3.0 Classification Methodologies

Eugene Rex L. Jalao, Ph.D.
Associate Professor
Department Industrial Engineering and Operations Research
University of the Philippines Diliman
@thephdataminer

Module 3 of the Business Intelligence and Analytics Certification of UP NEC and the UP Center for Business Intelligence

Outline for This Training

1. Introduction to Data Mining
2. Data Preprocessing
   – Case Study on Big Data Preprocessing using R
3. **Classification Methodologies**
   – **Case Study on Classification using R**
4. Regression Methodologies
   – Case Study: Regression Analysis using R
5. Unsupervised Learning
   – Case Study: Social Media Sentiment Analysis using R

This Session's Outline

- What is Classification?
- Frequency Table
  - Zero R
  - One R
  - Naïve Bayes
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - ANN
  - SVM
- Ensembles
  - Adaboost
  - Random Forests
- Model Evaluation
- Case Study

Required Input Dataset Structure

Attributes/Columns/Variables/Features $(p + 1)$

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	**No**
2	No	Married	100K	**No**
3	No	Single	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	95K	**Yes**
6	No	Married	60K	**No**
7	Yes	Divorced	220K	**No**
8	No	Single	85K	**Yes**
9	No	Married	75K	**No**
10	No	Single	90K	**Yes**

Rows/ Instances /Tuples /Objects (n)

Predictor Variables/Independent Variables/Control Variables

Response Variable/ Dependent Variable/ Class Variable/ Label Variable/ Target Variable

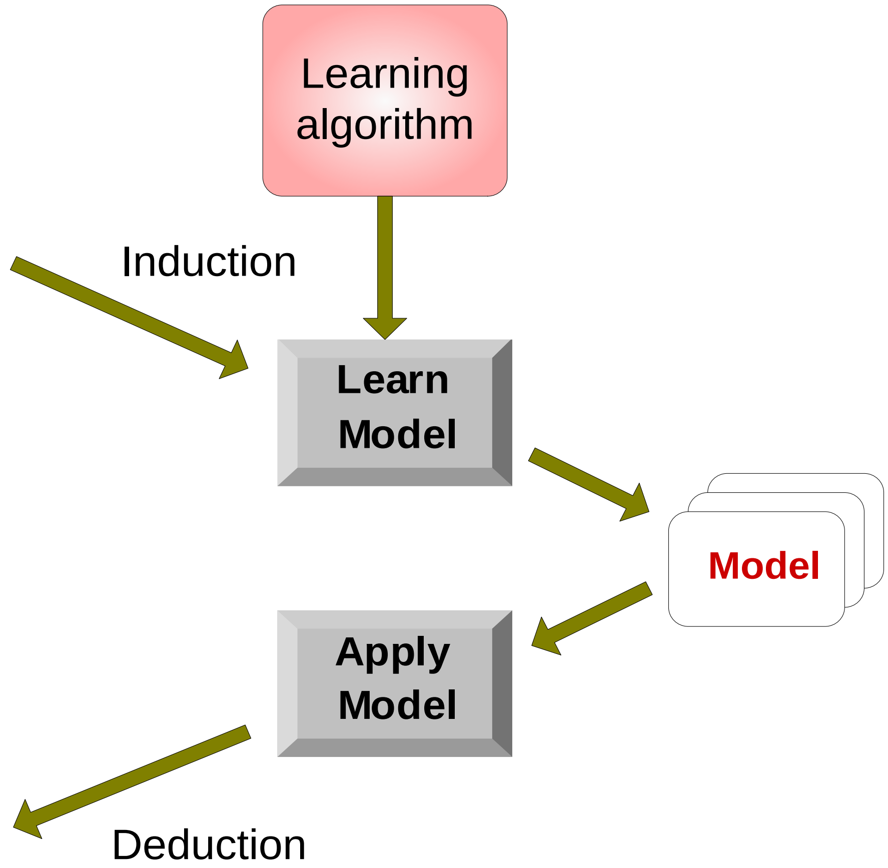
What is Classification?

- Given a collection of records
  - Multiple predictor variables usually $x_1, x_2, \ldots x_p$
  - One categorical response variable usually y
- Find a model for predicting the class variable from the predictor variables.
  - Use historical "training data" to build the model
- Goal: previously unseen records should be predicted a class as accurately as possible.
  - Use "testing data" to test the accuracy of the model

Illustrating 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

Golf Dataset

Outlook	Temperature Nominal	Humidity Nominal	Windy	Play (Class)
overcast	hot	high	false	yes
overcast	cool	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
rainy	mild	normal	false	yes
rainy	mild	high	true	no
sunny	hot	high	false	no
sunny	hot	high	true	no
sunny	mild	high	false	no
sunny	cool	normal	false	yes
sunny	mild	normal	true	yes

This Session's Outline

- What is Classification?
- Frequency Table
  - **Zero R**
  - One R
  - Naïve Bayes
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - ANN
  - SVM
- Ensembles
  - Adaboost
  - Random Forests
- Model Evaluation
- Case Study

ZeroR

- ZeroR is the simplest classification methodology which relies on the target and ignores all predictors.
- Simply predicts the majority class.
- Is useful for determining a baseline performance as a benchmark for other classification methods.
- For the Golf Data Set: Majority is Yes (9 Yes, 5 No)
  - Predict Everything to be Yes: Accuracy = 64%

This Session's Outline

- What is Classification?
- Frequency Table
  - Zero R
  - **One R**
  - Naïve Bayes
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - ANN
  - SVM
- Ensembles
  - Adaboost
  - Random Forests
- Model Evaluation
- Case Study

OneR

- OneR, short for "One Rule"
- Simple, yet accurate, classification algorithm that generates one rule for each predictor in the data
- The rule with the smallest total error as its "one rule“
- Algorithm:
  - For each predictor:
    - For each value of that predictor, make a rule as follows:
      - Count how often each value of target (class) appears
      - Find the most frequent class
      - Make the rule assign that class to this value of the predictor
    - Calculate the total error of the rules of each predictor
  - Choose the predictor with the smallest total error.

Golf Dataset

Outlook	Temperature Nominal	Humidity Nominal	Windy	Play (Class)
overcast	hot	high	false	yes
overcast	cool	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
rainy	mild	normal	false	yes
rainy	mild	high	true	no
sunny	hot	high	false	no
sunny	hot	high	true	no
sunny	mild	high	false	no
sunny	cool	normal	false	yes
sunny	mild	normal	true	yes

Generate Frequency Tables

Outlook Table		Actual Play Golf	
		Yes	No
Outlook	Sunny	2	3
	Overcast	4	0
	Rainy	3	2

Temperature Table		Actual Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1

Humidity Table		Actual Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1

Windy Table		Actual Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3

OneR Model

Humidity Table		Actual Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1

- IF Humidity = High THEN Play Golf = No
- IF Humidity = Normal THEN Play Golf = Yes

Confusion Matrix		Actual Play Golf	
		Yes	No
Predicted	Yes	6	1
	No	3	4

Accuracy = 71%

Prediction Confidence

- Level of Confidence we get for each prediction rule
- Example 1:
  - IF Humidity = High THEN Play Golf = No (4/7)
  - The rule is correct 7/11 times
- Example 2:
  - IF Humidity = Normal THEN Play Golf = No (1/7)
  - The rule is correct 7/8 times
- Hence we are more comfortable using Rule 2 in predicting play golf as compared to Rule 1.

This Session's Outline

- What is Classification?
- Frequency Table
  - Zero R
  - One R
  - **Naïve Bayes**
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - ANN
  - SVM
- Ensembles
  - Adaboost
  - Random Forests
- Model Evaluation
- Case Study

Naïve Bayes Classifier

- Uses a probabilistic framework for classification
- Review: Conditional Probability, Bayes theorem:

$$P(C|A) = P(A|C)\frac{P(C)}{P(A)}$$

- Interpreted as:
  - The probability of C happening given A is true is equal to the Probability of A happening given C is true times the ratio of the probability of C happening and probability of A happening

Example of Bayes Theorem

- Given Historical Data:
  - A doctor knows that if a patient has meningitis, the probability that he has a stiff neck is 50%.
  - Prior probability of any patient having meningitis is 1/50,000
  - Probability of any patient having stiff neck is 1/20
- If a patient has stiff neck, what's the probability he/she has meningitis?

$$P(M = Y|S = Y) = \frac{P(S = Y|M = Y)P(M = Y)}{P(S = Y)}$$

$$= \frac{0.5 * \frac{1}{50000}}{\frac{1}{20}} = 0.0002$$

Naïve Bayes Classifier

- Given a record with p attributes $(A_1, A_2, \dots, A_p)$
  - Goal is to predict class C, specifically, we want to find the value of C that maximizes $P(C|A_1, A_2, \dots, A_p)$
  - Example: $(A_{Outlook} = Rainy, A_{Temp} = Hot)$
- We choose the bigger probability:
  - $P(\text{Play} = \text{Yes}|A_{Outlook} = Rainy, A_{Temp} = Hot)$

    or
  - $P(\text{Play} = \text{No}|A_{Outlook} = Rainy, A_{Temp} = Hot)$
- How can we compute these directly from the data?

Naïve Bayes Classifier

- Using Bayes Formula:

$$P(\text{Play} = \text{Yes}|A_{Outlook} = Rainy, A_{Temp} = Hot) =$$
$$P(A_{Outlook} = Rainy|\text{Play} = \text{Yes})$$
$$* P\left(A_{Temp} = Hot\middle|\text{Play} = \text{Yes}\right) * P(\text{Play} = \text{Yes})/\text{constant}$$

$$P(\text{Play} = No|A_{Outlook} = Rainy, A_{Temp} = Hot) =$$
$$P(A_{Outlook} = Rainy|\text{Play} = \text{No})$$
$$* P\left(A_{Temp} = Hot\middle|\text{Play} = \text{No}\right) * P(\text{Play} = \text{No})/\text{constant}$$

How to Estimate Probabilities from Discrete Data?

Outlook	Temp	Play (Class)
overcast	hot	yes
overcast	cool	yes
overcast	mild	yes
overcast	hot	yes
rainy	mild	yes
rainy	cool	yes
rainy	cool	no
rainy	mild	yes
rainy	mild	no
sunny	hot	no
sunny	hot	no
sunny	mild	no
sunny	cool	yes
sunny	mild	yes

- Class: $P(C) = \frac{N_c}{N}$
  - e.g., $P(No) = 5/14$, $P(Yes) = 9/14$
- For discrete attributes:

$$P(A_i \mid C_k) = \frac{|A_{ik}|}{N_c}$$

  - where $|A_{ik}|$ is number of instances having attribute A_i and belongs to class C_k
  - Examples:

$$P(Outlook = Rainy|Play = Yes) = \frac{3}{9}$$

Example of a Naïve Bayes Classifier

- Given today's Outlook = Rainy, Temperature = Hot, Humidity = High and Windy = False, will I play golf?

  - $\boldsymbol{P(Play = Yes|Outlook = Rainy, Temp = Hot)}$

    $= P(Outlook = Rainy|Play = Yes) * P(Temp = Hot|Play = Yes) * P(Play = Yes)$

  - $\boldsymbol{P(Play = No|Outlook = Rainy, Temp = Hot)}$

    $= P(Outlook = Rainy|Play = No) * P(Temp = Hot|Play = No) * P(Play = No)$

How to Estimate Probabilities from Continuous Data?

- For continuous attributes:
  - Discretize the range into bins
    - one ordinal attribute per bin
    - violates independence assumption
  - Two-way split: $(A < v)$ or $(A > v)$
    - choose only one of the two splits as new attribute
- Example: $(A < 15)$ or $(A \geq 15)$

Original Data	Discretized Data
10	Low
20	High
15	High
11	Low

Naïve Bayes Classifier

- If one of the conditional probability is zero, then the entire expression becomes zero

$$Original: P(A_i|C) = \frac{N_{ic}}{N_c}$$

c: number of classes

p: prior probability

- Laplace Probability Estimation:

$$Laplace: P(A_i|C) = \frac{N_{ic} + 1}{N_c + c}$$

Business Scenario: Nursery Data

- The Nursery Data set originally developed to rank applications into nursery schools
  - parents: usual, pretentious, great_pret
  - has_nurs: proper, less_proper, improper, critical, very_crit
  - form: complete, completed, incomplete, foster
  - children: 1, 2, 3, more
  - housing: convenient, less_conv, critical
  - finance: convenient, inconv
  - social: non-prob, slightly_prob, problematic
  - health: recommended, priority, not_recom
  - *Rank:* not_recom, priority, reocommend, spec_prior, very_recom
- 12960 Profiles

Business Scenario: Nursery Data

- New Data:

parents	has_nur	form	children	housing	finance	social	health	rank
usual	proper	complete	1	convenient	inconv	problematic	priority	?
pretentious	critical	foster	2	critical	convenient	slightly_prob	not_recom	?

Business Scenario: Nursery Data

- Type the following lines of code in RStudio and run.

```
nursery = read.csv("nursery.csv")
NaiveBayes <-
make_Weka_classifier("weka/classifiers/bayes/NaiveBayes")
NBModel <- NaiveBayes(rank ~ parents + has_nur
                      + form  + children
                      + housing + finance + health
                      , data=nursery)
summary(NBModel)
nurserytest = read.csv("nurserytest.csv")
nurserytest$predictions = predict(NBModel, nurserytest)
nurserytest
```

Business Scenario: Nursery Data

- Results

```
> nurserytest
      parents   has_nur     form children     housing
1       usual    proper complete        1  convenient
2 pretentious critical    foster        2    critical
     finance         social     health rank predictions
1     inconv    problematic   priority   NA    priority
2 convenient slightly_prob not_recom   NA   not_recom
```

Advantages and Disadvantages Table: Naïve Bayes

Parameter	Advantages	Disadvantages
Complexity		
Assumptions		
Preprocessing		
Categorical/Numerical		
Parameters		
Speed of Algorithm		
Handling Outliers/Noise		
Handling Missing Data		
Interpretability		
Relative Predictive Power		
Stability of Model		
Optimality of Model		

This Session's Outline

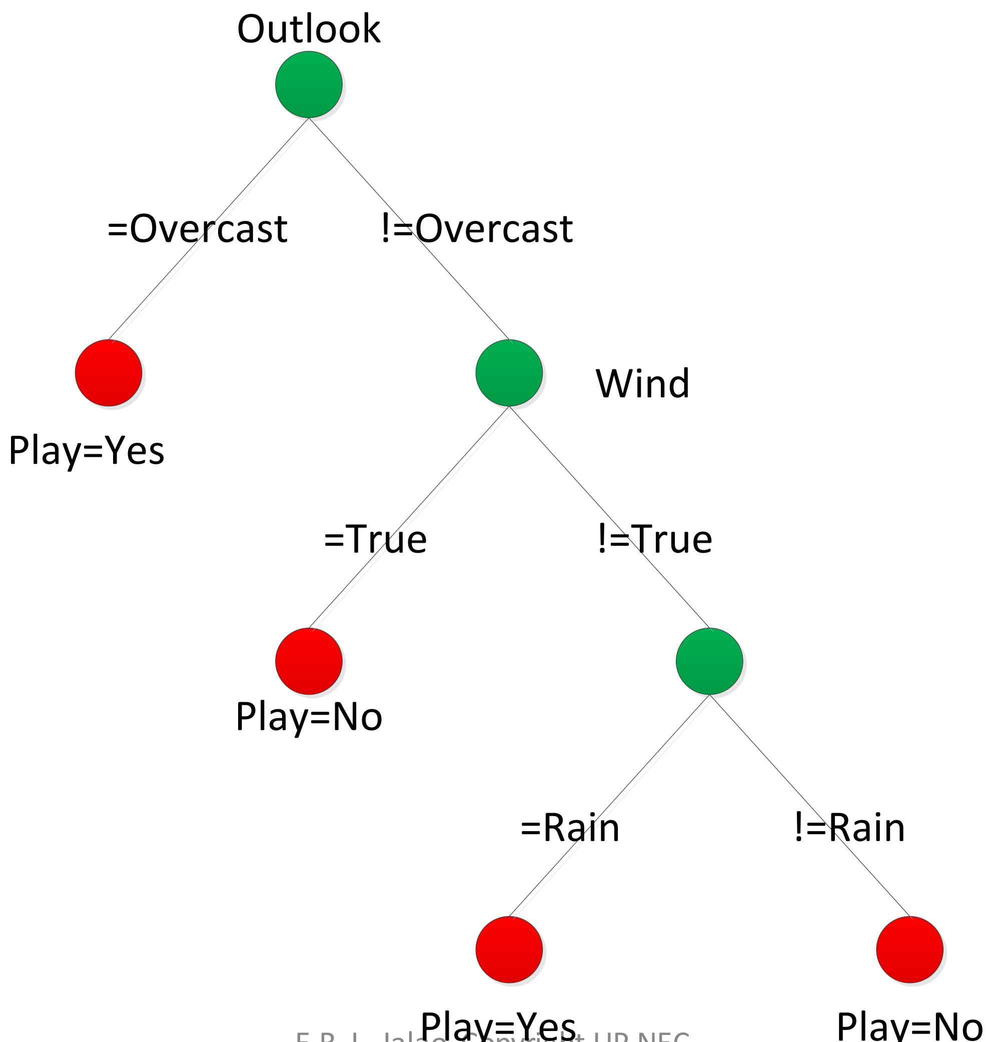
- What is Classification?
- Frequency Table
  - Zero R
  - One R
  - Naïve Bayes
  - **Decision Tree**
  - **Rule Based Classifiers**
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - ANN
  - SVM
- Ensembles
  - Adaboost
  - Random Forests
- Model Evaluation
- Case Study

Decision Trees

- Decision tree builds classification models in the form of a tree structure.
- It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed.
- The final result is a tree with decision nodes and leaf nodes.
  – A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy). Leaf node (e.g., Play) represents a classification or decision.

Decision Trees

- Example



Decision Tree Generation: The ID3 Algorithm

- The core algorithm for building decision trees called **ID3**
- Employs a top-down, greedy search through the space of possible branches with no backtracking.
- ID3 uses Entropy and Information Gain to construct a decision tree.
- A decision tree is built top-down from a parent node and involves partitioning the data into subsets that contain instances with similar values (homogenous).

Definition: Entropy

- Entropy: Measures homogeneity of a node.

$$Gini(t) = 1 - \sum_{j} p(j|t)^2$$

  - Where $p(j \mid t)$ is the relative frequency of class j at node t

The Tax Dataset

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Entropy Calculation Example

- Entropy of Class Tax Cheat

Actual Tax Cheat	
Yes	No
3	7

$$Gini(t) = 1 - \sum_{j} p(j|t)^2$$

$$Gini(t) = 1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2$$

$$Gini(t) = 0.42$$

- If the sample is completely homogeneous (e.g. 10/0) the gini index is zero.
- If the sample is an equally divided (e.g. 5/5) it has entropy of 0.5.

Definition: Information Gain

- Gini Split: Measures Reduction in Gini achieved because of the split.

$$Gini(i)_{Gain} = Gini(p) - \sum_{i=1}^{k} \frac{n_i}{n} Gini(c_i)$$

  – Where parent node, p is split into k partitions; n_i is number of records in partition i

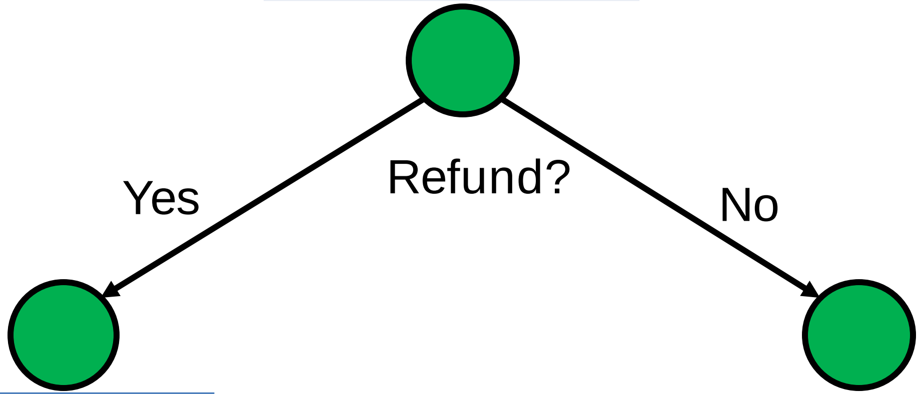
eljalao@up.edu.ph

Example: Gini Split

- Split on Refund:

Actual Tax Cheat	
Yes	No
3	7

$$Gini(t) = 0.42$$



Actual Tax Cheat	
Yes	No
0	3

Actual Tax Cheat	
Yes	No
3	4

Example: Gini Split

- Calculate Gini of Children Nodes

Actual Tax Cheat	
Yes	No
0	3

$$Gini(t_1) = 1 - \left(\frac{3}{3}\right)^2 - \left(\frac{0}{3}\right)^2$$

$$\boldsymbol{Gini(t_1) = 0}$$

Actual Tax Cheat	
Yes	No
3	4

$$Gini(t_2) = 1 - \left(\frac{3}{7}\right)^2 - \left(\frac{4}{7}\right)^2$$

$$\boldsymbol{Gini(t_2) = 0.48}$$

- Gini Child: $Gini\ Split(c) = 0 * \frac{3}{10} + 0.48 * \frac{7}{10} = 0.34$
- Gini Gain: $Gini\ Gain(c) = 0.42 - 0.34 = 0.08$

Computing Gini Index for Continuous Attributes

- For efficient computation: for each attribute,
  - Sort the attribute on values
  - Linearly scan these values, each time updating the count matrix and computing gini index
  - Choose the split position that has the least gini index

Cheat	No	No	No	Yes	Yes	Yes	No	No	No	No
	Taxable Income									
Sorted Values	60	70	75	85	90	95	100	120	125	220

Split Positions	55		65		72		80		87		92		97		110		122		172		230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		***0.300***		0.343		0.375		0.400		0.420	

ID3 Algorithm

- Constructing a decision tree is all about finding attribute that returns the highest information gain (i.e., the most homogeneous branches).

- *Step 1*: Calculate entropy of the parent class.

Actual Tax Cheat	
Yes	No
3	7

$$Gini(t) = 0.42$$

eljalao@up.edu.ph

ID3 Algorithm

- *Step 2*: Calculate the Information Gain for all Attributes.

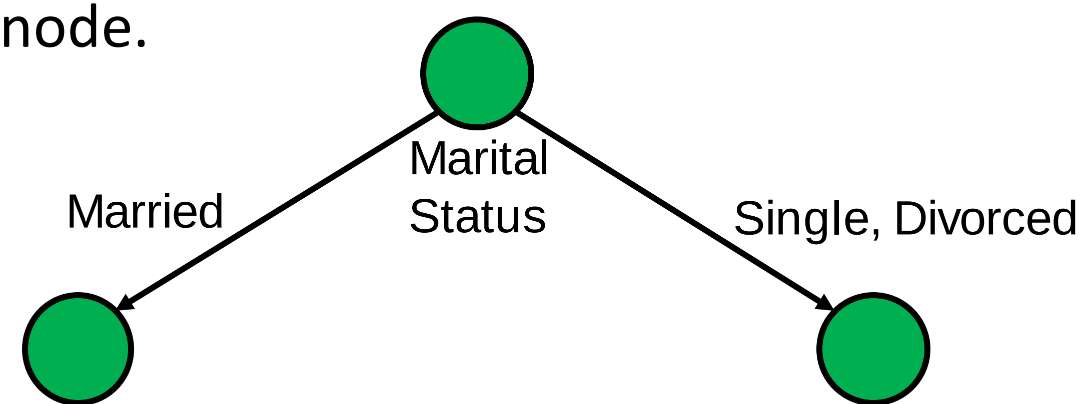
Outlook Table		Actual Tax Cheat	
		Yes	No
Refund	Yes	0	3
	No	3	4
Gain =0.08			

Taxable Income Table		Actual Tax Cheat	
		Yes	No
Taxable Income	> 97 K	0	4
	<= 97K	4	3
Gain = 0.12			

Marital Status Table		Actual Tax Cheat	
		Yes	No
Married	Yes	0	4
	No	3	3
Gain =0.12			

ID3 Algorithm

- *Step 3*: Choose attribute with the largest gini gain as the decision node.

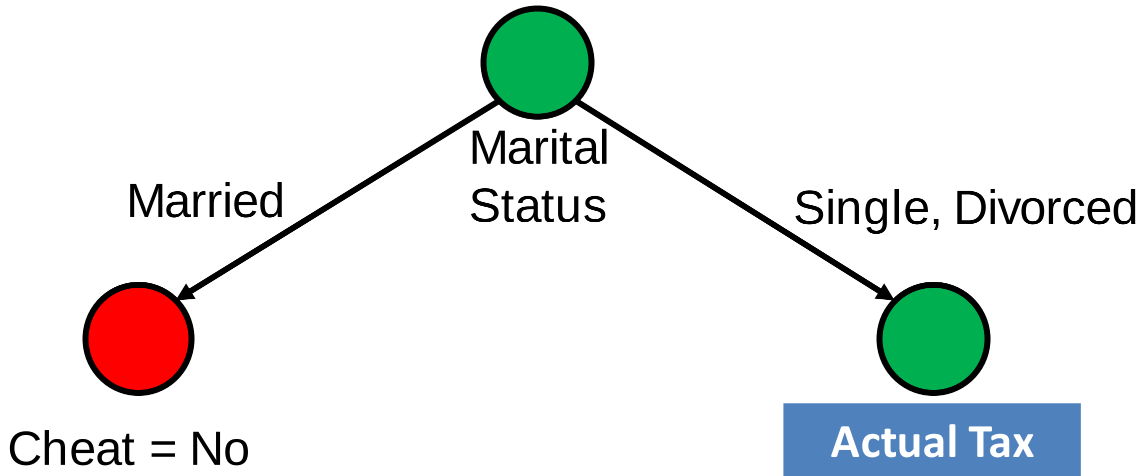


Actual Tax Cheat	
Yes	No
0	4

Actual Tax Cheat	
Yes	No
3	3

ID3 Algorithm

- *Step 4a*: Label a branch if entropy = 0 as a leaf node.



Actual Tax Cheat	
Yes	No
3	3

Information Gain

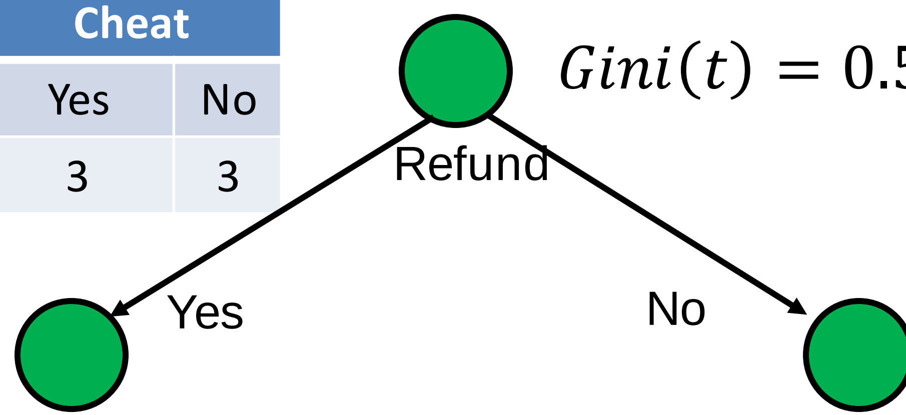
- *Step 4b*: The ID3 algorithm is run recursively on the non-leaf branches, until all data is classified.

