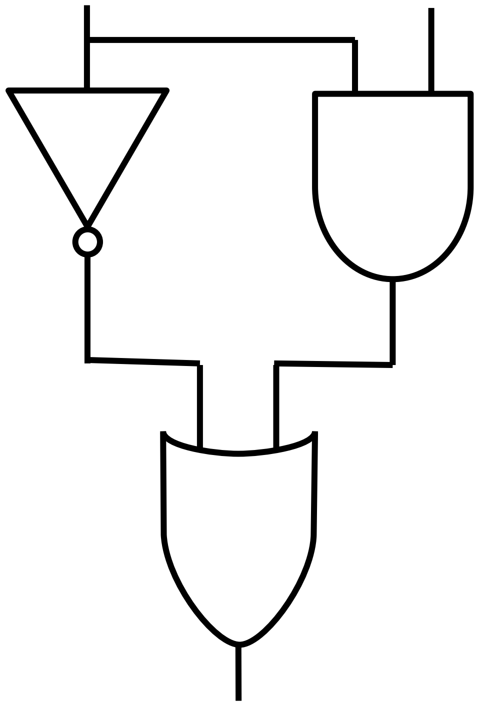
- The symbols $\neg$, $\wedge$ and $\vee$ are *logical connectives*
  - They stand for negation, conjunction and disjunction
- Other connectives can also be introduced
  - Such as implication ($\Longrightarrow$) and equivalence ($\Longleftrightarrow$)
  - But these can be defined in terms of the three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

$$\alpha \Longrightarrow \beta \equiv \neg\alpha \vee \beta$$

$$\alpha \Longleftrightarrow \beta \equiv (\alpha \Longrightarrow \beta) \wedge (\beta \Longrightarrow \alpha)$$

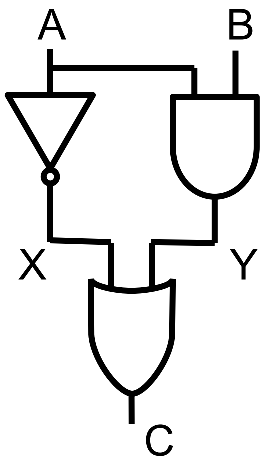
Syntax of Propositional Sentences

- 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



$$\Delta = \begin{cases} A & \Rightarrow \neg X \\ \neg A & \Rightarrow X \\ A \wedge B & \Rightarrow Y \\ \neg(A \wedge B) & \Rightarrow \neg Y \\ X \vee Y & \Rightarrow C \\ \neg(X \vee Y) & \Rightarrow \neg C \end{cases}$$

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
  - $A \land \neg A$ is inconsistent (will never hold)
  - $A \lor \neg A$ is valid (always hold)
  - A and $(A \Longrightarrow B)$ implies B
  - $A \lor B$ is equivalent to $B \lor A$

$$Burglary \lor Earthquake \Longrightarrow Alarm$$

$$Burglary \Longrightarrow Alarm$$

$$Alarm \land \neg Burglary \Longrightarrow Earthquake$$

Semantics of Propositional Sentences

- Yet it may not be obvious that $A \Longrightarrow B$ and $\neg B \Longrightarrow \neg A$ are equivalent
  - Or that $(A \Longrightarrow B) \wedge (A \Longrightarrow \neg B)$ 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
  - $\neg Burglary$ is true at world w_3
  - $Burglary \vee Earthquake$ 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 $w \vDash \alpha$ 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
  - $Mods(\alpha \wedge \beta) \;=\; Mods(\alpha) \cap Mods(\beta)$
  - $Mods(\alpha \vee \beta) = \; Mods(\alpha) \cup Mods(\beta)$
  - $Mods(\neg\alpha) \;= \overline{Mods(\alpha)}$

