이산수학

휴먼지능정보공학전공

학습 내용

- 명제
- 논리연산자
- 합성명제
- 명제함축(함축명제)
- 논리적동치
- 한정기호
- 명제함수
- 추론
- 파이썬코딩

명제

- 논리의 기본 구성 요소로써 참(True)이나 거짓(False)으로 구분할 수 있는 문장이나 수식을 의미
  - 영문자 p, q, r, …으로 표현
  - 명제는 T와 F의 2가지의 진리 값만을 가지므로 이진 논리
  - 문장 중 참이나 거짓으로 구분하기 어려운 기원문, 감탄문, 명령문 등은 명제로 활용하지 않음
- 하나의 문장이라 식으로 구성되어 있는 명제를 단순명제라고 하고, 여러 개의 단순 명제들이 논리 연산들과 결합되어 만들어진 명제를 합성명제라고 함
  - 논리 연산자로는 단항 연산자인 부정 연산자가 있고 이항 연산자인 AND(그리고), OR(또는) 등으로 구성

명제

- 명제 진리값
  - 명제의 참(True)이나 거짓(False)을 가리키는 값을 진릿값(Truth Value) 이라고 하고 T, F 또는 0, 1로 표현
  - 명제를 구성하는 단순 명제의 진리 값과 논리 연산자의 특성에 따라 값이 정해짐
- 단순 명제의 진리 값은 그 명제가 참이냐 거짓이냐에 따라 T 또는 F로 표시
  - 참(T), 거짓(F)
- 합성 명제의 진리 값은 복잡하게 도출되기 때문에 진리 표(truth table)를 사용하여 단계적으로 연 산하거나 논리법칙을 사용함으로써 원하는 합성 명제의 진리 값을 구할 수 있다.

논리연산자

- 단순명제들을 연결시켜 주는 역할을 하는 ∨, ∧, ~과 같은 논리연산자라고 함

논리연산자

- 부정(Negation)연산자 NOT
  - 명제 p 가 명제일 때 “p 가 아니다”도 명제가 됨
  - ¬ p 또는 ~ p로 표현

논리연산자

- 논리합(Disjunction) 연산자 OR
  - p, q 가 명제 일 때 p, q 의 진릿값이 모두 거짓(F)일 때만 거짓(F)이 되고, 그렇지 않으면 참(T)이 되는 명제이다. $p \vee q$로 표현

논리연산자

- 논리곱(Conjunction) 연산자 AND
  - 문장 p, q 가 명제 일 때 p, q 의 진릿값이 모두 참(T)일 때만 참(T)이 되고, 그렇지 않을 때는 거짓(F)이 되는 명제이다. $p \wedge q$ 로 표현

논리연산자

- 배타적 논리합(Exclusive OR) 연산자 XOR
  - 문장 p, q 가 명제일 때 p, q 의 두 진리값 중 하나만 참(T)일 때 참(T)이 되고, 그렇지 않으면 거짓(F)이 되는 명제이다. p ⊕ q 표현

합성명제

- 합성명제
  - 합성명제(Compound Proposition)는 하나 이상의 명제들이 결합되어 만들어진 명제
  - AND, OR, NOT 등의 논리 연산자(Logical Operators)를 이용해 명제를 결합하여 사용

합성명제

- 항진명제(Tautology)
  - 항진명제는 합성명제를 구성하는 명제의 진릿값에 상관없이 합성명제의 진릿값이 항 상 참(T)인 명제

합성명제

- 모순명제(Contradiction)
  - 모순명제는 합성명제를 구성하는 명제들의 진릿값에 상관없이 합성명제의 진릿값이 항상 거짓(F)인 명제

합성명제

- 사건명제
  - 사건명제(Contingency)는 항진명제도, 모순명제도 아닌 명제

합성명제

- 사건명제
  - 사건명제(Contingency)는 항진명제도, 모순명제도 아닌 명제

합성명제

- 합성명제 진리표
  - ~(p ^ q) v (r v q)

명제함축

- 명제함축
  - 명제의 함축은 문장 p, q가 명제일 때, 명제 p가 조건 또는 원인으로 제시되고, 명제 q 가 결론 또 는 결과로 제시되는 명제로써 p→q (p : 조건, 원인, q : 결론, 결과)로 표현
    - 명제의 함축은 p는 q를 함축한다로 읽거나 p 이면 q 이다 라고 읽음

명제함축

- 쌍방조건명제
  - 쌍방조건명제는 명제 p, q 가 명제일 때, 명제 p 와 q가 모두 조건이면서 결론인 명제로써 p↔q (p : 조건, 원인, q : 결론, 결과)로 표현
  - 쌍방쌍방조건명제는 p면 q 고, q면 p 다 라고 읽음

