BITS Pilani
Dubai Campus

Machine Learning

CS F464

Dr. Pranav M. Pawar

Contents

- Bayesian Classifier
- Gaussian Classifier
- Decision Tree
- Classification Accuracy Metrics

Probabilistic Classification

- Establishing a probabilistic model for classification
  - **Discriminative model**

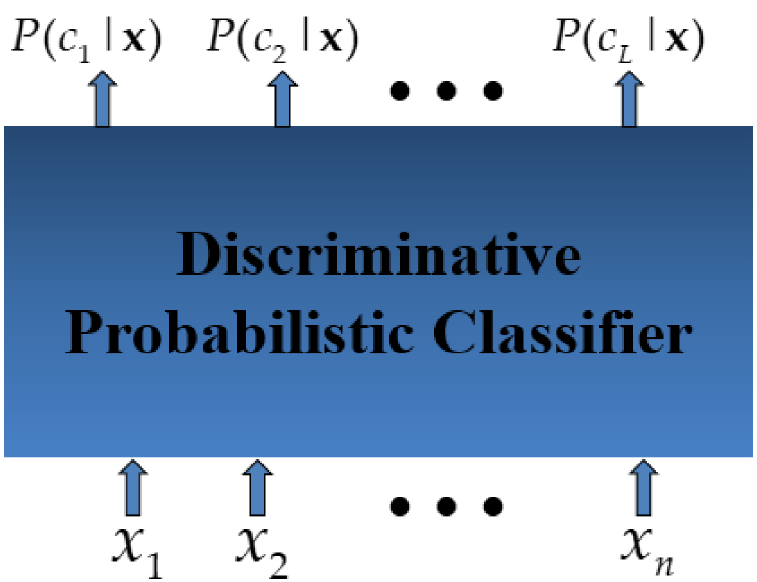
Vectors for teaching

$$P(C \mid \mathbf{X}) \quad C = c_1, \cdots, c_L, \mathbf{X} = (X_1, \cdots, X_n)$$

Probability of seeing a member of this class

What is a discriminative Probabilistic Classifier?

We want to know probabilities of classes for events **x**1



Probability that when they show me a fruit it will be an apple

$$\mathbf{x} = (x_1, x_2, \cdots, x_n)$$

We know events x1, … xn

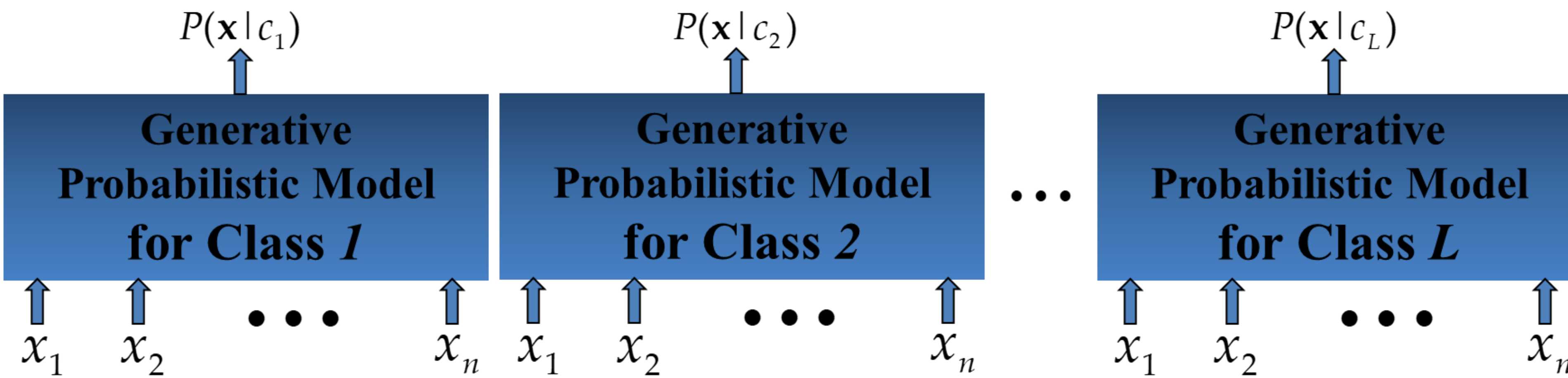
Probabilistic Classification

- Establishing a probabilistic model for classification (cont.)
  - **Generative model**

Probability that this fruit is an apple

$$P(\mathbf{X}|C) \quad C = c_1, \cdots, c_L, \mathbf{X} = (X_1, \cdots, X_n)$$

Probability that this fruit is an orange

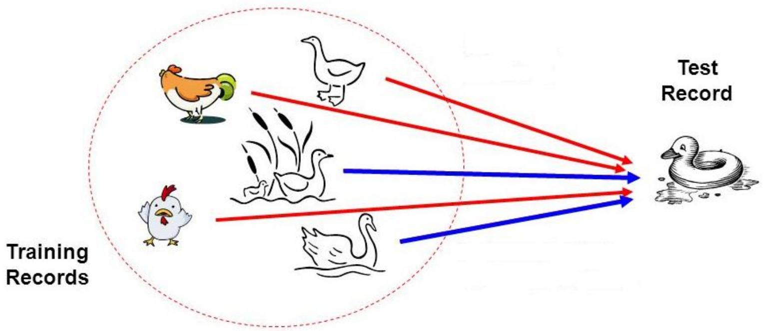


$$\mathbf{x} = (x_1, x_2, \cdots, x_n)$$

Vectors of random variables

Bayesian Classifier

- Principal
  - If it walks like a duck, quacks like a duck, then it is probably a duck.



