Fundamentals of Machine Learning

: 개괄적으로
빠르게 훑는다.. (기본적으로 다 뗄 예)

⊕ Statistically
Improved 된 Tool 소개

Il-Chul Moon
Dept. of Industrial and Systems Engineering
KAIST
icmoon@kaist.ac.kr

Weekly Objectives

- Learn the most classical methods of machine learning
  - Rule based approach
  - Classical statistics approach
  - Information theory appraoch
- Rule based machine learning
  - How to find the specialized and the generalized rules
  - Why the rules are easily broken
- Decision Tree
  - How to create a decision tree given a training dataset
  - Why the tree becomes a weak learner with a new dataset
- Linear Regression
  - How to infer a parameter set from a training dataset
  - Why the feature engineering has its limit

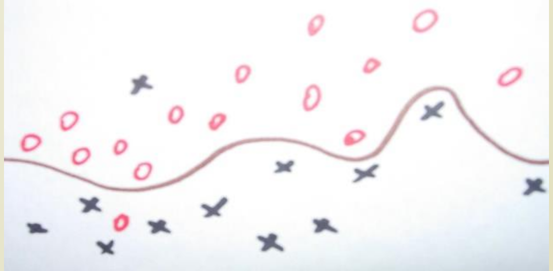
D.T
Forest
⋮

RULE BASED MACHINE LEARNING

From the Last Week

Supervised Learning

You know the true answers of some of instances



"배울수록 나아진다" = learning

- Definition of machine learning
  - A computer program is said to
    - learn from experience E
    - With respect to some class of tasks T
    - And performance measure P, if its performance at tasks in T, as measured by P, improves with experience E
- More experience → more thumbtack toss, more prior knowledge
  - Data: We have observed the sequence data of D with a_H and a_T
  - Our hypothesis
    - The gambling result of thumbtack follows the binomial distribution of θ
- Our first trial other than thumbtack
  - Rule based learning
  - Still, about choosing a better hypothesis

A Perfect World for Rule Based Learning

- Imagine
  - A perfect world with
    - No observation errors, No inconsistent observations
    - No stochastic elements in the system we observe (=Random effect)
    - Full information in the observations to regenerate the system
  - A perfect world of "EnjoySport"
- Observation on the people
  - Sky, Temp, Humid, Wind, Water, Forecast → EnjoySport

Training data is error-free, noise-free

Target function is deterministic

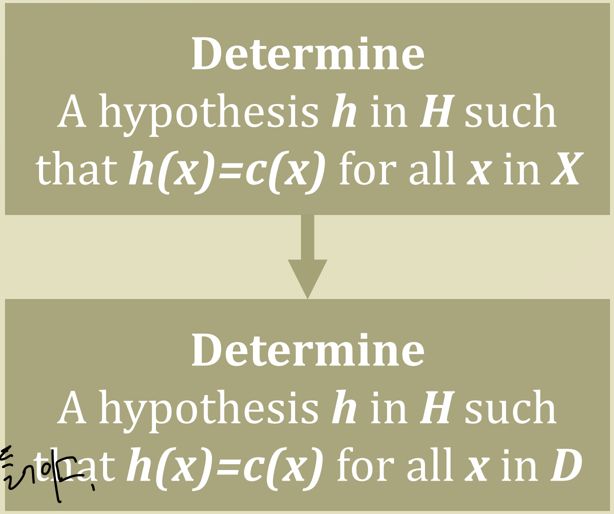
Target function is contained in hypotheses set

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Function Approximation → ML의 본질

- Machine Learning?
  - The effort of producing a better approximate function
  - Remember PAC Learning Theory?
- In the perfect world of EnjoySport
  - Instance ***X***
    - Input Features: O: <Sunny, Warm, Normal, Strong, Warm, Same>
    - Label: Y: <Yes>
  - Training Dataset ***D***
    - A collection of observations on the instance
  - Hypotheses ***H***
    - Potentially possible function to turn X into Y
    - $\boldsymbol{h_i}$: <Sunny, Warm, ?, ?, ?, Same> → Yes
    - How many hypotheses exist?
  - Target Function ***c*** (= 알아내야 / 목표로하는 함수) ∴이를 c로 만들어야함.
    - Unknown target function between the features and the label