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	~~No~~	~~Married~~	~~100K~~	~~No~~
3	No	Single	70K	No
4	~~Yes~~	~~Married~~	~~120K~~	~~No~~
5	No	Divorced	95K	Yes
6	~~No~~	~~Married~~	~~60K~~	~~No~~
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	~~No~~	~~Married~~	~~75K~~	~~No~~
10	No	Single	90K	Yes

Actual Tax Cheat	
Yes	No
3	3

Single, Divorced

$$Gini(t) = 0.5$$

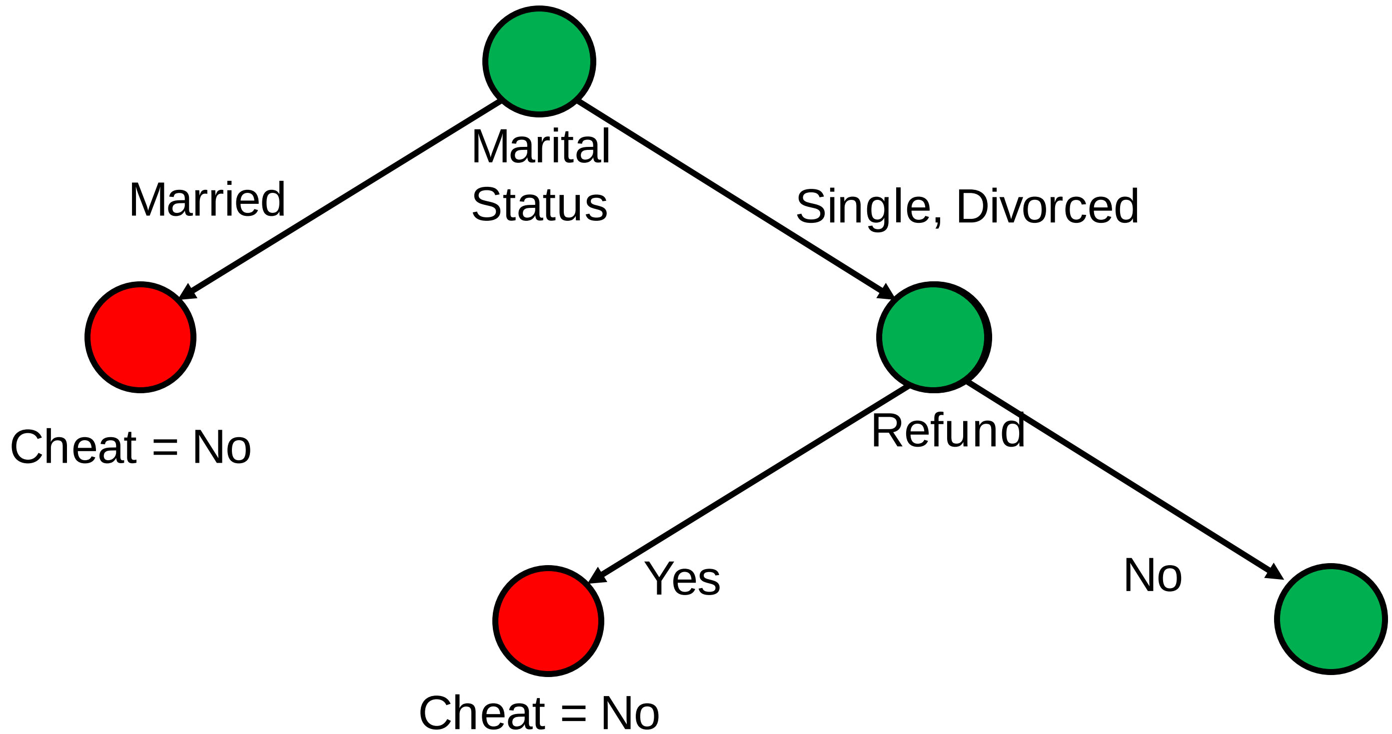


Actual Tax Cheat	
Yes	No
0	2

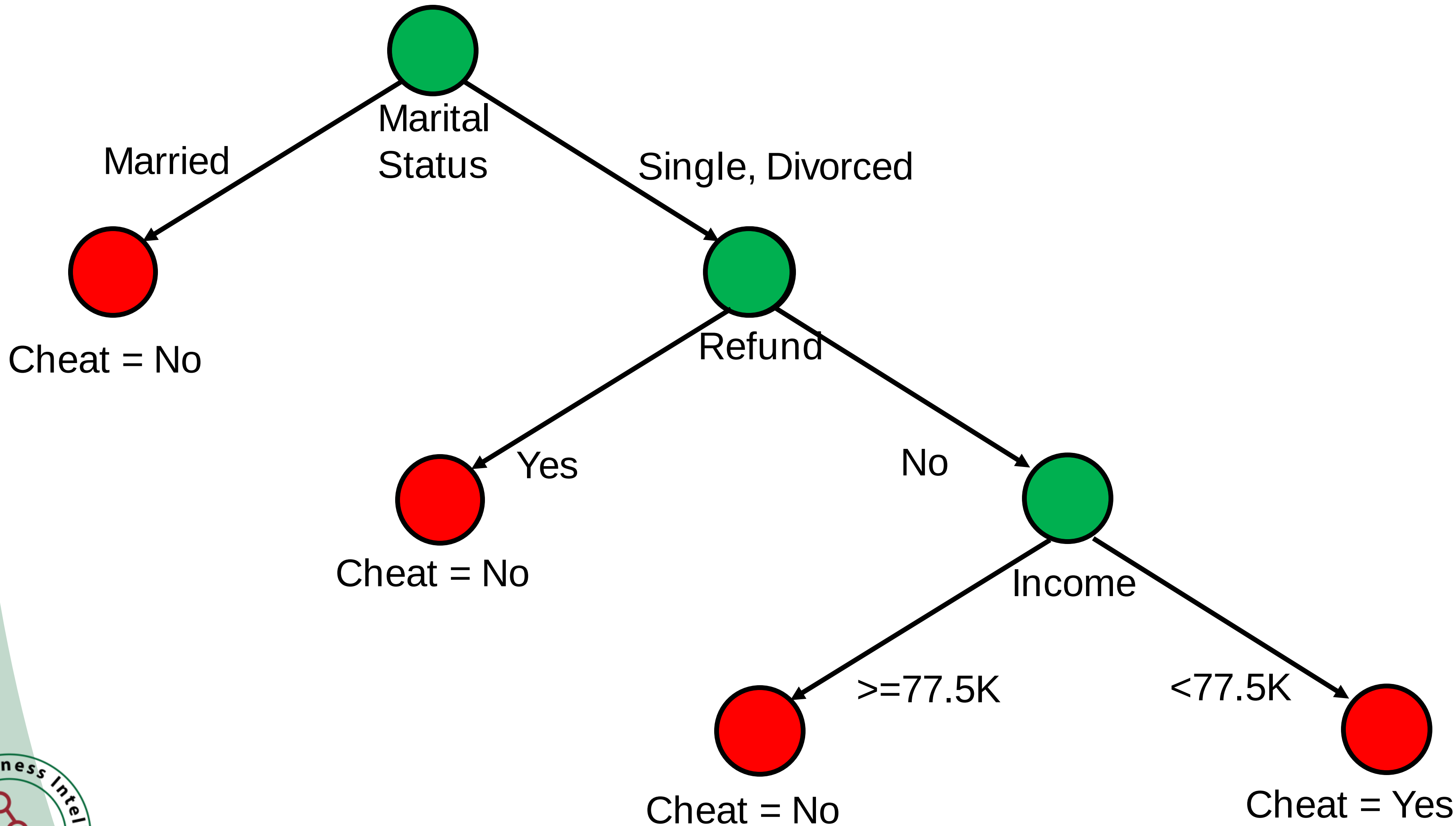
Actual Tax Cheat	
Yes	No
3	1

Information Gain

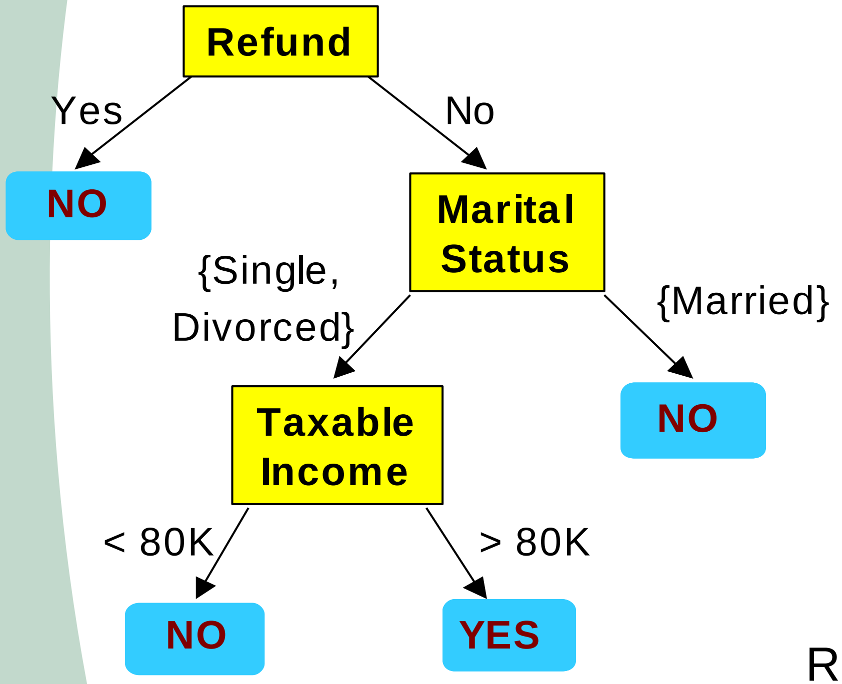
- *Step 5*: The ID3 algorithm is run recursively on the non-leaf branches, until all data is classified.



Final Decision Tree



From Decision Trees To Rules



Classification Rules

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

Rules are mutually exclusive and exhaustive

Rule set contains as much information as the tree

Rule Based Classifiers

- These are classifiers that utilizes **rules** to generate predictions
- Created using the **RIPPER** Algorithm
- Same principle as the decision trees
- Based on frequency tables

Characteristics of Rule-Based Classifier

- **Mutually exclusive** rules
  - Classifier contains mutually exclusive rules if the rules are independent of each other
  - Every record is covered by at most one rule

- **Exhaustive** rules
  - Classifier has exhaustive coverage if it accounts for every possible combination of attribute values
  - Each record is covered by at least one rule

How does Rule-based Classifiers Work?

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

R4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

R5: (Live in Water = sometimes) $\rightarrow$ Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
lemur	warm	yes	no	no	?
turtle	cold	no	no	sometimes	?
dogfish shark	cold	yes	no	yes	?

A lemur triggers rule R3, so it is classified as a mammal

A turtle triggers both R4 and R5

A dogfish shark triggers none of the rules. Usually a default rule is recommended

Rule Coverage and Accuracy

- Coverage of a rule:
  - **Fraction of records** that satisfy the antecedent of a rule
- Accuracy of a rule:
  - Fraction of records that satisfy **both** the antecedent and consequent of a rule

Tid	Refund	Marital Status	Taxable Income	Class
1	Yes	**Single**	125K	**No**
2	No	Married	100K	**No**
3	No	**Single**	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	95K	**Yes**
6	No	Married	60K	**No**
7	Yes	Divorced	220K	**No**
8	No	**Single**	85K	**Yes**
9	No	Married	75K	**No**
10	No	**Single**	90K	**Yes**

If (Status=Single) → Cheat= No

Coverage = 40%, Accuracy = 50%

Business Scenario: Bank Data

- A leading bank's marketing department would like to profile its clients to know which factors lead to the purchase of one of its flagship products: PEP (Personal Equity Plan)
- 600 Clients where gathered from the company's various databases each having variables such as: age, region, income, sex, married, children, car, save_act, current_act, and mortgage

Business Scenario: Bank Data

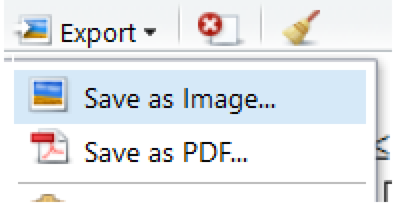
- Type the following lines of code in RStudio and run.

```
bankdata = read.csv("bankdata.csv")
J48Model <- J48(pep ~ age + sex+  region  + income
                + married + children  + car
                + save_act+ current_act+  mortgage
              , data=bankdata)
J48Model
plot(J48Model)
```

```
> J48Model
J48 pruned tree
------------------

children <= 1
|   children <= 0
|   |   married = NO
|   |   |   mortgage = NO: YES (48.0/3.0)
|   |   |   mortgage = YES
|   |   |   |   save_act = NO: YES (12.0)
|   |   |   |   save_act = YES: NO (23.0)
|   |   married = YES
|   |   |   save_act = NO
|   |   |   |   mortgage = NO
|   |   |   |   |   income <= 21506.2
|   |   |   |   |   |   age <= 41: NO (11.0/1.0)
|   |   |   |   |   |   age > 41: YES (5.0/1.0)
|   |   |   |   |   income > 21506.2: NO (20.0)
|   |   |   |   mortgage = YES: YES (25.0/3.0)
|   |   |   save_act = YES: NO (119.0/12.0)
|   children > 0
|   |   income <= 15538.8
|   |   |   age <= 41: NO (22.0/2.0)
|   |   |   age > 41: YES (2.0)
|   |   income > 15538.8: YES (111.0/5.0)
children > 1
|   income <= 30404.3: NO (124.0/12.0)
|   income > 30404.3
|   |   children <= 2: YES (51.0/5.0)
|   |   children > 2
|   |   |   income <= 44288.3: NO (19.0/2.0)
|   |   |   income > 44288.3: YES (8.0)

Number of Leaves  :     15

Size of the tree :      29
```

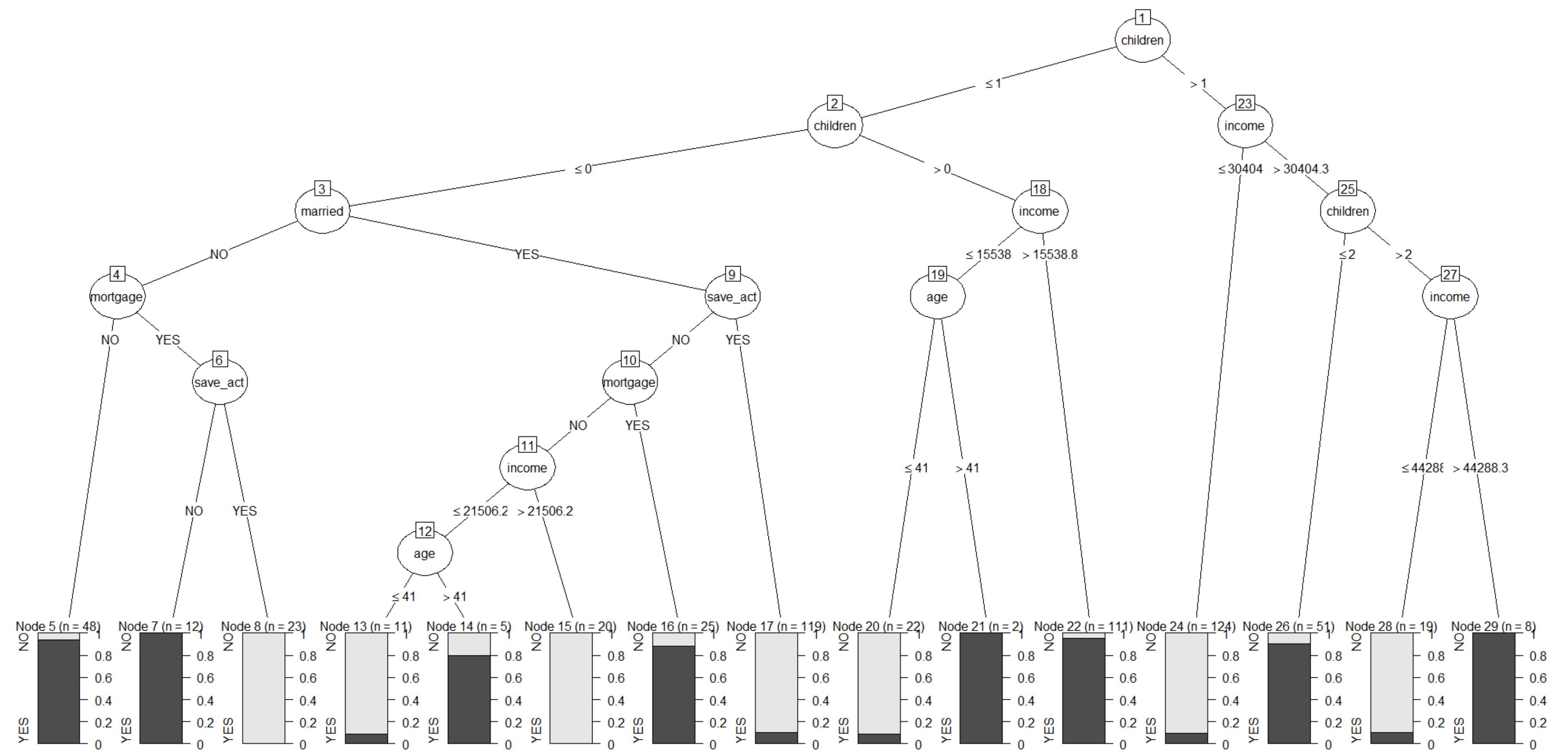
Business Scenario: Bank Data

- To export the plot, click on Export –>Save as Image



- Set the Width to 2000 and Height to 1000.
- Click on Save.
- An image of the plot is saved in the working directory.

Business Scenario: Bank Data



Business Scenario: Bank Data

- Type the following lines of code in RStudio and run.

```
JRipModel <- JRip(pep ~ age + sex+  region  + income
                  + married + children  + car
                  + save_act+ current_act+  mortgage
                  , data=bankdata)
JRipModel
```

Business Scenario: Bank Data

```
> JRipModel
JRIP rules:
===========

(income >= 29714.4) and (children >= 1) and (children <= 2) => pep=YES (102.0/7.0)
(children <= 1) and (save_act = NO) and (mortgage = YES) => pep=YES (46.0/7.0)
(children <= 1) and (children >= 1) and (income >= 15735.8) => pep=YES (53.0/1.0)
(married = NO) and (children <= 0) and (mortgage = NO) => pep=YES (48.0/3.0)
(children >= 1) and (income >= 45031.9) => pep=YES (8.0/0.0)
 => pep=NO (343.0/35.0)

Number of Rules : 6
```

Advantages and Disadvantages Table: Decision Trees/Rule Classifiers

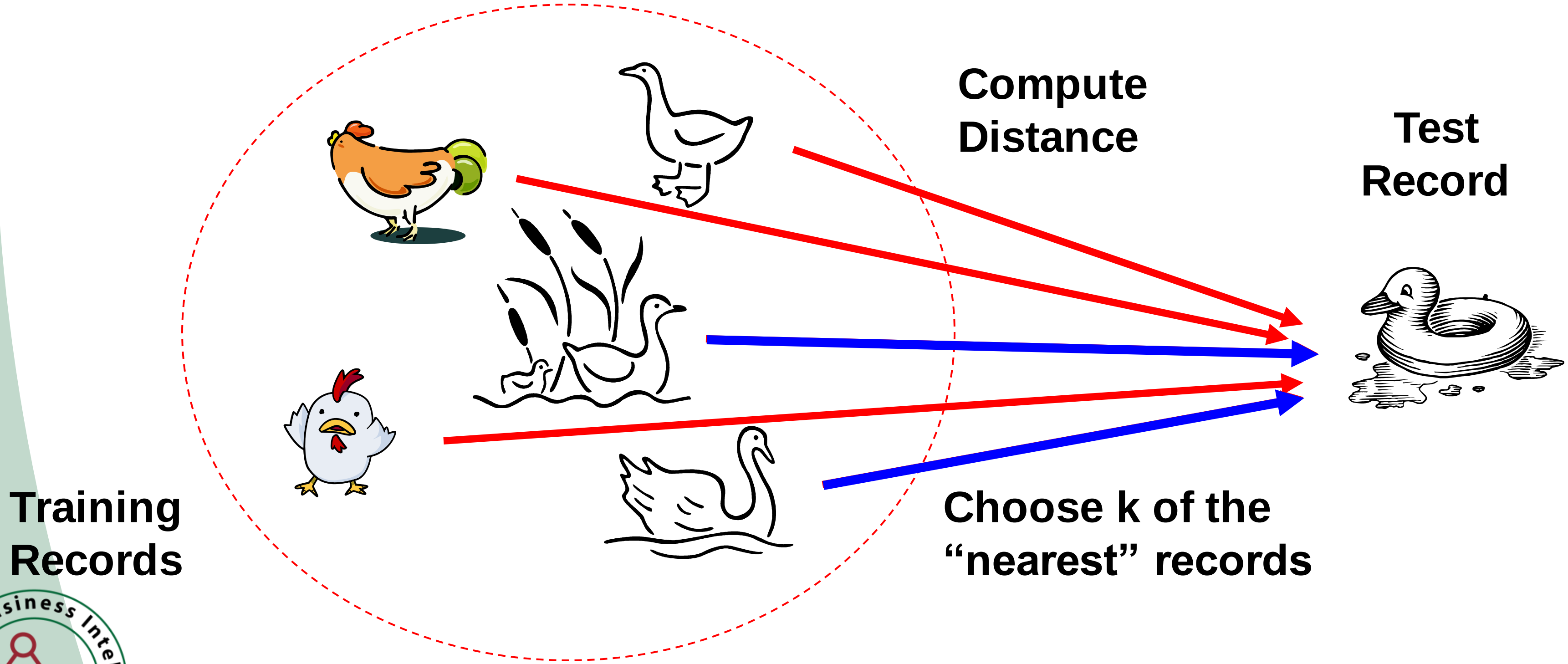
Parameter	Advantages	Disadvantages
Complexity		
Assumptions		
Preprocessing		
Categorical/Numerical		
Parameters		
Speed of Algorithm		
Handling Outliers/Noise		
Handling Missing Data		
Interpretability		
Relative Predictive Power		
Stability of Model		
Optimality of Model		

This Session's Outline

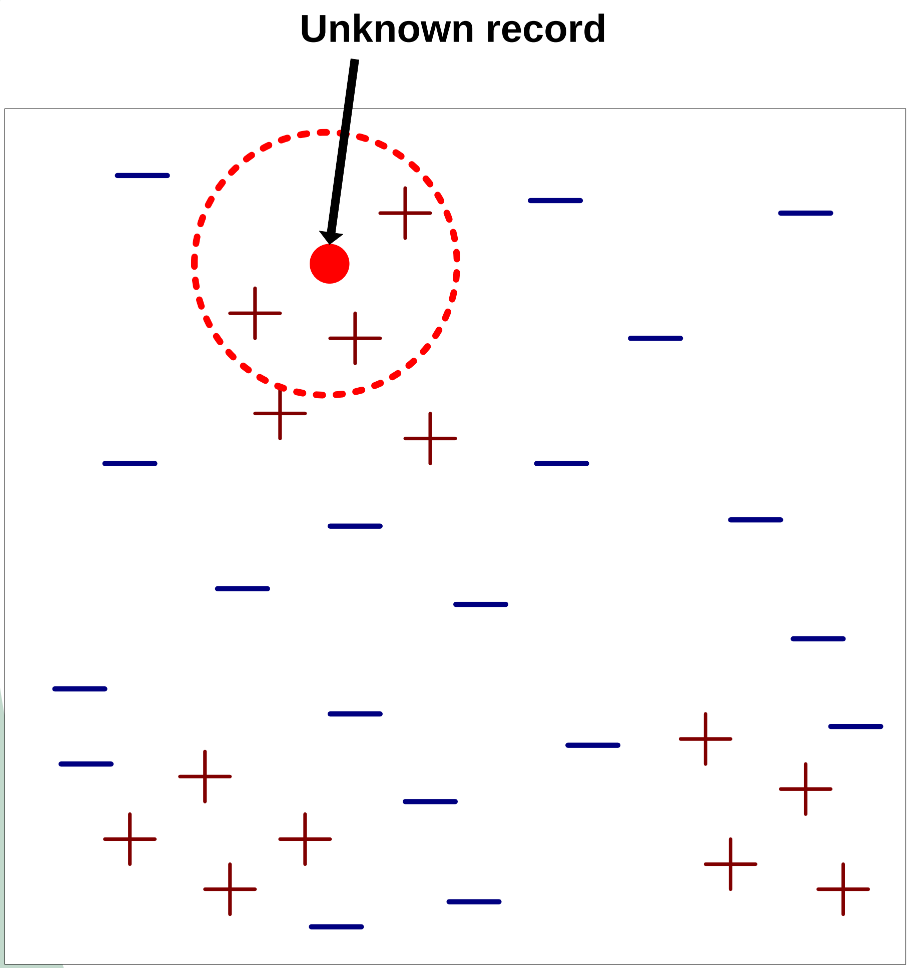
- What is Classification?
- Frequency Table
  - Zero R
  - One R
  - Naïve Bayes
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - **K-Nearest Neighbors**
- Perceptron Based
  - ANN
  - SVM
- Ensembles
  - Adaboost
  - Random Forests
- Model Evaluation
- Case Study

Nearest Neighbor Classifiers

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck

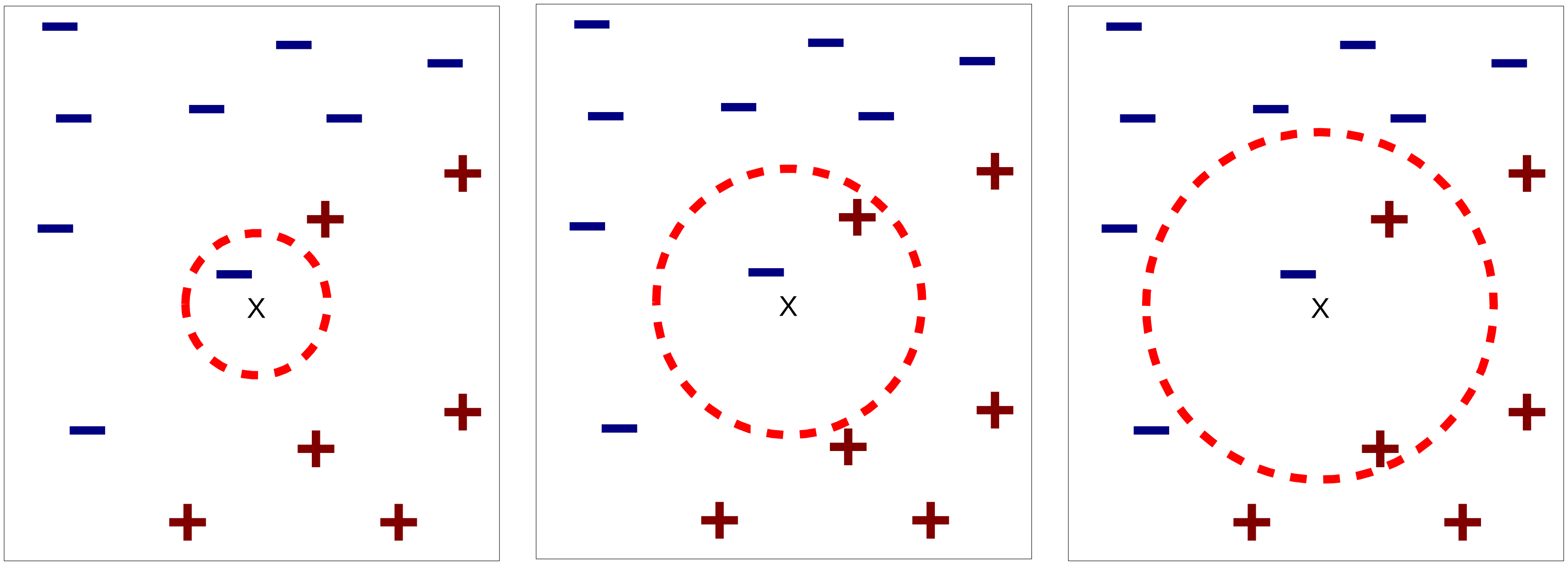


Nearest Neighbor Classifiers



- Requires three things
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve
- To classify an unknown record:
  - Compute distance to other training records
  - Identify *k* nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

Definition of Nearest Neighbor



(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

K-nearest neighbors of a record x are data points that have the k smallest distance to x

Nearest Neighbor Classification

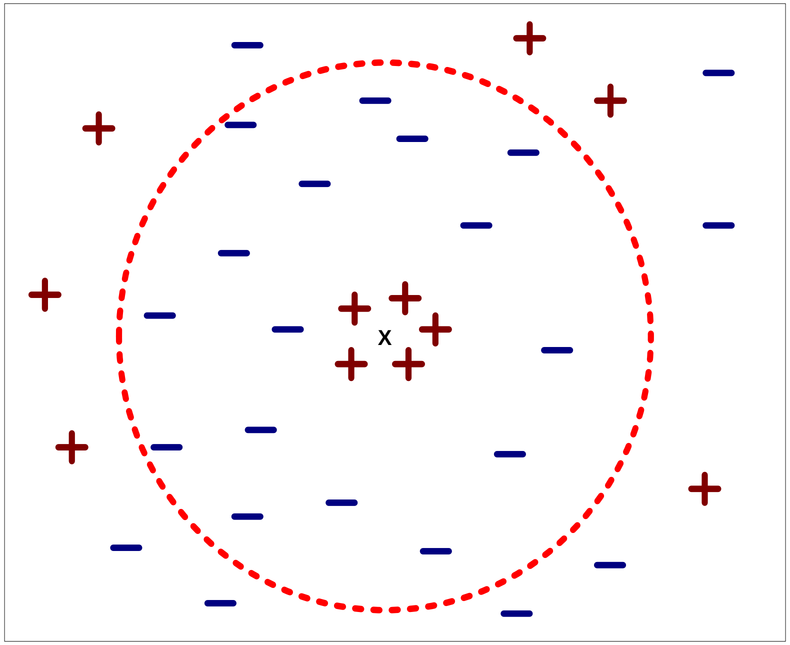
- Compute distance between two points:
  - Euclidean distance

$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- Determine the class from nearest neighbor list
  - take the majority vote of class labels among the k-nearest neighbors
  - Weigh the vote according to distance
    - weight factor, $w = \frac{1}{d^2}$

