Fundamentals of Machine Learning

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

Weekly Objectives

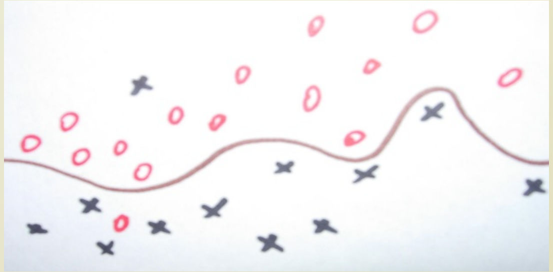
- Learn the most classical methods of machine learning
  - Rule based approach
  - Classical statistics approach
  - Information theory appraoch
- Rule based machine learning
  - How to find the specialized and the generalized rules
  - Why the rules are easily broken
- Decision Tree
  - How to create 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

RULE BASED MACHINE LEARNING

From the Last Week

Supervised Learning

You know the true answers of some of instances



- 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
    - 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

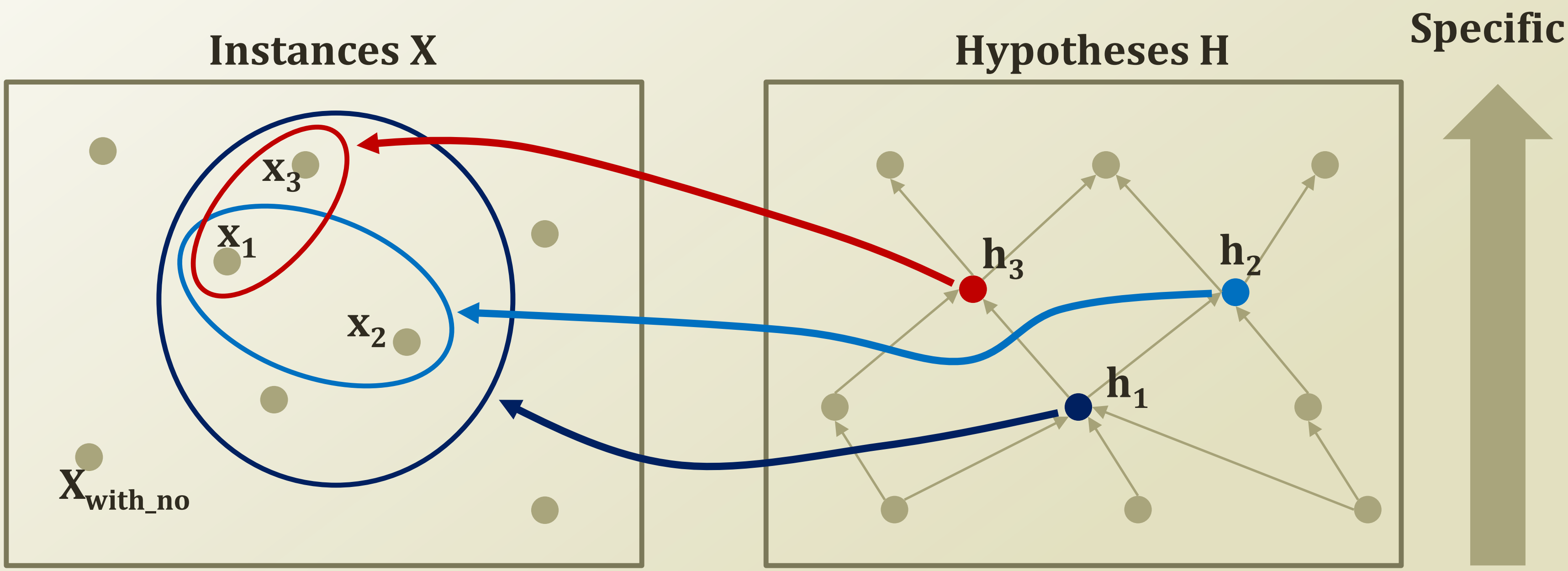
- Machine Learning?
  - The effort of producing a better approximate function
  - Remember PAC Learning Theory?
- In the perfect world of EnjoySport
  - Instance ***X***
    - Features: O: <Sunny, Warm, Normal, Strong, Warm, Same>
    - Label: Y: <Yes>
  - Training Dataset ***D***
    - A collection of observations on the instance
  - Hypotheses ***H***
    - Potentially possible function to turn X into Y
    - h_i: <Sunny, Warm, ?, ?, ?, Same> → Yes
    - How many hypotheses exist?
  - Target Function ***c***
    - Unknown target function between the features and the label