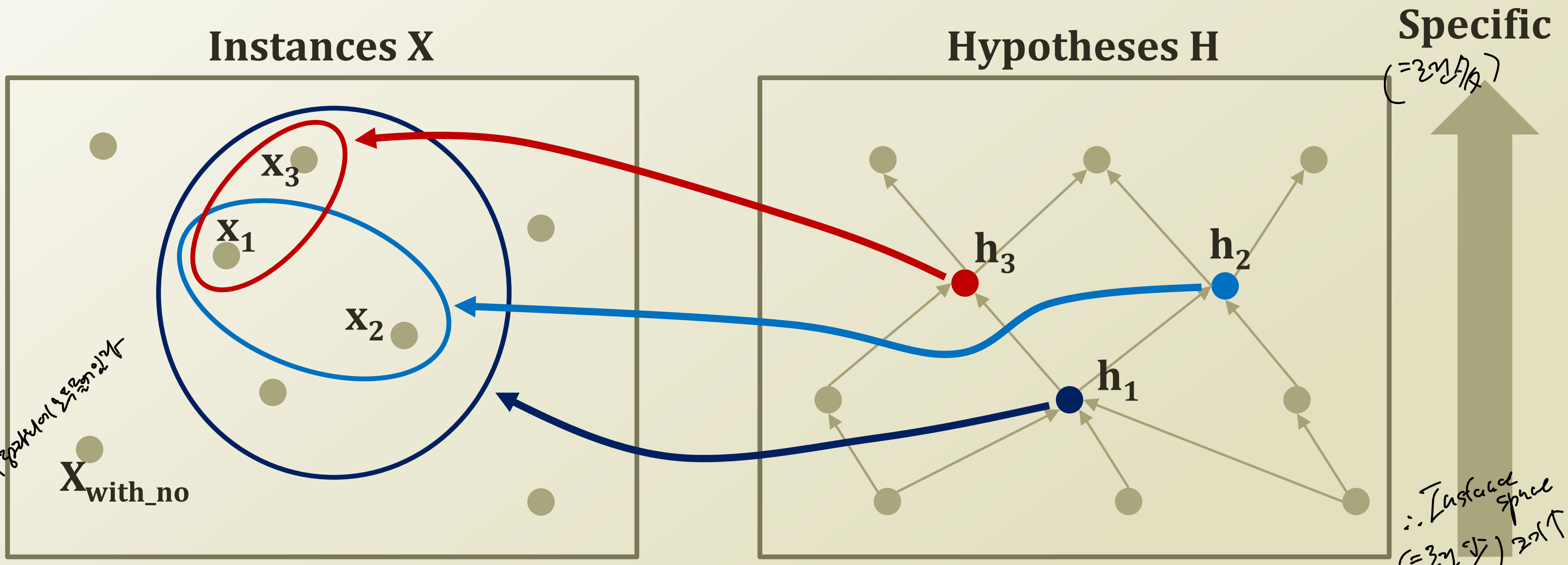
Data



Determine
A hypothesis ***h*** in ***H*** such that ***h(x)=c(x)*** for all ***x*** in ***X***

Determine
A hypothesis ***h*** in ***H*** such that ***h(x)=c(x)*** for all ***x*** in ***D***

Graphical Representation of Function Approximation



(=조건多)

∴ Instance space (=조건 少) 크기↑.

h₁.h₂.h₃ 모두에서 내가본 것과 = 제일 일반화이(동등함)인가

x_1: <Sunny, Warm, Normal, Strong, Warm, Same>

x_2: <Sunny, Warm, Normal, Light, Warm, Same>

x_3: <Sunny, Warm, Normal, Strong, Warm, Change>

h_1: <Sunny, ?, ?, ?, Warm, ?>

h_2: <Sunny, ?, ?, ?, Warm, Same>

h_3: <Sunny, ?, ?, Strong, Warm, ?>

∵ "Sunny & warm = 무조건 여가 된다" 를 가정

- What would be the better function approximation?
  - Generalization vs. Specialization

Find-S Algorithm

- Find-S Algorithm
  - Initialize h to the most specific in H
  - For instance x in D
    - if x is positive
      - For feature f in O
        - If f_i in h == f_i in x
          - Do nothing
        - Else
          - f_i in h = f_i in h ∪ f_i in x
  - Return h