Nearest Neighbor Classification

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes
  - Good Value for $k = \sqrt{n}$



Business Scenario: Delivery Time Data

- Management would like to determine whether a new order for soft drinks would be either a "Fast," "Medium" or "Slow" delivery from the Number of Cases and Distance.
- 25 Historical Deliveries where gathered and profiled whether it was "Fast," "Medium" or "Slow"

deltime	ncases	deldistance
Medium	7	560
Fast	3	220
Fast	3	340
Fast	4	80
Fast	6	150
Medium	7	330
Fast	2	110
Medium	7	210
Slow	30	1460
Slow	5	605
Slow	16	688
Slow	10	215
Fast	4	255
Medium	6	462
Slow	9	448
Slow	10	776
Fast	6	200
Medium	7	132
Fast	3	36
Slow	17	770
Medium	10	140
Slow	26	810
Medium	9	450
Medium	8	635
Fast	4	150

Delivery Time Data: 1 Nearest Neighbor

- New Order:
  - 11 Cases
  - 500 ft Distance
- Standardized New Order
  - **0.33 SCases**
  - **0.28 SDistance**
- Distance =

$$\sqrt{(0.33-(-0.26))^2 + (0.28-0.46)^2}$$

  - **Distance = 0.616**
- Closest Neighbor:
  - 9 Cases
  - 450 Ft Distance
- Prediction
  - Medium Delivery Time

deltime	ncases	deldistance	sncases	sdeldistance	eucdistance
Medium	**7**	**560**	**-0.26**	**0.46**	**0.616**
Fast	3	220	-0.84	-0.58	1.447
Fast	3	340	-0.84	-0.21	1.262
Fast	4	80	-0.69	-1.01	1.644
Fast	6	150	-0.40	-0.80	1.299
Medium	7	330	-0.26	-0.24	0.782
Fast	2	110	-0.98	-0.92	1.774
Medium	7	210	-0.26	-0.61	1.064
Slow	30	1460	3.09	3.23	4.042
Slow	5	605	-0.55	0.60	0.930
Slow	16	688	1.05	0.86	0.929
Slow	10	215	0.18	-0.60	0.888
Fast	4	255	-0.69	-0.47	1.266
Medium	6	462	-0.40	0.16	0.736
Slow	9	448	0.03	0.12	0.332
Slow	10	776	0.18	1.13	0.861
Fast	6	200	-0.40	-0.64	1.174
Medium	7	132	-0.26	-0.85	1.272
Fast	3	36	-0.84	-1.15	1.840
Slow	17	770	1.20	1.11	1.204
Medium	10	140	0.18	-0.83	1.117
Slow	26	810	2.51	1.23	2.379
Medium	9	450	0.03	0.13	0.329
Medium	8	635	-0.11	0.69	0.602
Fast	4	150	-0.69	-0.80	1.481

Delivery Time Data: 5 Nearest Neighbors

- New Order:
  - 11 Cases
  - 500 ft Distance
- Standardized
  - 0.33 SCases
  - 0.28 SDistance
- Closest Neighbors:
  - 4 Medium
  - 1 Slow
- Prediction
  - Medium Delivery Time

deltime	ncases	deldistance	sncases	sdeldistance	eucdistance
Medium	7	560	-0.26	0.46	0.610
Fast	3	220	-0.84	-0.58	1.447
Fast	3	340	-0.84	-0.21	1.262
Fast	4	80	-0.69	-1.01	1.644
Fast	6	150	-0.40	-0.80	1.299
Medium	7	330	-0.26	-0.24	0.782
Fast	2	110	-0.98	-0.92	1.774
Medium	7	210	-0.26	-0.61	1.064
Slow	30	1460	3.09	3.23	4.042
Slow	5	605	-0.55	0.60	0.930
Slow	16	688	1.05	0.86	0.929
Slow	10	215	0.18	-0.60	0.888
Fast	4	255	-0.69	-0.47	1.266
Medium	6	462	-0.40	0.16	0.736
Slow	9	448	0.03	0.12	0.332
Slow	10	776	0.18	1.13	0.861
Fast	6	200	-0.40	-0.64	1.174
Medium	7	132	-0.26	-0.85	1.272
Fast	3	36	-0.84	-1.15	1.840
Slow	17	770	1.20	1.11	1.204
Medium	10	140	0.18	-0.83	1.117
Slow	26	810	2.51	1.23	2.379
Medium	9	450	0.03	0.13	0.329
Medium	8	635	-0.11	0.69	0.602
Fast	4	150	-0.69	-0.80	1.481

Center for Business Intelligence

Advantages and Disadvantages Table: KNN

Parameter	Advantages	Disadvantages
Complexity		
Assumptions		
Preprocessing		
Categorical/Numerical		
Parameters		
Speed of Algorithm		
Handling Outliers/Noise		
Handling Missing Data		
Interpretability		
Relative Predictive Power		
Stability of Model		
Optimality of Model		

This Session's Outline

- What is Classification?
- Frequency Table
  - Zero R
  - One R
  - Naïve Bayes
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - **ANN**
  - SVM
- Ensembles
  - Adaboost
  - Random Forests
- Model Evaluation
- Case Study

The Perceptron

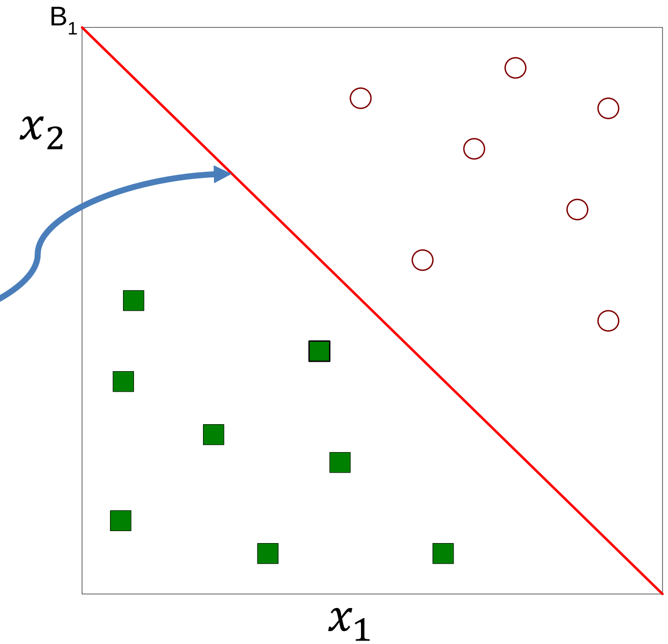
- Let $x_1, x_2, \ldots x_p$ be numerical variables
- Let a binary $y = Green\ or\ Red$
- Given data
  - Find a line that separates the data

$$\beta_o + \beta_1 x_1 + .. + \beta_p x_p = 0$$

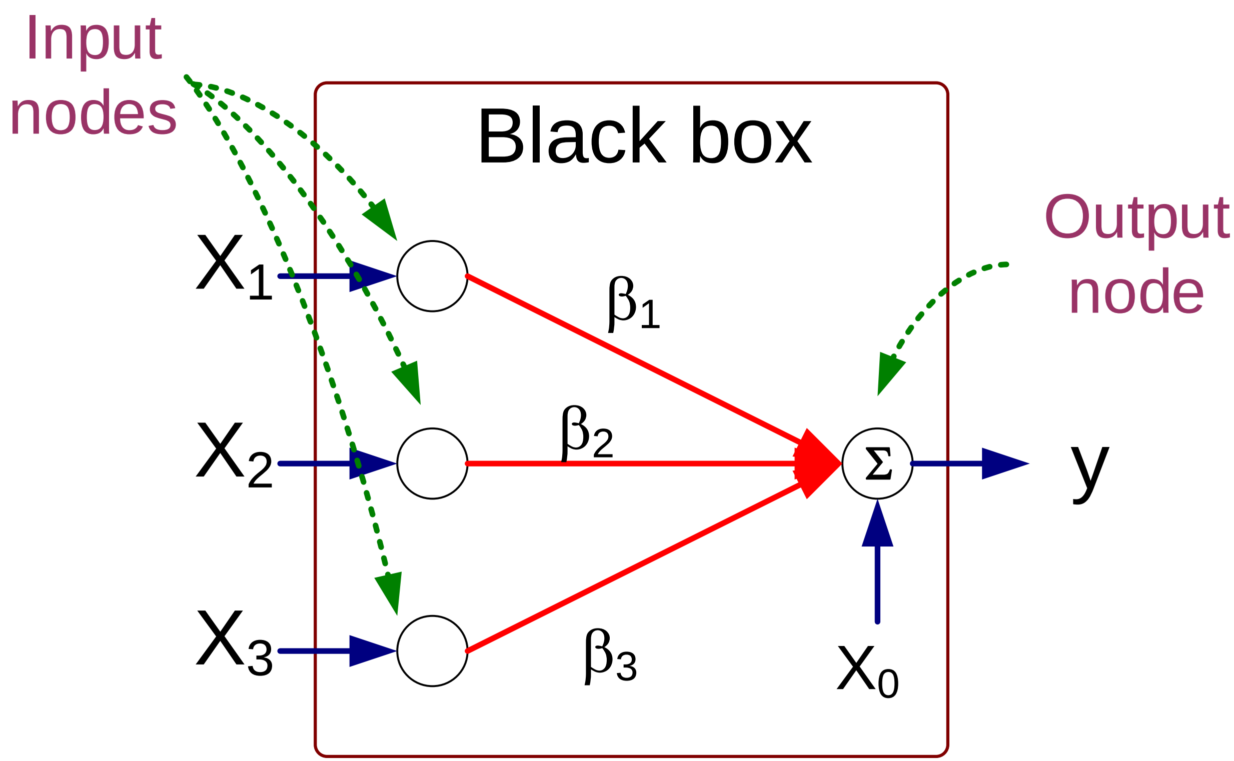
$$or$$

$$\beta^T x = 0$$

- Predict $\hat{y} = \begin{cases} Green & if\ (\beta^T x) < 0 \\ Red & if\ (\beta^T x) > 0 \end{cases}$



Perceptron in a Nutshell



$$\hat{y} = sign(\beta^T x)$$

Perceptron Learning Algorithm

- Find weights β using the Stochastic Gradient Descent methodology
  - Start with random β values
  - Update the β values as follows

$$\beta_{k+1} = \beta_k + \lambda(y_i - \hat{y}_i)(x_i)$$

  - λ is called the Step Size
    - λ may decrease with the number of iterations
- Iterate through each row and adjust β for each point misclassified

Simple Perceptron Induction Example

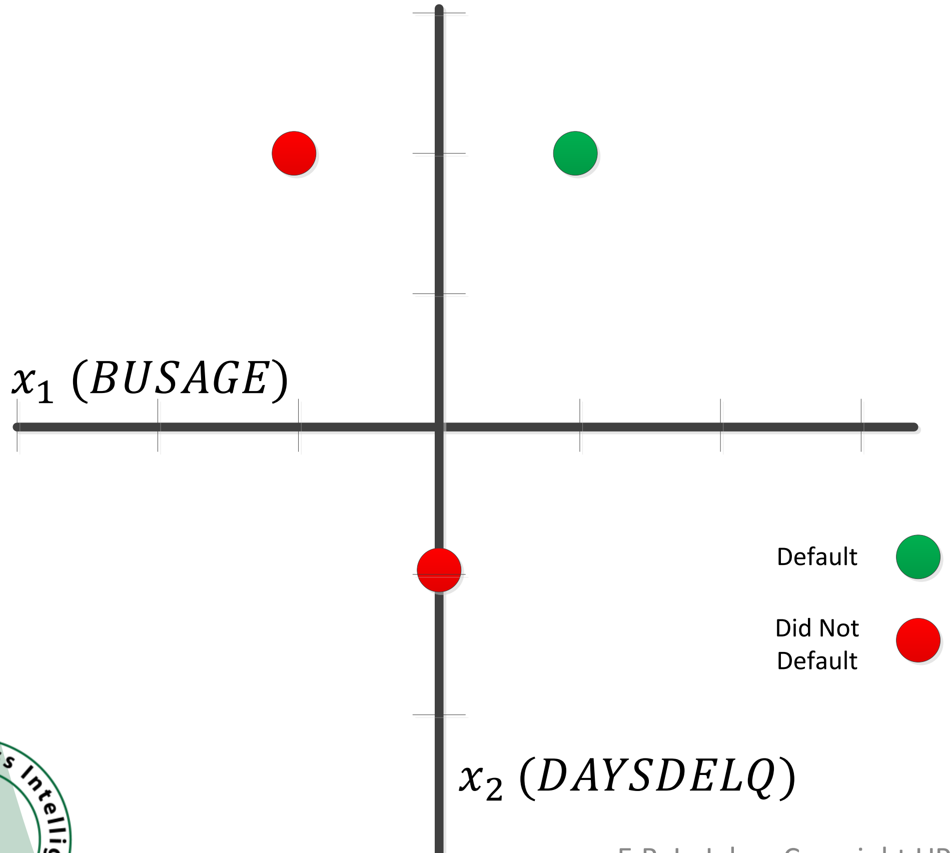
- Example: the training data set contains three examples
- Find a single-neuron perceptron

Business Age (Ave = 10, SD=3)	Number of Days Delinquent (Ave = 50, SD=10)	Default ?
13	70	Yes
7	70	No
10	40	No

sBUSAGE	sDAYSDELQ	DEFAULT
1	2	Yes
-1	2	No
0	-1	No

Simple Perceptron Induction Example

- Graphically:

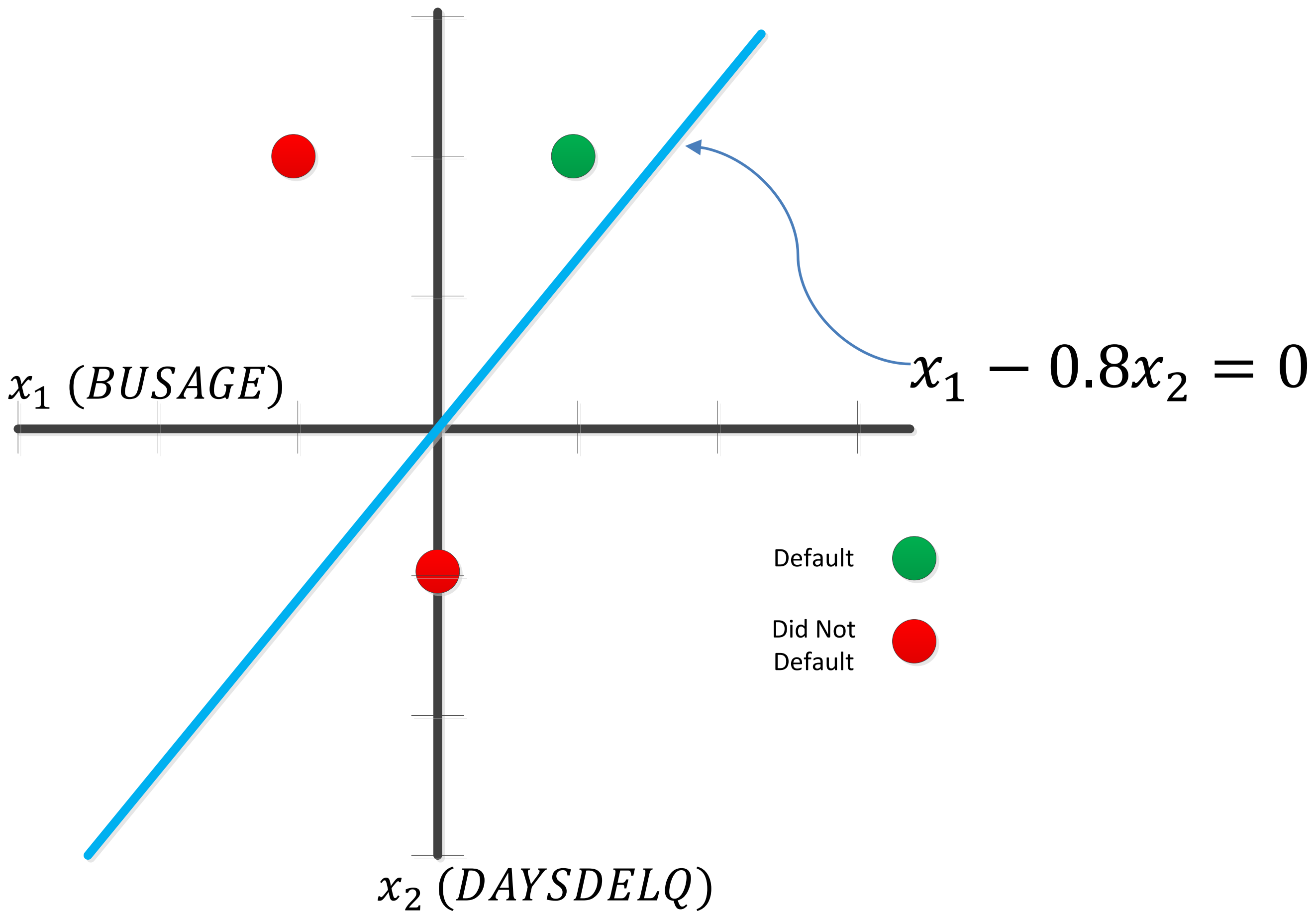


- Algebraically:

$$\left\{x = \begin{bmatrix}1\\2\end{bmatrix}, y = 1\right\}$$
$$\left\{x = \begin{bmatrix}-1\\2\end{bmatrix}, y = -1\right\}$$
$$\left\{x = \begin{bmatrix}0\\-1\end{bmatrix}, y = -1\right\}$$

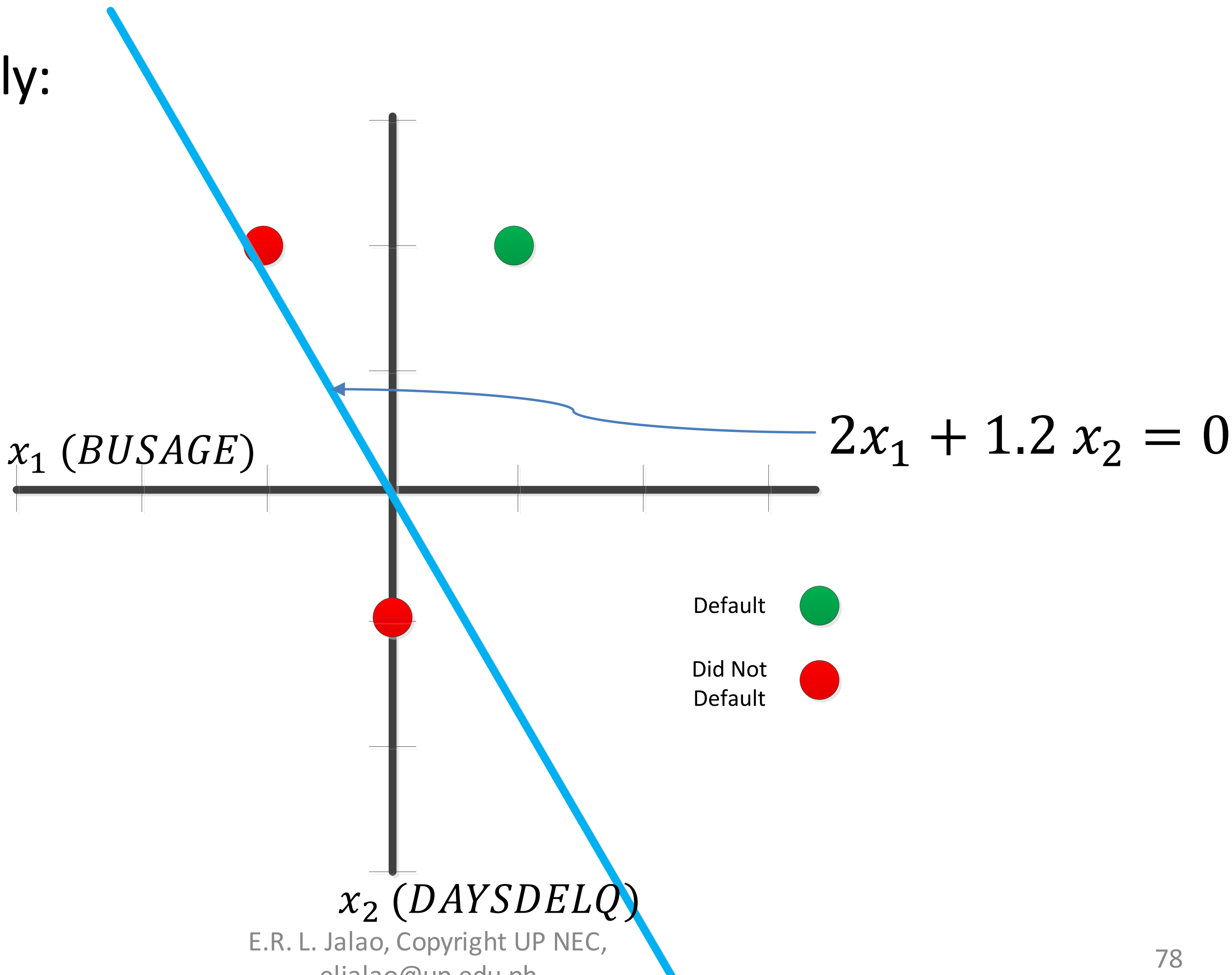
Simple Perceptron Induction Example

- Graphically:



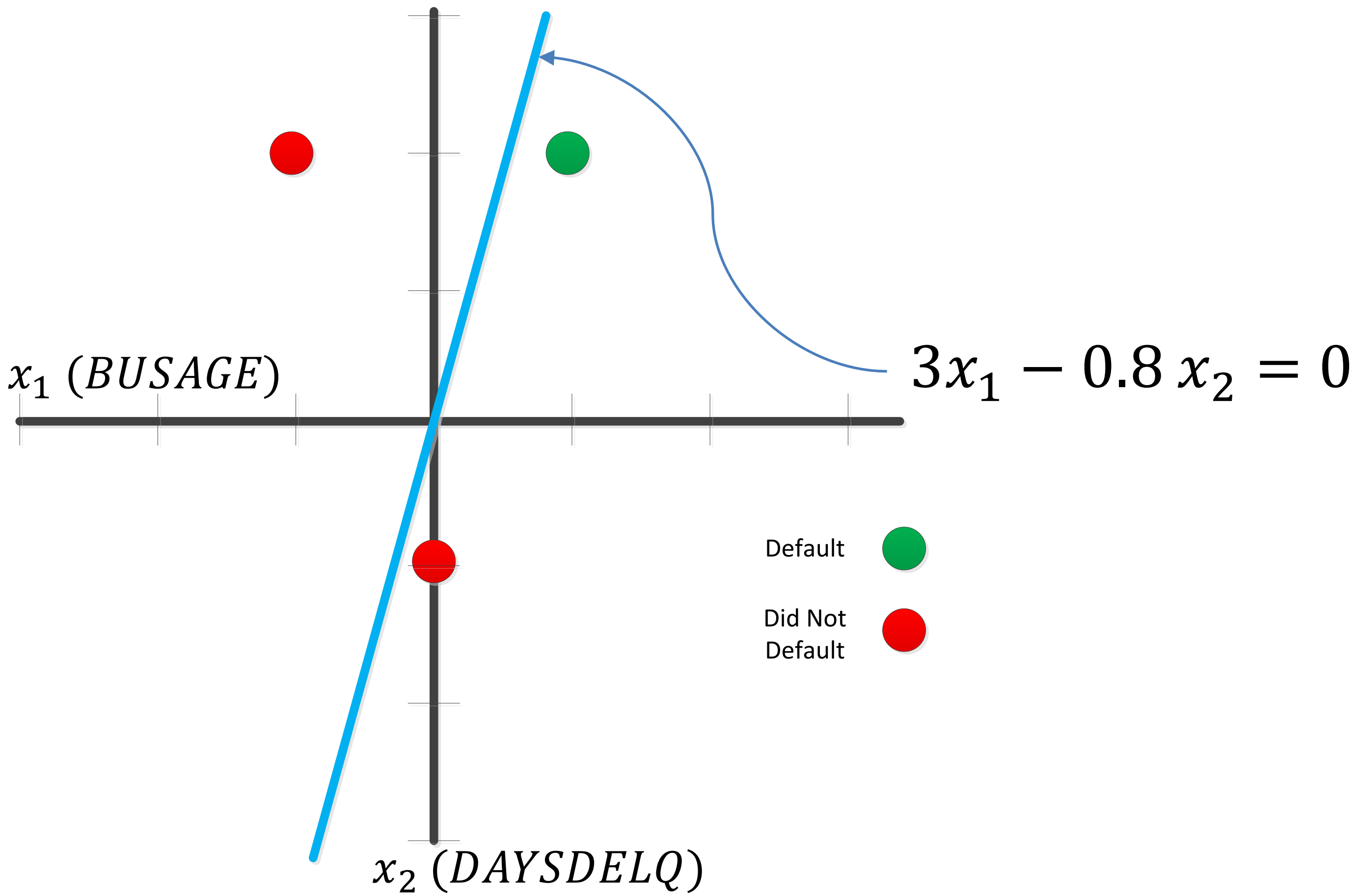
Simple Perceptron Induction Example

- Graphically:



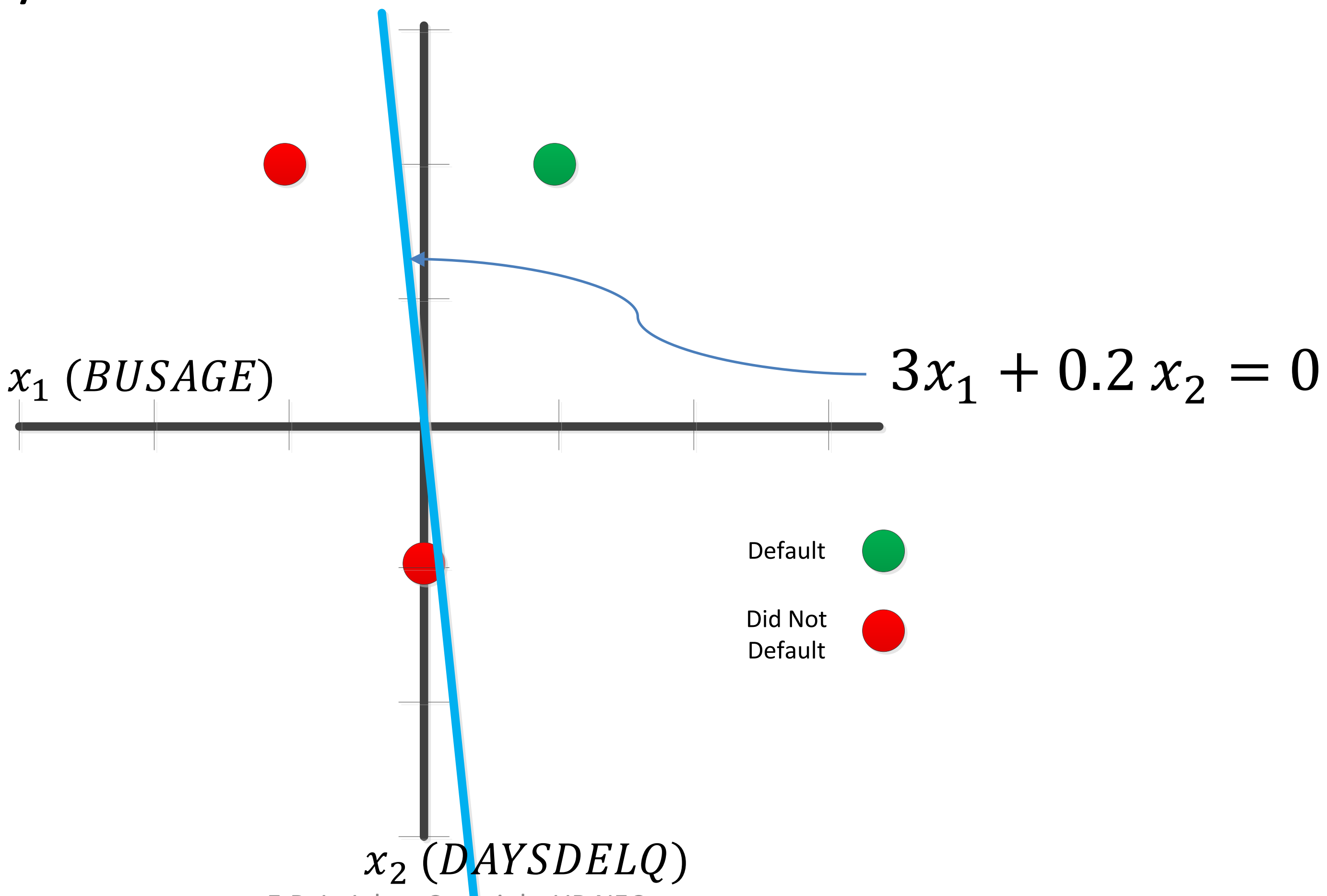
Simple Perceptron Induction Example

- Graphically:



