



ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN  
VIETNAM - KOREA UNIVERSITY OF INFORMATION AND COMMUNICATION TECHNOLOGY

한-베정보통신기술대학교

Nhân bản – Phụng sự – Khai phóng

## Chapter 4

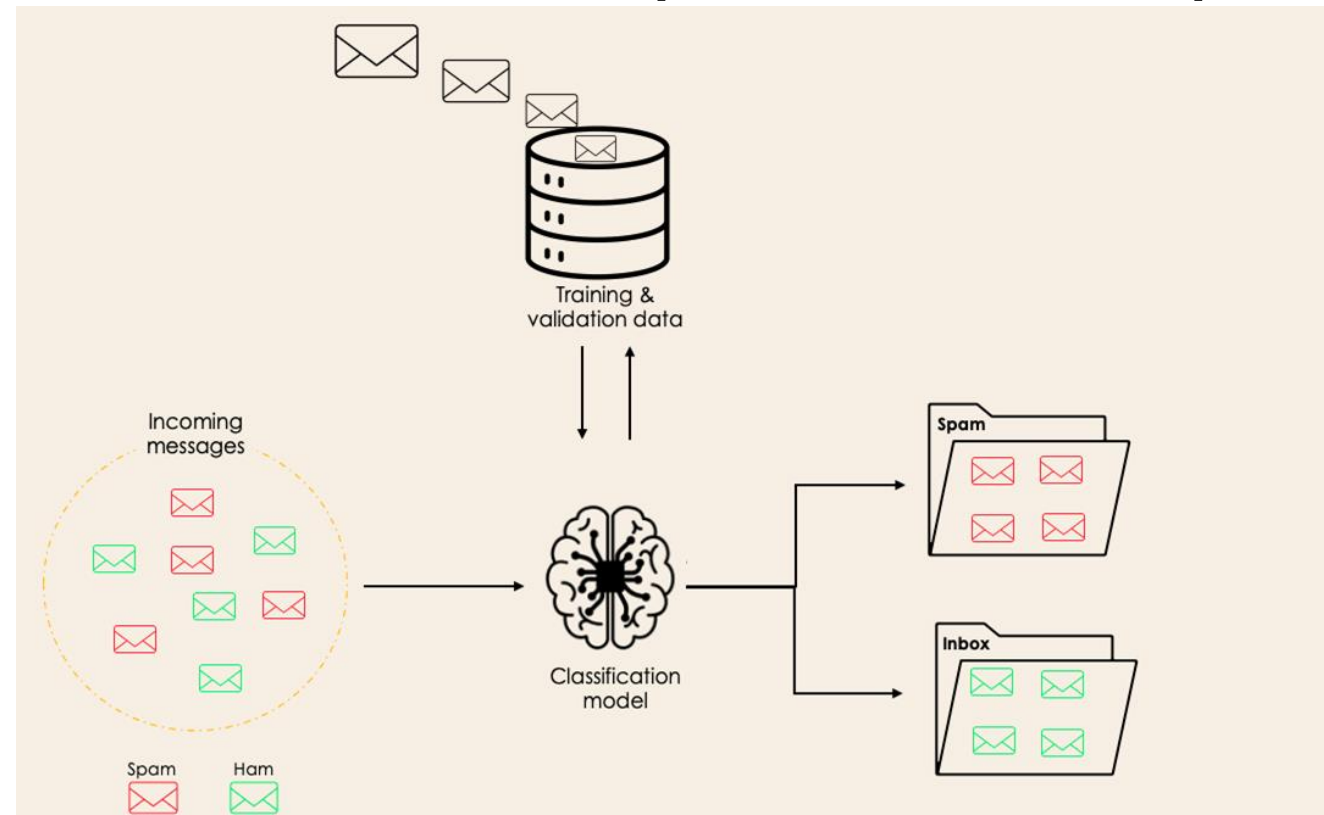
# Classification Techniques

Machine Learning

- **Classification Problems**
- **Classification Algorithms**
  - **K-Nearest Neighbors**
  - **Naïve Bayes Classification**
  - **Decision Tree**
  - **Support Vector Machines**
- **Metrics to measure Classification Performance**

- **Classification Problems**
- **Classification Algorithms**
  - K-Nearest Neighbors
  - Naïve Bayes Classification
  - Decision Tree
  - Support Vector Machines
- **Metrics to measure Classification Performance**

- **Classification** is a supervised task that requires the use of machine learning algorithms that learn how to assign a class label to examples from the problem domain



- Classification requires a training dataset with many examples of inputs and outputs from which to learn.
- A ML model will use the training dataset and will calculate how to best map examples of input data to specific class labels.
- The training dataset must be sufficiently representative of the problem and have many examples of each class label.

Highest Degree	Work Experience	Favorite Language	Needs Work Visa	Hire
Bachelors	Mobile Dev	Objective-C	TRUE	yes
Masters	Web Dev	Java	FALSE	yes
Masters	Mobile Dev	Java	TRUE	yes
PhD	Mobile Dev	Objective-C	TRUE	yes
PhD	Web Dev	Objective-C	TRUE	no
Bachelors	UX Design	Objective-C	TRUE	no
Bachelors	Mobile Dev	Java	FALSE	yes
PhD	Web Dev	Objective-C	FALSE	no
Bachelors	UX Design	Java	FALSE	yes
Masters	UX Design	Objective-C	TRUE	no
Masters	UX Design	Java	FALSE	yes
PhD	Mobile Dev	Java	FALSE	no
Masters	Mobile Dev	Java	TRUE	yes
Bachelors	Web Dev	Objective-C	FALSE	no