합집합

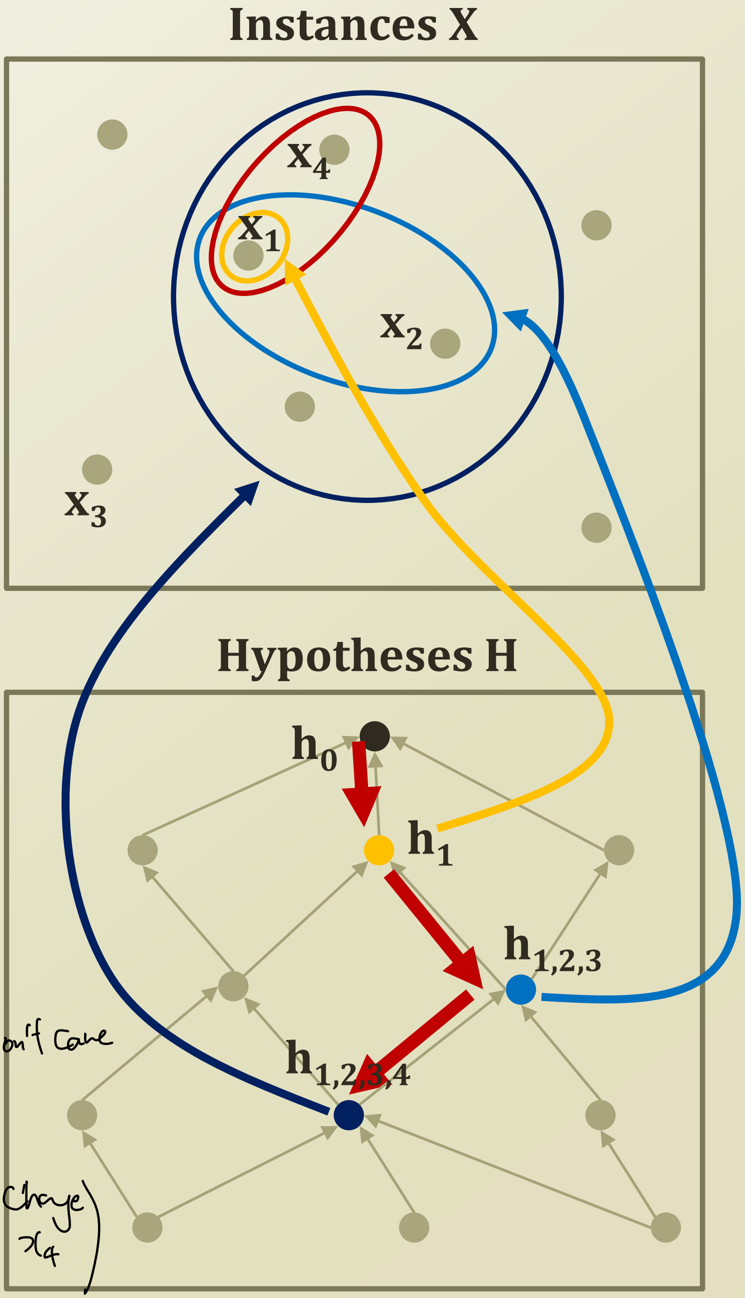
- Instances
  - x_1: <Sunny, Warm, Normal, Strong, Warm, Same>
  - x_2: <Sunny, Warm, Normal, Light, Warm, Same>
  - x_4: <Sunny, Warm, Normal, Strong, Warm, Change>
- Hypotheses
  - 시작=NULL h_0=<Ø, Ø, Ø, Ø, Ø, Ø> : 어떠한 경우도 만족 X.
  - h_1=<Sunny, Warm, Normal, Strong, Warm, Same>

→ Light ∪ STRONG = Don't Care

  - $h_{1,2,3}$=<Sunny, Warm, Normal, ?, Warm, Same>
  - $h_{1,2,3,4}$=<Sunny, Warm, Normal, ?, Warm, ?>

→ (Same ∪ Change)
$h_{1,2,3}$ x_4

- Any problems?
  - Many possible hs, and can't determine the converge : 딱 하나로 수렴할 때도 X.



Version Space

영역 = boundary를 정한다.

- Many hypotheses possible, and No way to find the convergence
- Need to setup the perimeter of the possible hypothesis
- The set of the possible (=범위) hypotheses == Version Space, ***VS***
  - General Boundary, ***G***
    - Is the set of the maximally general hypotheses of the version space
  - Specific Boundary, ***S***
    - Is the set of the maximally specific hypotheses of the version space
  - Every hypothesis, ***h***, satisifies
    - $VS_{H,D} = \{h \in H | \exists s \in S, \exists g \in G, g \geq h \geq s\}$