명제 역,이,대우

- 명제의 역, 이, 대우
  - 함축명제에 대해 역, 이, 대우를 구할 수 있음
  - 함축 $p \rightarrow q$의 역은 $q \rightarrow p$, 이는 $\neg p \rightarrow \neg q$, 대우는 $\neg q \rightarrow \neg p$로 표현
  - 주어진 명제만으로 증명하기 어려운 경우 역, 이, 대우 등을 활용하여 증명할 수 있음

명제 역,이,대우

- 명제의 역, 이, 대우
    - 오늘 눈이 오면 나는 커피를 마신다

논리적 동치

- 합성명제 p 와 q 의 진릿값이 서로 같은 경우를 의미하며 p ≡ q 로 표현
  - 논리적 동치를 이용하여 논리를 단순화 할 수 있음
- 합성명제의 진리표를 이용한 합성명제의 진릿값을 구하거나 논리동치법칙을 이용하여 합성명제의 논리적 동치를 확인할 수 있음
  - 합성명제의 논리적 동치를 이용하면 합성명제의 검증과정에서 시간과 비용을 줄일수 있음

논리적 동치

- 합성명제의 진리표를 이용한 합성명제의 진릿값을 구하거나 논리동치법칙을 이용하여 합성명제의 논리적 동치를 확인할 수 있음
  - 명제 $p \rightarrow q$ 와 $\neg p \vee q$ 는 어떤 관계

논리적 동치

- 합성명제의 진리표를 이용한 합성명제의 진릿값을 구하거나 논리동치법칙을 이용하여 합성명제의 논리적 동치를 확인할 수 있음
  - 명제 $p \rightarrow q$ 와 $\neg p \vee q$ 는 어떤 관계

논리적 동치

- 합성명제의 진리표를 이용한 합성명제의 진릿값을 구하거나 논리동치법칙을 이용하여 합성명제의 논리적 동치를 확인할 수 있음
  - $\neg(p \vee (\neg p \wedge q))$ 와 $\neg p \wedge \neg q$
  - $\neg(p \vee (\neg p \wedge q)) \equiv \neg(p \wedge \neg(\neg p \wedge q))$ 드모르간의 법칙
  - $\equiv \neg(p \wedge (\neg(\neg p) \vee \neg q))$ 드모르간의 법칙
  - $\equiv \neg(p \wedge p \vee \neg q))$ 이중 부정법칙
  - $\equiv (\neg p \wedge p) \vee (\neg p \wedge \neg q)$ 분배법칙
  - $\equiv F \vee (\neg p \wedge \neg q)$ 부정법칙
  - $\equiv (\neg p \wedge \neg q)$ 항등법칙

한정기호

- 명제논리
  - 명제 논리(Propositional Logic)는 주어와 술어를 구분하지 않고 전체를 하나의 식으로 처리하여 참 또는 거짓을 판별하는 법칙
  - 논리식을 이용해 명제를 기술함
  - 명제 또는 문장들 간의 논리적 관계만을 다룸

한정기호

- 술어논리
  - 술어 논리(Predicate Logic)는 주어와 술어로 구분하여 참 또는 거짓에 관한 법칙임
  - 명제에 ' 주어 ' 와 ' 술어 ' 의 구조가 존재하고, ' 주어 ' 가 될 수 있는 대상에 대한 한정 기호를 사용 할 수 있는 논리임
  - 하나의 술어는 하나 이상의 객체를 수식할 수 있으며 또한 객체는 상수가 사용될 수도 있고 변수가 사용 될 수도 있음
    - 소크라테스는 사람이다 명제에서 사람이다는 술어가 되고 소크라테스는 객체(상수)가 됨
  - 변수 x가 나타내는 객체의 집합 D를 정의역(domain)이라 함
  - p(x)를 변수 x에 대한 명제 술어라고 함
  - 명제 논리와 구분하여 명제 술어에 대한 논리를 술어 논리라고 함

한정기호

- 술어논리와 한정자
  - 명제 중에는 변수의 값에 따라 그 명제가 참이 되거나 거짓이 될 수 있는데 이러한 형태의 명제를 p(x)를 변수 x에 대한 명제 술어라고 하고 명제 술어에 대한 논리를 술어 논리라고 함
  - 명제 술어를 나타내는 방법 중에서 변수의 범위를 한정시키는 한정자를 사용할 수 있는데 한정자 에는'모든 것에 대하여(for all)'와'존재한다(there exist)'의 두 가지가 있음
    - 모든 것에 대하여는 기호 ∀를 사용하고 존재한다는 기호 ∃로 표현함
    - 전칭기호 ∀는 전체한정자(Universal Quantifier)라고 하며 논의영역에 속하는 모든 값을 의미함
    - 논의영역 U에 속하는 모든 x에 대해 명제 P(x) 는 참이 됨 : $\forall x P(x)$
    - 존재기호 ∃는 존재한정자(Existential Quantified)라고 하며 논의영역에 속하는 어떤 값을 의미함
    - 논의영역 U에 속하는 어떤 x 에 대해 명제 P(x)는 참이 됨: ∃xP(x