- A statistical classifier
  - Performs *probabilistic prediction, i.e.,* predicts class membership probabilities.
- Assumptions
  - The classes are mutually exclusive and exhaustive.
  - The attributes are independent given the class.
- Incremental
  - Each training example can incrementally increase/decrease the probability that a hypothesis is correct — prior knowledge can be combined with observed data
- Called “Naïve” classifier because of these assumptions.
  - Empirically proven to be useful.
  - Scales very well.

Basis

- We defined prior, conditional and joint probability for random variables
  - Prior probability: $P(X)$
  - Conditional probability: $P(X_1 | X_2), P(X_2 | X_1)$
  - Joint probability: $\mathbf{X} = (X_1, X_2), P(\mathbf{X}) = P(X_1, X_2)$
  - Relationship: $P(X_1, X_2) = P(X_2 | X_1)P(X_1) = P(X_1 | X_2)P(X_2)$
  - Independence: $P(X_2 | X_1) = P(X_2), P(X_1 | X_2) = P(X_1), P(X_1, X_2) = P(X_1)P(X_2)$
- Bayesian Rule

$$P(C | \mathbf{X}) = \frac{P(\mathbf{X} | C)P(C)}{P(\mathbf{X})} \Rightarrow Posterior = \frac{Likelihood \times Prior}{Evidence}$$

Probabilistic Classification using MAP rule

- MAP classification rule
  - **MAP**: **M**aximum **A** **P**osterior
  - Assign x to c^* if

$$P(C = c^* \mid \mathbf{X} = \mathbf{x}) > P(C = c \mid \mathbf{X} = \mathbf{x}) \quad c \neq c^*, \; c = c_1, \cdots, c_L$$

- **Method of** Generative classification with the MAP rule
  1. Apply Bay[es rule to get posteri]or probabilities

$$P(C = c_i \mid \mathbf{X} = \mathbf{x}) = \frac{P(\mathbf{X} = \mathbf{x} \mid C = c_i) P(C = c_i)}{P(\mathbf{X} = \mathbf{x})}$$
$$\propto P(\mathbf{X} = \mathbf{x} \mid C = c_i) P(C = c_i)$$
$$\text{for } i = 1, 2, \cdots, L$$

  2. Then apply the MAP rule

Navie Bayes Classification

- <u>Bayes classification</u>

$$P(C \mid \mathbf{X}) \propto P(\mathbf{X} \mid C)P(C) = P(X_1, \cdots, X_n \mid C)P(C)$$

Difficulty: learning the joint probability $P(X_1, \cdots, X_n \mid C)$

- <u>Naïve Bayes classification</u>
  - Assumption that all input attributes are conditionally independent!

$$\begin{aligned} P(X_1, X_2, \cdots, X_n \mid C) &= P(X_1 \mid X_2, \cdots, X_n, C)P(X_2, \cdots, X_n \mid C) \\ &= P(X_1 \mid C)P(X_2, \cdots, X_n \mid C) \\ &= P(X_1 \mid C)P(X_2 \mid C) \cdots P(X_n \mid C) \end{aligned}$$

Product of individual probabilities

  - MAP classification rule: for $\mathbf{x} = (x_1, x_2, \cdots, x_n)$

$$[P(x_1 \mid c^*) \cdots P(x_n \mid c^*)]P(c^*) > [P(x_1 \mid c) \cdots P(x_n \mid c)]P(c), \quad c \neq c^*, c = c_1, \cdots, c_L$$

Navie Bayes Classification

- Naïve Bayes Algorithm (for discrete input attributes) has two phases
  - **1. Learning Phase**: Given a training set S,

For each target value of c_i $(c_i = c_1, \cdots, c_L)$

$\hat{P}(C = c_i) \leftarrow$ estimate $P(C = c_i)$ with examples in $\mathbf{S}$; Prior

For every attribute value x_{jk} of each attribute X_j $(j = 1, \cdots, n;\ k = 1, \cdots, N_j)$

Liklihood $\hat{P}(X_j = x_{jk} \mid C = c_i) \leftarrow$ estimate $P(X_j = x_{jk} \mid C = c_i)$ with examples in $\mathbf{S}$;

Output: conditional probability tables; for $X_j, N_j \times L$ elements

  - **2. Test Phase**: Given an unknown instance $\mathbf{X}' = (a'_1, \cdots, a'_n)$,

Look up tables to assign the label c^* to $\mathbf{X}'$ if

$$[\hat{P}(a'_1 \mid c^*) \cdots \hat{P}(a'_n \mid c^*)]\hat{P}(c^*) > [\hat{P}(a'_1 \mid c) \cdots \hat{P}(a'_n \mid c)]\hat{P}(c), \quad c \neq c^*, c = c_1, \cdots, c_L$$

Example

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Example: Learning Phase

innovate achieve lead

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

$P(\text{Play}=\textit{Yes}) = 9/14$

$P(\text{Play}=\textit{No}) = 5/14$

We have four variables, we calculate for each

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

Temperature	Play=*Yes*	Play=*No*
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