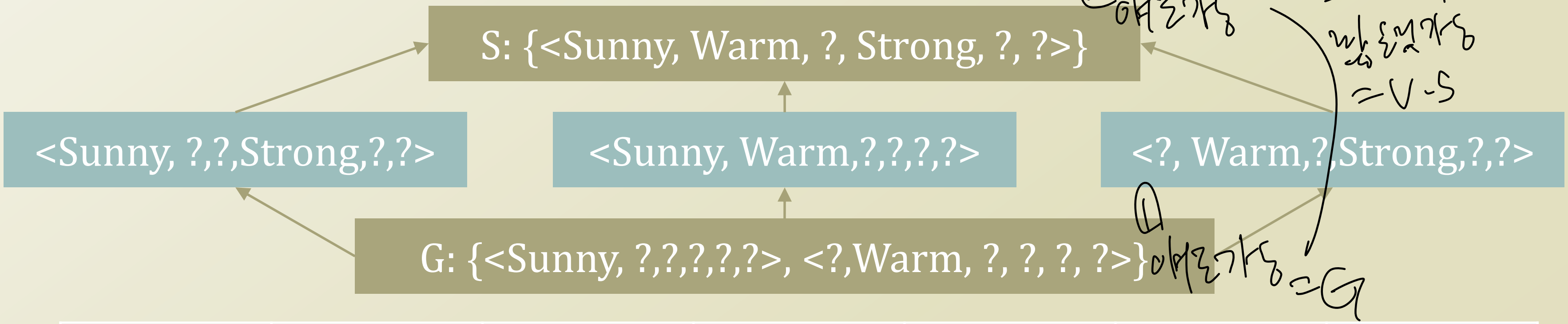
      where $x \geq y$ means x is more general or equal to y

S

중간 = Version

G

여러 H의 집합



S: {<Sunny, Warm, ?, Strong, ?, ?>}

<Sunny, ?,?,Strong,?,?>

<Sunny, Warm,?,?,?,?>

<?, Warm,?,Strong,?,?>

G: {<Sunny, ?,?,?,?,?>, <?,Warm, ?, ?, ?, ?>}

① 에러가능 = G

② 에러가능 = S

③ 그 사이에도 발생가능 = V-S

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Candidate Elimination Algorithm

= (가장 S / 가장 G) 찾고 V.S 도출 & 특정 h 찾기

- Candidate Elimination Algorithm
  - Initialize S to maximally specific h in H
  - Initialize G to maximally general h in H
  - For instance x in D → Data 이용해서 줄여나감 (G와 S를 수정해나감)
    - If y of x is positive : (가) 너무 빡빡 했다 → most S를 generalize (
      - Generalize S as much as needed to cover o in x
        이 instance feature 커버 할수없는 선 최소한 gen.
      - Remove any h in G, for which h(o)≠y → Error 발생시 드랍 ..
    - If y of x is negative : (가) 너무 헐렁 했다 → specialize.
      - Specialize G as much as needed to exclude o in x
      - Remove any h in S, for which h(o)=y
  - Generate h that satisfies $\exists s \in S, \exists g \in G, g \geq h \geq s$

S0: {<Ø, Ø, Ø, Ø, Ø, Ø>} : 무조건 안됨

G0: {<?,?,?,?,?,?>} : 무조건 됨

Progress of Candidate Elimination Algorithm

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

S_2: Don't Care

Initialize. : 부정확 X ∴ (x_1, x_2 만족시키려면 relax시켜야)