Simple Perceptron Induction Example

- Graphically:



Perceptron Induction Usage

- Final Equation of the line: $3x_1 + 0.2\,x_2 = 0$

BUSAGE	DAYSDELQ	DEFAULT
1	2	Yes
-1	2	No
0	-1	No

$$3(1) + 0.2(2) = 3.4 > 0$$

$$3(-1) + 0.2(2) = -2.6 < 0$$

$$3(0) + 0.2(-1) = -0.2 < 0$$

Perceptron Induction Issues

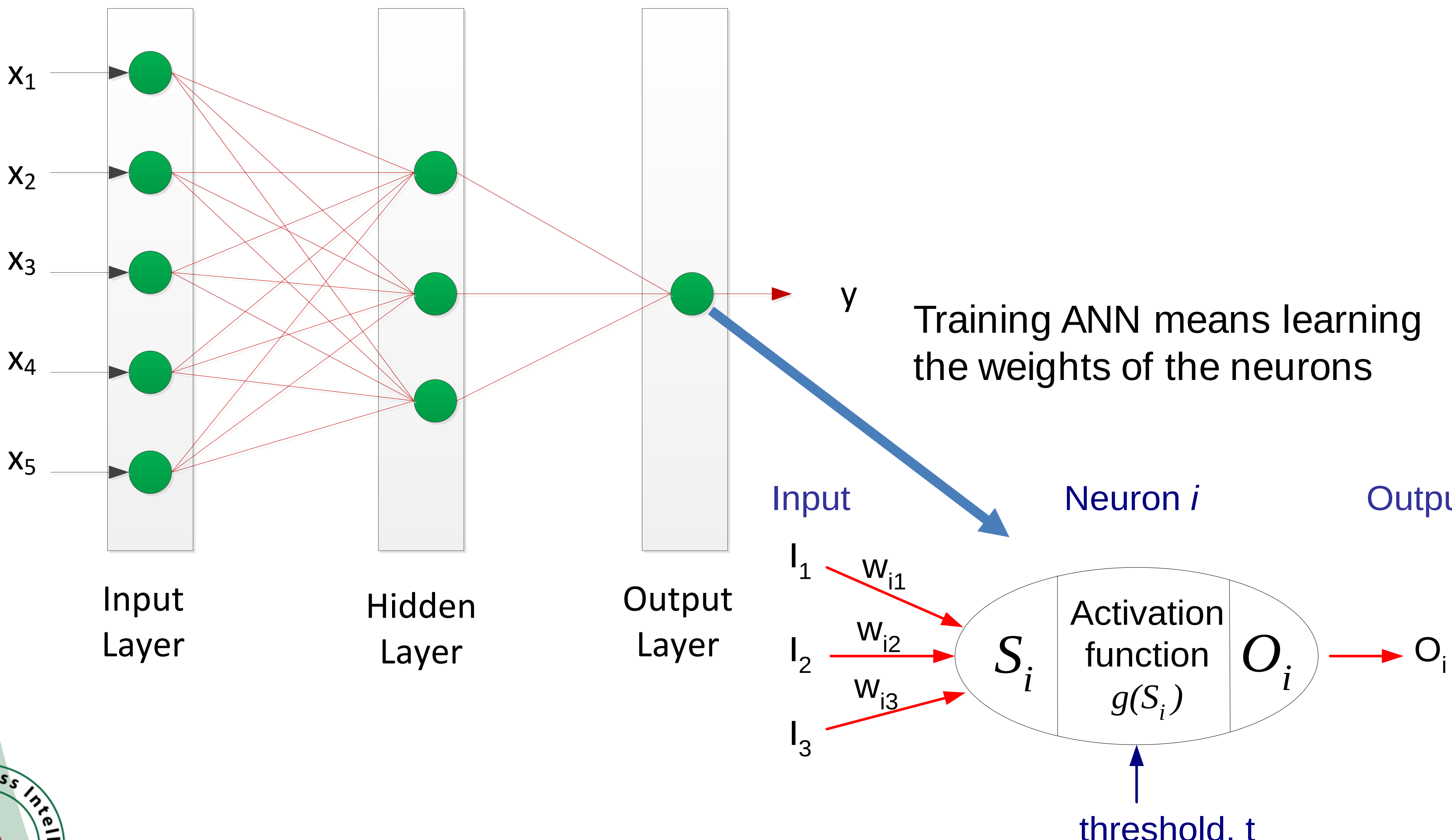
- Separable data implies there is a plane that classifies training data perfectly
- Algorithm converges for separable data, if λ is sufficiently small
- If separable, many solutions, depends on the initial w
- If not separable, does not converge: Needs ANN

Artificial Neural Networks

- Is a network of Perceptrons or Nodes that mimics a biological network of Neurons in a brain
- An ANN attempts to recreate the computational mirror of the biological neural network
- Each neuron takes many input signals, then based on an internal weighting system, produces a single output signal that's typically sent as input to another neuron.
- Finding the weights in an ANN constitute the bulk of the time done in learning the ANN.

General Structure of ANN

Training ANN means learning the weights of the neurons

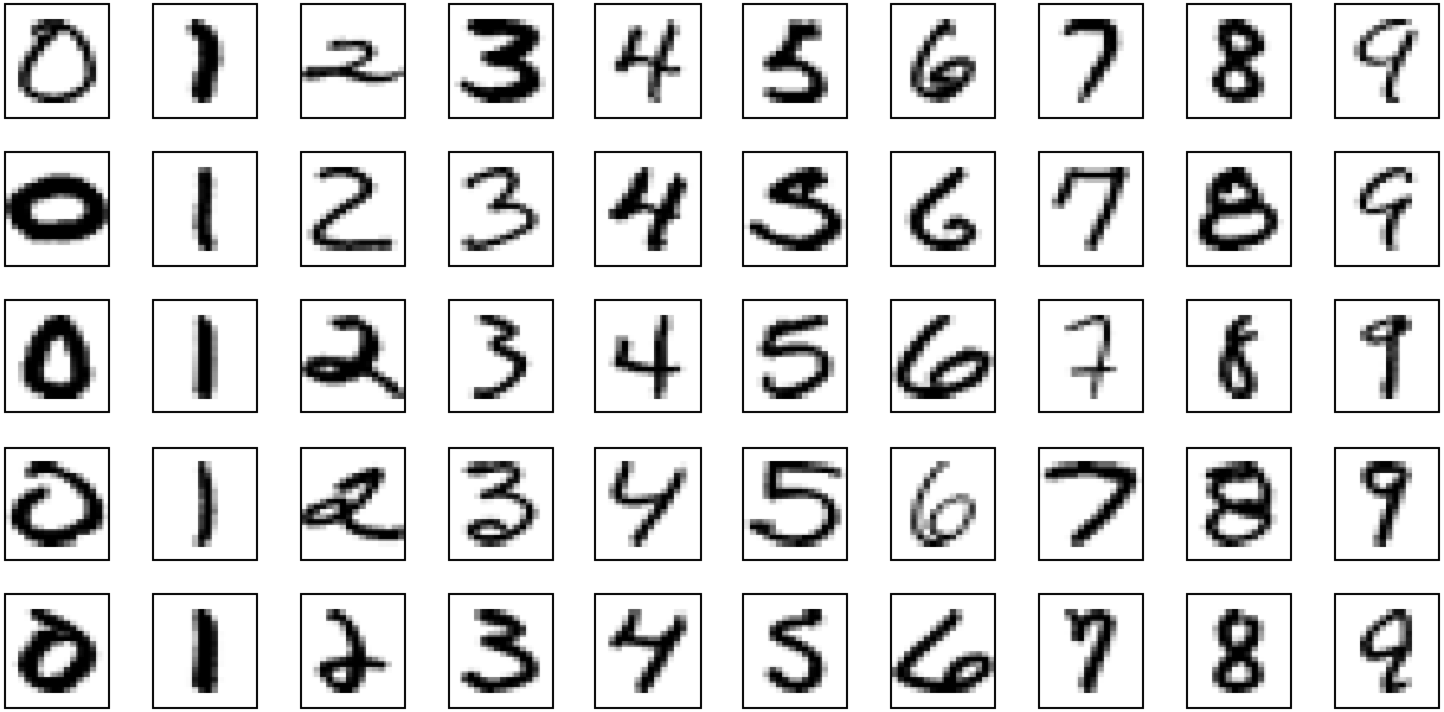


Choice of Hidden Layers

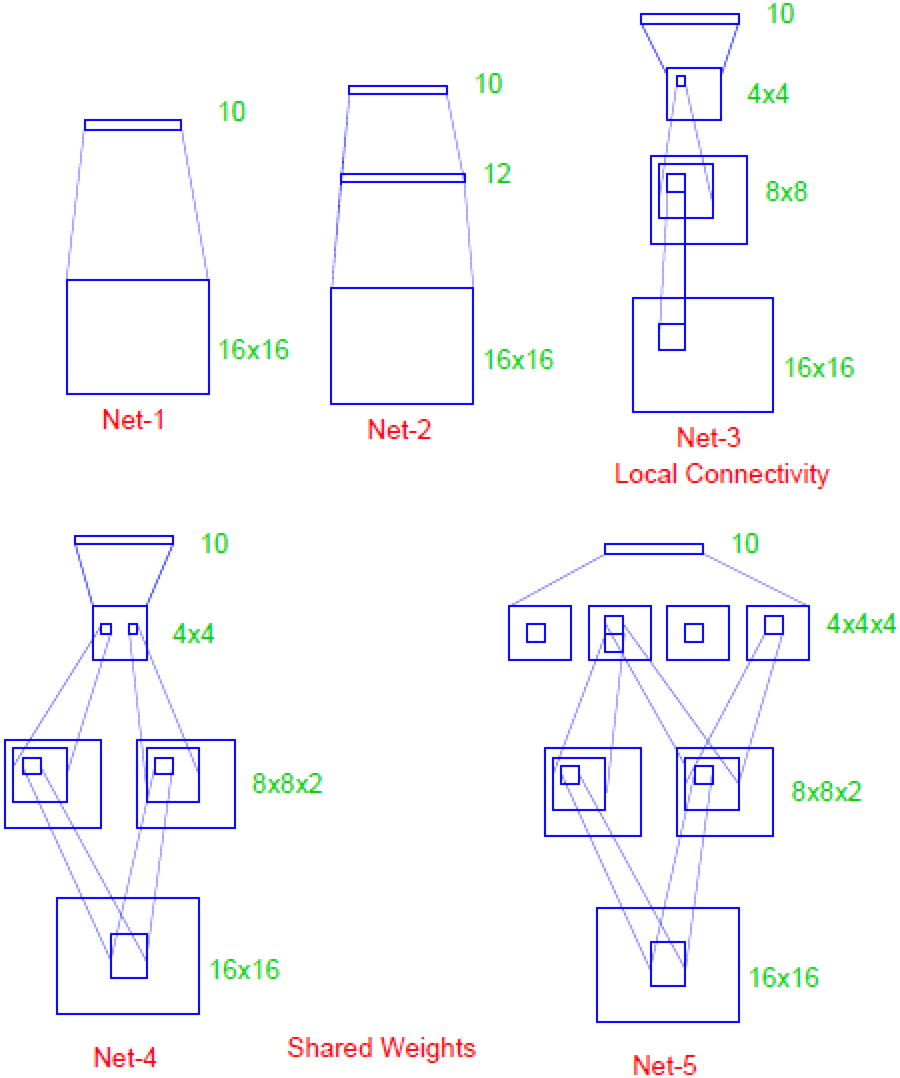
- Typically the number of **hidden units** is somewhere in the range of 5 to 100, with the number increasing with the number of inputs and number of training cases.
- Choice of the number of hidden layers is guided by **background knowledge and experimentation**. (Friedman et al.)

ANN ZIP code example

- A 16x16 8 bit greyscale image of numbers



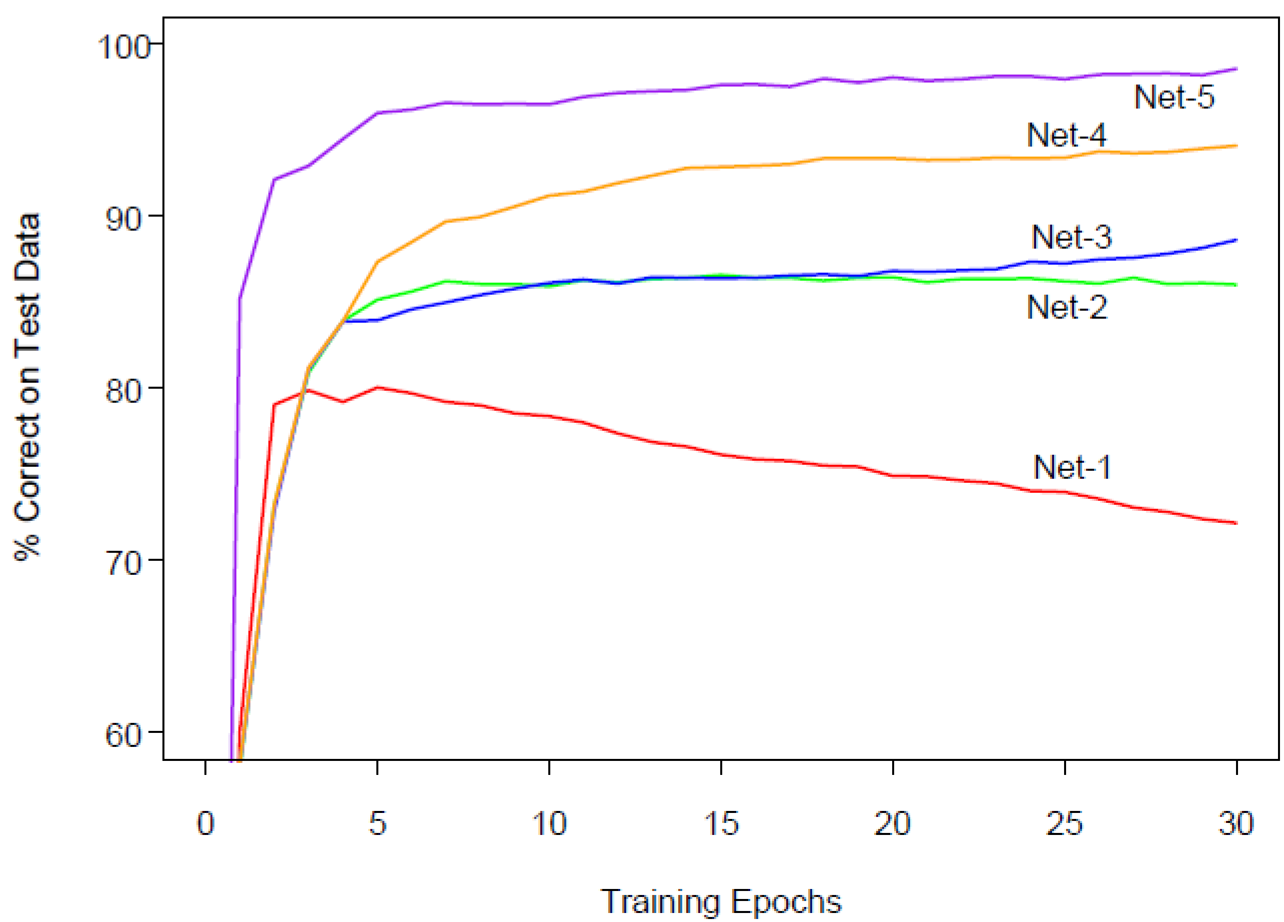
ANN ZIP code example



- Net-1: No hidden layer, equivalent to multinomial logistic regression.
- Net-2: One hidden layer, 12 hidden units fully connected.
- Net-3: Two hidden layers locally connected.
- Net-4: Two hidden layers, locally connected with weight sharing.
- Net-5: Two hidden layers, locally connected, two levels of weight sharing.

ᴾ NEC,

ANN ZIP code example



Business Scenario: Credit Scoring

- Credit scoring is the practice of analyzing a persons background and credit application in order to assess the creditworthiness of the person
- The variables *income* (yearly), *age*, *loan* (size in euros) and *LTI*(the loan to yearly income ratio) are available.
- The goal is to devise a model which *predicts,* whether or not a default will occur within 10 years.

http://www.r-bloggers.com/using-neural-networks-for-credit-scoring-a-simple-example/

Business Scenario: Credit Scoring

- New Data:

income	age	loan	LTI	default10yr
42710	46	6104	0.143	?
66953	19	8770	0.131	?
24904	57	15	0.001	?

Business Scenario: Credit Scoring

- Type the following lines of code in RStudio and run.

```
creditsetnumeric = read.csv("creditsetnumeric.csv")
creditsetnumeric$default10yr = as.factor(creditsetnumeric$default10yr)
MLP <-
make_Weka_classifier("weka/classifiers/functions/MultilayerPerceptron")
ANNModel <- MLP(default10yr ~ income + age +  loan  + LTI
                  , data=creditsetnumeric, control = Weka_control( H='5'))
creditsettest = read.csv("creditsettest.csv")
creditsettest$predictions = predict(ANNModel, creditsettest)
creditsettest
```

Business Scenario: Credit Scoring

```
> creditsettest
  income age loan   LTI default10yr predictions
1  42710  46 6104 0.143          NA           0
2  66953  19 8770 0.131          NA           1
3  24904  57   15 0.001          NA           0
```

Advantages and Disadvantages Table: ANN

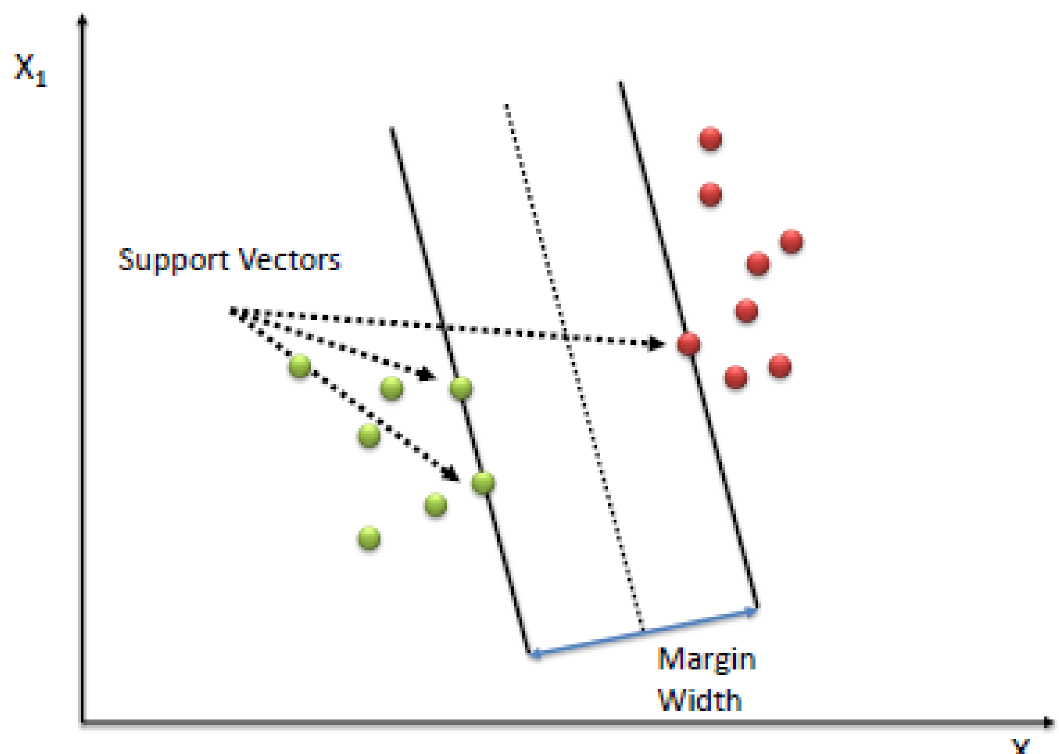
Parameter	Advantages	Disadvantages
Complexity		
Assumptions		
Preprocessing		
Categorical/Numerical		
Parameters		
Speed of Algorithm		
Handling Outliers/Noise		
Handling Missing Data		
Interpretability		
Relative Predictive Power		
Stability of Model		
Optimality of Model		

This Session's Outline

- What is Classification?
- Frequency Table
  - Zero R
  - One R
  - Naïve Bayes
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - ANN
  - **SVM**
- Ensembles
  - Adaboost
  - Random Forests
- Model Evaluation
- Case Study

Support Vector Machines

- A Support Vector Machine (SVM) performs classification by finding a plane that **maximizes the margin** between the two classes. The vectors (cases) that define the plane are the support vectors.

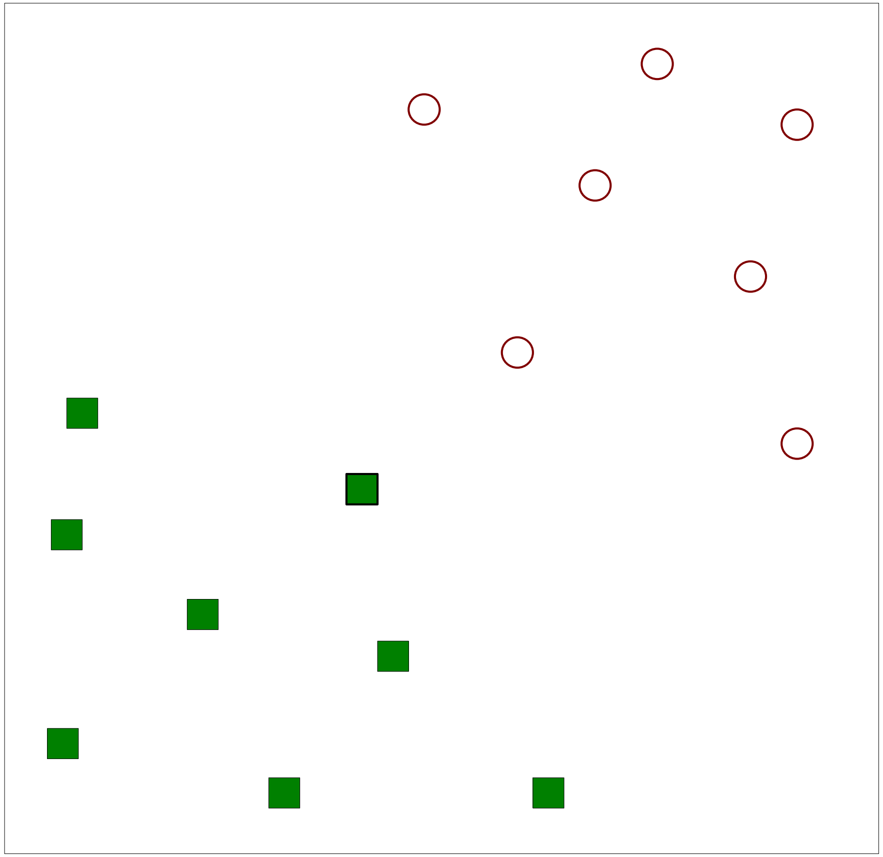


Recall: Limitations of Perceptrons

- Solution not unique if separable
- Data needs to be separable to converge
- Linear Model

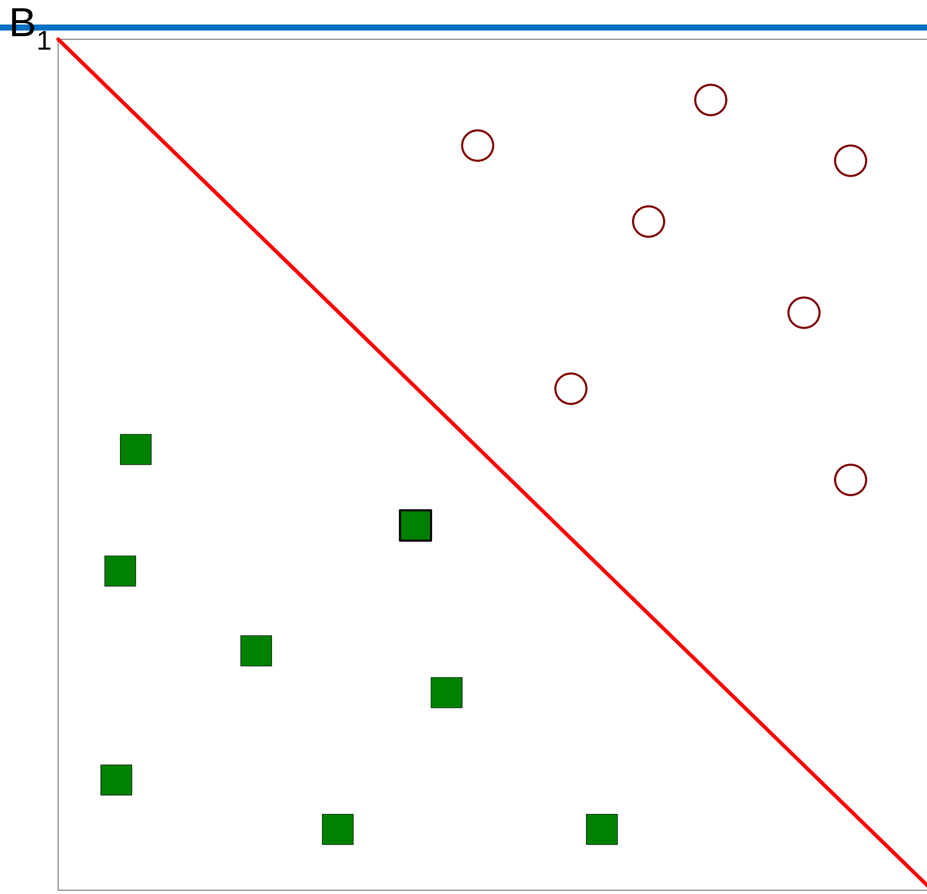
- These are solved using Support Vector Machines (SVM)

Support Vector Machines



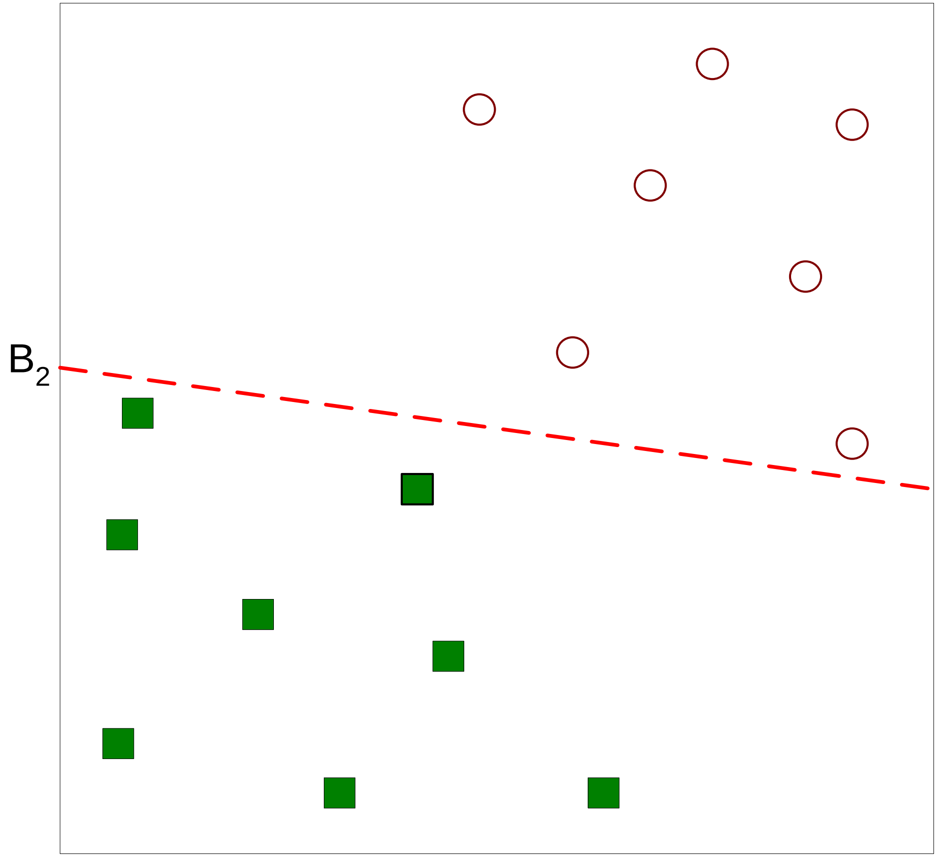
- Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