Humidity	Play=*Yes*	Play=No
High	3/9	4/5
Normal	6/9	1/5

Wind	Play=*Yes*	Play=*No*
Strong	3/9	3/5
Weak	6/9	2/5

Example: Testing Phase

- Test Phase
  - Given a new instance of variable values,

    $\mathbf{x}' = (\text{Outlook}=Sunny, \text{Temperature}=Cool, \text{Humidity}=High, \text{Wind}=Strong)$
  - Given calculated Look up tables

$P(\text{Outlook}=Sunny \mid \text{Play}=Yes) = 2/9$

$P(\text{Temperature}=Cool \mid \text{Play}=Yes) = 3/9$

$P(\text{Huminity}=High \mid \text{Play}=Yes) = 3/9$

$P(\text{Wind}=Strong \mid \text{Play}=Yes) = 3/9$

$P(\text{Play}=Yes) = 9/14$

$P(\text{Outlook}=Sunny \mid \text{Play}=No) = 3/5$

$P(\text{Temperature}=Cool \mid \text{Play}==No) = 1/5$

$P(\text{Huminity}=High \mid \text{Play}=No) = 4/5$

$P(\text{Wind}=Strong \mid \text{Play}=No) = 3/5$

$P(\text{Play}=No) = 5/14$

  - **Use the MAP rule to calculate Yes or No**

$P(Yes \mid \mathbf{x}'): [P(Sunny \mid Yes)P(Cool \mid Yes)P(High \mid Yes)P(Strong \mid Yes)]P(\text{Play}=Yes) = 0.0053$

$P(No \mid \mathbf{x}'): [P(Sunny \mid No)\, P(Cool \mid No)P(High \mid No)P(Strong \mid No)]P(\text{Play}=No) = 0.0206$

Given the fact $P(Yes \mid \mathbf{x}') < P(No \mid \mathbf{x}')$, we label $\mathbf{x}'$ to be "*No*".

Issues with Navie Bayes Classifier (1)

Events are correlated

1. Violation of Independence Assumption
   - For many real world tasks, $P(X_1,\cdots,X_n|C) \neq P(X_1|C)\cdots P(X_n|C)$
   - Nevertheless, naïve Bayes works surprisingly well anyway!

2. Zero conditional probability Problem
   - Such problem exists when no example contains the attribute value

     $X_j = a_{jk}, \hat{P}(X_j = a_{jk}|C = c_i) = 0$
   - In this circumstance, $\hat{P}(x_1|c_i)\cdots\hat{P}(a_{jk}|c_i)\cdots\hat{P}(x_n|c_i) = 0$ during test
   - For a remedy, conditional probabilities are estimated with

$$\hat{P}(X_j = a_{jk}|C = c_i) = \frac{n_c + mp}{n + m}$$

n_c : number of training examples for which $X_j = a_{jk}$ and $C = c_i$

n : number of training examples for which $C = c_i$

p : prior estimate (usually, $p = 1/t$ for t possible values of X_j)

m : weight to prior (number of "virtual" examples, $m \geq 1$)

Issues with Navie Bayes Classifier (2)

- What to do in the case of Continuous Valued Inputs?
  - Numberless values for an attribute
  - Conditional probability is then modeled with the normal distribution

$$\hat{P}(X_j \mid C = c_i) = \frac{1}{\sqrt{2\pi}\sigma_{ji}} \exp\left(-\frac{(X_j - \mu_{ji})^2}{2\sigma_{ji}^2} \right)$$

μ_{ji} : mean (avearage) of attribute values X_j of examples for which $C = c_i$

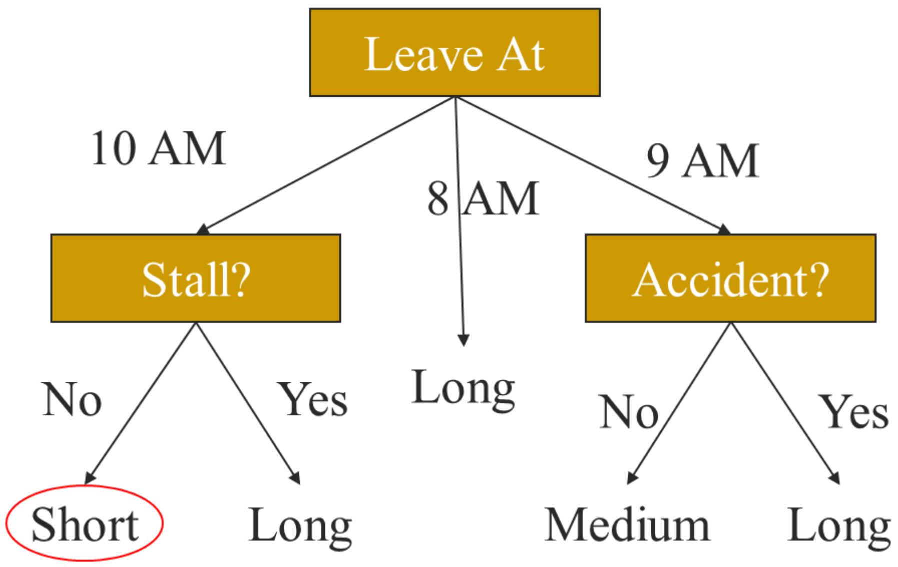
σ_{ji} : standard deviation of attribute values X_j of examples for which $C = c_i$