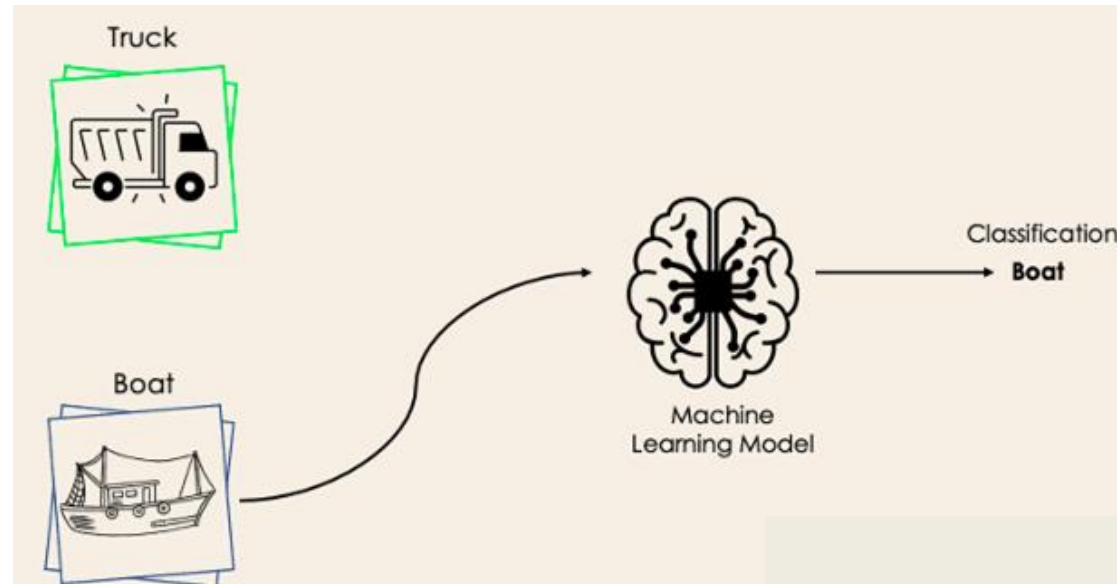
- Can we predict the class of the new example?

Highest Degree	Work Experience	Favorite Language	Needs Work Visa	Hire
Masters	UX Design	Java	TRUE	?



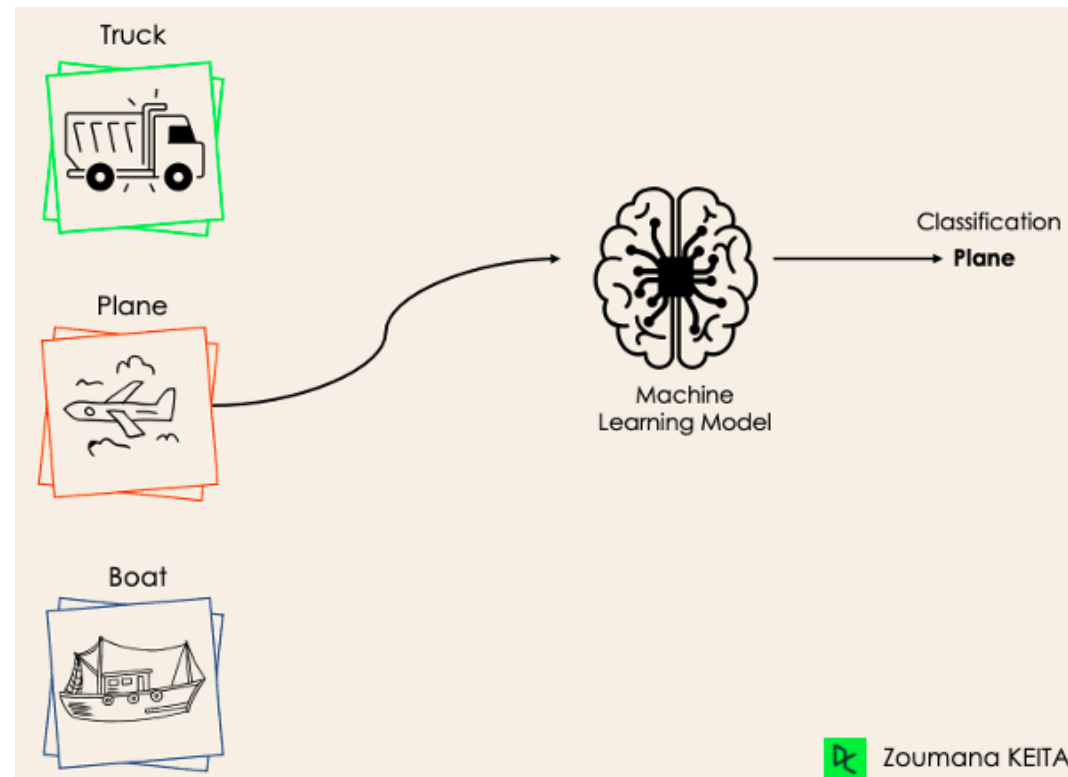
- Email Spam Detection
- Speech Recognition
- Identifications of Cancer tumor cells.
- Drugs Classification
- Biometric Identification, etc.

- **Binary Classification** classify the input data into two mutually exclusive categories.
- The training data in such a situation is labeled in a binary format: *true* and *false*; *positive* and *negative*; 0 and 1; *spam* and *not spam*, etc. depending on the problem being tackled.