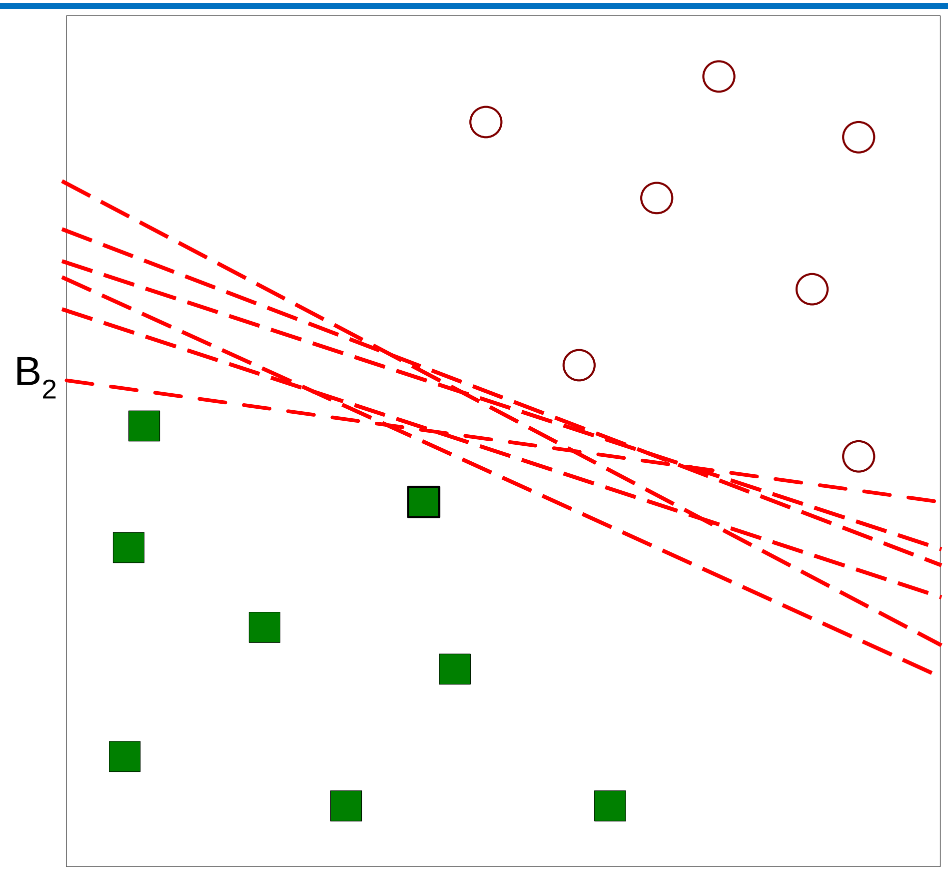
- One Possible Solution

Support Vector Machines



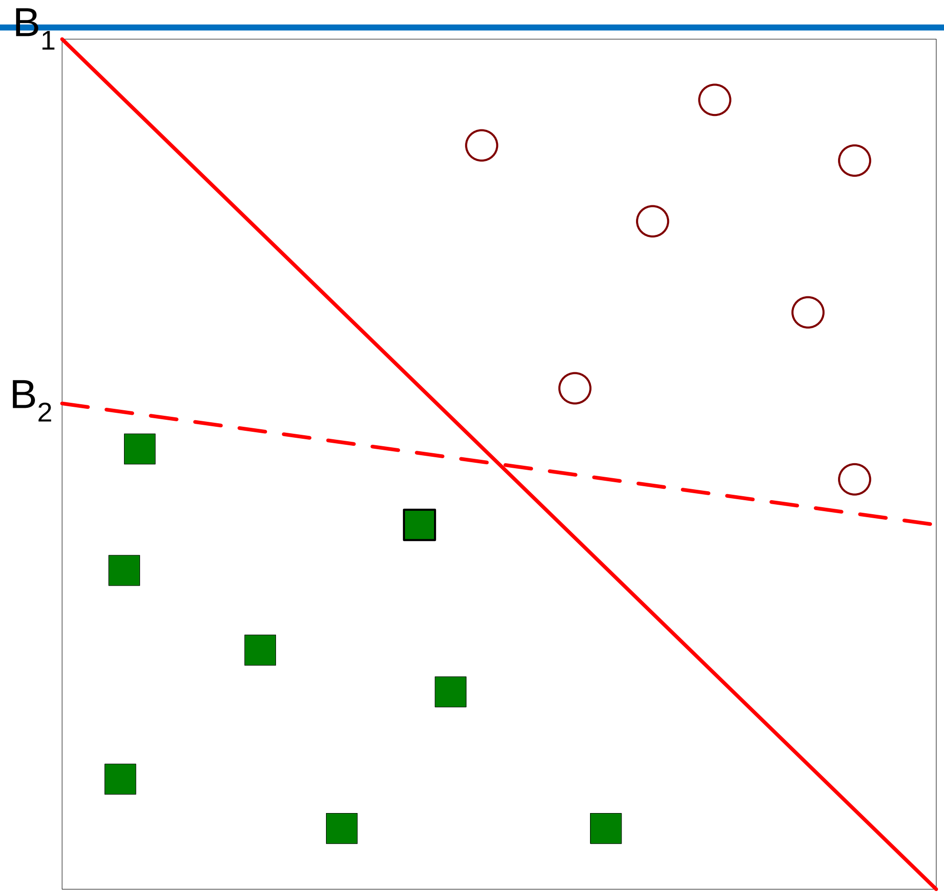
- Another possible solution

Support Vector Machines



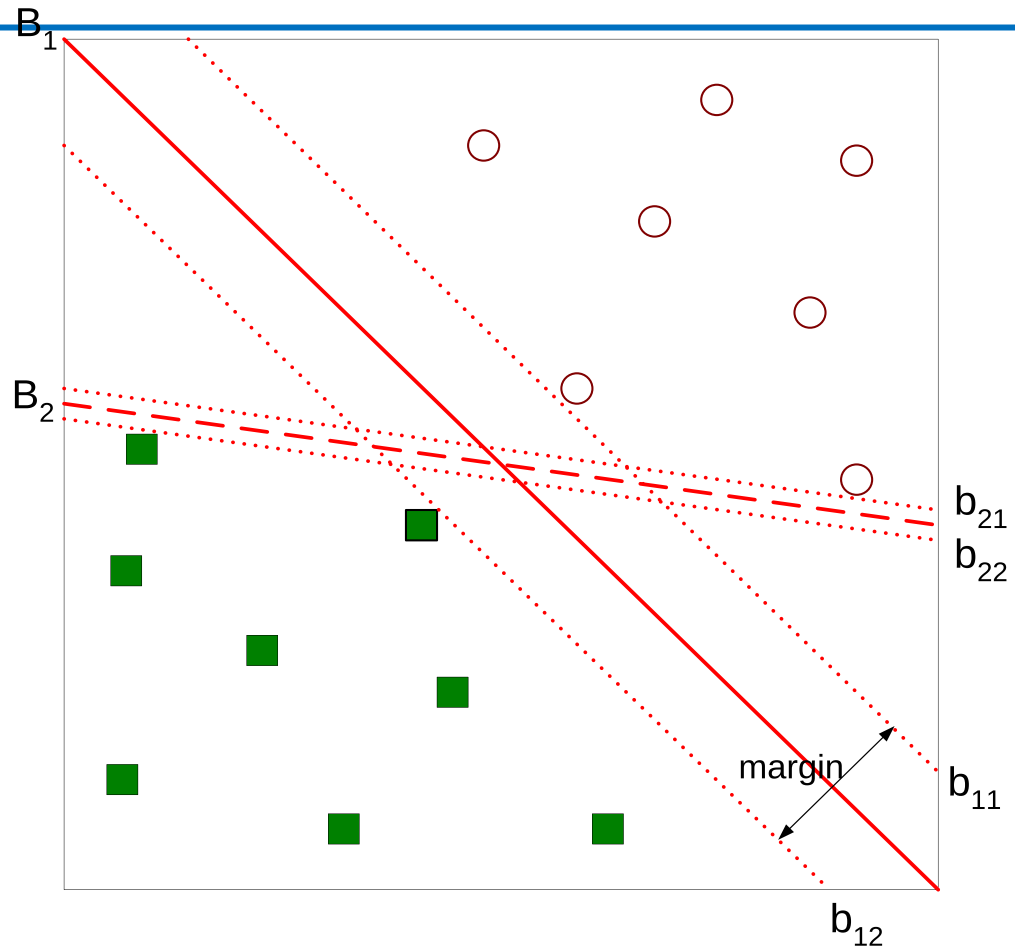
- Other possible solutions

Support Vector Machines



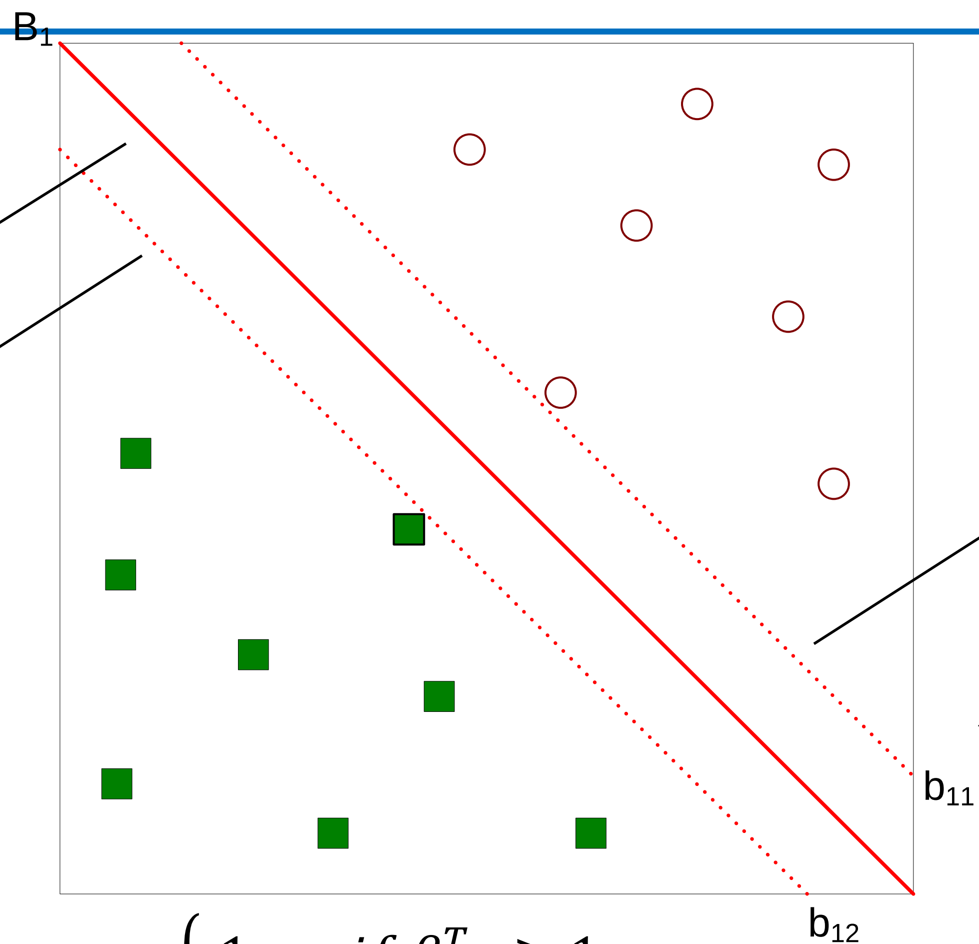
- Which one is better? B1 or B2?
- How do you define better?

Support Vector Machines



- Find hyperplane maximizes the margin => B1 is better than B2

Support Vector Machines



$$\beta^T x = 0$$

$$\beta^T x + b = -1$$

$$\beta^T x + b = 1$$

$$Margin = \frac{2}{\|\beta\|^2}$$

$$f(x) = \begin{cases} 1 & if\ \beta^T x \geq 1 \\ -1 & if\ \beta^T x \leq -1 \end{cases}$$

eljalao@up.edu.ph
103

Support Vector Machines

- We want to maximize: $Margin = \frac{2}{\|\beta\|^2}$
  – Which is equivalent to minimizing: $Z(\beta) = \frac{\|\beta\|^2}{2}$
  – But subjected to the following constraint:

$$y_i(\beta^T x_i + b) \geq 1, \ \ \forall(y_i, x_i), i = 1,2\ldots n$$

  - If $y_i = 1$, prediction $\beta^T x_i + b = 1$ and if $y_i = -1$ then $\beta^T x_i + b = -1$

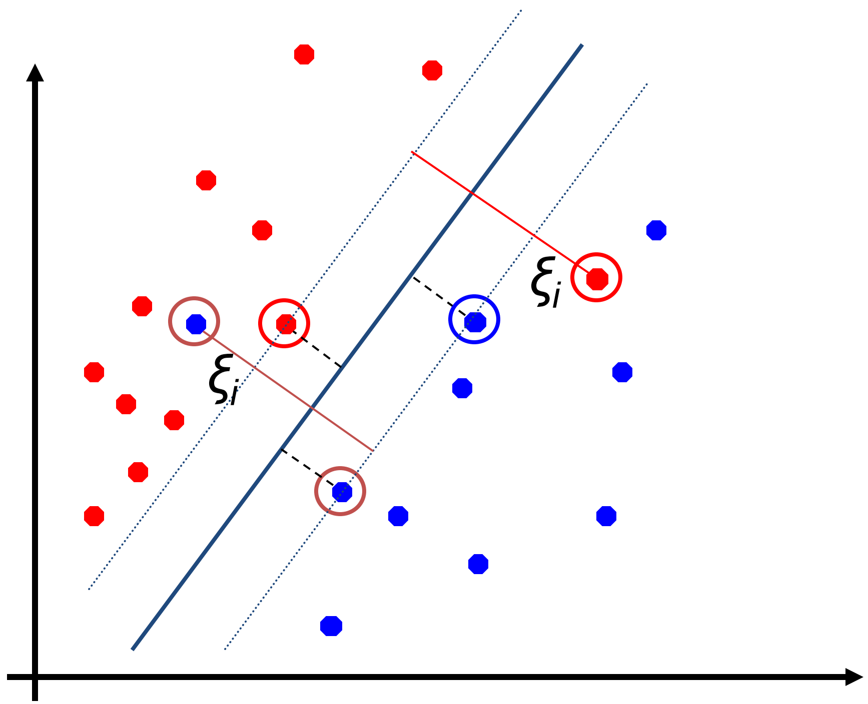
  – This is a constrained optimization problem
    - Numerical approaches to solve it (e.g., quadratic programming)

Limitations of Perceptrons

- Solution not unique if separable
- Data needs to be separable to converge
- Linear Model

Support Vector Machines

- What if the problem is not linearly separable?



Support Vector Machines

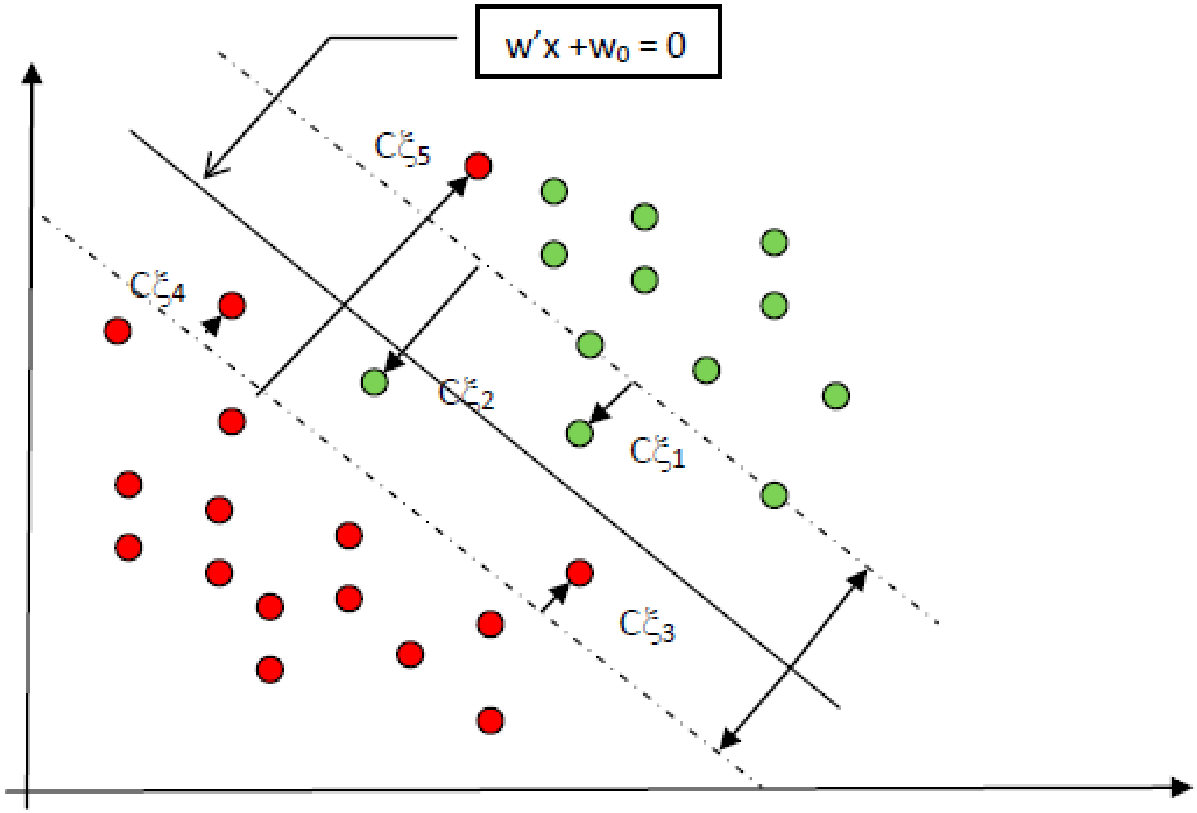
- What if the problem is not linearly separable?
  - Introduce slack variables
    - Need to minimize:

$$Min\ Z(\beta) = \frac{\|\beta\|^2}{2} + \lambda\left(\sum_{i=1}^{N} \xi_i^k\right)$$

- Subject to:

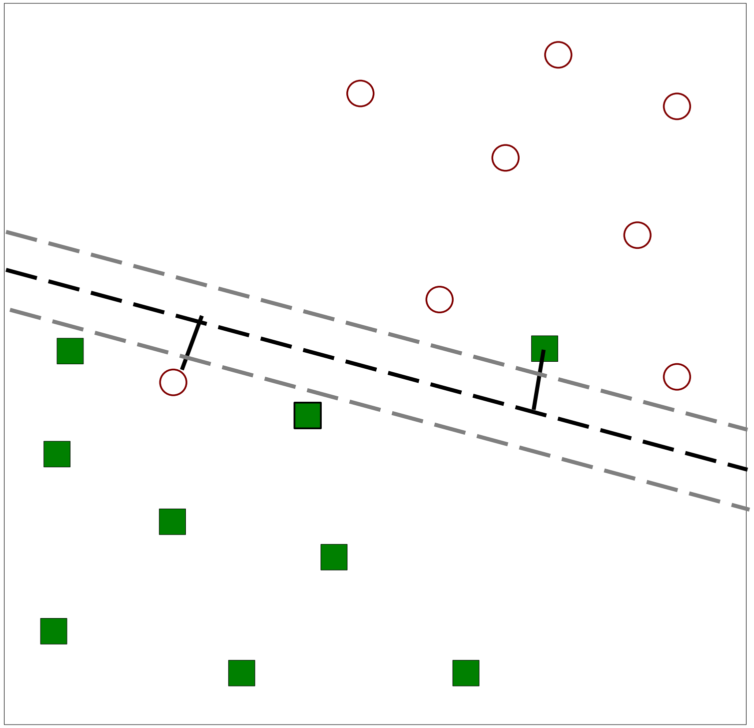
$$y_i(\beta^T x_i + b) \geq 1 - \xi_i, \qquad \forall(y_i, x_i), i = 1,2 \ldots n$$

Support Vector Machines



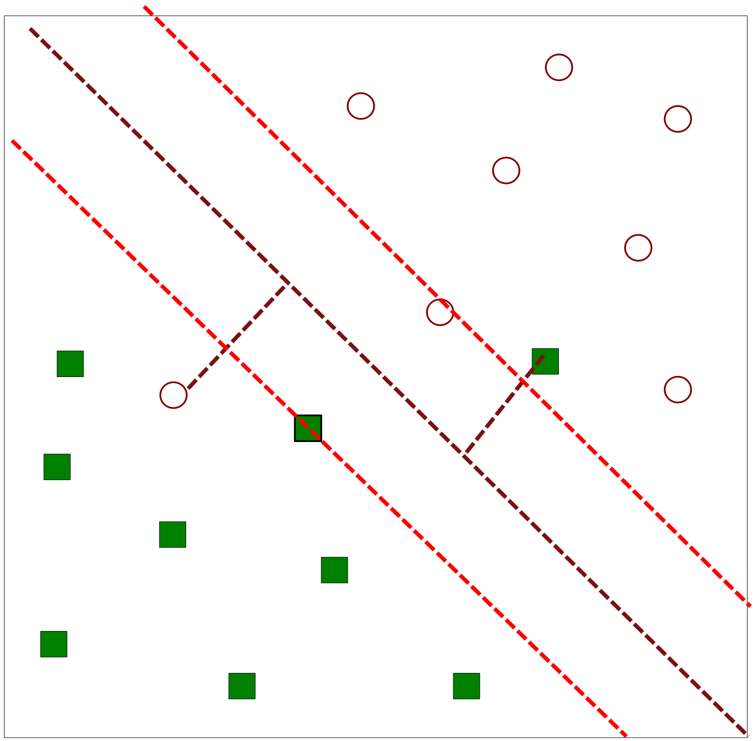
The Tuning/Complexity Parameter λ

- If λ is Large: Focus on Minimizing Errors



The Tuning/Complexity Parameter λ

- If λ is Small: Focus on Maximizing Margin

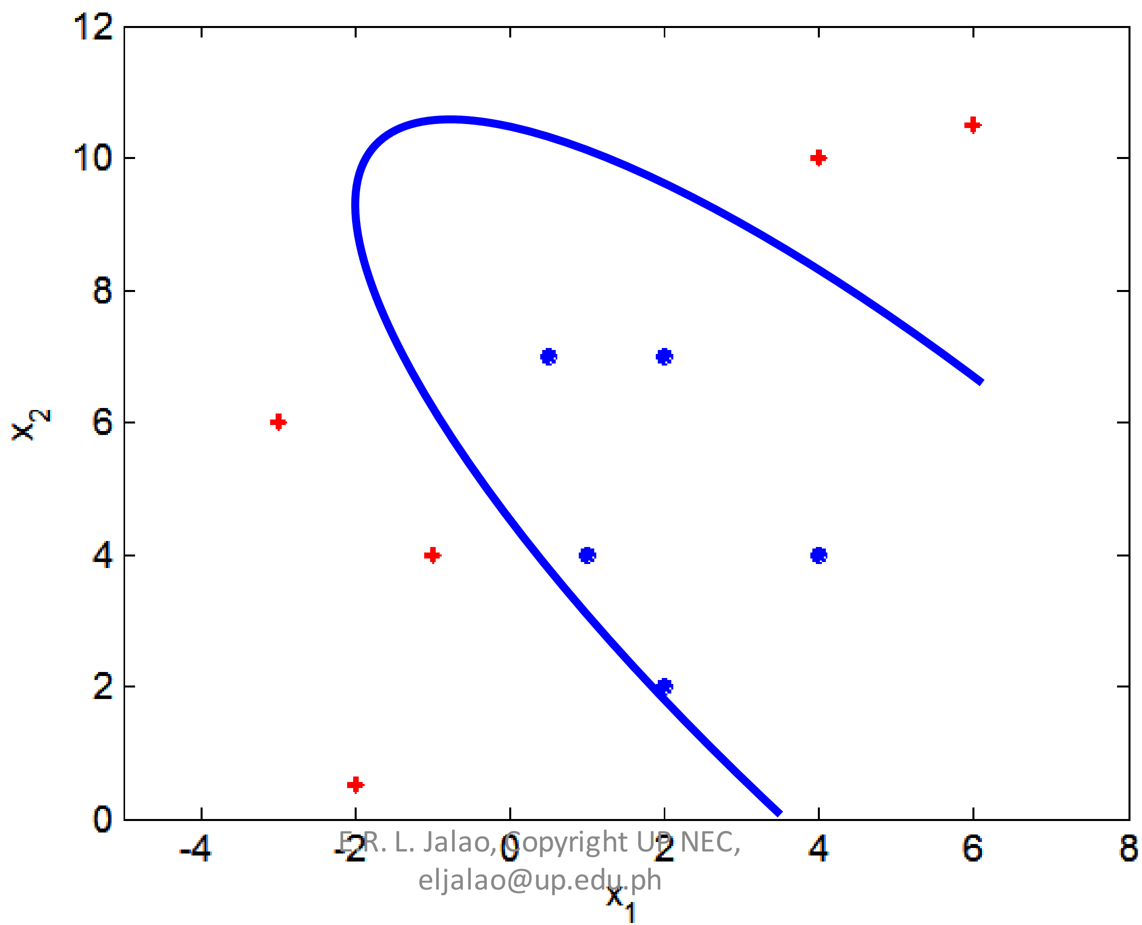


Limitations of Perceptrons

- Solution not unique if separable
- Data needs to be separable to converge
- Linear Model

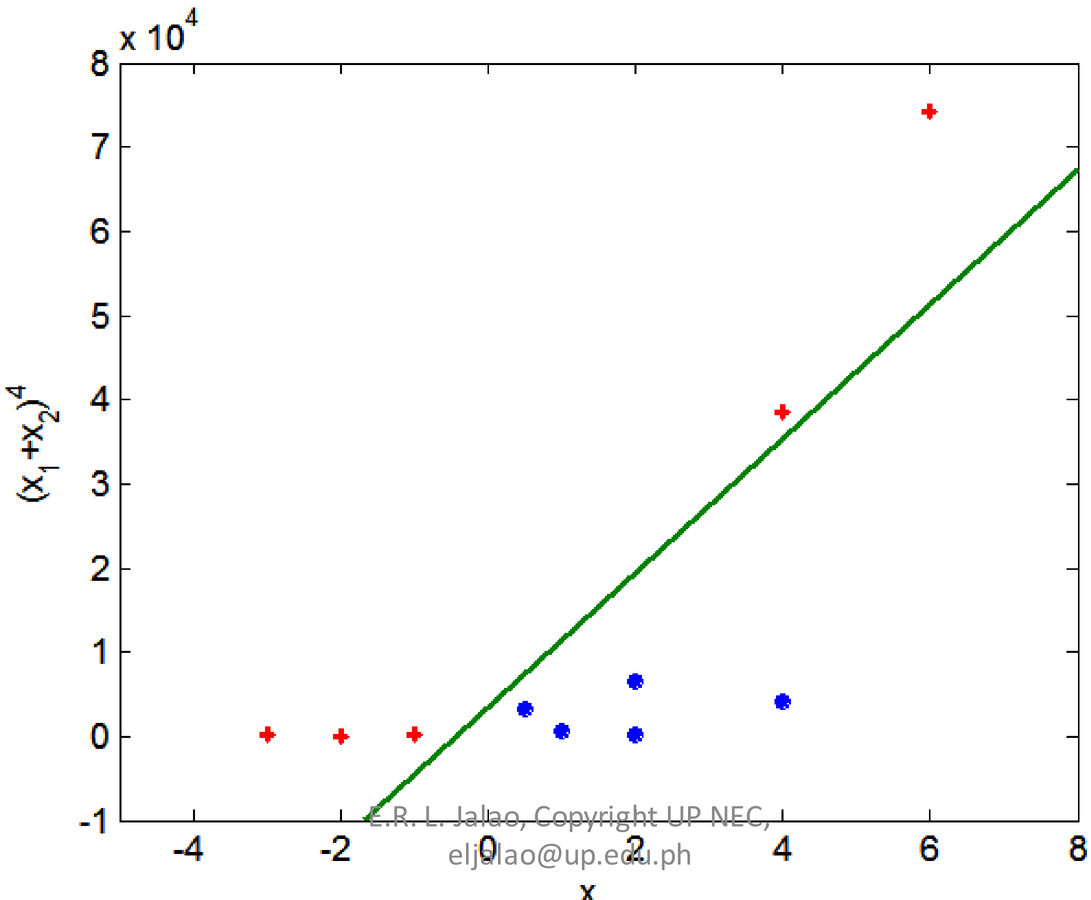
Nonlinear Support Vector Machines

- What if decision boundary is not linear?