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



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, ?>

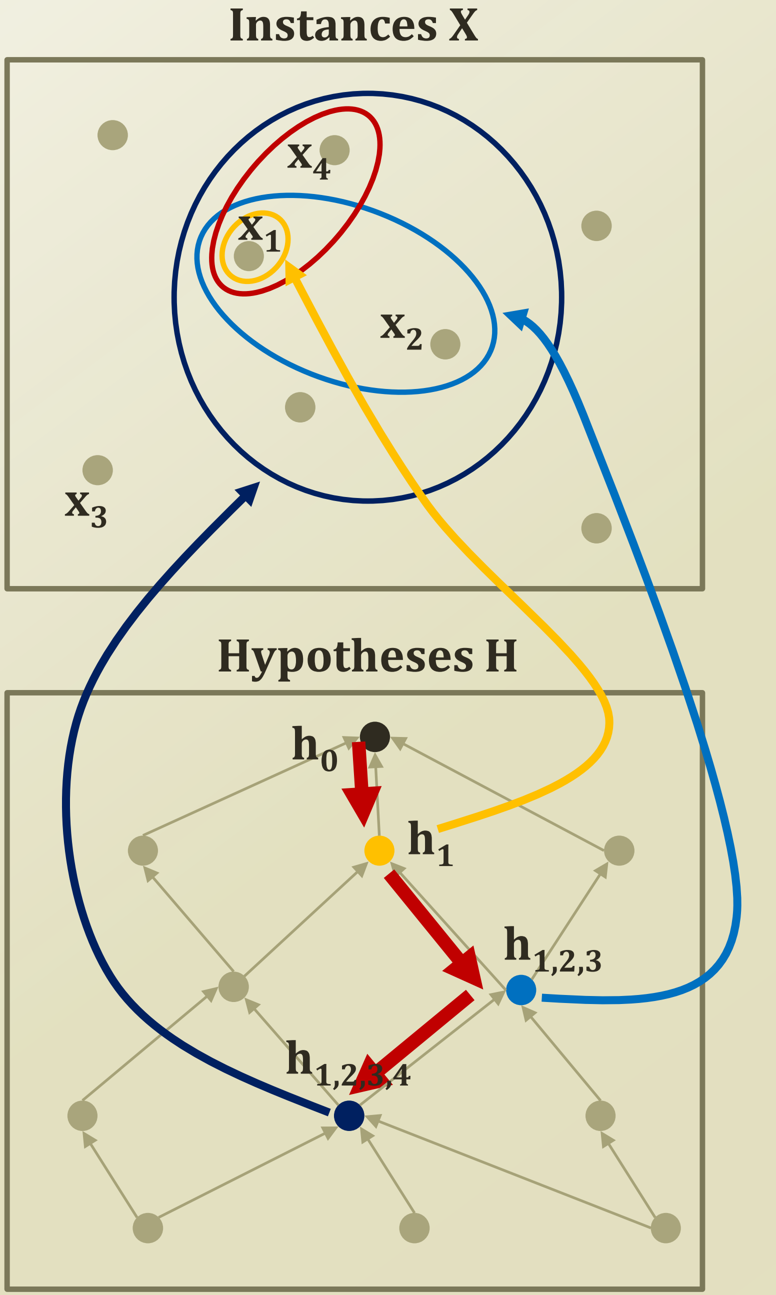
h_2: <Sunny, ?, ?, ?, Warm, Same>

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

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

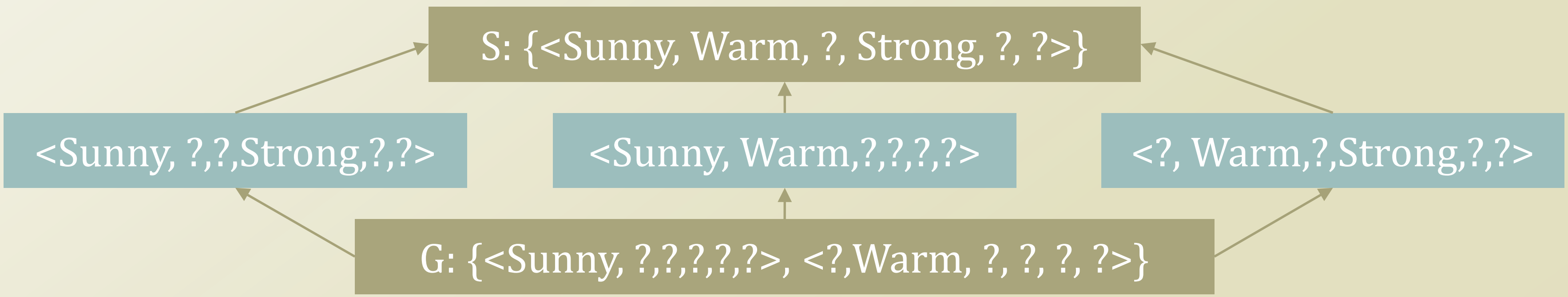
Find-S Algorithm

- 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
- 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
  - h_0=<Ø, Ø, Ø, Ø, Ø, Ø>
  - h_1=<Sunny, Warm, Normal, Strong, Warm, Same>
  - $h_{1,2,3}$=<Sunny, Warm, Normal, ?, Warm, Same>
  - $h_{1,2,3,4}$=<Sunny, Warm, Normal, ?, Warm, ?>
- Any problems?
  - Many possible hs, and can't determine the converge



Version Space

- Many hypotheses possible, and No way to find the convergence
- Need to setup the perimeter of the possible hypothesis
- The set of the possible 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
    - $\mathrm{VS}_{\mathrm{H,D}} = \{h \in H \mid \exists s \in S, \exists g \in G, g \geq h \geq s\}$
      $where\ x \geq y\ means\ x\ is\ more\ general\ or\ equal\ to\ y$



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

- Candidate Elimination Algorithm
  - Initialize S to maximally specific h in H
  - Initialize G to maximally general h in H
  - For instance x in D
    - If y of x is positive
      - Generalize S as much as needed to cover o in x
      - Remove any h in G, for which h(o)≠y
    - If y of x is negative
      - 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>}

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

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

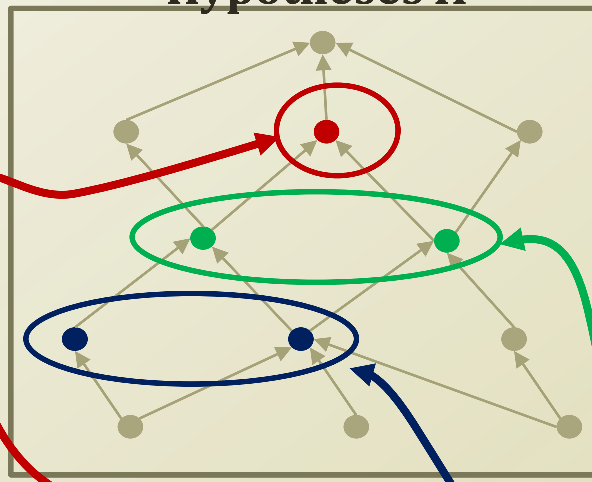
S2, S3: {<Sunny, Warm, ?, Strong, Warm, Same>}

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

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

Progress of Candidate Elimination Algorithm

Hypotheses H



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 ***h*** s

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

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

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, ?, ?>}

h_1= <Sunny, ?,?,Strong,?,?>

h_2= <Sunny, Warm,?,?,?,?>

h_3= <?, 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
  - <Sunny, Warm, Normal, Strong, Cool, Change>
  - <Rainy, Cold, Normal, Light, Warm, Same>
  - <Sunny, Warm, Normal, Light, Warm, Same>
- 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?