S0: {<Ø, Ø, Ø, Ø, Ø, Ø>}

$+x_1$

S1: {<Sunny, Warm, Normal, Strong, Warm, Same>}

$+x_2$

S2: {<Sunny, Warm, ?, Strong, Warm, Same>}

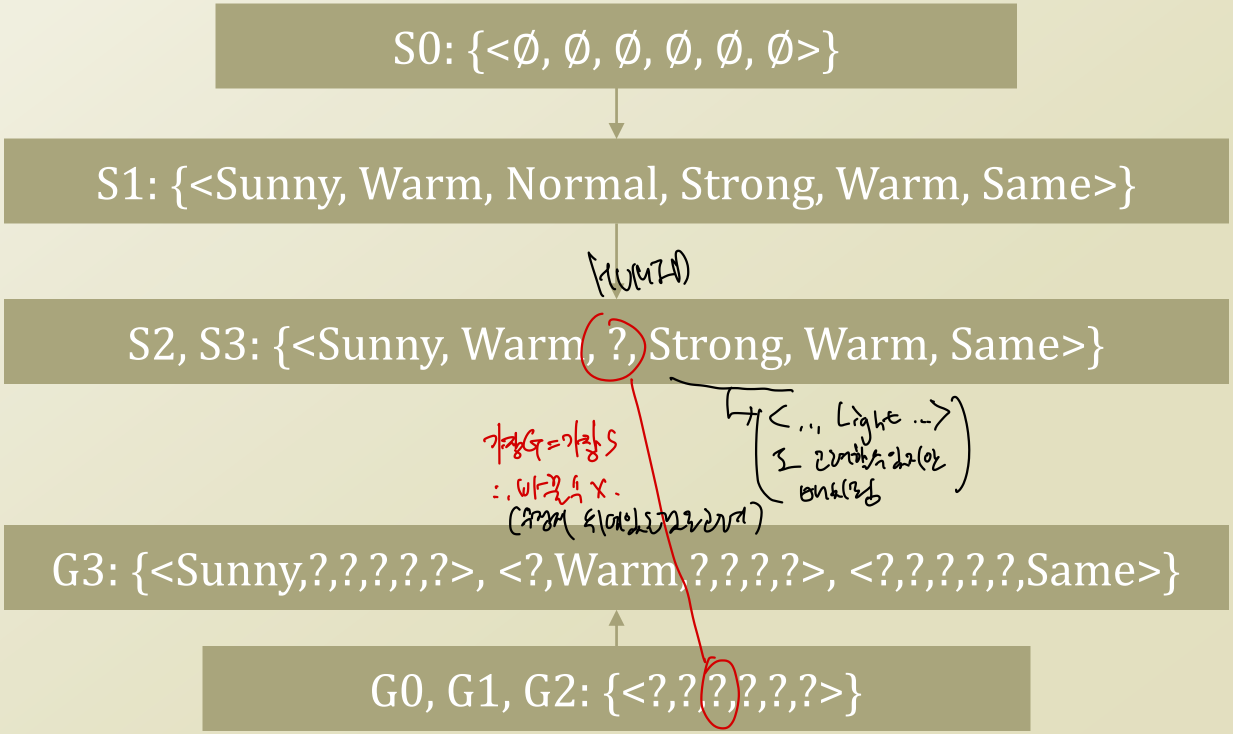
gen

Init. : 부정확 O → x_1, x_2 만족.

G0, G1, G2: {<?,?,?,?,?,?>}

Progress of Candidate Elimination Algorithm

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes



S0: {<Ø, Ø, Ø, Ø, Ø, Ø>}

S1: {<Sunny, Warm, Normal, Strong, Warm, Same>}

(HUMID)

S2, S3: {<Sunny, Warm, ?, Strong, Warm, Same>}

→ <.., Light ..> 도 고려할수있지만 배제됨

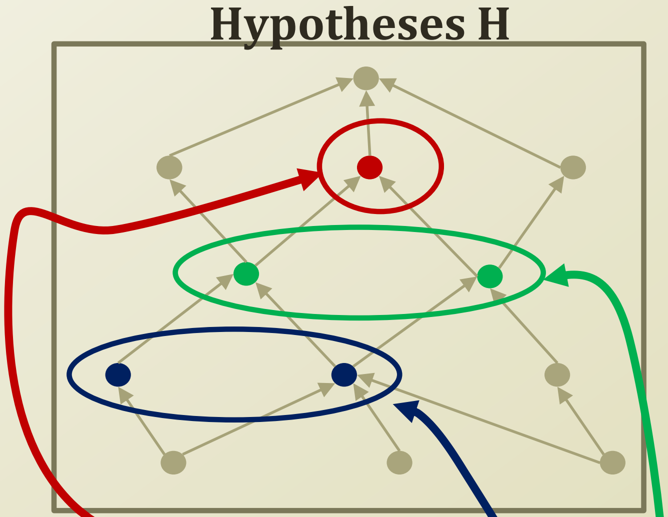
가장G=가장S
∴ 바꿀수 X.
(주변에 뒤에있는것고려안해서)

3Case로나눠짐,

G3: {<Sunny,?,?,?,?,?>, <?,Warm,?,?,?,?>, <?,?,?,?,?,Same>}

G0, G1, G2: {<?,?,?,?,?,?>}

Progress of Candidate Elimination Algorithm



Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

S0: {<Ø, Ø, Ø, Ø, Ø, Ø>}

S1: {<Sunny, Warm, Normal, Strong, Warm, Same>}

S2, S3: {<Sunny, Warm, ?, Strong, Warm, Same>}

S4: {<Sunny, Warm, ?, Strong, ?, ?>}

Still many ***hs*** : 이후 True ftn Co딩! (=Rule Based)