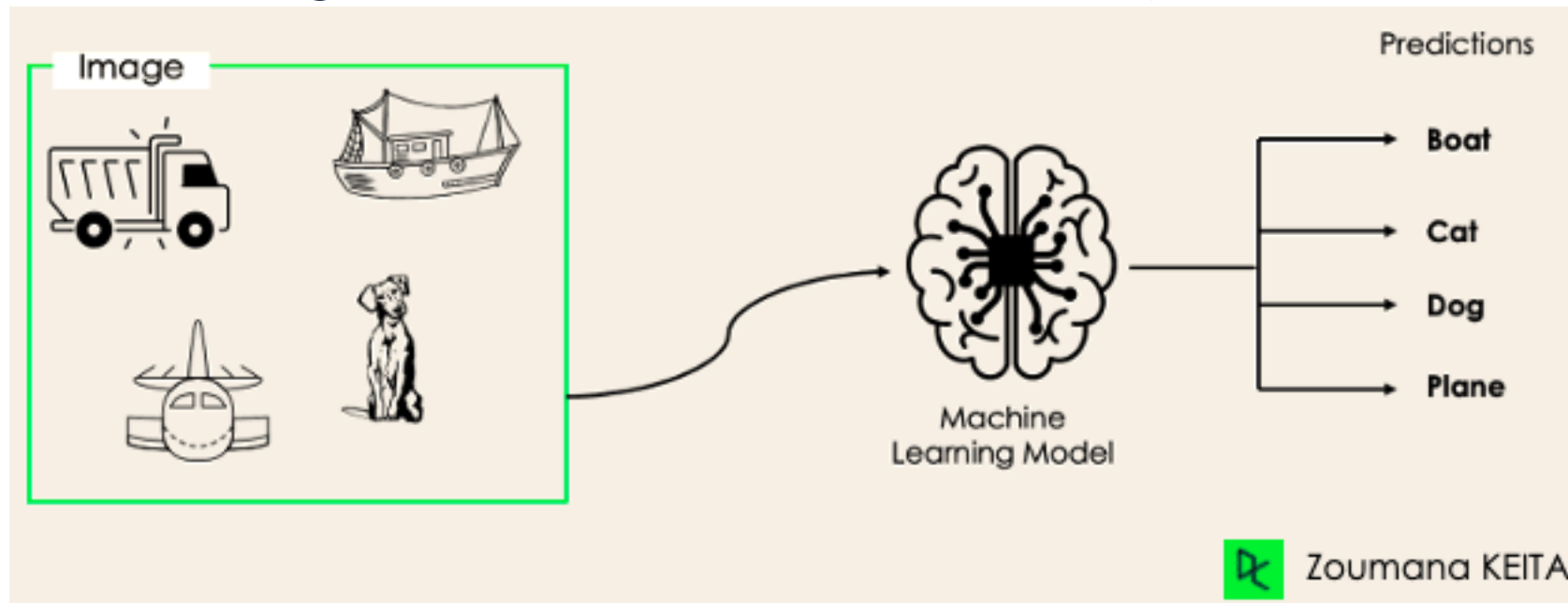




- **Multi-class classification** is where each data sample is assigned one and only one label from more than two classes.




- **Multi-Label classification** refers to predicting zero or more labels to each data sample.
- Example: auto-tagging in Natural Language Processing, where a given text can contain multiple topics or in computer vision, an image can contain multiple objects.









Instance	Target
X1	
X2	
X3	

a

Instance	Target
X1	
X2	
X3	

b

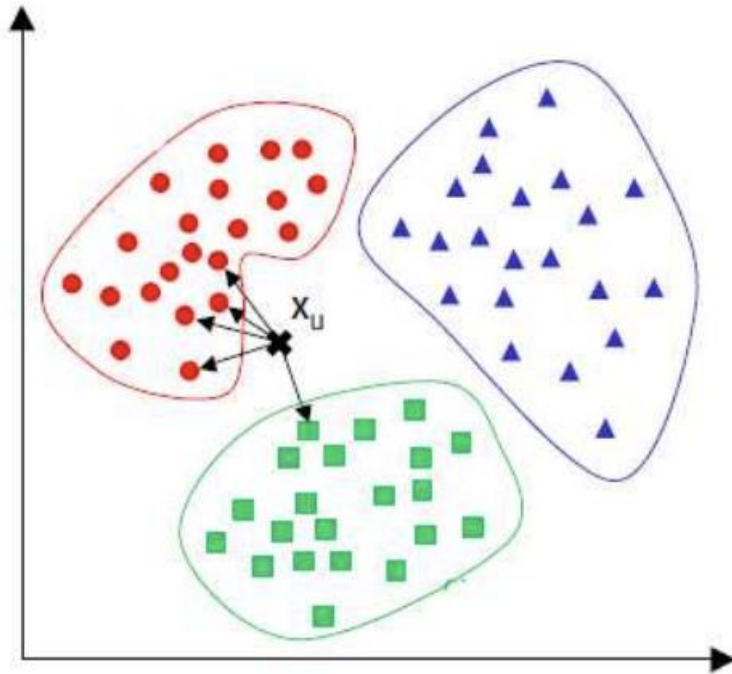
Instance	Target
X1	 
X2	  
X3	

c

????

- Classification Problems
- **Classification Algorithms**
  - **K-Nearest Neighbors**
  - Naïve Bayes Classification
  - Decision Tree
  - Support Vector Machines
- Metrics to measure Classification Performance

- Given training data  $D = \{(x_1, y_1), \dots, (x_N, y_N)\}$  and a test point
- Prediction Rule: Look at the  $K$  most similar training examples
- For classification: assign the majority class label (majority voting)
- For regression: assign the average response



- The algorithm requires:
  - ◆ **Parameter  $K$ :** number of nearest neighbors to look for
  - ◆ **Distance function:** To compute the similarities between examples
- Special Case: 1-Nearest Neighbor

- Compute the test point's distance from each training point
- Sort the distances in ascending (or descending) order
- Use the sorted distances to select the K nearest neighbors
- Use majority rule (for classification) or averaging (for regression)



- K-NN is called a ***non-parametric*** method
- Unlike other supervised learning algorithms, K-Nearest Neighbors doesn't learn an explicit mapping  $f$  from the training data
- It simply uses the training data at the test time to make predictions

- The K-NN algorithm requires computing distances of the test example from each of the training examples.
- Several ways to compute distances.
- The choice depends on the type of the features in the data.
  - ♦ Real-valued features ( $\mathbf{x}_i \in \mathbb{R}^D$ ): **Euclidean distance** is commonly used.