- 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***

  → **a correct *h* can be removed by the noise**

  → **Cannot say yes and no**

Training data is error-free, noise-free

Target function is deterministic

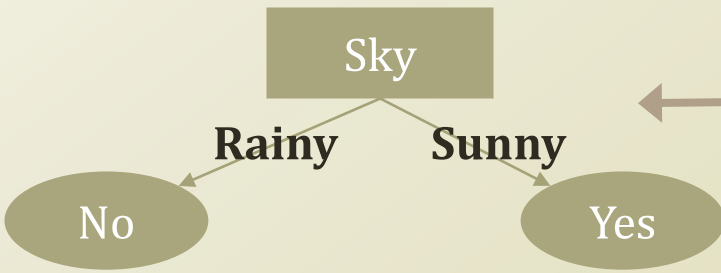
Target function is contained in hypotheses set

DECISION TREE

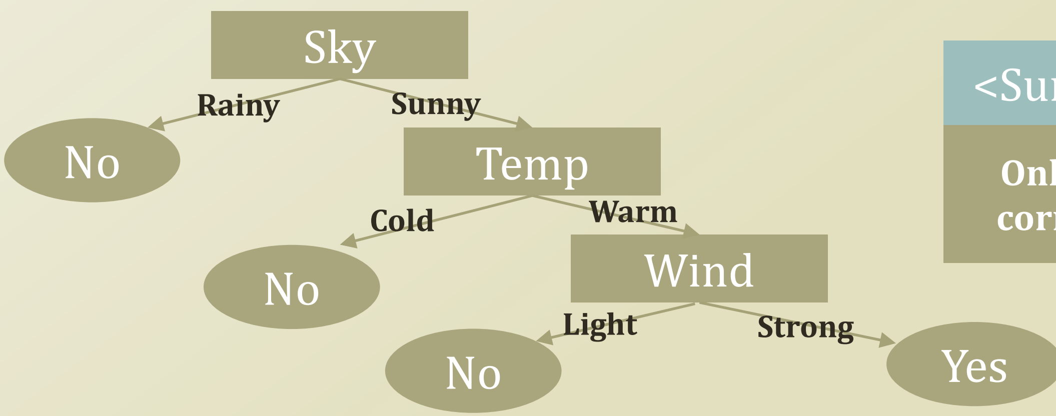
Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Because we live with noises…

- We need a better learning method
  - We need to have more robust methods given the noises
  - We need to have more concise presentations of the hypotheses
- One alternative is a decision tree



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



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

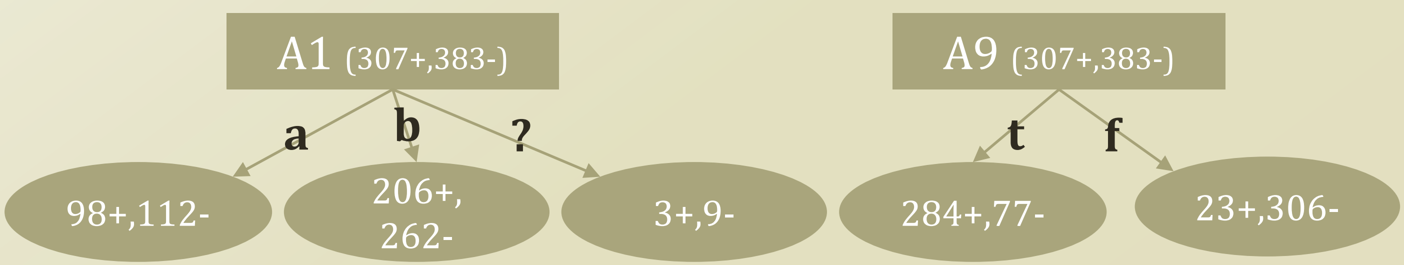
Only one potential decision tree corresponding to the hypothesis

Credit Approval Dataset

- http://archive.ics.uci.edu/ml/datasets/Credit+Approval
- To protect the confidential information, the dataset is anonymized
  - Feature names and values, as well
- A1: b, a.
  A2: continuous.
  A3: continuous.
  A4: u, y, l, t.
  A5: g, p, gg.
  A6: c, d, cc, i, j, k, m, r, q, w, x, e, aa, ff.
  A7: v, h, bb, j, n, z, dd, ff, o.
  A8: continuous.
  A9: t, f.
  A10: t, f.
  A11: continuous.
  A12: t, f.
  A13: g, p, s.
  A14: continuous.
  A15: continuous.
  C: +,- (class attribute)