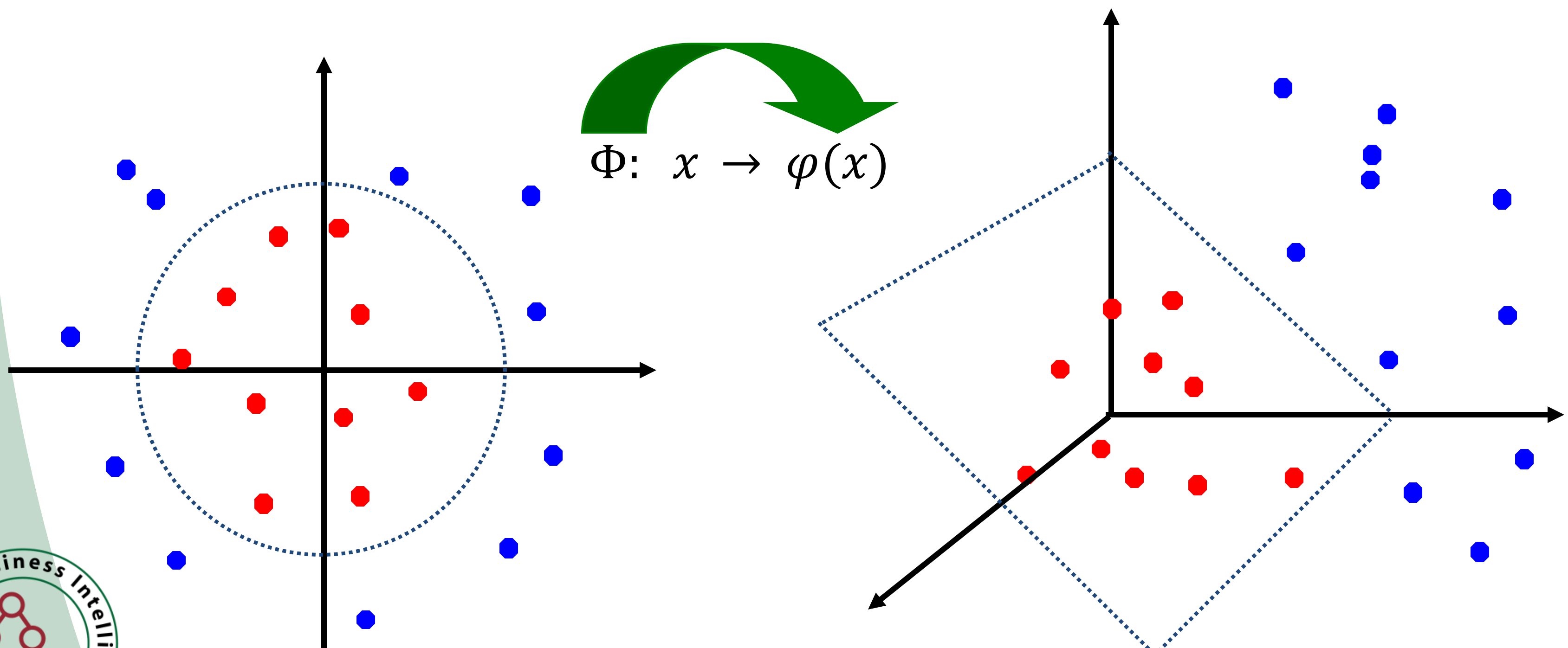
Nonlinear Support Vector Machines

- Transform data into higher dimensional space



Non-linear SVMs: Feature spaces

- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



Kernels

- Kernel methods transform the training data into higher dimensional spaces such that the data can be more easily separated
- Kernel Trick: A complex function can be approximated by a Kernel:
  - E.g. $f(x_1, x_2) = 1 + \sqrt{2}x_1 + \sqrt{2}x_2 + x_1^2 + x_2^2 + \sqrt{2}x_1x_2$
  - Kernel: $k(x_1, x_2) = (1 + x_1x_2^t)^2$

Business Scenario: Credit Scoring

- We will be using the credit scoring dataset again.

income	age	loan	LTI	default10yr
42710	46	6104	0.143	?
66953	19	8770	0.131	?
24904	57	15	0.001	?

http://www.r-bloggers.com/using-neural-networks-for-credit-scoring-a-simple-example/

Business Scenario: Credit Scoring

- Type the following lines of code in RStudio and run.

```
creditsetnumericSMO = read.csv("creditsetnumeric.csv")
creditsetnumericSMO$default10yr =
  as.factor(creditsetnumericSMO$default10yr)
SVMModel <- SMO(default10yr ~ income + age +  loan  + LTI
                , data=creditsetnumericSMO,
                control = Weka_control(C='1', K = list("PolyKernel", E=2)))
creditsettestSMO = read.csv("creditsettest.csv")
creditsettestSMO$default10yr =
  as.factor(creditsettestSMO$default10yr)
creditsettestSMO$predictions = predict(SVMModel,
                                       creditsettestSMO)
creditsettestSMO
```


eljalao@up.edu.ph

Business Scenario: Credit Scoring

```
> creditsettestSMO
  income age loan   LTI default10yr predictions
1  42710  46 6104 0.143        <NA>           0
2  66953  19 8770 0.131        <NA>           1
3  24904  57   15 0.001        <NA>           0
```

Advantages and Disadvantages Table: SVM

Parameter	Advantages	Disadvantages
Complexity		
Assumptions		
Preprocessing		
Categorical/Numerical		
Parameters		
Speed of Algorithm		
Handling Outliers/Noise		
Handling Missing Data		
Interpretability		
Relative Predictive Power		
Stability of Model		
Optimality of Model		

119

This Session's Outline

- What is Classification?
- Frequency Table
  - Zero R
  - One R
  - Naïve Bayes
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - ANN
  - SVM
- **Ensembles**
  - Adaboost
  - Random Forests
- Model Evaluation
- Case Study

Ensemble Methods

- Construct a set of classifiers from the training data
- Predict class label of previously unseen records by aggregating predictions made by multiple classifiers
- Wisdom of the Crowd
- Types of Ensembles
  - Parallel Ensemble
  - Serial Ensemble

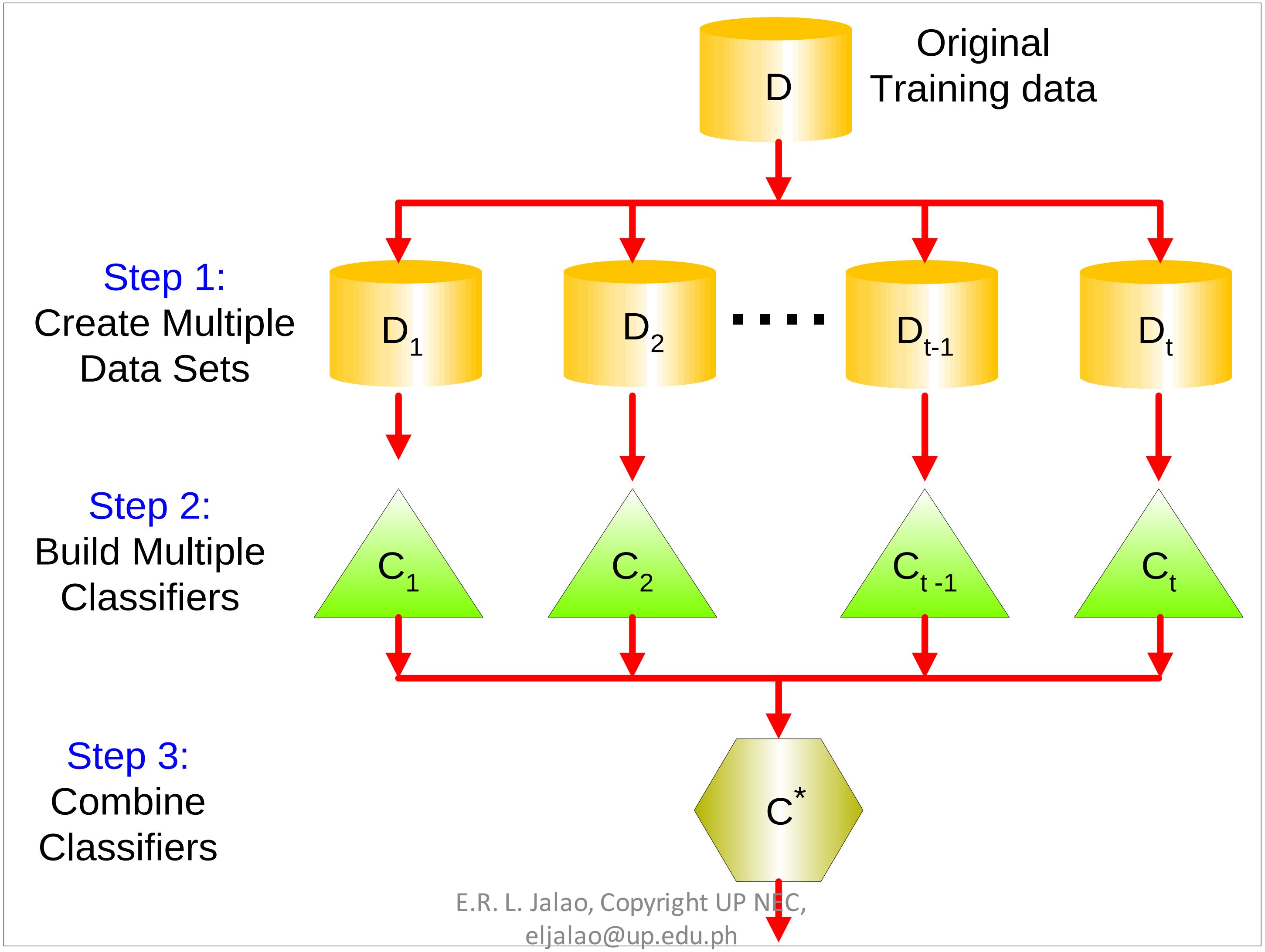
eljalao@up.edu.ph

Parallel Ensembles

- Parallel: Combines approx. independent, diverse base learners
- Different learners should make different errors
- Ensemble can outperform any one of its components
- Bagging, Random Forest (RF) examples

122

General Idea



Generation of Datasets: Bagging

- Sampling with replacement

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

- Build classifier on each bootstrap sample
- Each sample has probability $\left(1-\frac{1}{n}\right)^n$ of NOT being selected
- $\approx \frac{1}{3}$ is not inside the bag: OOB

eljalao@up.edu.ph

Bagging

- Example Initial Data

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
y	1	1	1	-1	-1	-1	-1	1	1	1

- Round 1: Random Data Selected

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	1	1

- Round 1 Optimal Split
  - if $x \leq 0.35 \rightarrow y = 1, x > 0.35 \rightarrow y = -1$

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
$\hat{y}$	1	1	1	1	-1	-1	-1	-1	-1	-1

Bagging

- Round 2: Random Data Selected

x	0.2	0.4	0.5	0.6	0.7	0.7	0.7	0.8	0.9	0.9
y	1	-1	-1	-1	-1	-1	-1	1	1	1

- Round 2 Optimal Split:
  - if $x \leq 0.75 \rightarrow y = -1, x > 0.75 \rightarrow y = 1$

x	0.2	0.4	0.5	0.6	0.7	0.7	0.7	0.8	0.9	0.9
$\hat{y}$	-1	-1	-1	-1	-1	-1	-1	1	1	1

Bagging

- Round 3: Random Data Selected

x	0.1	0.1	0.1	0.3	0.3	0.4	0.8	0.9	0.9	1
y	1	1	1	1	1	-1	1	1	1	1

- Round 3 Optimal Split:
  $y = 1$

x	0.1	0.1	0.1	0.3	0.3	0.4	0.8	0.9	0.9	1
$\hat{y}$	1	1	1	1	1	1	1	1	1	1

Bagging Ensemble

- Ensemble Model
  - If $x \leq 0.35 \rightarrow y = 1, x > 0.35 \rightarrow y = -1$
  - If $x \leq 0.75 \rightarrow y = -1, x > 0.75 \rightarrow y = 1$
  - $y = 1$

Round	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
1	1	1	1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1	1	1	1
3	1	1	1	1	1	1	1	1	1	1
Majority	1	1	1	-1	-1	-1	-1	1	1	1
True	1	1	1	-1	-1	-1	-1	1	1	1

This Session's Outline

- What is Classification?
- Frequency Table
  - Zero R
  - One R
  - Naïve Bayes
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - ANN
  - SVM
- Ensembles
  - **Adaboost**
  - Random Forests
- Model Evaluation
- Case Study

Serial Ensemble

- Serial: New learner uses previously built learners
  - Iteratively reweight the training data to focus on previous errors—known as **boosting**
  - Combine as linear combination –Adaboost Algorithm (1996)
  - Dramatic increase in accuracy with even weak base learners—can reduce bias and variance
  - Initially, all n records are assigned equal weights
  - Unlike bagging, weights may change at the end of boosting round

Boosting

- Records that are wrongly classified will have their weights increased
- Records that are classified correctly will have their weights decreased

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

- Example 4 is hard to classify
- Its weight is increased, therefore it is more likely to be chosen again in subsequent rounds

Adaboost - Adaptive Boosting

- Instead of resampling, uses training set re-weighting
  - Each training sample uses a weight to determine the probability of being selected for a training set.
- AdaBoost is an algorithm for constructing a "strong" classifier as linear combination of "simple" "weak" classifier

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

- Final classification based on weighted vote of weak classifiers

Business Scenario: Bank Data

- Type the following lines of code in RStudio and run.

```
bankdata = read.csv("bankdata.csv")
AdaboostModel <- AdaBoostM1(pep ~ age + sex+  region  + income
                 + married  + children  + car
                 +  save_act+ current_act+  mortgage
                 , data=bankdata,
                 control = Weka_control(W = list(J48)))
adaboostpred = bankdata
adaboostpred$predictions =predict(AdaboostModel)
adaboostpred[1:10,]
```


eljalao@up.edu.ph

Business Scenario: Bank Data

```
> adaboostpred[1:10,]
        id age    sex     region   income married children car save_act
1  ID12101  48 FEMALE INNER_CITY 17546.00      NO        1  NO       NO
2  ID12102  40   MALE       TOWN 30085.10     YES        3 YES       NO
3  ID12103  51 FEMALE INNER_CITY 16575.40     YES        0 YES      YES
4  ID12104  23 FEMALE       TOWN 20375.40     YES        3  NO       NO
5  ID12105  57 FEMALE      RURAL 50576.30     YES        0  NO      YES
6  ID12106  57 FEMALE       TOWN 37869.60     YES        2  NO      YES
7  ID12107  22   MALE      RURAL  8877.07      NO        0  NO       NO
8  ID12108  58   MALE       TOWN 24946.60     YES        0 YES      YES
9  ID12109  37 FEMALE   SUBURBAN 25304.30     YES        2 YES       NO
10 ID12110  54   MALE       TOWN 24212.10     YES        2 YES      YES
   current_act mortgage pep predictions
1           NO       NO YES         YES
2          YES      YES  NO          NO
3          YES       NO  NO          NO
4          YES       NO  NO          NO
5           NO       NO  NO          NO
6          YES       NO YES         YES
7          YES       NO YES         YES
8          YES       NO  NO          NO
9           NO       NO  NO          NO
10         YES       NO  NO          NO
```

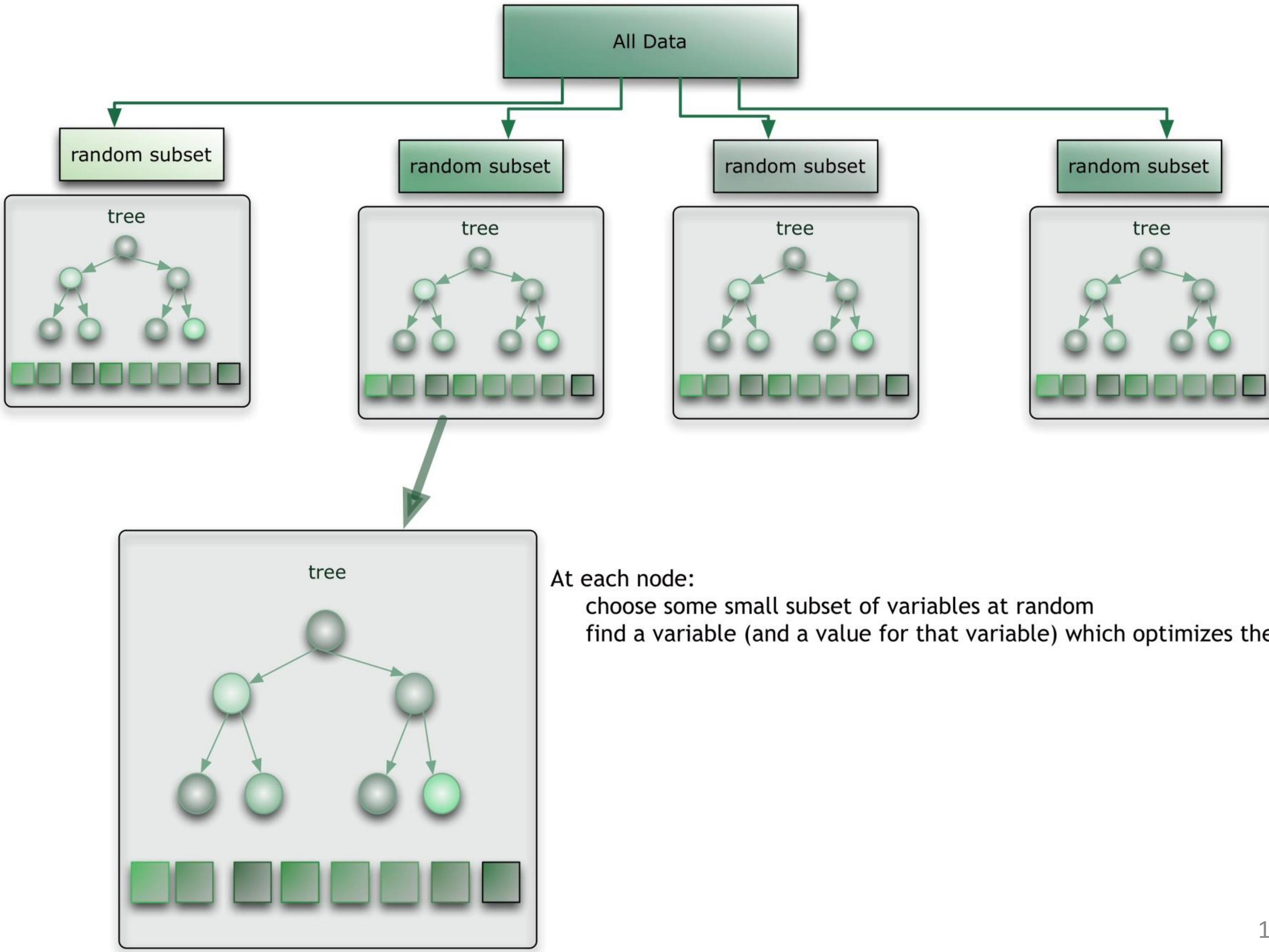
This Session's Outline

- What is Classification?
- Frequency Table
  - Zero R
  - One R
  - Naïve Bayes
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - ANN
  - SVM
- Ensembles
  - Adaboost
  - **Random Forests**
- Model Evaluation
- Case Study

Random Forests

- The random forest (Breiman, 2001) is an ensemble of decision trees
- Trees are combined by average (regression) or voting (classification)
- Tree provides a class probability estimates so that weighted votes can be used

Random Forests



At each node:

choose some small subset of variables at random

find a variable (and a value for that variable) which optimizes the split

137

Random Forests Algorithm

- For some number of trees T
- Sample N cases at random with replacement to create a subset of the data at each node:
  - For some number p, p predictor variables are selected at random from all the predictor variables.
  - The predictor variable that provides the best split, according to some objective function, is used to do a binary split on that node.
  - At the next node, choose another p variables at random from all predictor variables and do the same.

Random Forests

- Selecting p variables randomly reduces correlation between trees (variance)
- Brieman suggests **three possible values** for m: ½$\sqrt{p}$, $\sqrt{p}$, and $2\sqrt{p}$

Business Scenario: Census Income

- Predict whether a person's income exceeds $50K/yr based on various demographic data like: age, workclass, education, marital status, occupation, relationship, race, sex, capital gains, hours per week and country.
- Sample Data:

age	workclass	finalweight	education	education-num	marital-status	occupation	relationship	race	sex	capitalgain	capitalloss	hoursperweek	country	income
27	Private	116358	Some-college	10	Never-married	Craft-repair	Own-child	Asian-Pac-Islander	Male	0	1980	40	Philippines	<=50K
29	State-gov	71592	Some-college	10	Never-married	Adm-clerical	Unmarried	Asian-Pac-Islander	Female	0	0	40	Philippines	<=50K
32	Private	270335	Bachelors	13	Married-civ-spouse	Adm-clerical	Other-relative	White	Male	0	0	40	Philippines	>50K

140

Business Scenario: Income Data

- Type the following lines of code in RStudio and run.

```
adult = read.csv("adult.csv")
RF <- make_Weka_classifier("weka/classifiers/trees/RandomForest")
RFModel <- RF(income ~ age+workclass
              +finalweight+education+education.num
              +marital.status+occupation+relationship
              +race+sex+capitalgain+capitalloss+hoursperweek
              +country,
              data=adult, control = Weka_control(K =1))
adult$predictions = predict(RFModel, adult, se.fit=TRUE)
adult[1:10,]
```

Advantages and Disadvantages Table: Random Forests

Parameter	Advantages	Disadvantages
Complexity		
Assumptions		
Preprocessing		
Categorical/Numerical		
Parameters		
Speed of Algorithm		
Handling Outliers/Noise		
Handling Missing Data		
Interpretability		
Relative Predictive Power		
Stability of Model		
Optimality of Model		

eljalao@up.edu.ph
142

This Session's Outline

- What is Classification?
- Frequency Table
  - Zero R
  - One R
  - Naïve Bayes
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - ANN
  - SVM
- Ensembles
  - Adaboost
  - Random Forests
- **Model Evaluation**
- Case Study

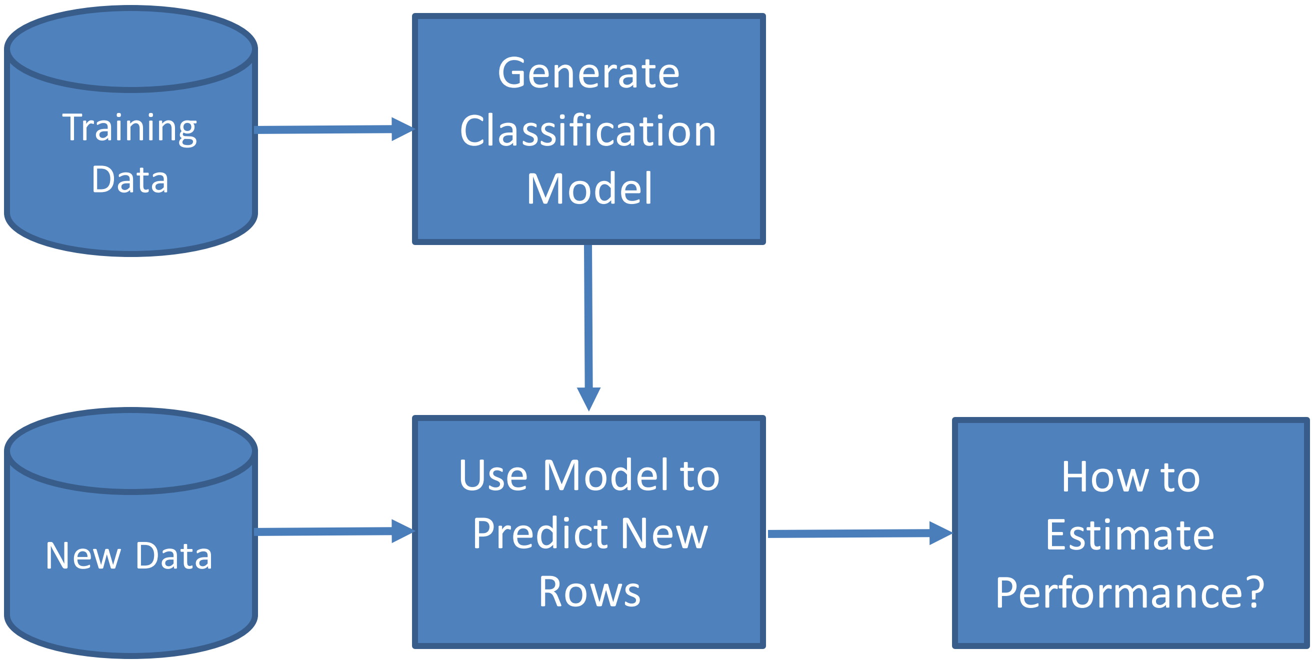
Model Evaluation

- How to evaluate the performance of a model?
  - Metrics for Performance Evaluation
- How to obtain reliable estimates?
  - Methods for Performance Evaluation
  - Overfitting and Underfitting
- How to compare the relative performance among competing models?
  - Methods for Model Comparison

eljalao@up.edu.ph

Model Evaluation

- Model Evaluation is a methodology that helps to find the best model that represents our data and how well the chosen model will work in the future.



eljalao@up.edu.ph

Definition of Terms

- Error:

$$error = 1\ if\ \hat{y} \neq y$$

  - If predicted value of the classifier model is not equal to the actual value, then we define that as an error.
- We would like to minimize error
- Given a binary classification model, how do we count the error predictions?