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{m=1}^D (x_{im} - x_{jm})^2} = \sqrt{\|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2 - 2\mathbf{x}_i^T \mathbf{x}_j}$$

→ Some other distance measures

◆ Binary-valued features

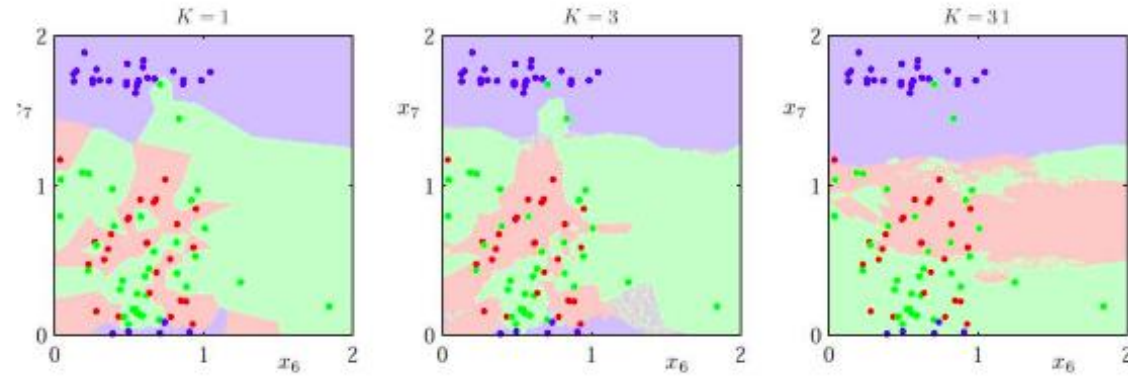
$$d(x_i, x_j) = \sum_{m=1}^D \mathbb{I}(x_{im} \neq x_{jm})$$

- Use Hamming distance:
- Hamming distance counts the number of features where the two examples disagree

◆ Mixed feature types (some real-valued and some binary-valued)?

- Can use mixed distance measures
- E.g., Euclidean for the real part, Hamming for the binary part

◆ Can  $d(x_i, x_j) = \sum_{m=1}^D w_m d(x_{im}, x_{jm})$  :

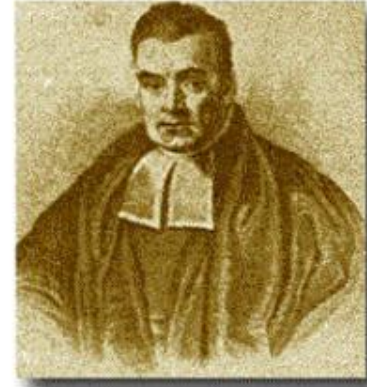


- Small K
  - ◆ Creates many small regions for each class
  - ◆ May lead to non-smooth) decision boundaries and overfit
- Large K
  - ◆ Creates fewer larger regions
  - ◆ Usually leads to smoother decision boundaries (caution: too smooth decision boundary can underfit)
  - ◆ Choosing K
- Often data dependent and heuristic based
  - ◆ Or using cross-validation (using some held-out data)
  - ◆ In general, a K too small or too big is bad!

- What's nice
  - ◆ Simple and intuitive; easily implementable
  - ◆ Asymptotically consistent (a theoretical property)
    - With infinite training data and large enough K, K-NN approaches the best possible classifier (Bayes optimal)
- What's not so nice..
  - ◆ Store all the training data in memory even at test time
    - Can be memory intensive for large training datasets
    - An example of non-parametric, or memory/instance-based methods
    - Different from parametric, model-based learning models
  - ◆ Expensive at test time:  $O(ND)$  computations for each test point
    - Have to search through all training data to find nearest neighbors
    - Distance computations with N training points (D features each)
  - ◆ Sensitive to noisy features

- Classification Problems
- **Classification Algorithms**
  - K-Nearest Neighbors
  - **Naïve Bayes Classification**
  - Decision Tree
  - Support Vector Machines
- Metrics to measure Classification Performance





**Bayes, Thomas (1763)** An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London*, 53:370-418

→ Bayes Rule: 
$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

→ Prior :  $P(A)$

→ Posterior :  $P(A|B)$

...by no means merely a curious speculation in the doctrine of chances, but necessary to be solved in order to a sure foundation for all our reasonings concerning past facts, and what is likely to be hereafter.... necessary to be considered by any that would give a clear account of the strength of analogical or inductive reasoning...

→ Bayes Rule:  $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$

$$P(A|B) = \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|\bar{A})P(\bar{A})}$$

→ A = you got flu                      B = you just coughed

$$P(A) = 0.05, \quad P(B|A) = 0.8, \quad P(B|\bar{A}) = 0.2$$

→ What is  $P(\text{flu} | \text{cough}) = P(A|B)$ ?

# Naïve Bayes Classification: in a Nutshell

→ Bayes Rule:  $P(Y = y_k | X_1, \dots, X_n) = \frac{P(Y = y_k) P(X_1, \dots, X_n | Y = y_k)}{\sum_j P(Y = y_j) P(X_1, \dots, X_n | Y = y_j)}$

→ If  $X_i$  and  $X_j$  are conditionally independent given  $Y$ , for all  $i \neq j$

$$P(Y = y_k | X_1, \dots, X_n) = \frac{P(Y = y_k) \prod_i P(X_i | Y = y_k)}{\sum_j P(Y = y_j) \prod_i P(X_i | Y = y_j)}$$

→ So, to pick the most probably  $Y$  for  $X^{\text{new}} = (X_1^{\text{new}}, X_2^{\text{new}}, \dots, X_n^{\text{new}})$

$$Y^{\text{new}} = \operatorname{argmax}_{y_k} P(Y = y_k) \prod_i P(X_i^{\text{new}} | Y = y_k)$$

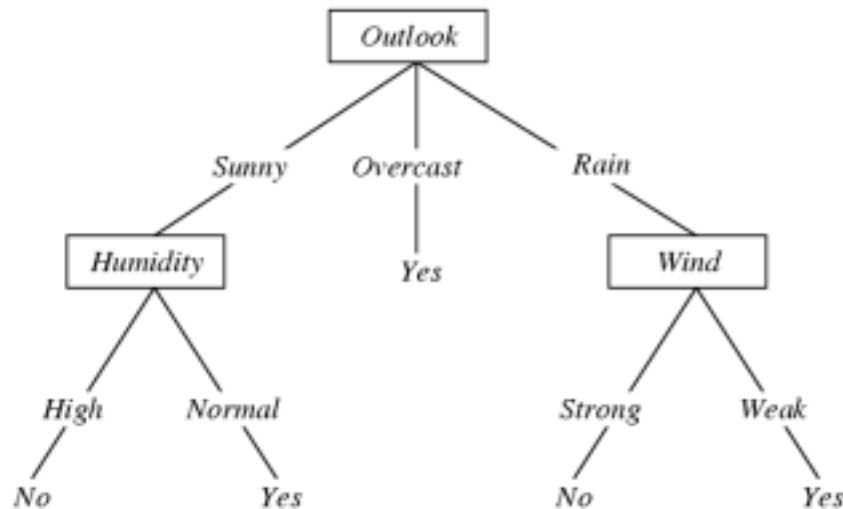
- Classification Problems
- **Classification Algorithms**
  - K-Nearest Neighbors
  - Naïve Bayes Classification
  - **Decision Tree**
  - Support Vector Machines
- Metrics to measure Classification Performance