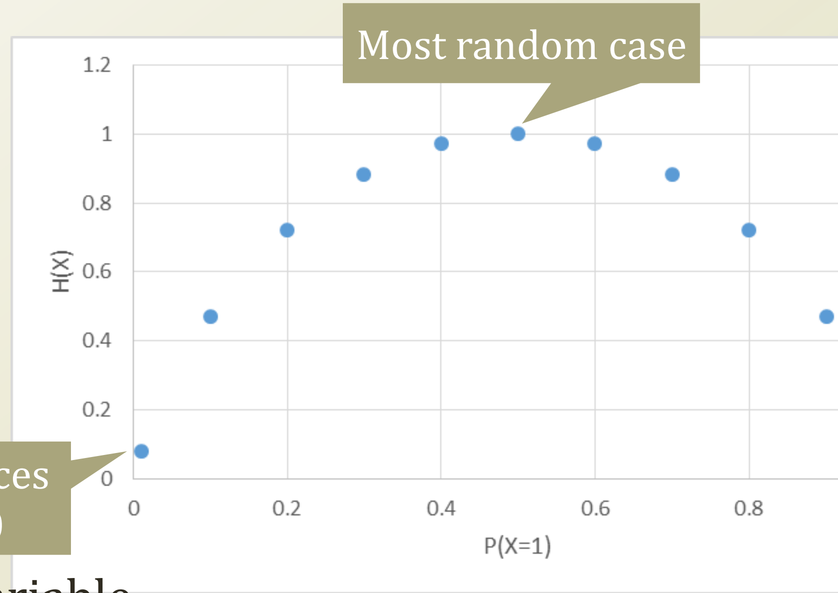
Some Counting Result

- 690 instances total
- 307 positive instances
- Considering A1
  - 98 positive when a
  - 112 negative when a
  - 206 positive when b
  - 262 negative when b
  - 3 positive when ?
  - 9 negative when ?
- Considering A9
  - 284 positive when t
  - 77 negative when t
  - 23 positive when f
  - 306 negative when f



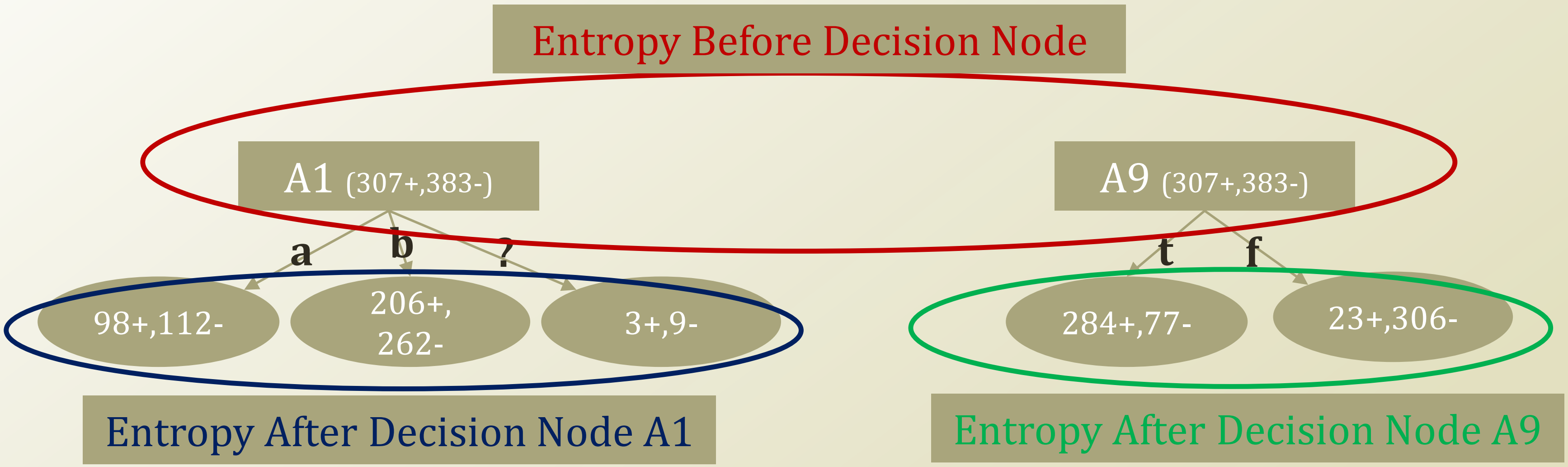
Which is a better attribute to include in the feature set of the hypothesis?

Entropy



- Better attribute to check?
  - Reducing the most uncertainty
  - Then, how to measure the uncertainty of a feature variable
- Entropy of a random variable
  - Features are random variables
  - Higher entropy means more uncertainty
  - $H(X) = -\sum_X P(X=x) log_b P(X=x)$
- Conditional Entropy
  - We are interested in the entropy of the class given a feature variable
  - Need to introduce a given condition in the entropy
  - $H(Y|X) = \sum_X P(X=x) H(Y|X=x)$
    $= \sum_X P(X=x)\{-\sum_Y P(Y=y|X=x) log_b P(Y=y|X=x)\}$