Metrics for Performance Evaluation

- Confusion Matrix: A way to tabulate errors

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Metrics for Performance Evaluation

		ACTUAL CLASS	
PREDICTED CLASS		Class=Yes	Class=No
	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

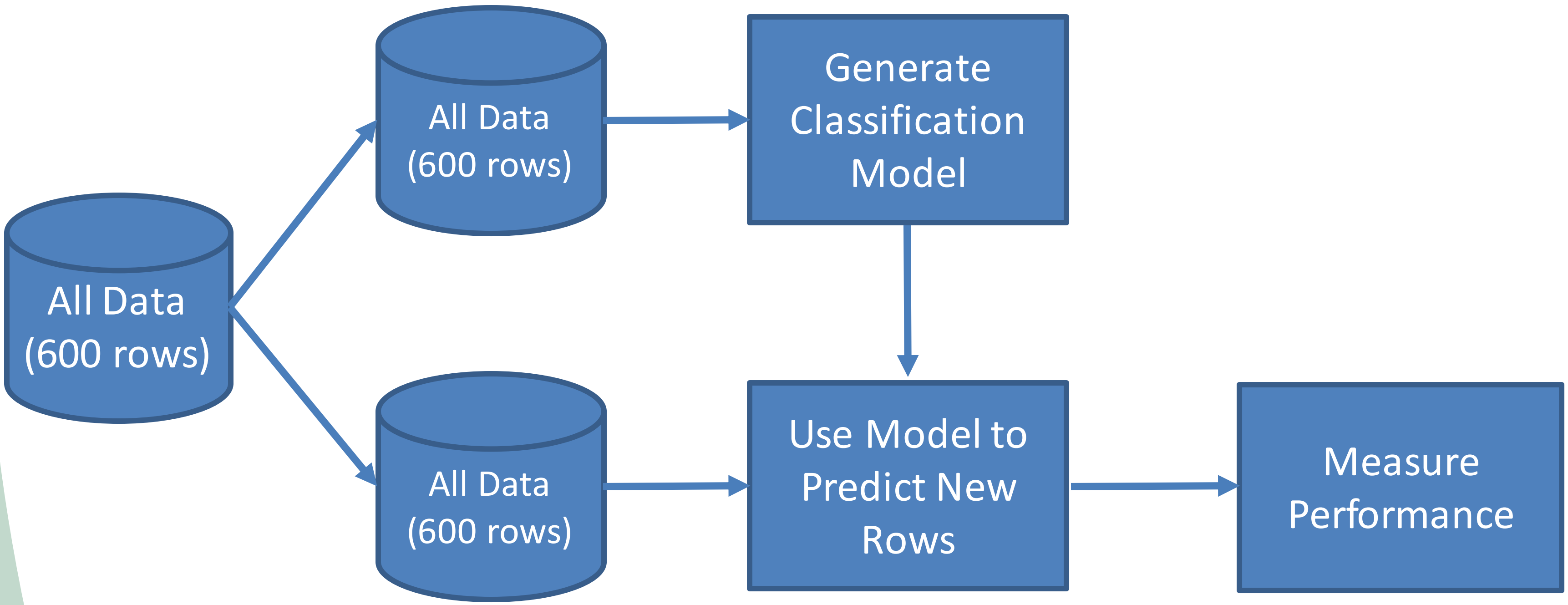
- Most widely-used metric:

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

eljalao@up.edu.ph

Re-Substitution Errors

- Re-substitution errors: error on training data



Re-Substitution Errors

- Type the following lines of code in RStudio and run.

```
bankdata = read.csv("bankdata.csv")
J48Model <- J48(pep ~ age + sex+  region  + income
              + married + children  + car
              + save_act+ current_act+  mortgage
              , data=bankdata)
summary(J48Model)
```

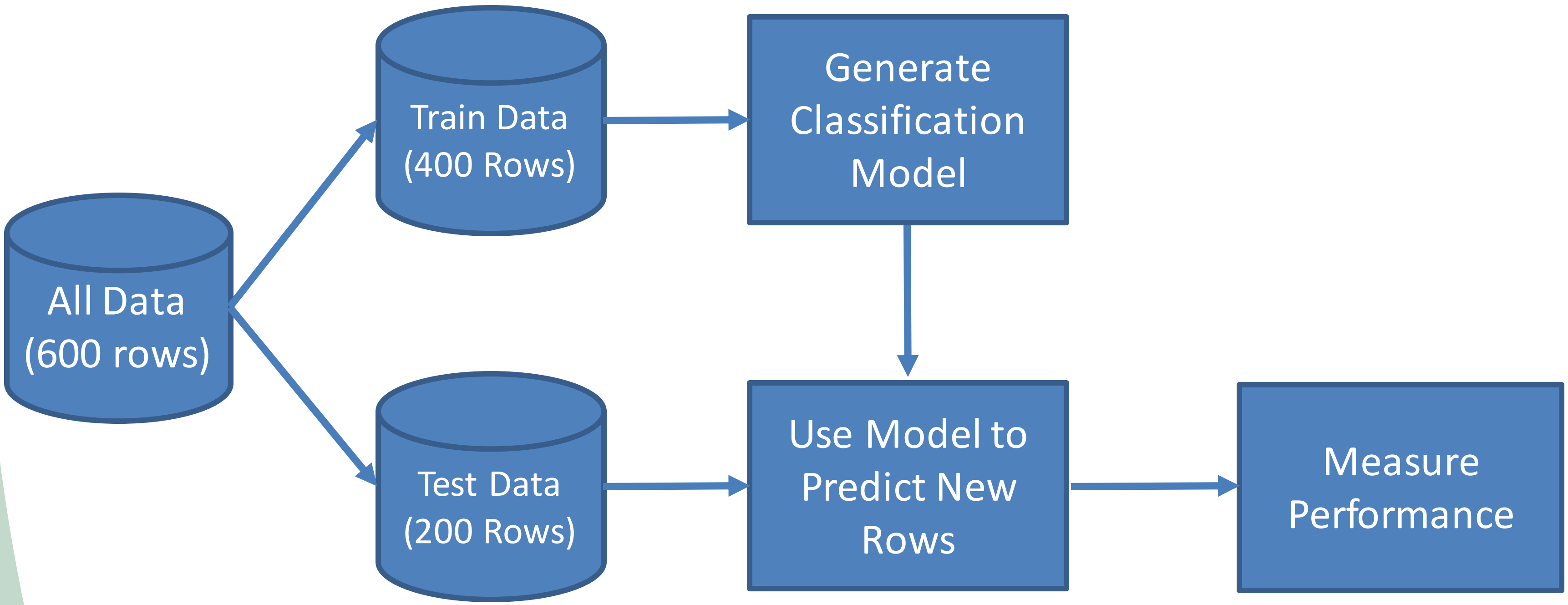
Re-Substitution Errors

```
> summary(J48Model)

=== Summary ===

Correctly Classified Instances         554               92.3333 %
Incorrectly Classified Instances        46                7.6667 %
Kappa statistic                          0.845
Mean absolute error                      0.1389
Root mean squared error                  0.2636
Relative absolute error                 27.9979 %
Root relative squared error             52.9137 %
Total Number of Instances              600

=== Confusion Matrix ===

   a   b   <-- classified as
 309  17 |   a = NO
  29 245 |   b = YES
```


eljalao@up.edu.ph

Generalization Errors

- Generalization errors: error on unseen data



Generalization Errors

- Type the following lines of code in RStudio and run.

```
bankdata = read.csv("bankdata.csv")
sample <- floor(2/3 * nrow(bankdata))
set.seed(123)
train_ind <- sample(seq_len(nrow(bankdata)),
                    size = sample)
bankdatatrain <- bankdata[train_ind, ]
bankdatatest <- bankdata[-train_ind, ]
J48ModelHoldout <- J48(pep ~ age + sex+ region  + income
                + married + children  + car
                + save_act+ current_act+  mortgage
                , data=bankdatatrain)
evaluate_Weka_classifier(J48ModelHoldout,
                         newdata = bankdatatest)
```

Generalization Errors

```
=== Summary ===

Correctly Classified Instances         178               89      %
Incorrectly Classified Instances        22               11      %
Kappa statistic                          0.7727
Mean absolute error                      0.1696
Root mean squared error                  0.3252
Relative absolute error                 34.7989 %
Root relative squared error             65.8951 %
Total Number of Instances              200

=== Confusion Matrix ===

   a   b   <-- classified as
 107   9 |   a = NO
  13  71 |   b = YES
```

Model Evaluation

- How to evaluate the performance of a model?
  - Metrics for Performance Evaluation
- **How to obtain reliable estimates?**
  - Methods for Performance Evaluation
  - Overfitting and Underfitting
- How to compare the relative performance among competing models?
  - Methods for Model Comparison

eljalao@up.edu.ph

Methods for Performance Evaluation

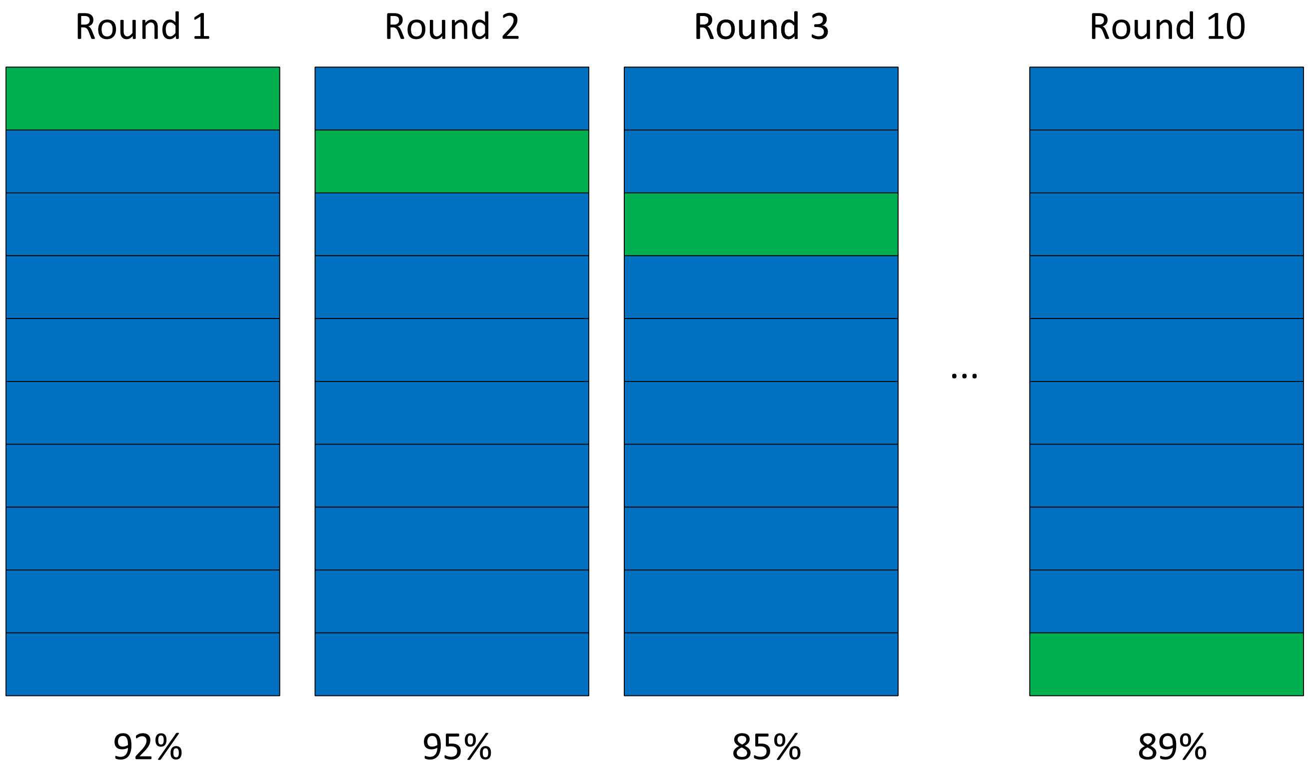
- Disclaimer: Performance of a model may depend on other factors besides the learning algorithm:
  - Class distribution
  - Cost of misclassification
  - Size of training and test sets

eljalao@up.edu.ph
156

Methods of Estimation

- Holdout
  - Reserve x% for training and 100-x% for testing
- Cross validation
  - Estimates the performance of the Model generated using all data.
  - Partition data into k disjoint subsets
  - k-fold: train on $k - 1$ partitions, test on the remaining one
  - If $k = 10$, 11 Decision Tree Models will be created. One for each fold, and 1 using all data.

Cross Validation



Overall Accuracy = Ave(Round 1, Round 2...)

Where:

Training Data

Testing Data

eljalao@up.edu.ph

Cross Validation

- Type the following lines of code in RStudio and run.

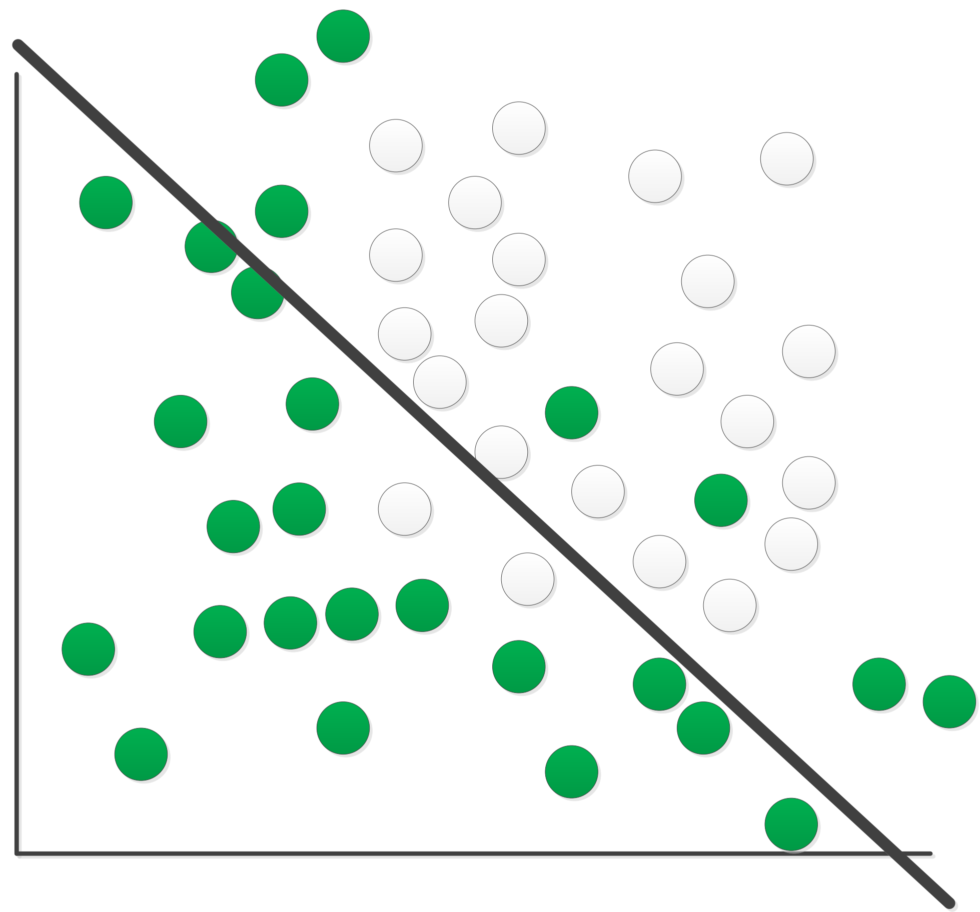
```
bankdata = read.csv("bankdata.csv")
J48Model <- J48(pep ~ age + sex+  region  + income
                + married + children  + car
                + save_act+ current_act+  mortgage
                , data=bankdata)
summary(J48Model)
evaluate_Weka_classifier(J48Model,
                         newdata = bankdata,
                         numFolds = 10, class = TRUE, seed=1)
```

Cross Validation

```
=== 10 Fold Cross Validation ===

=== Summary ===

Correctly Classified Instances         539               89.8333 %
Incorrectly Classified Instances        61               10.1667 %
Kappa statistic                          0.7942
Mean absolute error                      0.167
Root mean squared error                  0.305
Relative absolute error                 33.6511 %
Root relative squared error             61.2344 %
Total Number of Instances              600

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.929    0.139    0.889      0.929    0.909      0.795    0.883     0.863     NO
                 0.861    0.071    0.911      0.861    0.886      0.795    0.883     0.847     YES
Weighted Avg.    0.898    0.108    0.899      0.898    0.898      0.795    0.883     0.856

=== Confusion Matrix ===

   a   b   <-- classified as
 303  23 |   a = NO
  38 236 |   b = YES
```

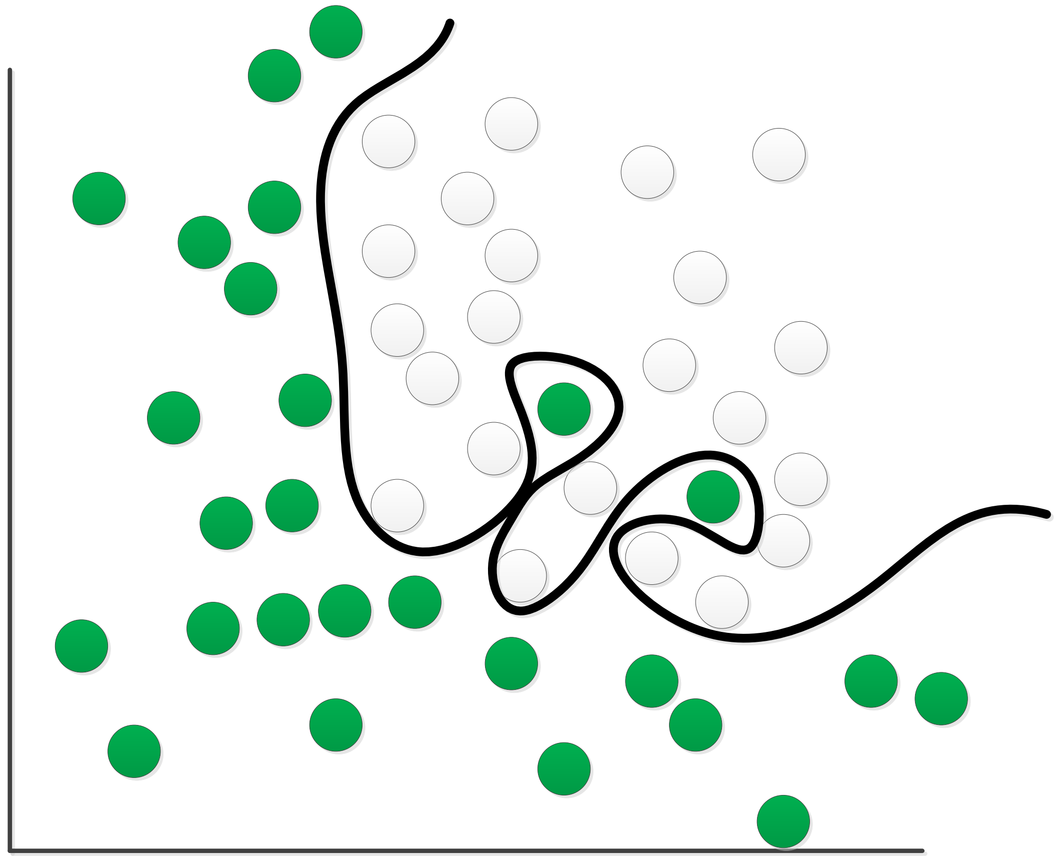
Underfitting and Overfitting

- Underfit
  - Simple Model
  - Lots of Errors
  - Stable Prediction



Underfitting and Overfitting

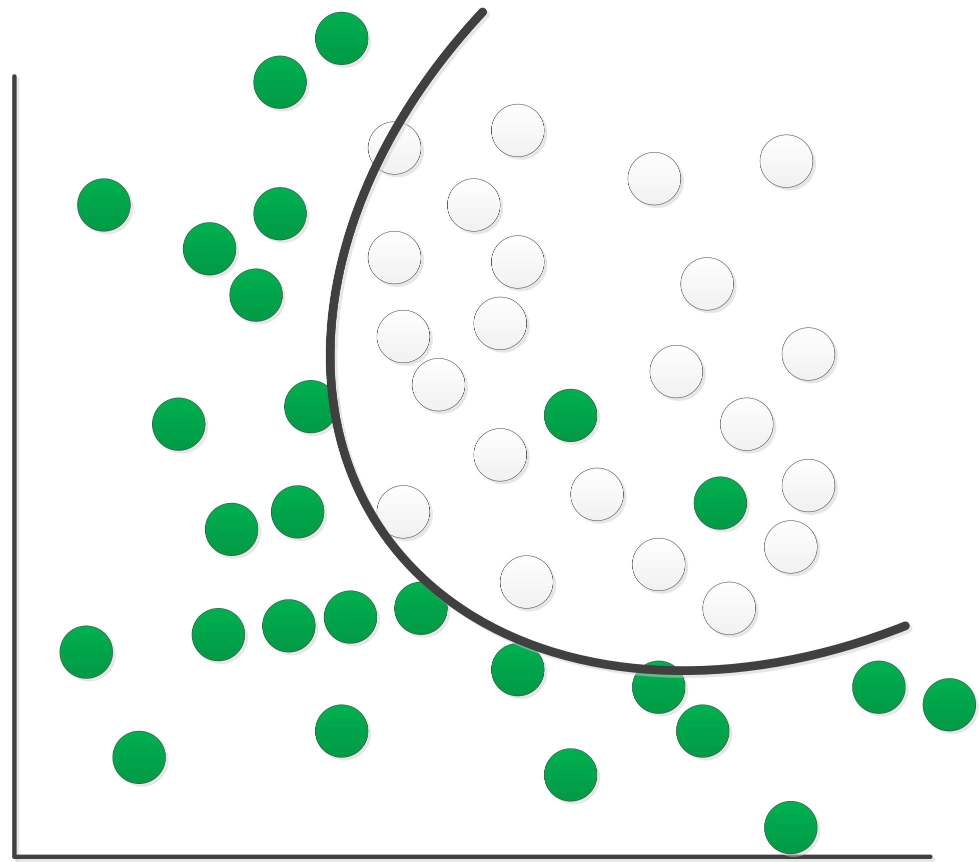
- Overfit
  - Complex Model
  - No Errors
  - Unstable Prediction
  - Predicts Even Noise



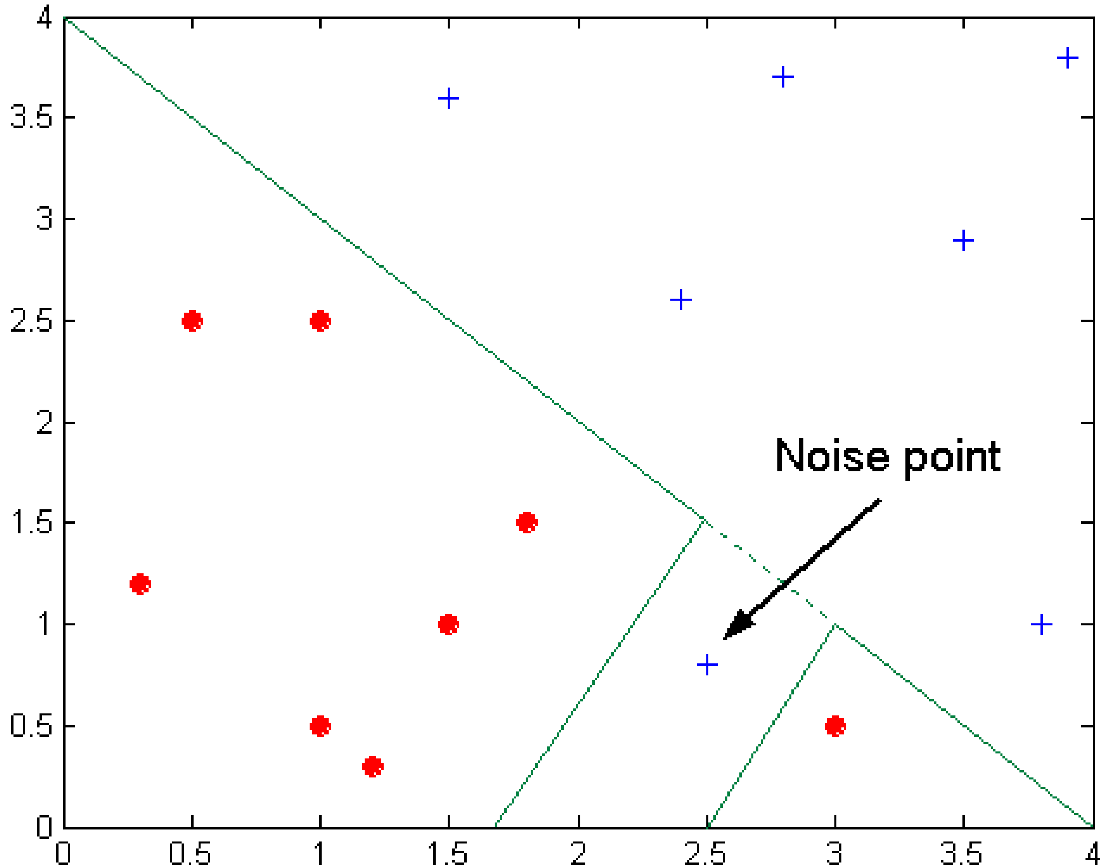
162

Underfitting and Overfitting

- Just Right
  - Small Errors
  - Stable Predictions

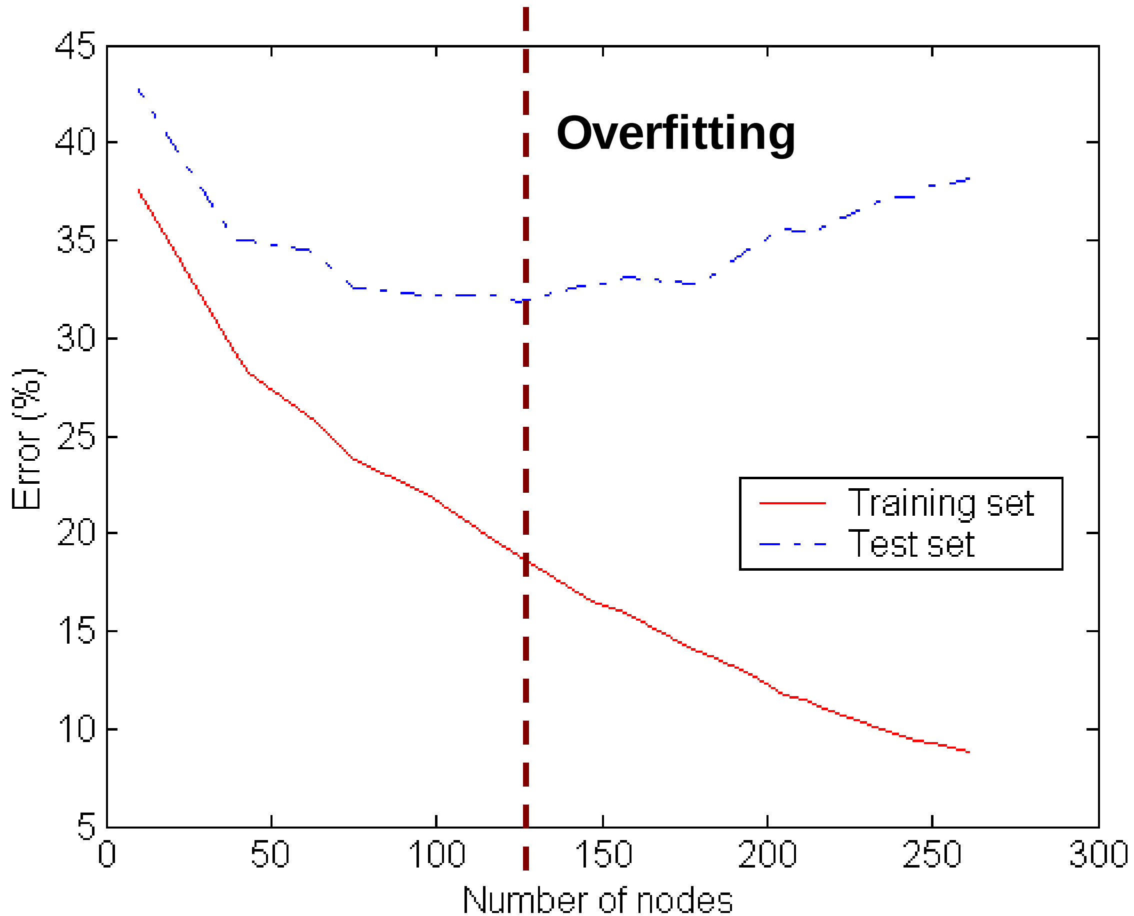


Overfitting due to Noise



Decision boundary is distorted by noise point

Underfitting and Overfitting



Underfitting: when model is too simple, both training and test errors are large

Notes on Overfitting

- Overfitting results in classification models that are more complex than necessary
- Training error no longer provides a good estimate of how well the tree will perform on previously unseen records
- Need new ways for estimating errors

How to Address Overfitting

- Post-pruning
  - Grow model to its entirety
  - Trim the complexity of the model
  - If generalization error improves after trimming, replace the complex model with the less complex model

167

Addressing Overfitting and Underfitting

- Fitting a complex model:

```
bankdata = read.csv("bankdata.csv")
J48Overfit <- J48(pep ~ age + sex+  region  + income
               + married + children  + car
               + save_act+ current_act+  mortgage
               , data=bankdata, control = Weka_control(U =TRUE))
J48Overfit
summary(J48Overfit)
evaluate_Weka_classifier(J48Overfit,
                        newdata = bankdata,
                        numFolds = 10, class = TRUE,seed =1)
```


eljalao@up.edu.ph

Overfitting Model

- Complexity:
- Training Error:

```
Number of Leaves  :      64

Size of the tree :       113
```

```
> summary(J48Overfit)

=== Summary ===

Correctly Classified Instances         572               95.3333 %
Incorrectly Classified Instances        28                4.6667 %
```

- Testing Error

```
> evaluate_Weka_classifier(J48Overfit,
+ newdata = bankdata,
+ numFolds = 10, class = TRUE,seed =1)
=== 10 Fold Cross Validation ===

=== Summary ===

Correctly Classified Instances         519               86.5     %
Incorrectly Classified Instances        81               13.5     %
```

Addressing Overfitting and Underfitting

- Fitting a underfit model:

```
bankdata = read.csv("bankdata.csv")
J48ModelUnderfit <- J48(pep ~ age + sex+  region  + income
                        + married  + children  + car
                        +  save_act+ current_act+  mortgage
                        , data=bankdata, control = Weka_control(M = 30))
J48ModelUnderfit
summary(J48ModelUnderfit)
evaluate_Weka_classifier(J48ModelUnderfit,
                        newdata = bankdata,
                        numFolds = 10, class = TRUE,seed =1)
```


eljalao@up.edu.ph

Underfit Model

- Complexity:

```
Number of Leaves  :      9

Size of the tree :      17
```

- Training Error:

```
> summary(J48ModelUnderfit)

=== Summary ===

Correctly Classified Instances         501               83.5     %
Incorrectly Classified Instances        99               16.5     %
```

- Testing Error

```
> evaluate_Weka_classifier(J48ModelUnderfit,
+ newdata = bankdata,
+ numFolds = 10, class = TRUE,seed =1)
=== 10 Fold Cross Validation ===

=== Summary ===

Correctly Classified Instances         482               80.3333 %
Incorrectly Classified Instances       118               19.6667 %
```

Addressing Overfitting and Underfitting

- Fitting a pruned model:

```
bankdata = read.csv("bankdata.csv")
J48ModelwPrunning <- J48(pep ~ age + sex+ region  + income
                        + married  + children  + car
                        +  save_act+ current_act+  mortgage
                        , data=bankdata)
J48ModelwPrunning
summary(J48ModelwPrunning)
evaluate_Weka_classifier(J48ModelwPrunning,
                         newdata = bankdata,
                         numFolds = 10, class = TRUE,
                         seed =1)
```

Pruned Model

- Complexity:
- Training Error:

```
Number of Leaves  :      15

Size of the tree :       29
```