- Learning Phase: for $\mathbf{X} = (X_1, \cdots, X_n),\ C = c_1, \cdots, c_L$

  Output: $n \times L$ normal distributions and $P(C = c_i)\ i = 1, \cdots, L$
- Test Phase: for $\mathbf{X}' = (X'_1, \cdots, X'_n)$
  1. Calculate conditional probabilities with all the normal distributions
  2. Apply the MAP rule to make a decision

Decision Tree

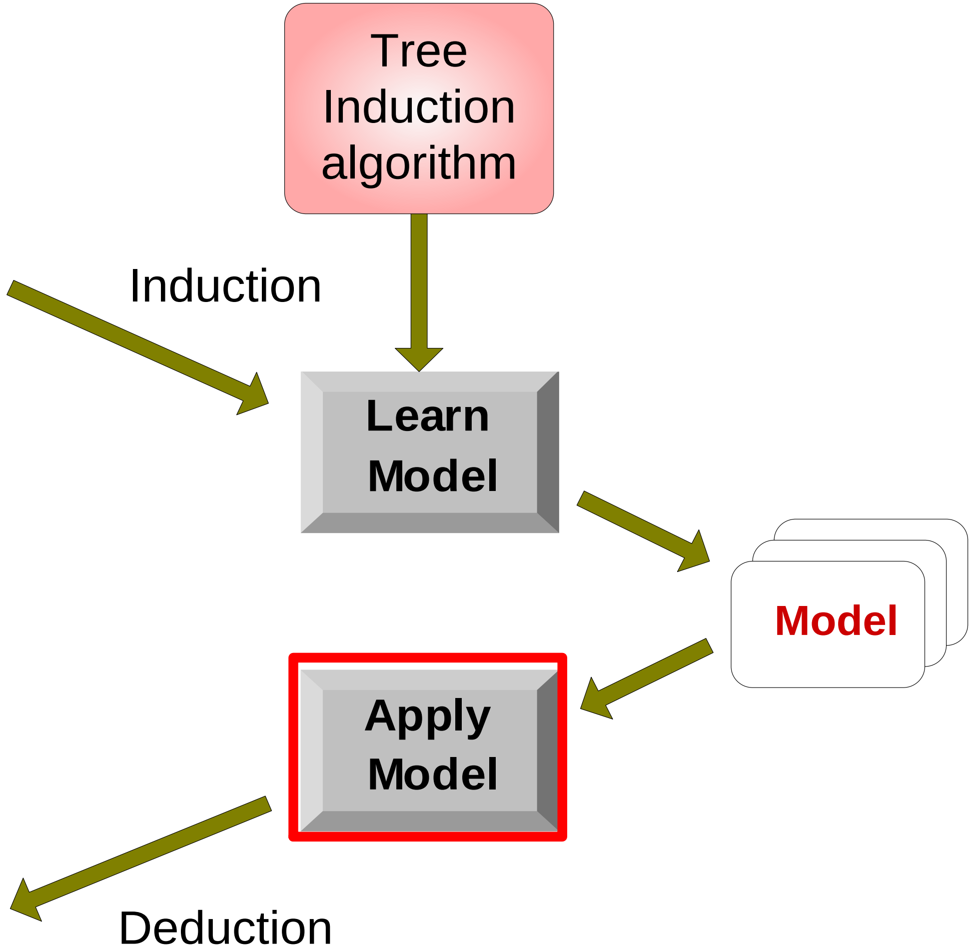
- An inductive learning task
  - Use particular facts to make more generalized conclusions
- A predictive model based on a branching series of Boolean tests
  - These smaller Boolean tests are less complex than a one-stage classifier
- In this decision tree, do a series of Boolean decisions and follow the corresponding branch
  - Did we leave at 10 AM?
  - Did a car stall on the road?
  - Is there an accident on the road?
- By answering each of these yes/no questions, we then came to a conclusion on how long our commute might take



Decision Tree Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

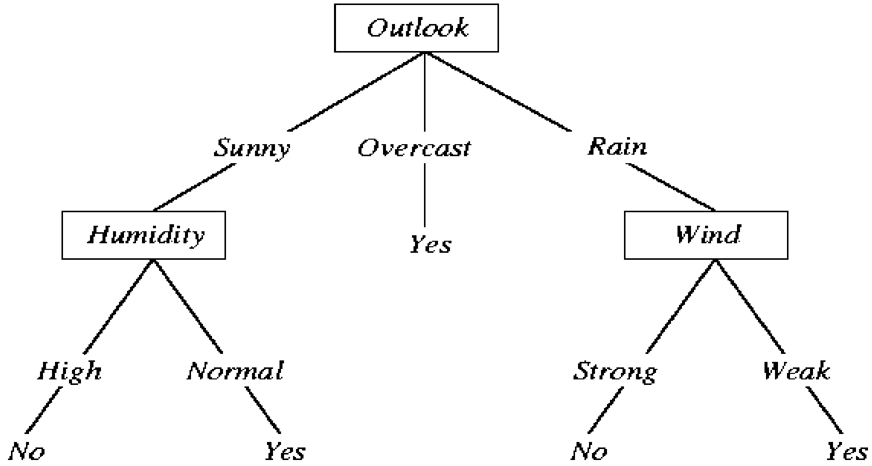


Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

Decision Tree Classification

- Decision tree for Play Tennis Dataset



- Classifies instances into one of a discrete set of possible categories
- Learned function represented by tree
- Each node in tree is test on some attribute of an instance
- Branches represent values of attributes
- Follow the tree from root to leaves to find the output value.