Information Gain

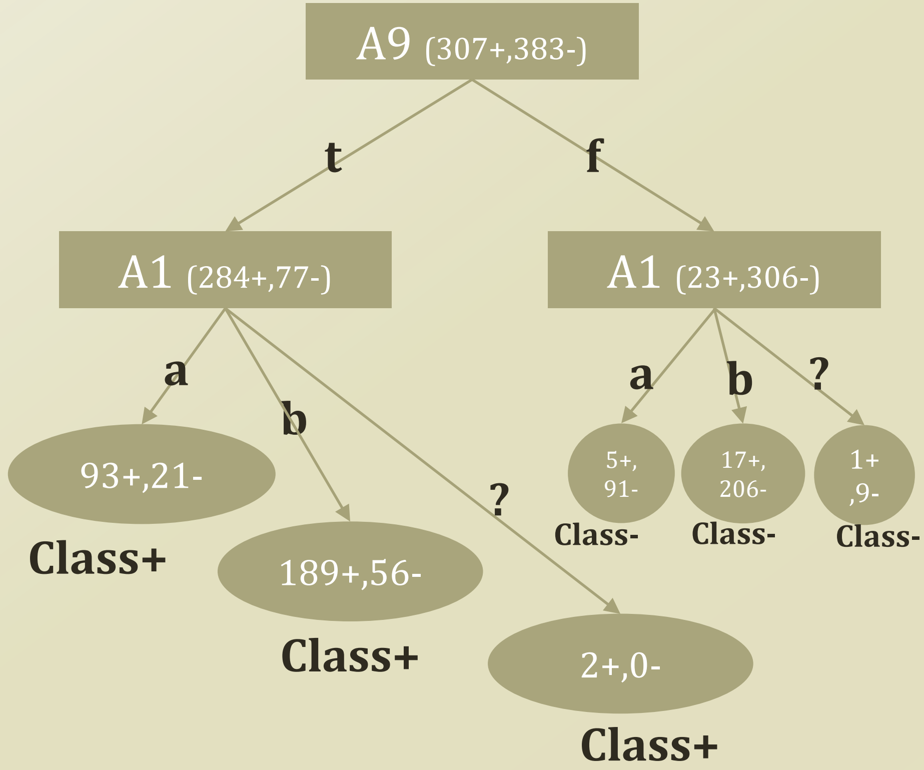


- Let's calculate the entropy values
  - H(Y)= $-\sum_{Y\in\{+,-\}} P(Y=y)log_2 P(Y=y)$
  - H(Y|A1)=$\sum_{X\in\{a,b,?\}}\sum_{Y\in\{+,-\}} P(A1=x,Y=y)log_2\frac{P(A1=x)}{P(A1=x,Y=y)}$
  - H(Y|A9) =$\sum_{X\in\{t,f\}}\sum_{Y\in\{+,-\}} P(A9=x,Y=y)log_2\frac{P(A9=x)}{P(A9=x,Y=y)}$
- What's the difference before and after?
  - $IG(Y,A_i)=H(Y)-H(Y|A_i)$
- Who is the winner?

Top-Down Induction Algorithm

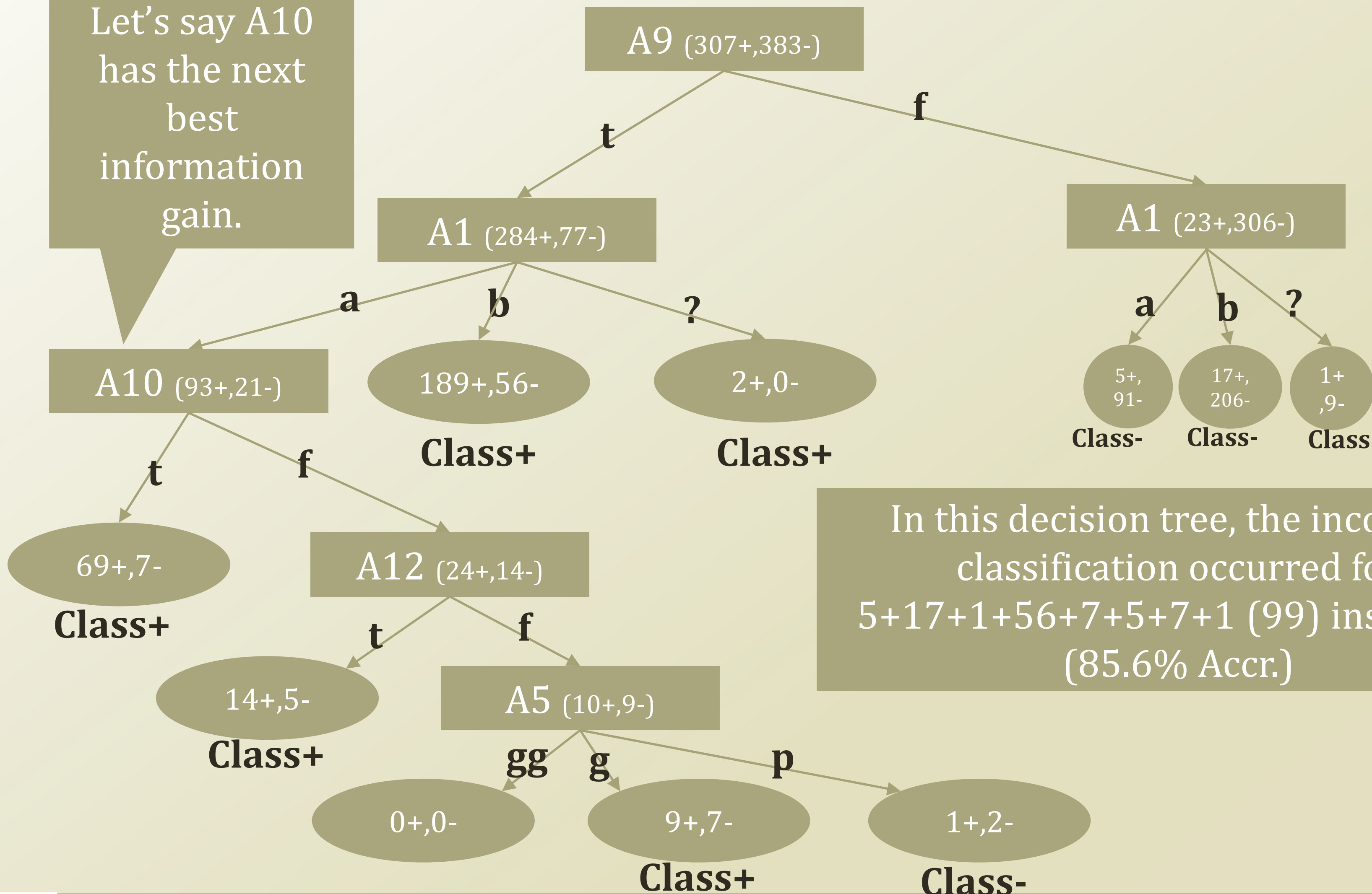
- Many, many variations in learning a decision tree
  - ID3, C4.5 CART….
- One example: ID3 algorithm
- ID3 algorithm
  - Create an initial open node
  - Put instances in the initial node
  - Repeat until no open node
    - Select an open node to split
    - Select a best variable to split
    - For values of the selected variable
      - Sort instances with the value of the selected variable
      - Put the sorted items under the branch of the value of the variable
      - If the sorted items are all in one class
        - Close the leaf node of the branch

Only using A1 and A9, we have 21+56+0+5+17+1 (100) instances classified inaccurately. (85.5% Accr.)