Example: learn concept Play Tennis (i.e., decide whether our friend will play tennis or not in a given day)

Simple Training Data Set

	Day	Outlook	Temperature	Humidity	Wind	Play Tennis	
example	D1	Sunny	Hot	High	Weak	No	label
	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	

- Each internal node: test one (discrete-valued) attribute  $X_i$
- Each branch from a node: corresponds to one possible values for  $X_i$
- Each leaf node: predict  $Y$  (or  $P(Y=1 | x \in \text{leaf})$ )
- Example: A Decision tree for  $f$ : PlayTennis?



Day	Outlook	Temperature	Humidity	Wind	Play Tennis
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

label

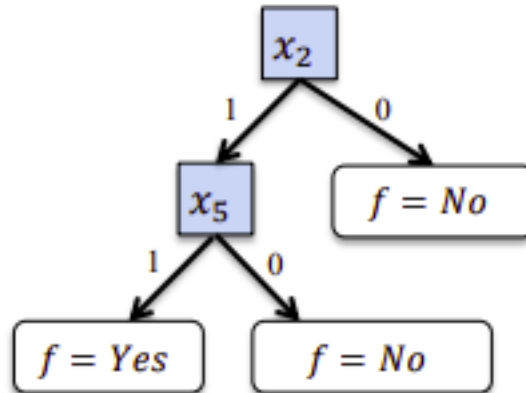
Hot, Hi

E.g.,  
 $f(x)=\text{Yes}$ .

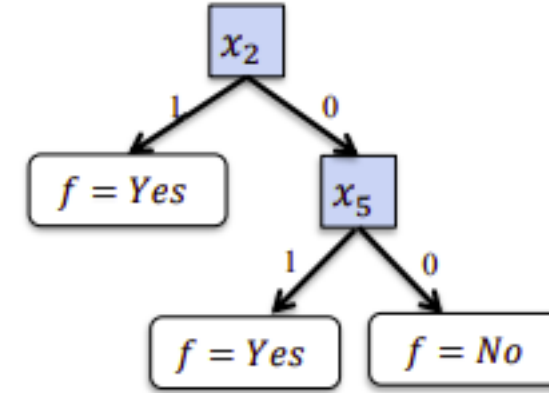


- Suppose  $X = \langle x_1, \dots, x_n \rangle$
- where  $x_i$  are boolean-valued variables
- How would you represent the following as DTs?

$$f(x) = x_2 \text{ AND } x_5 ?$$



$$f(x) = x_2 \text{ OR } x_5$$



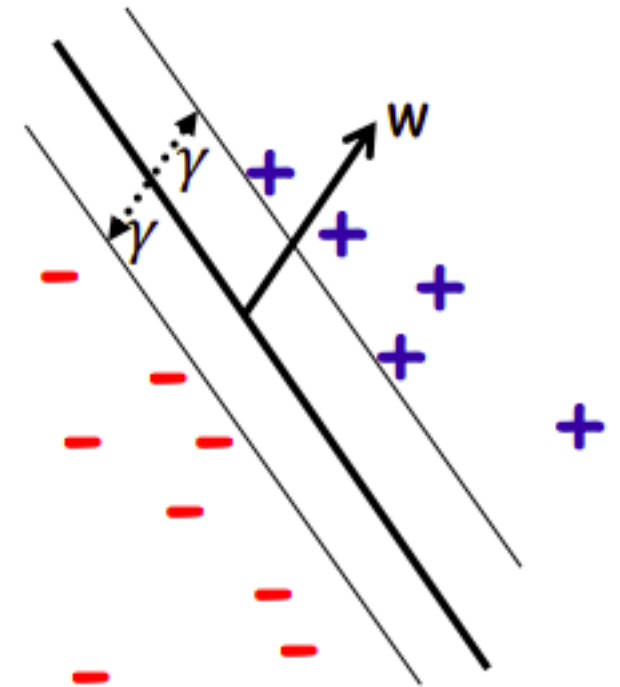
Hwk: How would you represent  $X_2 X_5 \vee X_3 X_4 (\neg X_1)$  ?

- Input: Training labeled examples  $\{(x^{(i)}, y^{(i)})\}$  of unknown target function  $f$
- ◆ Examples described by their values on some set of features or attributes
    - E.g. 4 attributes: Humidity, Wind, Outlook, Temp – e.g.,
    - Set of possible instances  $X$  (a.k.a instance space)
  - ◆ Unknown target function  $f : XY$ 
    - e.g.,  $Y=\{0,1\}$  label space
    - e.g., 1 if we play tennis on this day, else 0
- Output: Hypothesis  $h \in H$  that (best) approximates target function  $f$
- ◆ Set of function hypotheses  $H=\{h \mid h : XY\}$ 
    - each hypothesis  $h$  is a decision tree