Decision Tree Learning Steps

1. Choose an attribute from our dataset.
2. Calculate the significance of the attribute in the splitting of the data.
3. Split the data based on the value of the best attribute.
4. Go to step 1

How we can determine attribute for classification ?

Information Gain

- *Information gain* is our metric for how well one attribute A^i classifies the training data.
- Information gain for a particular attribute =
  Information about target function,
  given the value of that attribute.
  (conditional entropy)
- Mathematical expression for information gain:

$$\boxed{Gain(S, A_i) = H(S) - \sum_{v \in Values(A_i)} P(A_i = v) H(S_v)}$$

entropy

Entropy for value v

Entropy (1)

- Measure of randomness and uncertainty.
- For an ensemble of random events: $\{A_1, A_2, \ldots, A_n\}$, occurring with probabilities: $\boldsymbol{z} = \{P(A_1), P(A_2), \ldots, P(A_n)\}$

$$\boxed{H = -\sum_{i=1}^{n} P(A_i)\log_2(P(A_i))}$$

$$\boxed{\text{(Note:} \quad 1 = \sum_{i=1}^{n} P(A_i) \quad \text{and} \quad 0 \le P(A_i) \le 1 \quad)}$$

- *If you consider the self-information of event, i, to be: -log2(P(Ai))*
- *Entropy is weighted average of information carried by each event.*
- *For two states: Positive examples and Negative examples from set S*

$$H(S) = -\, p_{+}\log_2(p_{+}) - p_{-}\log_2(p_{-})$$

Entropy (2)

- If an event always occurs, $P(A_i)=1$, then it carries no information.

  $-\log_2(1) = 0$

- If an event rarely occurs (e.g. $P(A_i)=0.001$), it carries a lot of info.

  $-\log_2(0.001) = 9.97$

- **The less likely the event, the more the information it carries.**

ID3 (Iterative Dichotomizer 3) algorithm

- Calculate the entropy for all training examples
  - positive and negative cases
  - p_+ = #pos/Tot p_- = #neg/Tot
  - $H(S) = -p_+ log_2(p_+) - p_- log_2(p_-)$
- Determine which ***single* attribute** best classifies the training examples using information gain.
  - For each attribute find:
  - Use attribute with greatest information gain as a root

$$\boxed{Gain(S, A_i) = H(S) - \sum_{v \in Values(A_i)} P(A_i = v) H(S_v)}$$

Entropy before split

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

14 cases

9 positive cases

- Step 1: *Calculate* ***entropy*** for all cases:

$N_{Pos} = 9$ $N_{Neg} = 5$ $N_{Tot} = 14$

entropy

$$H(S) = -(9/14)*\log_2(9/14) - (5/14)*\log_2(5/14) = 0.940$$

- Step 2: Loop over all attributes, calculate gain:
  - **Attribute = *Outlook***
    - Loop over values of *Outlook*

Outlook = Sunny

$N_{Pos} = 2$ $N_{Neg} = 3$ $N_{Tot} = 5$

$H(\text{Sunny}) = -(2/5)*\log_2(2/5) - (3/5)*\log_2(3/5) = 0.971$

Outlook = Overcast

$N_{Pos} = 4$ $N_{Neg} = 0$ $N_{Tot} = 4$

$H(\text{Overcast}) = -(4/4)*\log_2 4/4 - (0/4)*\log_2(0/4) = 0.00$

Want to select best separation of values for all selected attributes. Approximate this by selecting an attribute with the highest information gain.

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Outlook = Rain

$N_{Pos} = 3$ $N_{Neg} = 2$ $N_{Tot} = 5$

$H(Rain) = -(3/5)*\log_2(3/5) - (2/5)*\log_2(2/5) = 0.971$

- Calculate **Information Gain** for attribute Outlook

$$\begin{aligned} Gain(S,Outlook) = H(S) &- N_{Sunny}/N_{Tot}*H(Sunny) \\ &- N_{Over}/N_{Tot}*H(Overcast) \\ &- N_{Rain}/N_{Tot}*H(Rainy) \end{aligned}$$

$Gain(S,Outlook) = 0.940 - (5/14)*0.971 - (4/14)*0 - (5/14)*0.971$ $Gain(S,Outlook) = 0.246$

– ***Attribute = Temperature***

- (Repeat process looping over {Hot, Mild, Cool})

$Gain(S,Temperature) = 0.029$

– Attribute = *Humidity*

- (Repeat process looping over {High, Normal})

  Gain(S,*Humidity*) = 0.029

– Attribute = *Wind*

- (Repeat process looping over {Weak, Strong})

  Gain(S,*Wind*) = 0.048

Find attribute with greatest information gain:

Gain(S,*Outlook*) = 0.246, Gain(S,*Temperature*) = 0.029

Gain(S,*Humidity*) = 0.029, Gain(S,*Wind*) = 0.048

∴ Outlook is root node of tree

- Iterate algorithm to find attributes which best classify training examples under the values of the root node
- Example continued
  - Take three subsets:
    - *Outlook* = Sunny (N$_{Tot}$ = 5)
    - *Outlook* = Overcast (N$_{Tot}$ = 4)
    - *Outlook* = Rainy (N$_{Tot}$ = 5)
  - For each subset, repeat the above calculation looping over all attributes other than *Outlook*

– For example:

- *Outlook* = Sunny (N_{Pos} = 2, N_{Neg}=3, N_{Tot} = 5) H=0.971
  - *Temp* = Hot (N_{Pos} = 0, N_{Neg}=2, N_{Tot} = 2) H = 0.0
  - *Temp* = Mild (N_{Pos} = 1, N_{Neg}=1, N_{Tot} = 2) H = 1.0
  - *Temp* = Cool (N_{Pos} = 1, N_{Neg}=0, N_{Tot} = 1) H = 0.0

  Gain(S_{Sunny},*Temperature*) = 0.971 - (2/5)*0 - (2/5)*1 - (1/5)*0

  Gain(S_{Sunny},*Temperature*) = 0.571

  Similarly:

  Gain(S_{Sunny},*Humidity*) = 0.971

  Gain(S_{Sunny},*Wind*) = 0.020

  ∴ Humidity classifies *Outlook*=Sunny instances best and is placed as the node under Sunny outcome.