If you want more….

Let's say A10 has the next best information gain.

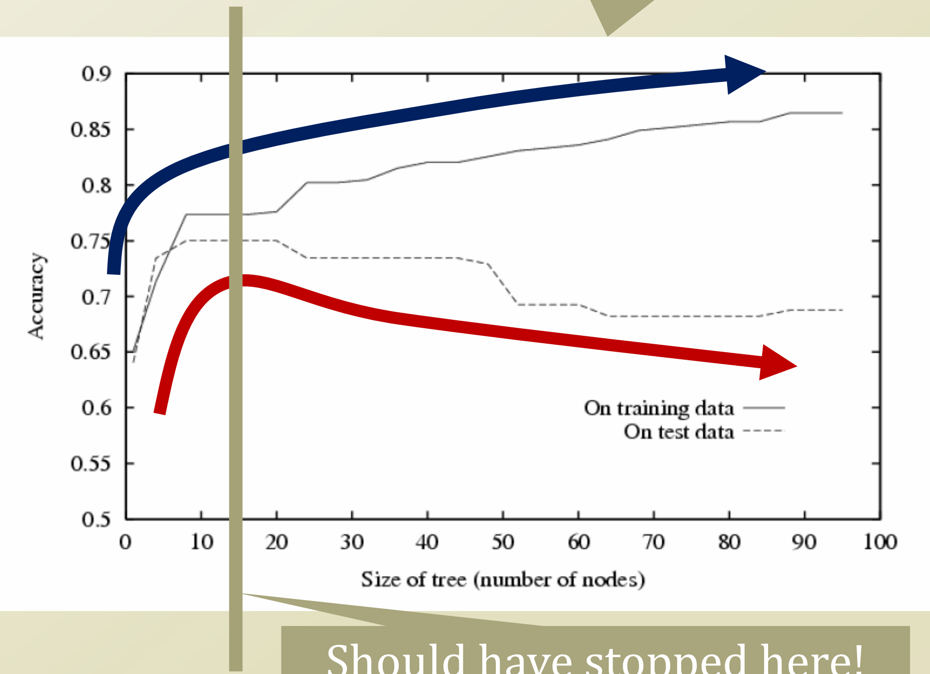


In this decision tree, the incorrect classification occurred for 5+17+1+56+7+5+7+1 (99) instances. (85.6% Accr.)

Problem of Decision Tree

- We did better in the given dataset!
  - Only in the given experience, a.k.a. Training dataset
- What if we deploy the created decision tree in the field?
  - World has so much noise and inconsistencies.
  - The training dataset will not be a perfect sample of the real world
    - Noise
    - Inconsistencies

Typical result of decision tree



Should have stopped here!

Knowing when to stop is a pretty difficult task. How to do it?

- Pruning by divided dataset?
- Path length penalty?

Why we are not interested in these?

Believe the small dataset?
(5/6→ Head with 83.3% prob?)

- Rule based machine learning algorithms
  - Easy to implement
  - Easily interpretable
    - Particularly, decision tree
- Their weaknesses
  - Fragile
    - Assume the perfect world in the dataset
    - Any new observations, contradicting to the training, will cause problems
  - Convergence
    - Convergence only guaranteed in the perfect dataset
    - Once there is a noise, there is a possibility that the true hypothesis can be ruled out.
    - Also, very hard to tell when to stop in some cases
- Still used in many places
  - Easy → Wide audience and users → Many applications → Better result???
- Need a white knight as a savior
  - Should be able to handle noisy datasets
  - Robust to errors

LINEAR REGRESSION

How about statistical approach?

- http://archive.ics.uci.edu/ml/datasets/Housing
- Housing dataset
  - 13 numerical independent values
  - 1 numerical dependent value
- How to create an approximated function?
  - Do you remember that the machine learning is the function approximation process?
- Here,
  - Our hypothesis is
    - The house value will be the linearly weighted sum of the feature values.
    - $h: \hat{f}(x;\theta) = \theta_0 + \sum_{i=1}^{n} \theta_i x_i = \sum_{i=0}^{n} \theta_i x_i$
    - $\boldsymbol{n}$ is the number of the feature values.
    - Two aspects: the linearly weight sum (the model), the parameter $\boldsymbol{\theta}$
  - The first effort is finding the better $\boldsymbol{\theta}$, just like the thumbtack