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

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
  - $\neg(Earthquake \lor Burglary)$ is true at worlds w_7, w_8
    $$Mods(\neg(Earthquake \lor Burglary)) = \overline{Mods(Earthquake) \cup Mods(Burglary)}$$
  - $\neg(Earthquake \lor Burglary) \lor Alarm$ is true at worlds w_1, w_3, w_5, w_7, w_8
  - $(Earthquake \lor Burglary) \implies Alarm$ is true at worlds w_1, w_3, w_5, w_7, w_8
  - $\neg Burglary \land Burglary$ 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

$Mods(\alpha) \neq \emptyset$

$Mods(\alpha) = \emptyset$

- 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 $\vDash \alpha$

$Mods(\alpha) = \Omega$

$Mods(\alpha) \neq \Omega$

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 $\vDash$ is also used to indicate implication between sentences
  - α implies or *entails* β

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

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

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

$Mods(\alpha) \subseteq \mathrm{Mods}(\beta)$

$\alpha \vDash \beta$

Equivalences

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

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
$\alpha \Longrightarrow \beta$	$\neg\beta \Longrightarrow \neg\alpha$	contraposition
$\alpha \Longrightarrow \beta$	$\neg\alpha \vee \beta$	definition of $\Longrightarrow$
$\alpha \Leftrightarrow \beta$	$(\alpha \Longrightarrow \beta) \wedge (\beta \Longrightarrow \alpha)$	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	$\alpha \land \neg\beta$ is unsatisfiable
α implies β	$\alpha \implies \beta$ is valid
α and β are equivalent	$\alpha \iff \beta$ is valid
α and β are mutually exclusive	$\alpha \land \beta$ is unsatisfiable
α and β are exhaustive	$\alpha \lor \beta$ is valid

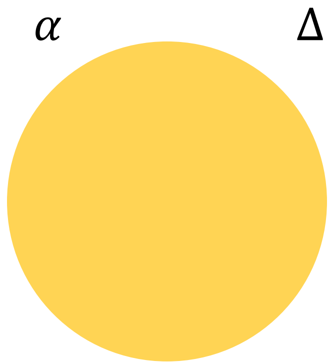
Monotonicity

- Consider the earthquake-burglary-alarm example
  - Suppose we introduce the sentence $\alpha: (Earthquake \vee Burglary) \Longrightarrow Alarm$
- 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
$\boldsymbol{w_1}$	true	true	true	true
w_2	true	true	false	false
$\boldsymbol{w_3}$	true	false	true	true
w_4	true	false	false	false
$\boldsymbol{w_5}$	false	true	true	true
w_6	false	true	false	false
$\boldsymbol{w_7}$	false	false	true	true
$\boldsymbol{w_8}$	false	false	false	true

Monotonicity

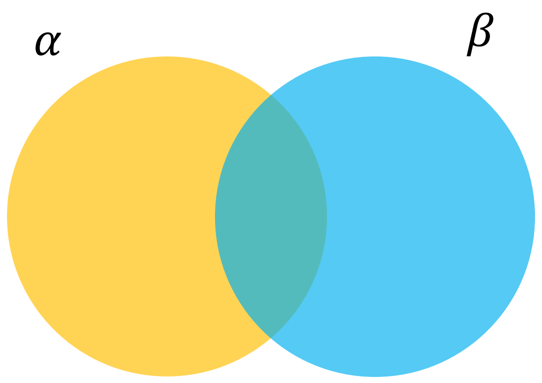
- Suppose we learn
  - $\beta: Earthquake \Longrightarrow Burglary$
  - $Mods(\beta) = \{w_1, w_2, w_5, w_6, w_7, w_8\}$
- Our state of belief rules out w_3
  - $Mods(\alpha \wedge \beta) = Mods(\alpha) \cap Mods(\beta)$



world	Earthquake	Burglary	Alarm	α	β
$\boldsymbol{w_1}$	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
$\boldsymbol{w_5}$	false	true	true	true	true
w_6	false	true	false	false	true
$\boldsymbol{w_7}$	false	false	true	true	true
$\boldsymbol{w_8}$	false	false	false	true	true