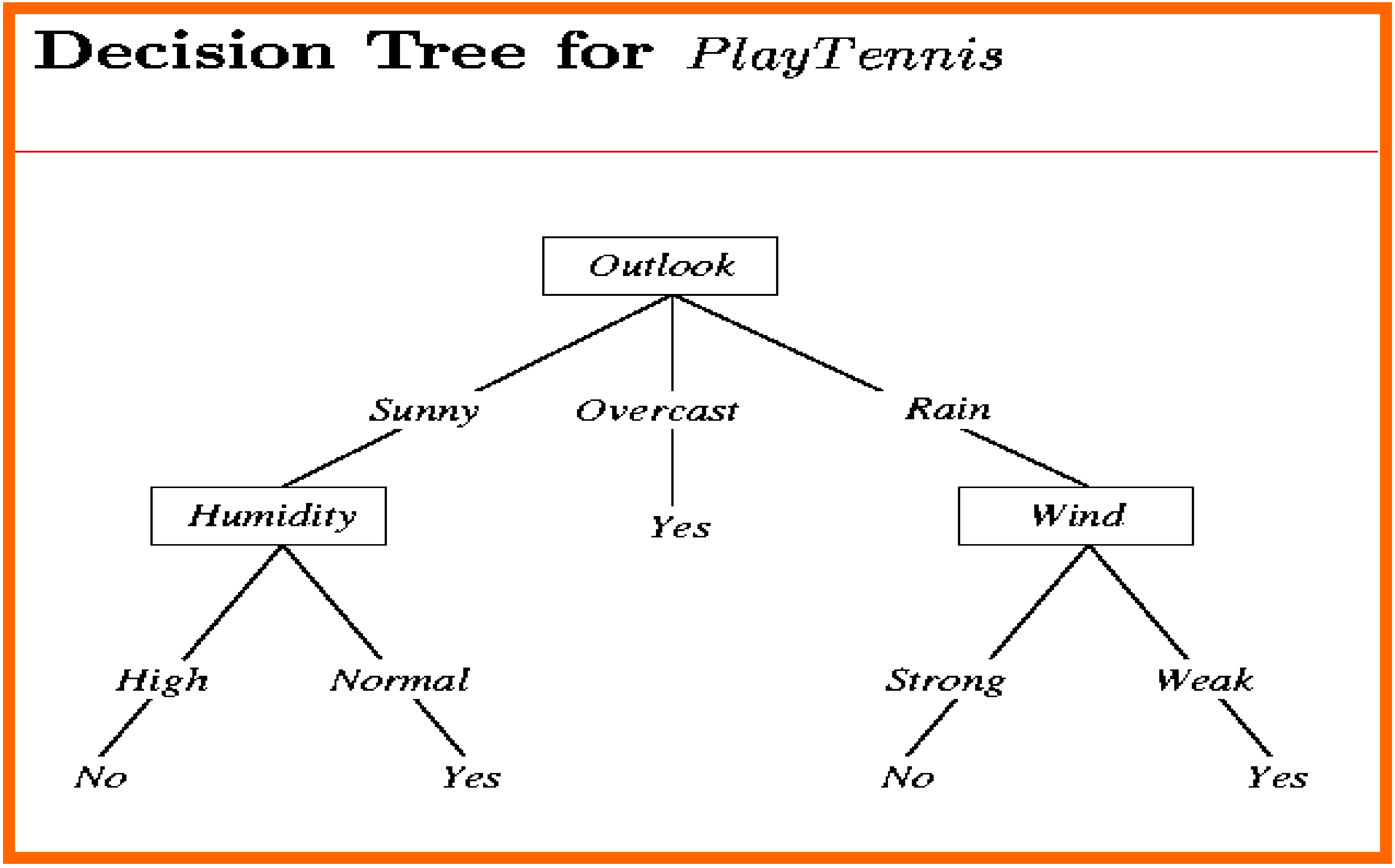
– Repeat this process for *Outlook* = Overcast &Rainy

- **Important:**
  - Attributes are excluded from consideration if they appear higher in the tree
- Process continues for each new leaf node until:
  - Every attribute has already been included along path through the tree

  or

  - Training examples associated with this leaf all have same target attribute value.