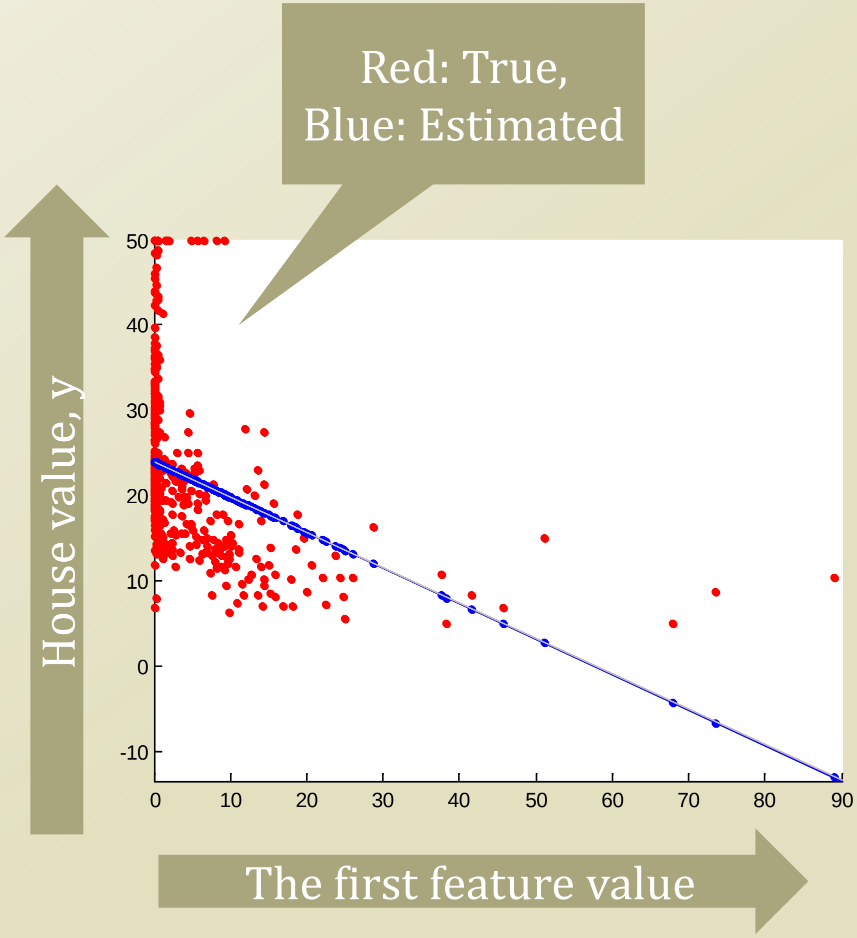
KAIST

Finding $\boldsymbol{\theta}$ in Linear Regression

- To make the hypothesis better, we need to find the better $\boldsymbol{\theta}$
  - $h: \hat{f}(x;\theta) = \sum_{i=0}^{n} \theta_i x_i \rightarrow \hat{f} = X\theta$
  - $X = \begin{pmatrix} 1 & \cdots & x_n^D \\ \vdots & \ddots & \vdots \\ 1 & \cdots & x_n^D \end{pmatrix}, \theta = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \cdots . \\ \theta_n \end{pmatrix}$
- The reality would be the noisy, so….
  - $f(x;\theta) = \sum_{i=0}^{n} \theta_i x_i + e = y \rightarrow f = X\theta + e = Y$
- The difference is the error from the noise, so let's make it minimum
  - $\hat{\theta} = argmin_\theta (f - \hat{f})^2 = argmin_\theta (Y - X\theta)^2$
    $= argmin_\theta (Y - X\theta)^T\ (Y - X\theta) = argmin_\theta (Y - X\theta)^T\ (Y - X\theta)$
  $= argmin_\theta (\theta^T X^T X\theta - 2\theta^T X^T Y + Y^T Y) = argmin_\theta (\theta^T X^T X\theta - 2\theta^T X^T Y)$
- Now, we need to optimize $\boldsymbol{\theta}$

Optimized θ

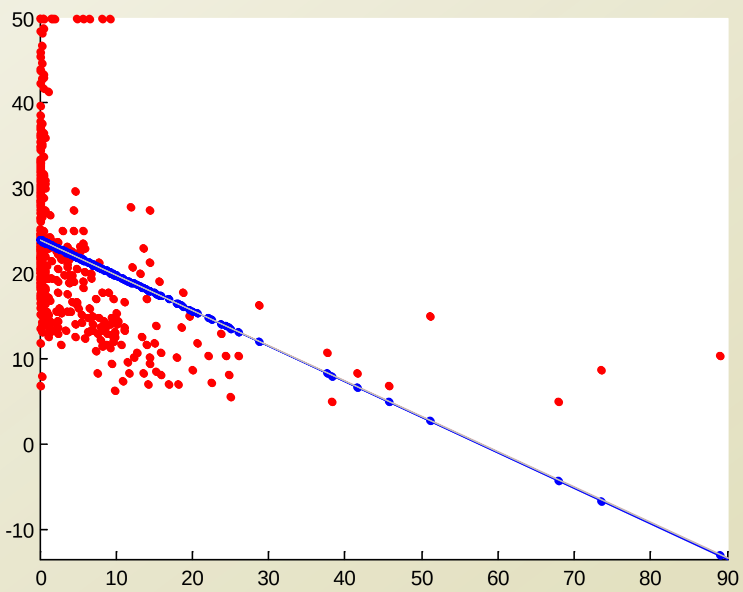
- $\hat{\theta} = argmin_{\theta}(\theta^T X^T X\theta - 2\theta^T X^T Y)$
- Same technique as in Thumbtack
  - $\nabla_{\theta}(\theta^T X^T X\theta - 2\theta^T X^T Y)=0$
  - $2X^T X\theta - 2X^T Y = 0$
  - $\theta = (X^T X)^{-1} X^T Y$
- Great! We know X and Y, so we can compute θ
- Let's calculate and watch the performance!
  - For demonstration purpose, we limit the dimension to the constant and the first feature
- MLE if error follows the normal distribution