한정기호

- (1) $\neg(\forall P(x))$ (2) $\exists x(\neg P(x))$ (3) $\exists x(\forall y P(x,y))$ (4) $\forall x \forall y P(x,y)$
  - (1) $\neg(\forall x P(x))$: 모든 x에 대해 P(x)를 만족하지 않는다.
  - (2) $\exists x(\neg P(x))$: P(x) 가 성립하지 않는 어떤 x가 존재한다.
  - (3) $\exists x(\forall y P(x,y))$: 모든 y에 대해 P(x,y)를 만족하는 어떤 x가 존재한다.
  - (4) $\forall x \forall y P(x,y)$: 모든 x에 대해 모든 y가 P(x,y)를 만족한다

한정기호

- 논의영역 D가 D = (x | 0< x ≦3, x는 양의 정수)고 명제함수 P(x)가 x 3 < 16 일 때 다음 명제함수의 진리값
  - (1) ∀xP(x) ▪ (2) ∃xP(x)
    - p(1) = 1< 16 (거짓)
    - (1) ∀xP(x) : 거짓
    - (2) ∃xP(x) : 참

명제함수

- 명제함수
  - 명제함수(Propositional Function) P(x)는 논의영역 D에 속하는 변수 x를 포함하는 문장임
  - 변수를 포함한 문장을 명제함수라고 하는데 변수가 포함된 문자이라도 변수의 값이나 범위가 주어져서 참, 거짓을 판별할수 있으므로 명제가 성립됨
  - 논의영역(Universe of Discourse)은 명제에 포함된 x가 속하게 될 범위를 의미함

명제함수

- (1) P(x, y)가 y - 11 = -(x2+2x+1) 일 때 P(2, 2) 진리값
  - (-2)-11 = - (4 + 4 +1)
  - -9 = -9
  - ∴ 참

명제함수

- (2) $Q(x,y)$: $x = 2y$ 일 때, $Q(1,2)$과 $Q(2,1)$ 의 진리값
  - $Q(1,2)$: $1 \neq 2\times2 = 4$ ∴ 거짓
  - $Q(2,1)$: $2 = 2\times1 = 2$ ∴ 참

명제함수

- $P(x,y)$ 가 $x^2 = y^2$ 일 때 다음 명제의 진릿값
  - (1) $\forall x \forall y P(x,y)$ (2) $\exists x \forall y P(x,y)$
    - (1) 모든 x 에 대해 모든 y가 $P(x,y)$를 만족해야 $\forall x \forall y P(x,y)$가 참이 됨. 그러나 $|x| \geq |y|$ 인 경우, $x = 4$, $y = 1$과 같은 경우는 성립하지 않으므로 $\forall x \forall y P(x,y)$ 는 거짓
    - (2) 모든 y 에 대해 $P(x,y)$를 만족하는 x가 하나라도 존재하면 $\exists x \forall y P(x,y)$ 는 참이 됨. 그러나 $|x| < |y|$ 인 경우에만 $P(x,y)$를 만족하므로 모든 y 에 대해 만족하는 x 가 존재할 수 없음. 그러므로 $\exists x \forall y P(x,y)$ 는 거짓

추론

- 추론
  - 추론은 어떤 명제가 참인 것을 근거로 하여 다른 명제가 참임을 유도하는 방식임
  - 근거가 되는 명제가 가정 또는 전제(Hypothesis)가 되고, 유도되는 명제가 결론이 됨
    - 주어진 전제가 참이고 결론도 참인 추론을 유효 추론이라 함
    - 주어진 전제가 참이나 추론의 결론이 거짓이 되는 추론을 허위 추론이라 함

추론

- 유효추론
  - 주어진 전제가 참이고 결론도 참인 추론을 유효 추론이라 함
    - 명제 p : “여름은 덥다”
    - 명제 q : “겨울은 춥다”
    - 여름이 더우면 겨울은 춥다.
    - 여름은 덥다.
    - ∴ 겨울은 춥다.
    - $p \rightarrow q$
    - P
    - ∴ q