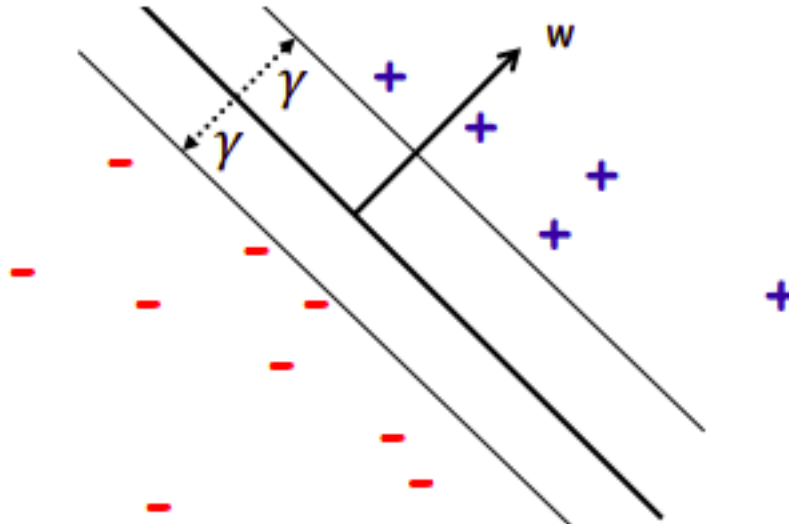
- Classification Problems
- **Classification Algorithms**
  - K-Nearest Neighbors
  - Naïve Bayes Classification
  - Decision Tree
  - **Support Vector Machines**
- Metrics to measure Classification Performance

- If large margin, # mistakes Peceptron makes is small (independent on the dim of the ambient space)!
- Large margin can help prevent overfitting.
  - ♦ If large margin  $\gamma$  and if alg. produces a large margin classifier, then amount of data needed depends only on  $R/\gamma$  [Bartlett & Shawe-Taylor '99].
- Ideas: Directly search for a large margin classifier!!

## Support Vector Machines (SVMs)



- Definition: The margin of example  $x$  w.r.t. a linear sep.  $w$  is the distance from  $x$  to the plane  $w \cdot x = 0$ .
- Definition: The margin  $\gamma w$  of a set of examples  $S$  wrt a linear separator  $w$  is the smallest margin over points  $x \in S$ .
- Definition: The margin  $\gamma$  of a set of examples  $S$  is the maximum  $\gamma w$  over all linear separators  $w$ .



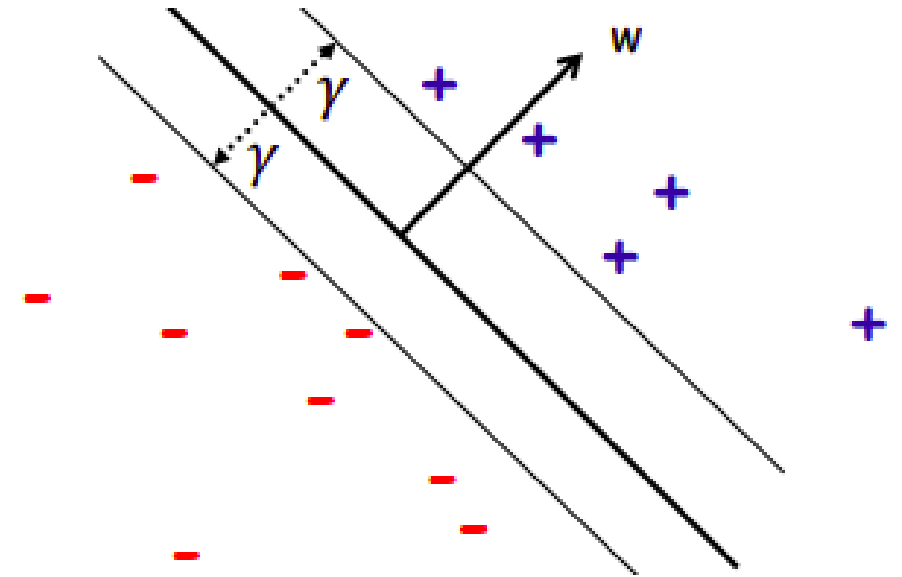
- Directly optimize for the maximum margin separator: SVMs
- First, assume we know a lower bound on the margin  $\gamma$

**Input:**  $\gamma, S=\{(x_1, y_1), \dots, (x_m, y_m)\};$

**Find:** some  $w$  where:

- $\|w\|^2 = 1$
- For all  $i, y_i w \cdot x_i \geq \gamma$

**Output:**  $w$ , a separator of margin  $\gamma$  over  $S$



The case where the data is truly linearly separable by margin  $\gamma$



→ Directly optimize for the maximum margin separator: SVMs

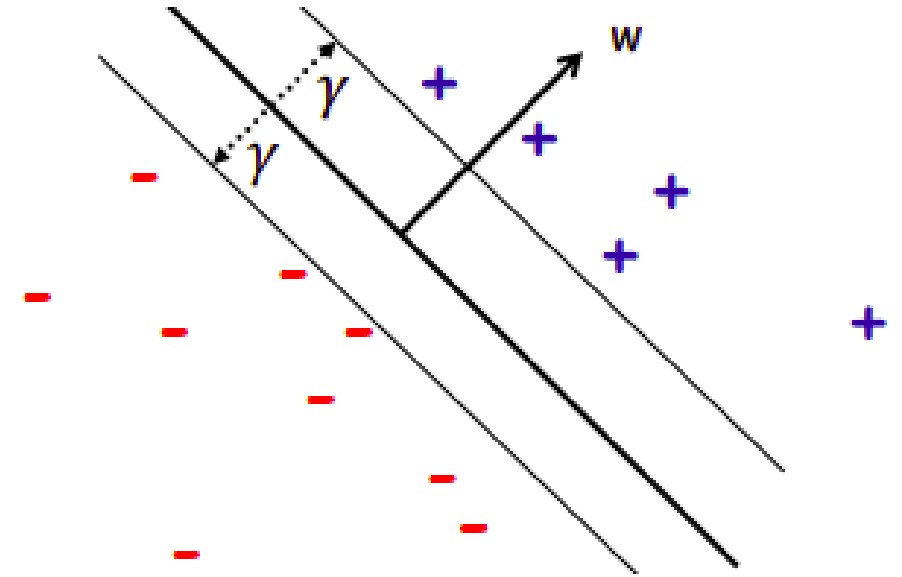
E.g., search for the best possible  $\gamma$