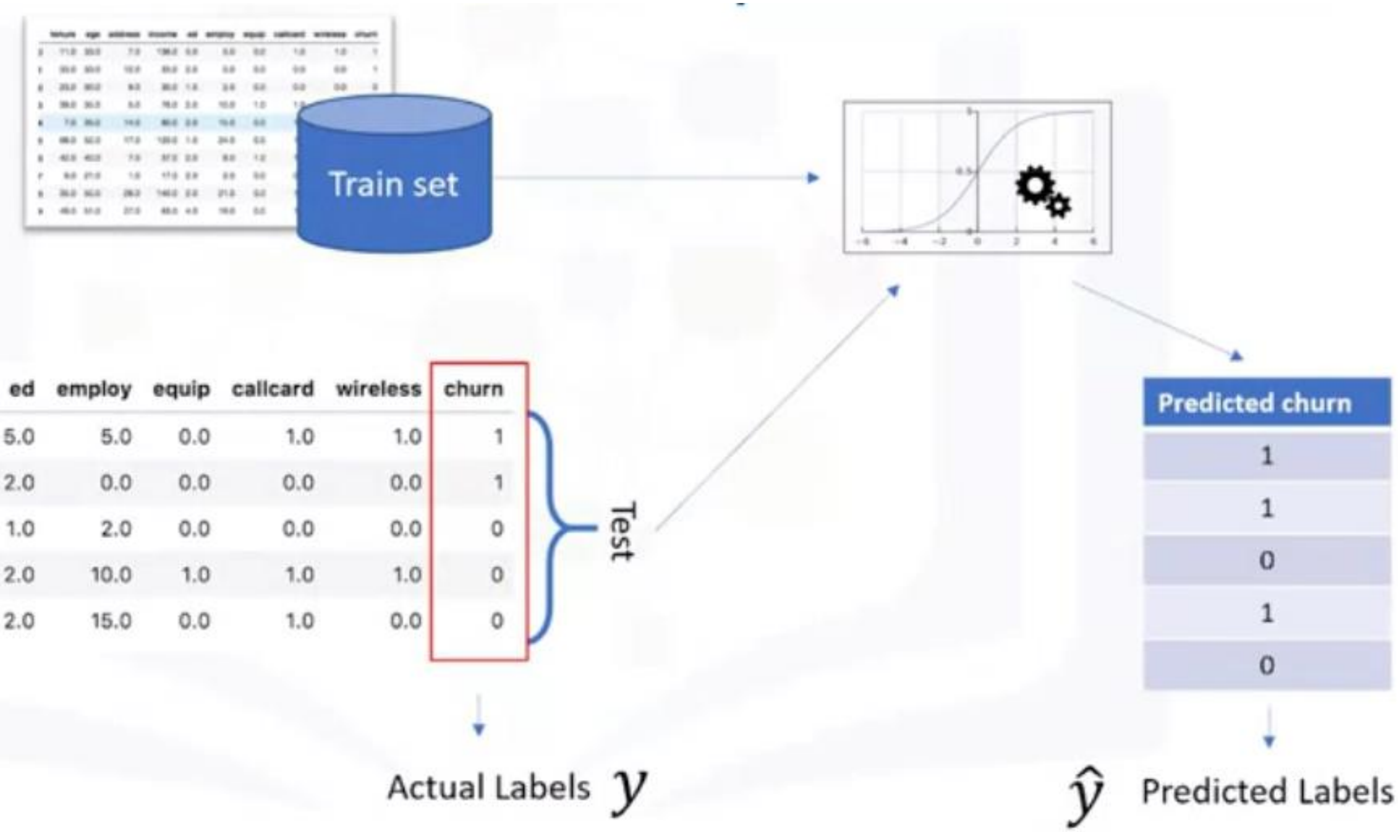
– End up with tree:



Other Decision Tree Learning Algorithm

- CART (Classification and Regression Tree)
- C 4.5

Classification Accuracy



	tenure	age	address	income	ed	employ	equip	callcard	wireless	churn
0	11.0	33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	1
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	1
2	23.0	30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	0
3	38.0	35.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	0
4	7.0	35.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	0

Predicted churn
1
1
0
1
0

Accuracy Measures (1)