전제	결론	전제
p	q	p -->q
T	T	T
T	F	F
F	T	T
F	F	T

추론

- 유효추론
  - (1) p∨(q∨r)
  - ¬ r
  - ∴ p∨q

p	q	r	$q \vee r$	전제		결론
				$p \vee (q \vee r)$	$\neg r$	$p \vee q$
T	T	T	T	T	F	T
T	T	F	T	T	T	T
T	F	T	T	T	F	T
T	F	F	F	T	T	T
F	T	T	T	T	F	T
F	T	F	T	T	T	T
F	F	T	T	T	F	F
F	F	F	F	F	T	F

추론

- 유효추론
  - 추론법칙

법칙	추론법칙	항진명제
논리곱	p q ∴ p ^ q	없음
선언적부가	p ∴ p v q	p → (p v q)
단순화	p ^ q ∴ p	(p ^ q) → p
긍정논법	p p → q ∴ q	(p ^ (p → q)) → q
부정논법	p q ∴ p ^ q	(~q ^ (p → q)) → ~p
선언적 삼단논법	p q ∴ p ^ q	(p v q) ^ ~p) → q
가설적 삼단논법	p q ∴ p ^ q	((p → q) ^ (q → r)) → (p→r)

추론

- (1) 긍정논법
  - 일호가 수학을 공부하면, 이호는 영어를 공부한다.
  - 일호가 수학을 공부한다.
  - ∴ ______________________________

    - (1) p : 일호는 수학을 공부한다.
    - q : 이호는 영어를 공부한다.
    - 전제는 p → q와 q로 표현될 수 있음
    - 긍정논법이므로 결론은 q다.
    - 그러므로 "이호는 영어를 공부한다"가 됨

추론

- 다음과 같이 전제로 주어진 명제가 항상 참이라고 할 때, 휴대폰이 어디에 있는지 찾아보세요
  - (a) 휴대폰이 서랍에 있었다면, 출근할 때 휴대폰을 보았다.
  - (b) 내가 아침을 먹었다면, 휴대폰은 서랍에 있다.
  - (c) 나는 샤워를 했거나 아침을 먹었다.
  - (d) 내가 소파에 앉아 있었다면,휴대폰 옷 주머니 속에 있다.
  - (e) 내가 샤워를 했다면, 내 휴대폰은 가방 속에 있다.
  - (f) 내가 출근할 때, 나는 휴대폰을 보지 못했다

추론

- def ch02_exam1():

```
def ch02_exam1():
        bool_p = [True,True,False,False]
        bool_q = [True,False,True,False]
        print("=========================")
        print("p₩t q₩t p^q")
        for i in range(len(bool_p)):
                P = bool_p[i]
                Q = bool_q[i]
                PandQ = P and Q
                print("{0}₩t {1}₩t {2} ".format(P,Q,PandQ))
        print("=========================")
        print("p₩t q₩t pvq")
        for i in range(len(bool_p)):
                P = bool_p[i]
                Q = bool_q[i]
                PorQ = P or Q
                print("{0}₩t {1}₩t {2} ".format(P,Q,PorQ))
        print("=========================")
        print("p₩t q₩t p+q")
        for i in range(len(bool_p)):
                P = bool_p[i]
                Q = bool_q[i]
                PxorQ = operator.xor(P,Q)
                print("{0}₩t {1}₩t {2} ".format(P,Q,PxorQ))
ch02_exam1()
```

추론

- def ch02_exam2():

```
def ch02_exam2():
	bool_p = [True,True,True,True,False,False,False,False]
	bool_q = [True,True,False,False,True,True,False,False]
	bool_r = [True,False,True,False,True,False,True,False]
	print("==========================")
	print("p₩t q₩t r₩t ~(p^q)₩t₩t (q v r)₩t ~(p^q) v (q v r)")
	for i in range(len(bool_p)):
		P = bool_p[i]
		Q = bool_q[i]
		R = bool_r[i]
		notPandQ = not(P and Q)
		QorR = Q or R
		notPandQorQorR = notPandQ or QorR
		print("{0}₩t {1}₩t {2}₩t {3}₩t₩t {4}₩t₩t
		{5}₩t₩t".format(P,Q,R,notPandQ,QorR,notPandQorQorR))
ch02_exam2()
```

논리연산

- 논리연산자
  - 단순명제들을 연결시켜 주는 역할을 하는 ∨, ∧, ~과 같은 논리연산자라고 함
  - 부정(Negation)연산자 NOT
    - 명제 p 가 명제일 때 "p 가 아니다"도 명제가 됨
    - ¬ p 또는 ~ p로 표현

학습 내용 요약

- 명제
- 논리연산자
- 합성명제
- 명제함축(함축명제)
- 논리적동치
- 한정기호
- 명제함수
- 추론
- 파이썬코딩