**Input:**  $\gamma$ ,  $S=\{(x_1, y_1), \dots, (x_m, y_m)\}$ ;

**Find:** some  $w$  where:

- $\|w\|^2 = 1$
- For all  $i$ ,  $y_i w \cdot x_i \geq \gamma$

**Output:**  $w$ , a separator of margin  $\gamma$  over  $S$



The case where the data is truly linearly separable by margin  $\gamma$

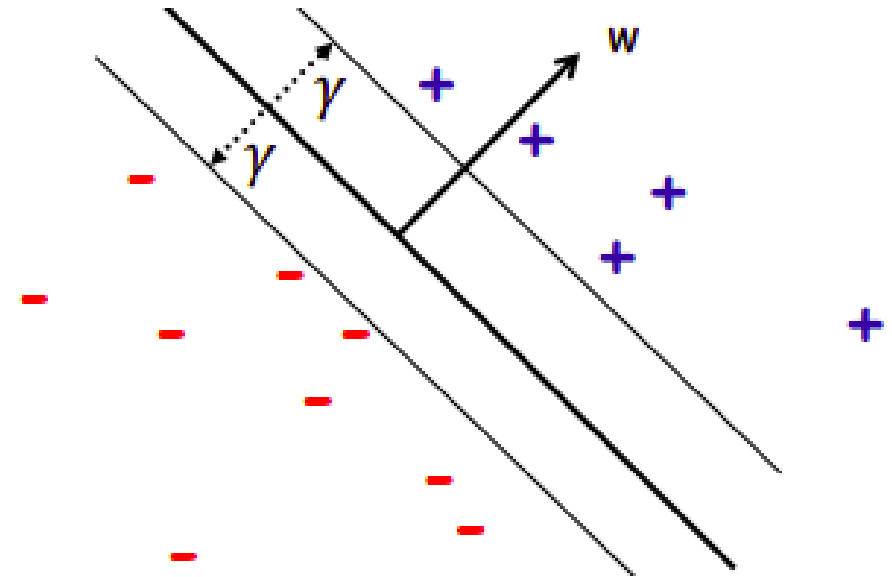
→ Directly optimize for the maximum margin separator: SVMs

Input:  $\gamma$ ,  $S=\{(x_1, y_1), \dots, (x_m, y_m)\}$ ;

Maximize  $\gamma$  under the constraint:

- $\|w\|^2 = 1$
- For all  $i$ ,  $y_i w \cdot x_i \geq \gamma$

objective function



Famous example of constrained optimization: linear programming, where objective fn is linear, constraints are linear (in)equalities

# Support Vector Machines: Geometric Margin

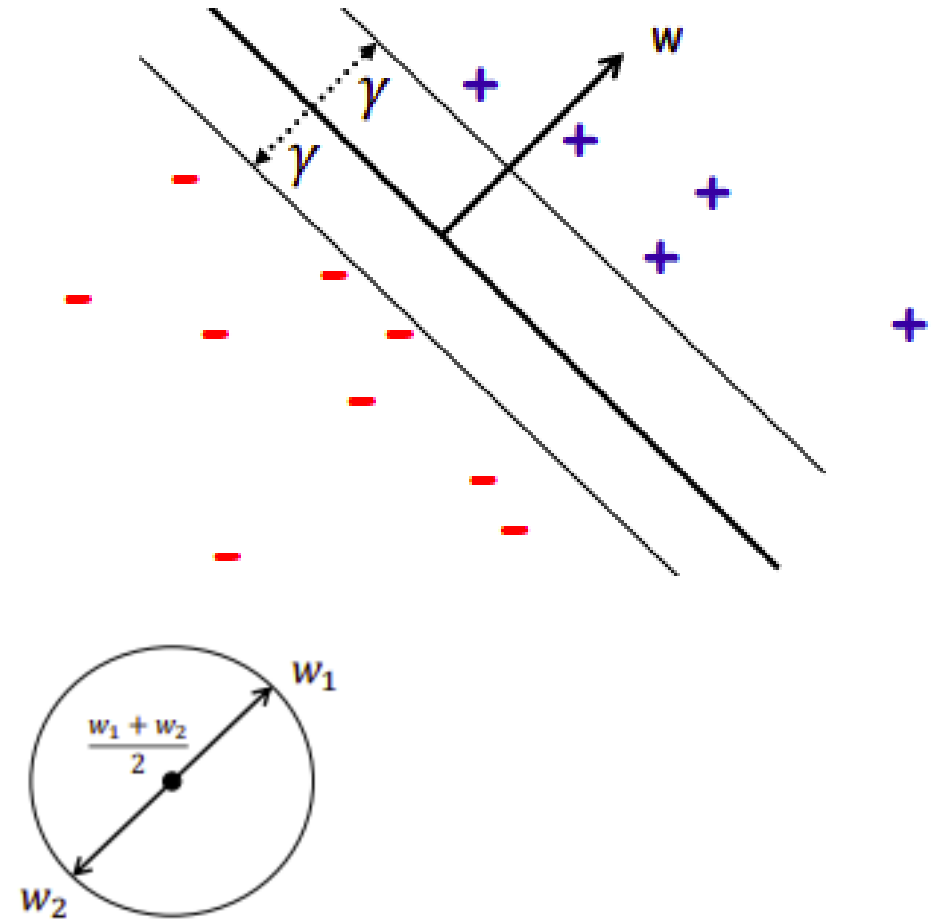
→ Directly optimize for the maximum margin separator: SVMs

**Input:**  $\gamma$ ,  $S=\{(x_1, y_1), \dots, (x_m, y_m)\}$ ;

**Maximize  $\gamma$**  under the constraint:

- $\|w\|^2 = 1$
- For all  $i$ ,  $y_i w \cdot x_i \geq \gamma$

This constraint is non-linear.  
In fact, it's even non-convex



- Classification Problems
- Classification Algorithms
  - K-Nearest Neighbors
  - Naïve Bayes Classification
  - Decision Tree
  - Support Vector Machines
- **Metrics to measure Classification Performance**

- Confusion Matrix
- Accuracy
- Precision
- Recall
- F1 score

- describe the performance of a classification model on a set of test data for which the true values (actual classes) are known.