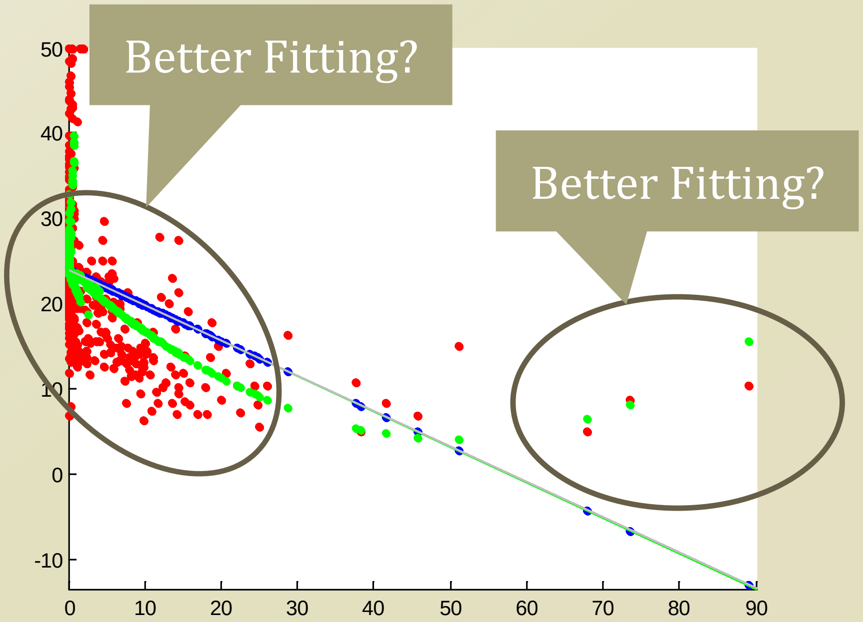


If you want more….

- Actually, you can increase the number of features, a.k.a. dimension
  - You have a value *x* in the previous fitted figure
  - How about adding x^2, x^3, x^4….?
- You can improve the result!
  - Is that right?
- We are going to come back!

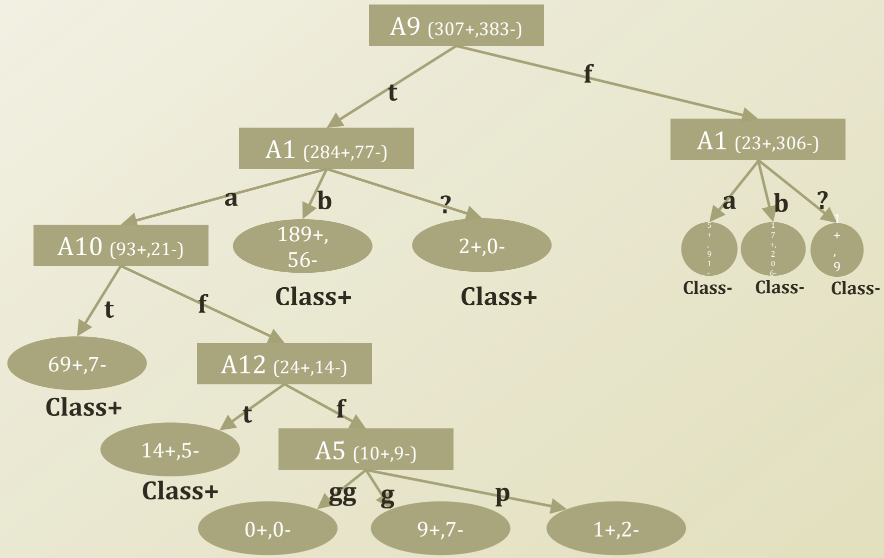
$$h: \hat{f}(x;\theta) = \sum_{i=0}^{n}\sum_{j=1}^{m} \theta_{i,j}\phi_j(x_i)$$

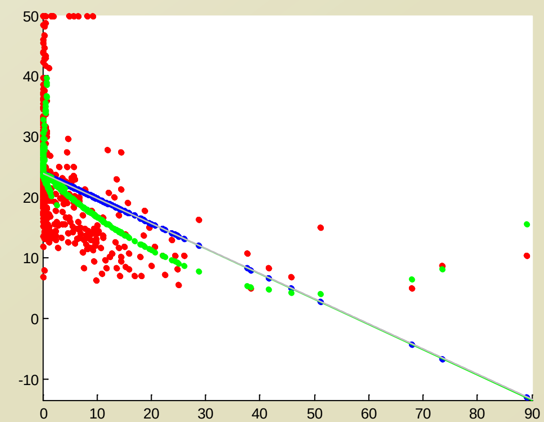


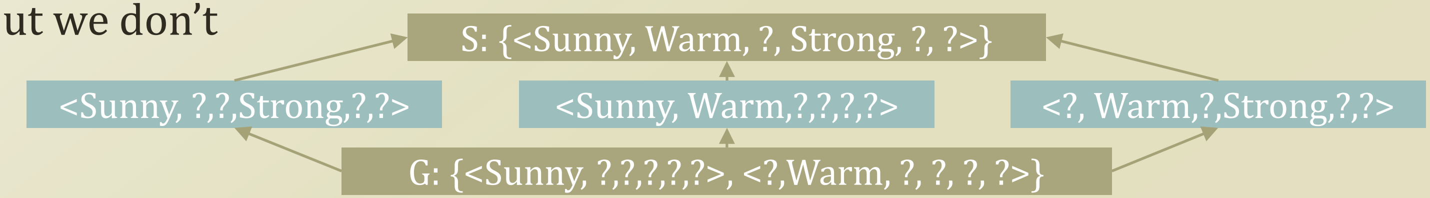


Too Brittle to Be Used Naively

- What we are doing
  - Approximating a function to the dataset
  - The function can be
    - Discrete logics
    - Statistical model
  - Often, the function type is given
    - The parameters of the functions are the target of the analysis
- Alternatives in finding the parameter
  - MLE or MAP
  - Engineering the features
  - Setting the generalization and the specialization level
- Best choice among the alternatives
  - If we have the perfect data, we will know
  - But we don't







Acknowledgement

- This slideset is greatly influenced
  - By Prof. Tom Mitchell at CMU
  - By Prof. Eric P. Xing at CMU

Further Readings

- Bishop Chapter 14.4
- Mitchell Chapter 1,2,3