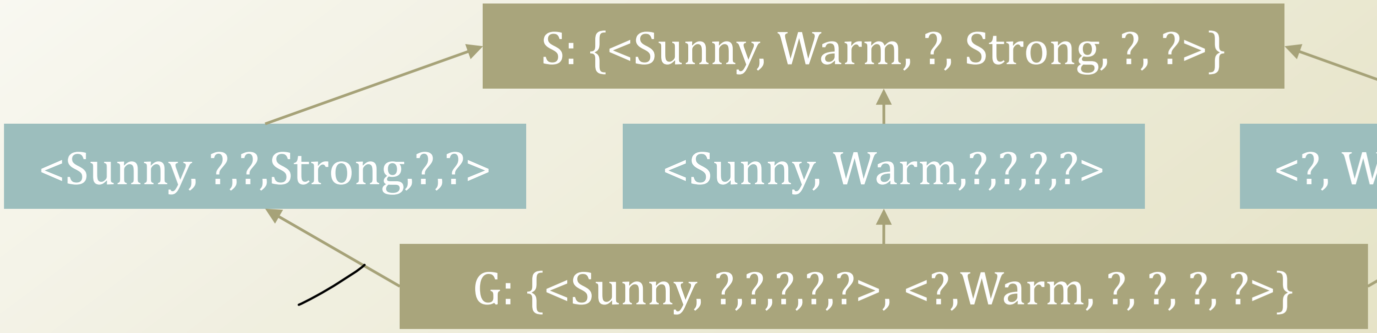
G4: {<Sunny,?,?,?,?,?>, <?,Warm,?,?,?,?>}

G3: {<Sunny,?,?,?,?,?>, <?,Warm,?,?,?,?>, ~~<?,?,?,?,?,Same>~~} : 얘네 만족하면 pos-여야 나오겠는데 chge. Yes였으므로 제거.

G0, G1, G2: {<?,?,?,?,?,?>}

How to classify the next instance?

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

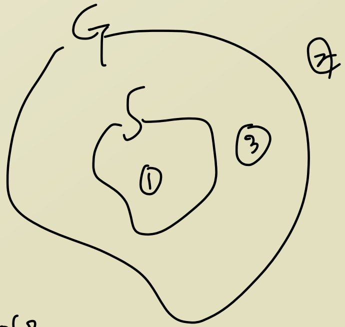

S: {<Sunny, Warm, ?, Strong, ?, ?>}

<Sunny, ?,?,Strong,?,?>

<Sunny, Warm,?,?,?,?>

<?, Warm,?,Strong,?,?>

G: {<Sunny, ?,?,?,?,?>, <?,Warm, ?, ?, ?, ?>}



- Somehow, we come up with the version space
  - A subset of ***H*** that satisfies the training data, ***D***
- Imagine a new instance kicks in

Instances

① <Sunny, Warm, Normal, Strong, Cool, Change> → S 만족 ∴ 다 만족.

② <Rainy, Cold, Normal, Light, Warm, Same> → G 불만족 ∴ 다 불만족.

③ <Sunny, Warm, Normal, Light, Warm, Same> → G 만족, S 불만족. → 아직 data evidence X [illegible]

∴ 결정X Case가! 갈라짐

= 여러 Case 판단해야 적용가능

⊕ 몇개의 V.S 만족하는지 비율 구하고 Voting 해봄으로

- How to classify these?
  - Which ***h*** to apply from the subset?
  - Or, a classification by all of ***h***s in the subset
  - How many are ***h***s satisfied?

Is this working? : Ideal 에서만.

- Will the candidate-elimination algorithm converge to the correct hypothesis?
  - Converge? → Able to select a hypothesis
  - Correct? → The hypothesis is true in the observed system
- Given the assumption, yes and yes
  - No observation errors, No inconsistent observations
  - No stochastic elements in the system we observe
  - Full information in the observations to regenerate the system
- However, we don't live in the perfect world
  - Any noise in ***o*** of ***x*** in ***D***
  - Decision factor other than ***o*** of ***x*** (다른 factor)

→ **a correct *h* can be removed by the noise** : 잘못되게 eliminate해버림.

→ **Cannot say yes and no** ∴ noise를 해결하기위해서 다양한 learning algo : ex) DT

Training data is error-free, noise-free

Target function is deterministic

Target function is contained in hypotheses set