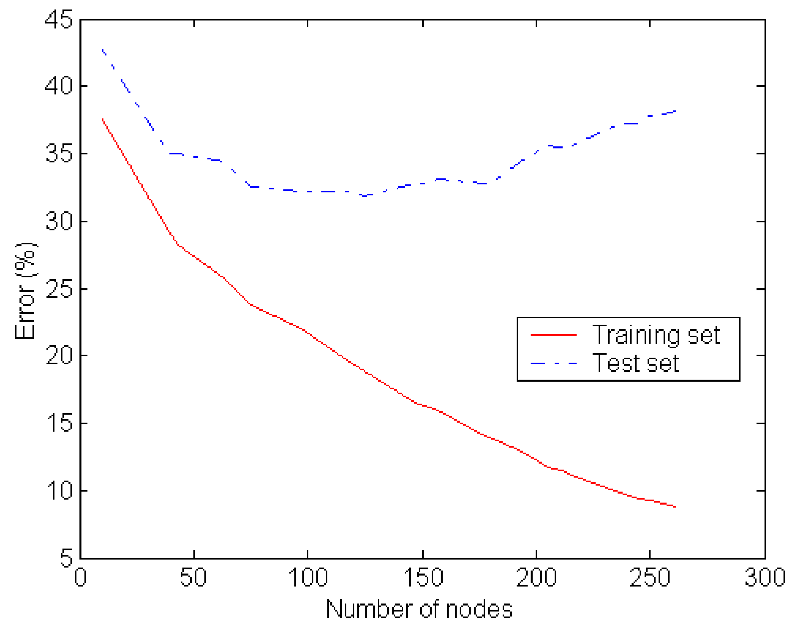
```
> summary(J48ModelWPrunning)

=== Summary ===

Correctly Classified Instances         554               92.3333 %
Incorrectly Classified Instances        46                7.6667 %
```

- Testing Error:

```
> evaluate_Weka_classifier(J48ModelWPrunning,
+                          newdata = bankdata,
+                          numFolds = 10, class = TRUE,
+                          seed =1)
=== 10 Fold Cross Validation ===

=== Summary ===

Correctly Classified Instances         539               89.8333 %
Incorrectly Classified Instances        61               10.1667 %
```

Model Summaries

Type of Model	Training Error	Testing Error
Underfit Model	16.5%	19.67%
Just Right Model	7.67%	10.16%
Overfit Model	4.67%	13.50%



174

Model Evaluation

- How to evaluate the performance of a model?
  - Metrics for Performance Evaluation
- How to obtain reliable estimates?
  - Methods for Performance Evaluation
  - Overfitting and Underfitting
- **How to compare the relative performance among competing models?**
  - Methods for Model Comparison

Limitation of Accuracy

- Consider a 2-class problem
  - Number of Class "–" examples = 9990
  - Number of Class "+" examples = 10

- If model predicts everything to be class "–", accuracy is 9990/10000 = 99.9 %
  - Accuracy is misleading because model does not detect any class "+" example

Cost Matrix

	PREDICTED CLASS		
ACTUAL CLASS	C(i|j)	**Class=Yes**	**Class=No**
	Class=Yes	C(Yes|Yes)	C(No|Yes)
	Class=No	C(Yes|No)	C(No|No)

C(i|j): Cost of misclassifying class j example as class i

Computing Cost of Classification

Cost Matrix	PREDICTED CLASS		
ACTUAL CLASS	C(i|j)	+	-
	+	-1	100
	-	1	0

Model M_1	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	255	40
	-	60	145

Accuracy = 80%

Cost = 3805

Model M_2	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	250	45
	-	5	200

Accuracy = 90%

Cost = 4255

Computing Cost of Classification

- With Cost

```
bankdata = read.csv("bankdata.csv")
J48Model <- J48(pep ~ age + sex+  region  + income
                + married + children  + car
                + save_act+ current_act+  mortgage
                , data=bankdata)
summary(J48Model)
evaluate_Weka_classifier(J48Model, cost = matrix(c(0,2,1,0), ncol = 2),
                         newdata = bankdata,
                         numFolds = 10, class = TRUE, seed=1)
```

- Result:

```
=== Summary ===

Correctly Classified Instances         539               89.8333 %
Incorrectly Classified Instances        61               10.1667 %
Kappa statistic                          0.7942
Total Cost                              99
```


eljalao@up.edu.ph

Cost-Sensitive Measures

$$\text{Precision (p)} = \frac{a}{a+c}$$

$$\text{Recall (r)} = \frac{a}{a+b}$$

$$\text{F-measure (F)} = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

- Precision is biased towards C(Yes|Yes) & C(Yes|No)
- Recall is biased towards C(Yes|Yes) & C(No|Yes)
- F-measure is biased towards all except C(No|No)

$$\text{Weighted Accuracy} = \frac{w_1 a + w_4 d}{w_1 a + w_2 b + w_3 c + w_4 d}$$

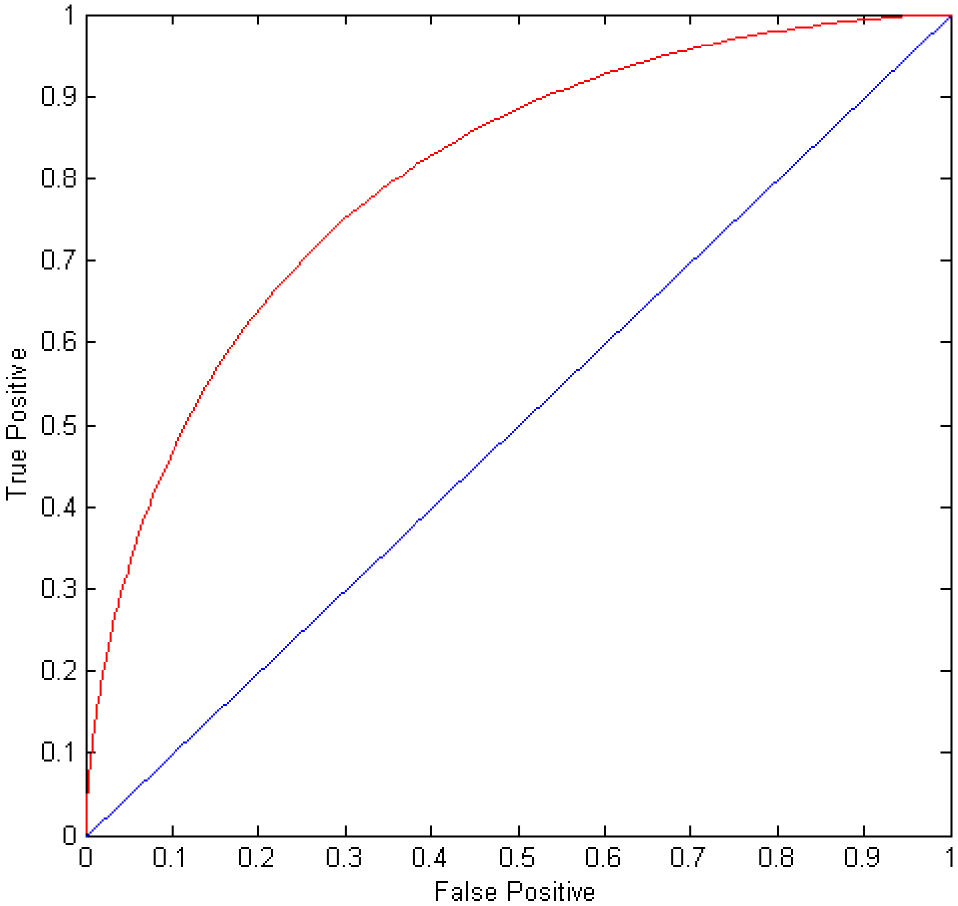
E.R. L. Jalao, Copyright UP NEC,
eljalao@up.edu.ph

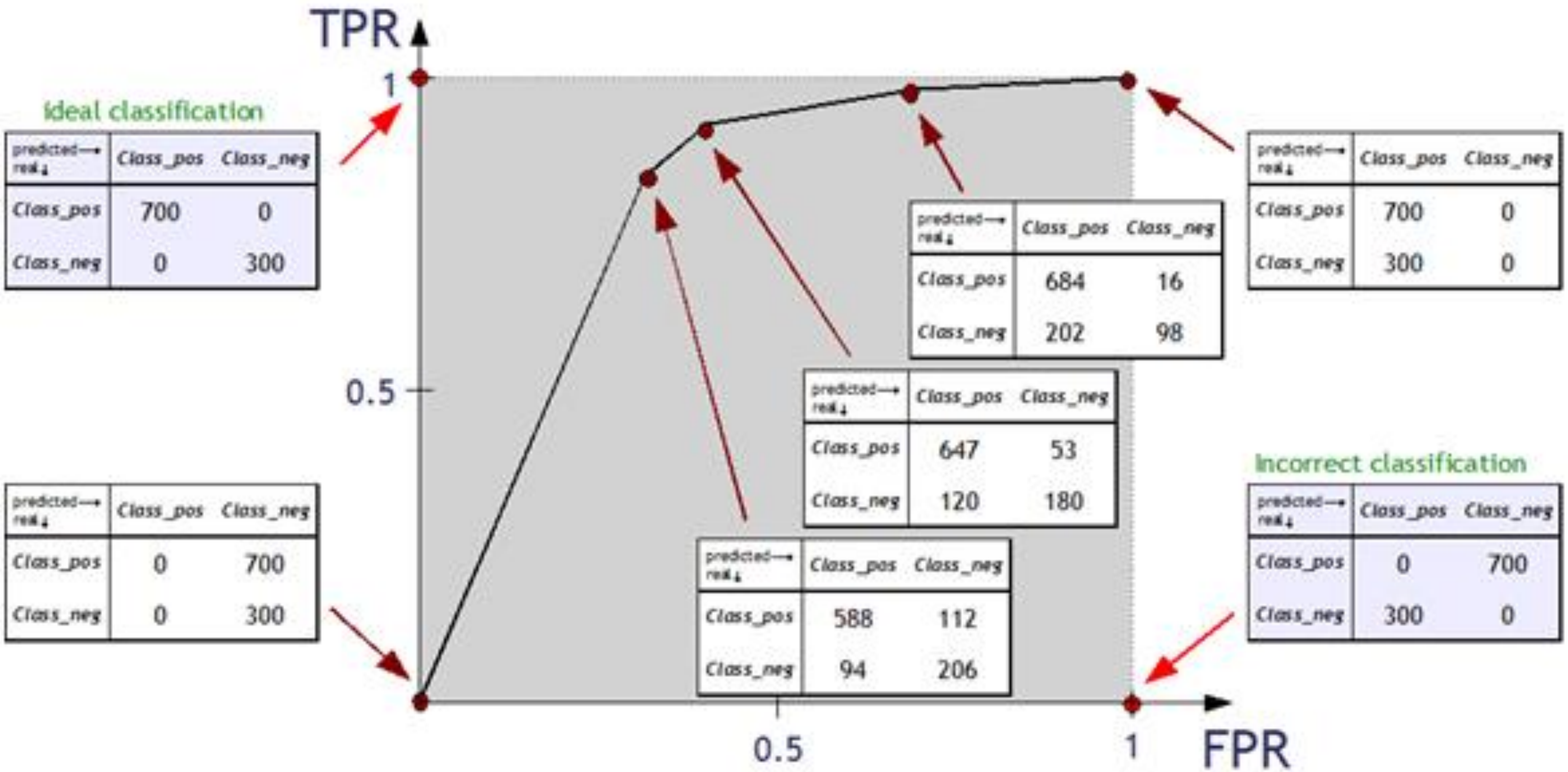
ROC (Receiver Operating Characteristic)

- Developed in 1950s for signal detection theory to analyze noisy signals
  - Characterize the trade-off between positive hits and false alarms
- ROC curve plots TP (on the y-axis) against FP (on the x-axis)
- Performance of each classifier represented as a curve with varying TP and FP performance

ROC Curve

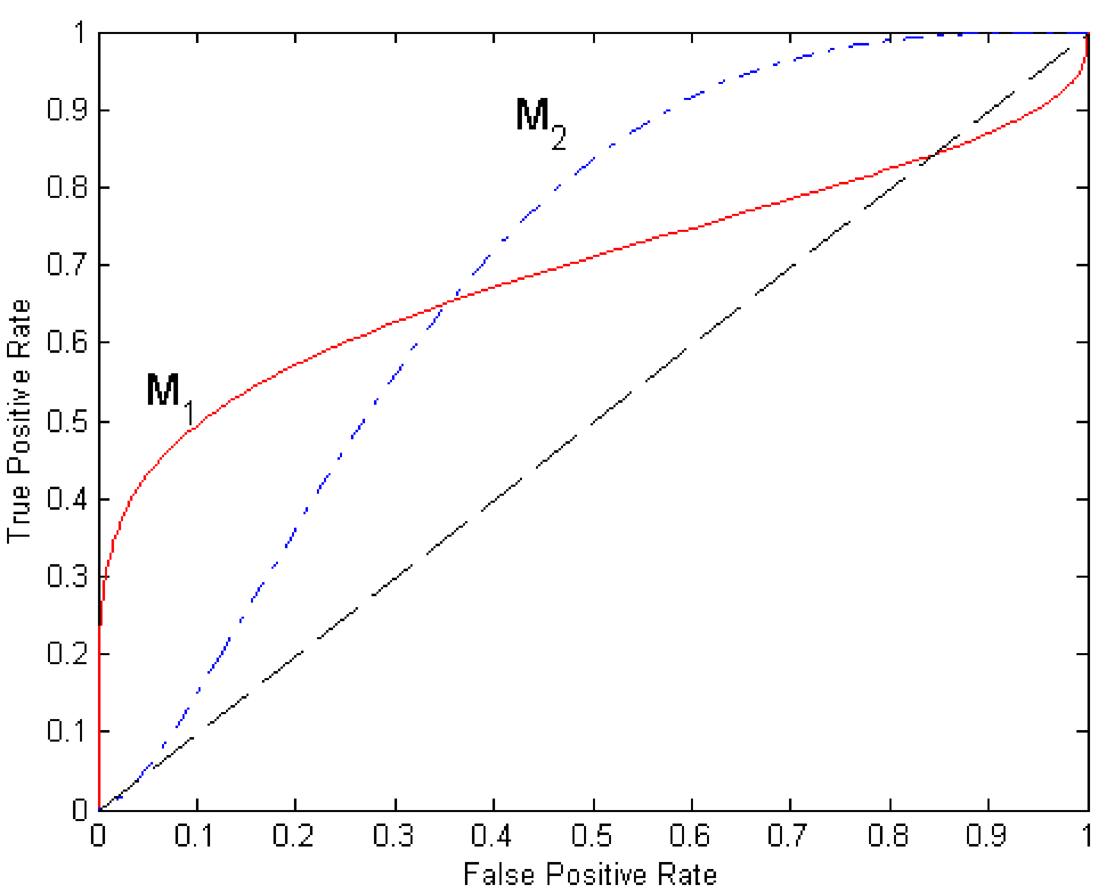
- (TP,FP):
- (0,0): declare everything to be negative class
- (1,1): declare everything to be positive class
- (1,0): ideal
- Diagonal line:
  - Random guessing
  - Below diagonal line:
    - prediction is opposite of t





http://www.datasciencecentral.com/profiles/blogs/how-to-assess-quality-and-correctness-of-classification-models

Using ROC for Model Comparison



- No model consistently outperform the other
  - M_1 is better for small FPR
  - M_2 is better for large FPR
- Area Under the ROC curve
  - Ideal:
    - Area = 1
  - Random guess:
    - Area = 0.5

Practical Application of ROCs

- Predicting **Churn**
  - Must have high true positive rate = very low false negative rate
    - Must be able to capture all Churn Customers
  - Ok to have high false positive rate
    - Predicting that you will Churn but actually will Not Churn
  - **Choose Model 2**
- Predicting **Flu**
  - Must have **low false positive** rate
    - Predicting You Have Flu but actually have no Flu
  - Must have an acceptable true positive rate
    - Must be able to detect Flu at least 60% of the time
  - **Choose Model 1**

Using ROC Curves

- Type the following lines of code in RStudio and run.

```
bankdata = read.csv("bankdata.csv")
sample <- floor(0.67 * nrow(bankdata))
set.seed(123)
train_ind <- sample(seq_len(nrow(bankdata)),
                    size = sample)
bankdatatrain <- bankdata[train_ind, ]
bankdatatest <- bankdata[-train_ind, ]
J48Model <- J48(pep ~ age + sex+  region  + income
                + married + children  + car
                + save_act+ current_act+  mortgage
                , data=bankdatatrain)
JRipModel <- JRip(pep ~ age + sex+  region  + income
                  + married + children  + car
                  + save_act+ current_act+  mortgage
                  , data=bankdatatrain)
```

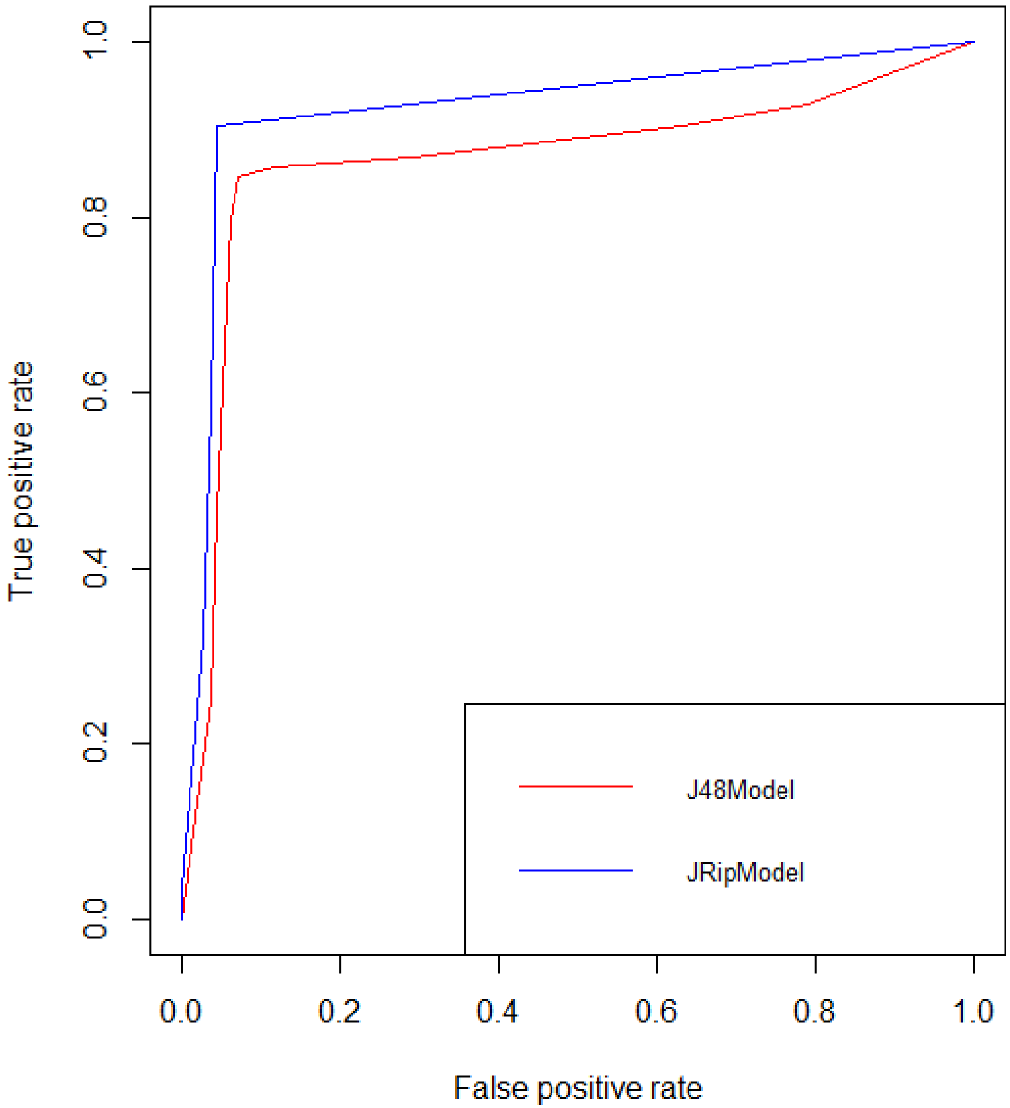

eljalao@up.edu.ph

Using ROC Curves

- Type the following lines of code in RStudio and run.

```
library("ROCR")
labels = ifelse(bankdatatest$pep=="YES",1,0)
predictions = cbind(predict(J48Model,
                            newdata =bankdatatest,
                            type = c("probability"))[,c("YES")],
                    predict(JRipModel,
                            newdata =bankdatatest,
                            type = c("probability"))[,c("YES")])
labels = cbind(labels,labels)
pred2 <- prediction( predictions, labels)
perf2 <- performance(pred2,"tpr","fpr")
plot(perf2, col=list("red", "blue"))
legend("bottomright", legend=c("J48Model", "JRipModel"),
       col=c("red", "blue"), lty=1:1, cex=0.8)
```

Using ROC Curves



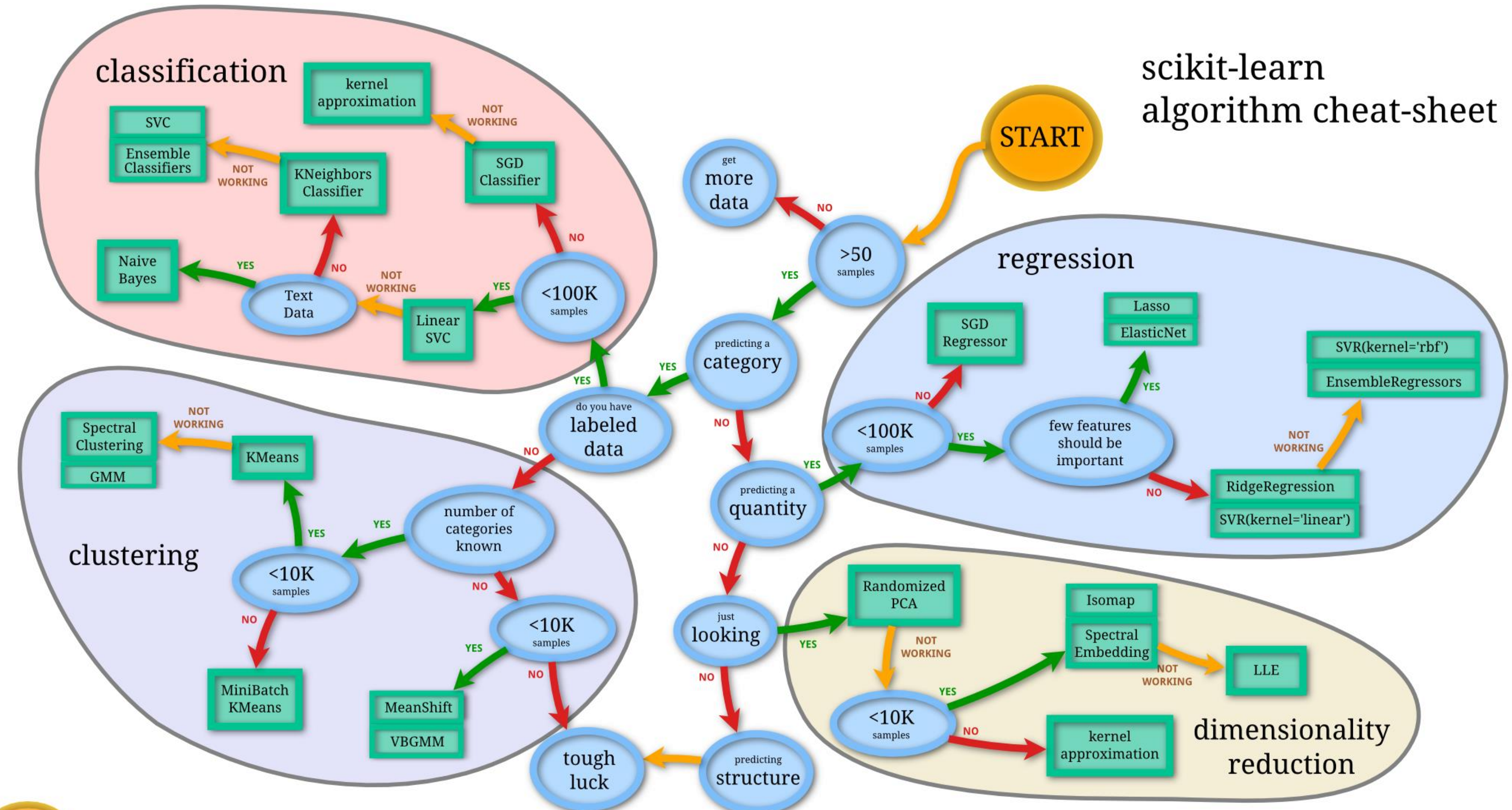
This Session's Outline

- What is Classification?
- Frequency Table
  - Zero R
  - One R
  - Naïve Bayes
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - ANN
  - SVM
- Ensembles
  - Adaboost
  - Random Forests
- Model Evaluation
- **Case Study**

Case 2: Classification Using R

- Selecting the Best Classification Model for the Churn Data Using R

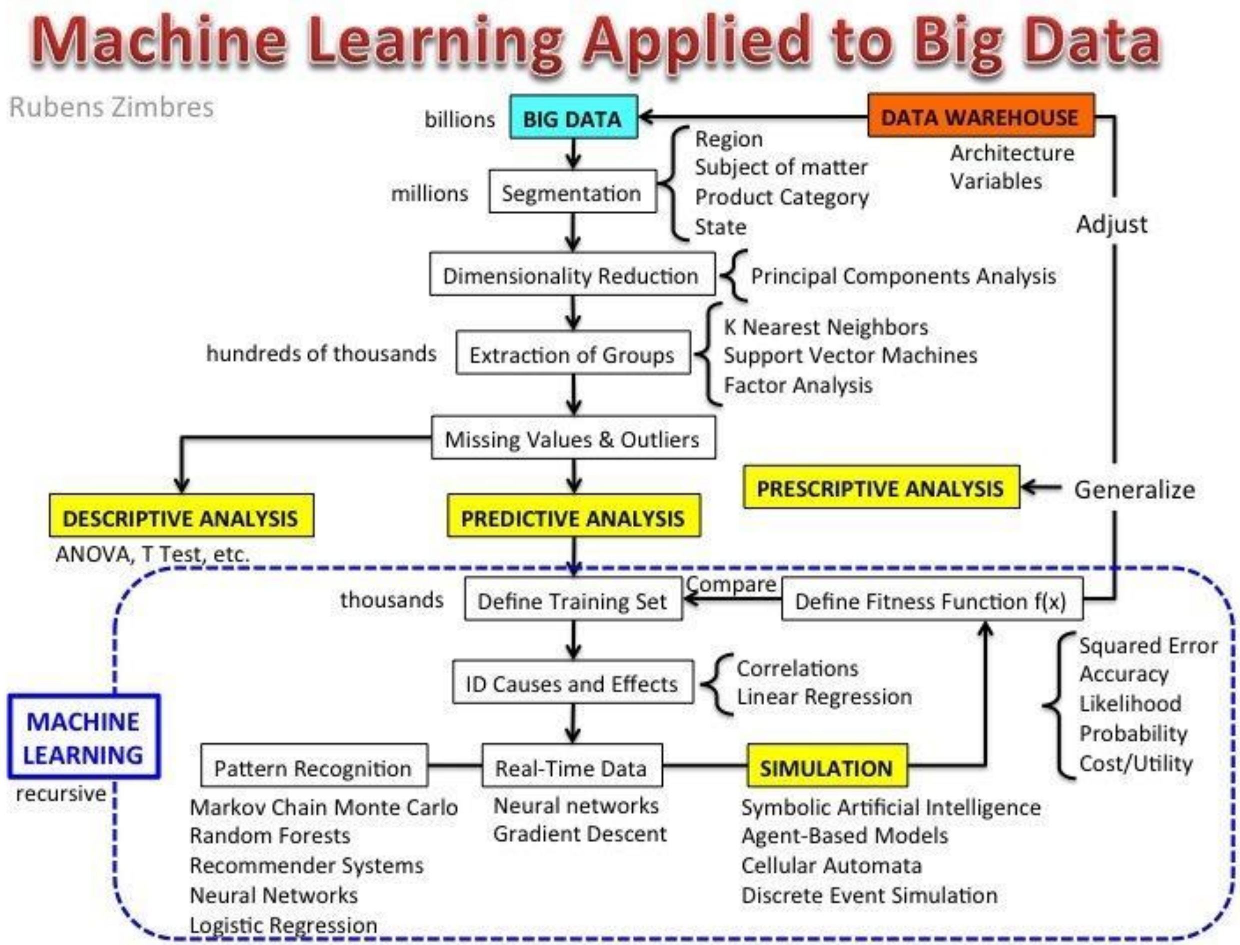
Algorithm Cheat Sheet



Back

http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

Algorithm Cheat Sheet



http://www.datasciencecentral.com/profiles/blogs/key-tools-of-big-data-for-transformation-review-case-study

This Session's Outline

- What is Classification?
- Frequency Table
  - Zero R
  - One R
  - Naïve Bayes
  - Decision Tree
  - Rule Based Classifiers
- Similarity
  - K-Nearest Neighbors
- Perceptron Based
  - ANN
  - SVM
- Ensembles
  - Adaboost
  - Random Forests
- Model Evaluation
- Case Study

References

- Data Mining Overview: http://www.saedsayad.com/
- Tan et al. Intro to Data Mining Notes
- Runger, G. IEE 520 notes
- http://axon.cs.byu.edu/Dan/678/miscellaneous/SVM.example.pdf
- G. Runger, ASU IEE 578