Monotonicity

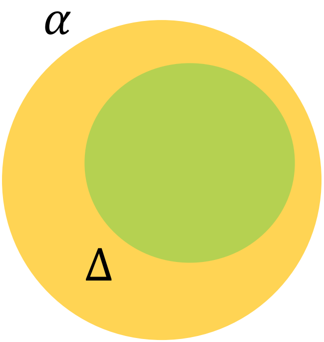
- Suppose we learn
  - $\beta: Earthquake \Longrightarrow Burglary$
  - $Mods(\beta) = \{w_1, w_2, w_5, w_6, w_7, w_8\}$
- Our state of belief rules out w_3
  - $Mods(\alpha \wedge \beta) = Mods(\alpha) \cap Mods(\beta)$



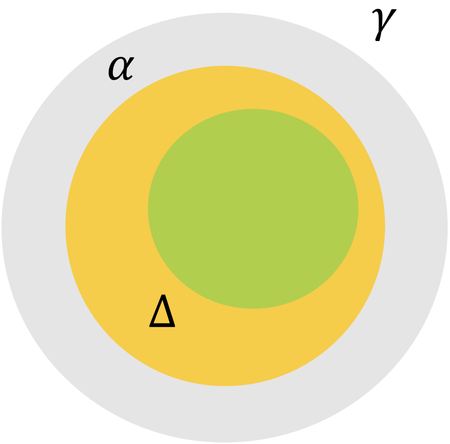
$\Delta = \alpha \wedge \beta$

world	Earthquake	Burglary	Alarm	α	β
$\boldsymbol{w_1}$	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
$\boldsymbol{w_5}$	false	true	true	true	true
w_6	false	true	false	false	true
$\boldsymbol{w_7}$	false	false	true	true	true
$\boldsymbol{w_8}$	false	false	false	true	true

Monotonicity

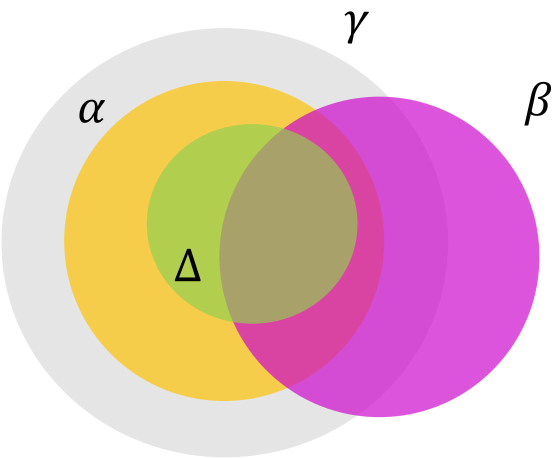


$\Delta \vDash \alpha$



$\alpha \vDash \gamma$

$\Delta \vDash \gamma$



$\Delta \vDash \alpha \wedge \beta$

$\Delta \vDash \gamma$

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 $X = true$ and $\neg X$ to mean $X = false$
- 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

$$Burglary \Longrightarrow Alarm = high$$
$$Burglary = true \Longrightarrow Alarm = high$$

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, them $\neg\alpha$, $\alpha \wedge \beta$, 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

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

- We simplify the notation by
  - Use simply a instead of $A = a$
  - Replace the operator $\wedge$ by a comma

$$a, b, c$$

- A *trivial instantiation* is an instantiation to an empty set of variables

$$\top$$

- We will denote variables by upper-case letters (A)
  - Values by lower-case letter (a)
  - Cardinalities (number of values) by $|A|$

Variable Instantiation

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

$A = \{a_1, a_2, a_3\}$
$B = \{b_1, b_2\}$
$C = \{c_1, c_2\}$
$\boldsymbol{D} = \{A, B, C\}$
$\boldsymbol{D}^{\#} = 12$

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