## Binary-class Classification

		Actual Classes	
		POSITIVE	NEGATIVE
Predicted Classes	POSITIVE	TRUE POSITIVE (TP)	FALSE POSITIVE (FP)
	NEGATIVE	FALSE NEGATIVE (FN)	TRUE NEGATIVE (TN)

A confusion matrix for a binary classifier in disease diagnosis in which "yes" would mean they have the disease, and "no" would mean they don't have the disease.

n=165	Predicted: NO	Predicted: YES
	Actual: NO	Actual: YES
	50	10
	5	100

## Multi-class Classification

		Actual Classes			
		a	b	c	d
Predicted Classes	a	50	3	0	0
	b	26	8	0	1
	c	20	2	4	0
	d	12	0	0	1



## Multi-label Classification

True labels

SERVICE	FOOD	ANECDOTES	PRICE	AMBIENCE
1	0	0	0	0
0	1	1	0	0
0	1	0	0	0
1	0	0	0	0
0	0	1	0	0

Predicted labels

SERVICE	FOOD	ANECDOTES	PRICE	AMBIENCE
0	1	0	0	0
1	1	0	0	0
0	0	0	1	0
1	0	0	0	0
1	0	0	0	0

- evaluates the performance of the classification models, when they make predictions on test data.
- measures how good generated classification model is.
- calculate the different metrics for evaluating the model, such as accuracy, precision, etc.

- determined as the number of correct predictions to the total number of predictions.

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

$$\text{Accuracy} = (100 + 50) / 165 = 0.91$$

n=165	Predicted: NO	Predicted: YES
Actual: NO	50	10
Actual: YES	5	100

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{All Samples}}$$

## Multi-class Classification

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

		Actual Classes			
		a	b	c	d
Predicted Classes	a	50	3	0	0
	b	26	8	0	1
	c	20	2	4	0
	d	12	0	0	1

$$\text{Accuracy} = \frac{50 + 8 + 4 + 1}{127} = \frac{63}{127} = 49,6\%$$

# Accuracy in Multilabel Classification

	True labels					Predicted labels				
TEXT	SERVICE	FOOD	ANECDOTES	PRICE	AMBIENCE	SERVICE	FOOD	ANECDOTES	PRICE	AMBIENCE
but the staff was so horrible to us	1	0	0	0	0	0	1	0	0	0
to be completely fair the only redeeming facto...	0	1	1	0	0	1	1	0	0	0
the food is uniformly exceptional with a very ...	0	1	0	0	0	0	0	0	1	0
where gabriela personally greets you and recomm...	1	0	0	0	0	1	0	0	0	0
for those that go once and dont enjoy it all i...	0	0	1	0	0	1	0	0	0	0

Total number of predictions (TNP) = 4

Total number of correct predictions (TNCP) = 1

$$\text{Accuracy} = \text{TNCP} / \text{TNP} = 1/4 = 0.25$$

- use the Accuracy metric when the target variable classes in data are approximately balanced (unbiased)
- in the case of imbalanced data (biased), where one class is much larger than another, the accuracy can be highly misleading.

There is a model for a disease prediction in which, out of 100 persons, only 5 persons have a disease, and 95 people don't have one. If the classifier predicts **everyone with no disease**, the ***Accuracy*** value is 95% (TP= 0; TN= 95; FP=0; FN=5; All examples=100)

=> **not correct**



60% of classes in a fruit image dataset are of Apple, 40% are Mango (approximately balanced data)  
if the classifier predict whether the image is of Apple or Mango, a prediction with 97% of accuracy is credible.

- **Precision** measures how good the model is at correctly identifying the positive class

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

## Actual Labels

## Predicted Labels

	urgent	normal	spam	
urgent	8	10	1	$\text{precision}_u = \frac{8}{8+10+1}$
normal	5	60	50	$\text{precision}_n = \frac{60}{5+60+50}$
spam	3	30	200	$\text{precision}_s = \frac{200}{3+30+200}$

- **Recall** measures how good the model is at correctly predicting all the positive observations in the dataset

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

		Actual Labels		
		urgent	normal	spam
Predicted Labels	urgent	8	10	1
	normal	5	60	50
	spam	3	30	200
		$\text{recall}_u = \frac{8}{8+5+3}$	$\text{recall}_n = \frac{60}{10+60+30}$	$\text{recall}_s = \frac{200}{1+50+200}$

- **F1 Score (F-measure)** is the **harmonic mean** of precision and recall.

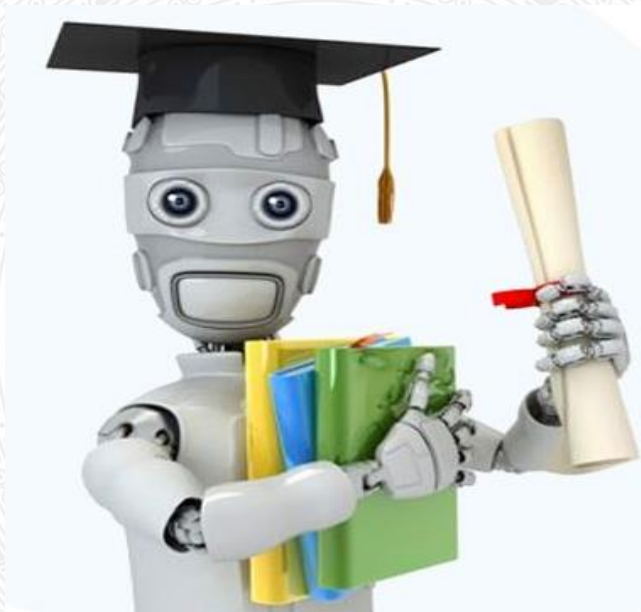
$$F1\ Score = 2 \times \frac{recall \times precision}{recall + precision}$$

- used to compare the performance of two classifiers when determines which one produces better results.

- **Classification Problems**
- **Classification Algorithms**
  - **K-Nearest Neighbors**
  - **Naïve Bayes Classification**
  - **Decision Tree**
  - **Support Vector Machines**
- **Metrics to measure Classification Performance**







**Enjoy the Course...!**