- Jacquard Index

y: Actual labels

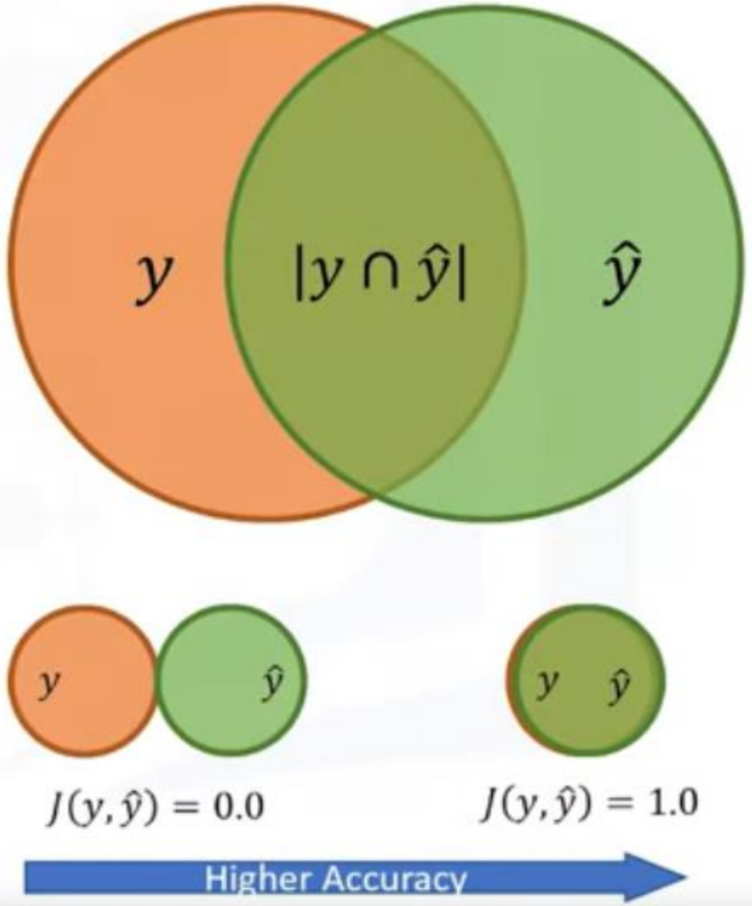
$\hat{y}$: Predicted labels

$$J(y,\hat{y}) = \frac{|y \cap \hat{y}|}{|y \cup \hat{y}|} = \frac{|y \cap \hat{y}|}{|y| + |\hat{y}| - |y \cap \hat{y}|}$$

y: [0, 0, 0, 0, 0, 1, 1, 1, ,1 ,1]

$\hat{y}$: [1, 1, 0, 0, 0, 1, 1, 1, ,1 ,1]

$$J(y,\hat{y}) = \frac{8}{10+10-8} = 0.66$$



Accuracy Measures (2)

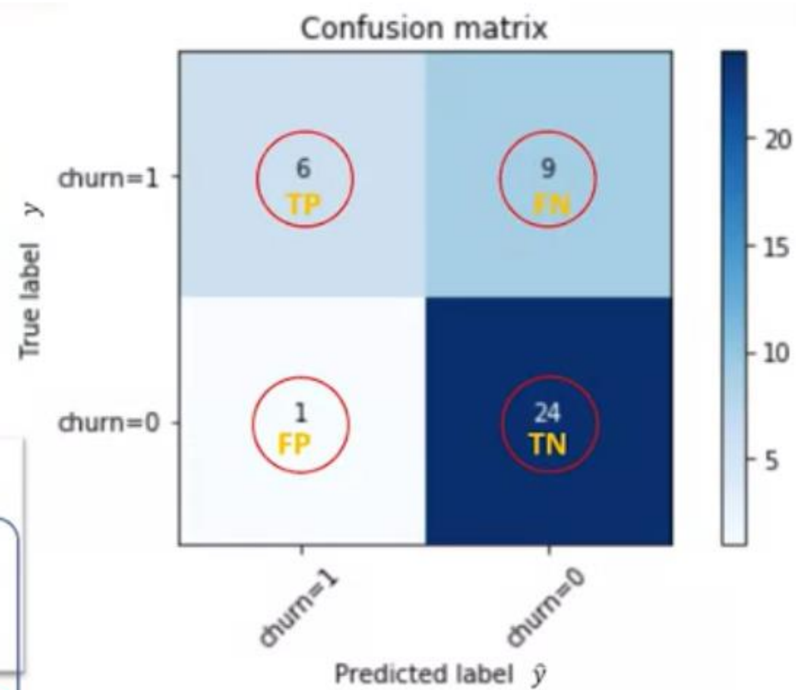
- F1-score
  - Precision = TP / (TP + FP)
  - Recall = TP / (TP + FN) Sensitivity
  - F1-score = 2x (prc x rec) / (prc+rec)

F1-score: 0.00 ... 0.20 .... 0.55 .... 0.83 ... 1.00

Higher Accuracy

	precision	recall	f1-score
Churn = 0	0.73	0.96	0.83
Churn = 1	0.86	0.40	0.55

Avg Accuracy = 0.72

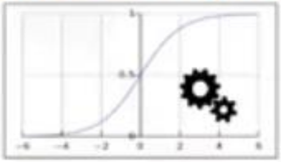


true positive (TP) : A test result that correctly indicates the presence of a condition or characteristic
true negative (TN): A test result that correctly indicates the absence of a condition or characteristic
false positive (FP): A test result which wrongly indicates that a particular condition or attribute is present
false negative (FN): A test result which wrongly indicates that a particular condition or attribute is absent.

Accuracy Measures (3)

- Log Loss

Performance of a classifier where the predicted output is a probability value between 0 and 1.



Test set

	tenure	age	address	income	ed	employ	equip	callcard	wireless	churn
0	11.0	33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	1
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	1
2	23.0	30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	0
3	38.0	35.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	0
4	7.0	35.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	0

Test

Actual Labels y

Predicted churn	LogLoss
0.91	0.11
0.13	2.04
0.04	0.04
0.23	0.26
0.43	0.56

$LogLoss = 0.60$

$\hat{y}$ Predicted Probability

$$LogLoss = -\frac{1}{n}\sum(y \times \log(\hat{y}) + (1-y) \times \log(1-\hat{y}))$$

LogLoss: $\boxed{0.00}$ ... 0.35 .... 0.60 ... $\boxed{1.00}$

Higher Accuracy

Sources

- Chapter 6, Tom M. Mitchell, Machine Learning, The McGraw-Hill Companies, 1st edition 2013.
- http://web.cecs.pdx.edu/~mperkows/CLASS_479/
- Machine Learning course by Ke Chen from University of Manchester and YangQiu Song from MSRA.
- Chapter 4, Tom M. Mitchell, Machine Learning, The McGraw-Hill Companies, 1st edition 2013.
- “Machine learning with Python course”, IBM

innovate achieve lead

BITS Pilani
Dubai Campus

Thank You!