Text Categorization

Arjun Mukherjee†

Course webpage:
http://www.cs.uh.edu/~arjun/courses/nlp

† Contains contents from [Liu, 2008] and lecture slides of B.Liu, and various other sources. Referenced in place.

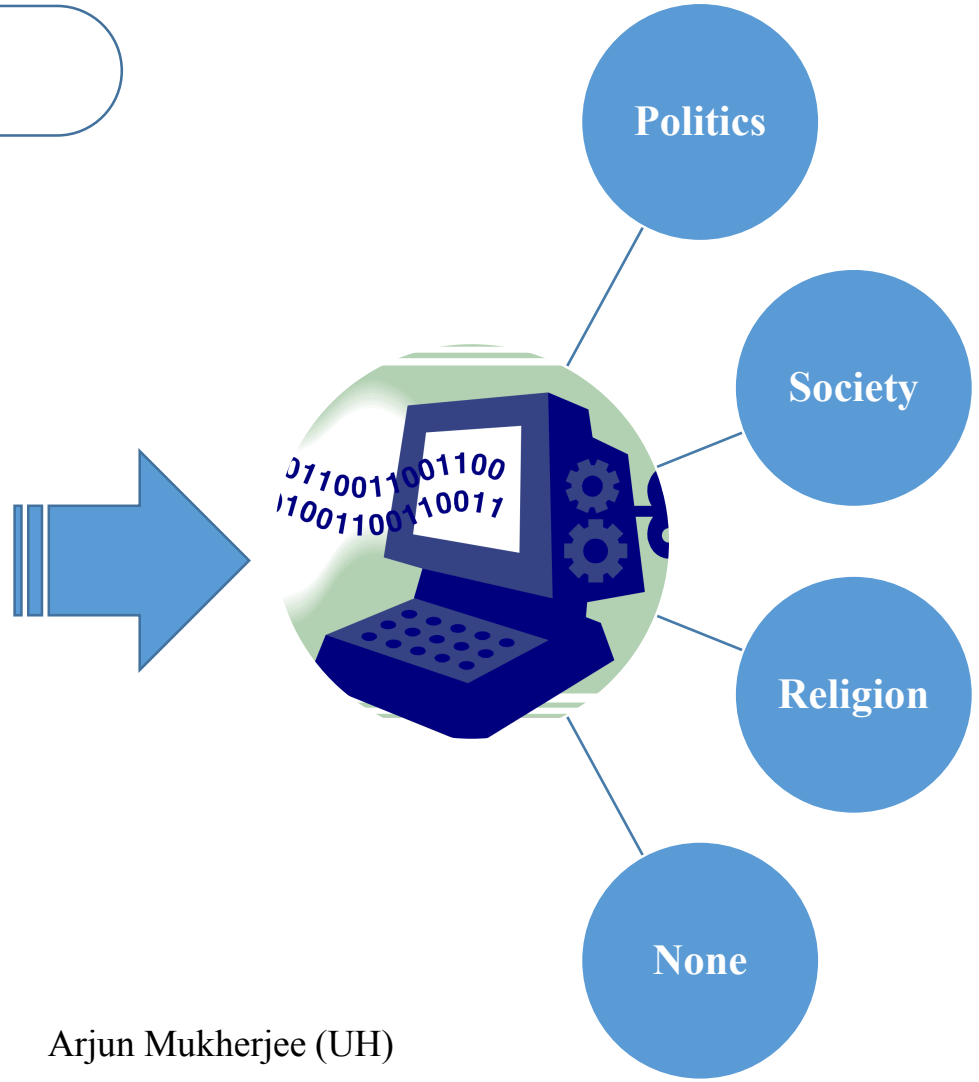
Text Classification

- **Problem:** Suppose you are given a random news article. What algorithm would you use to make a computer categorize it into (a) Politics (b) Religion (c) Society or (d) None?

The New York Times

WEDNESDAY, JULY 11, 2007

A plurality of voters think Barack Obama is the worst president since World War II, a new poll says. Obama's predecessor, former President George W. Bush, came in at second-worst with 28 percent, and Richard Nixon was in third place with 13 percent of the vote. After Jimmy Carter, who 8 percent of voters said was the worst president in the time period, no other president received more than 3 percent.



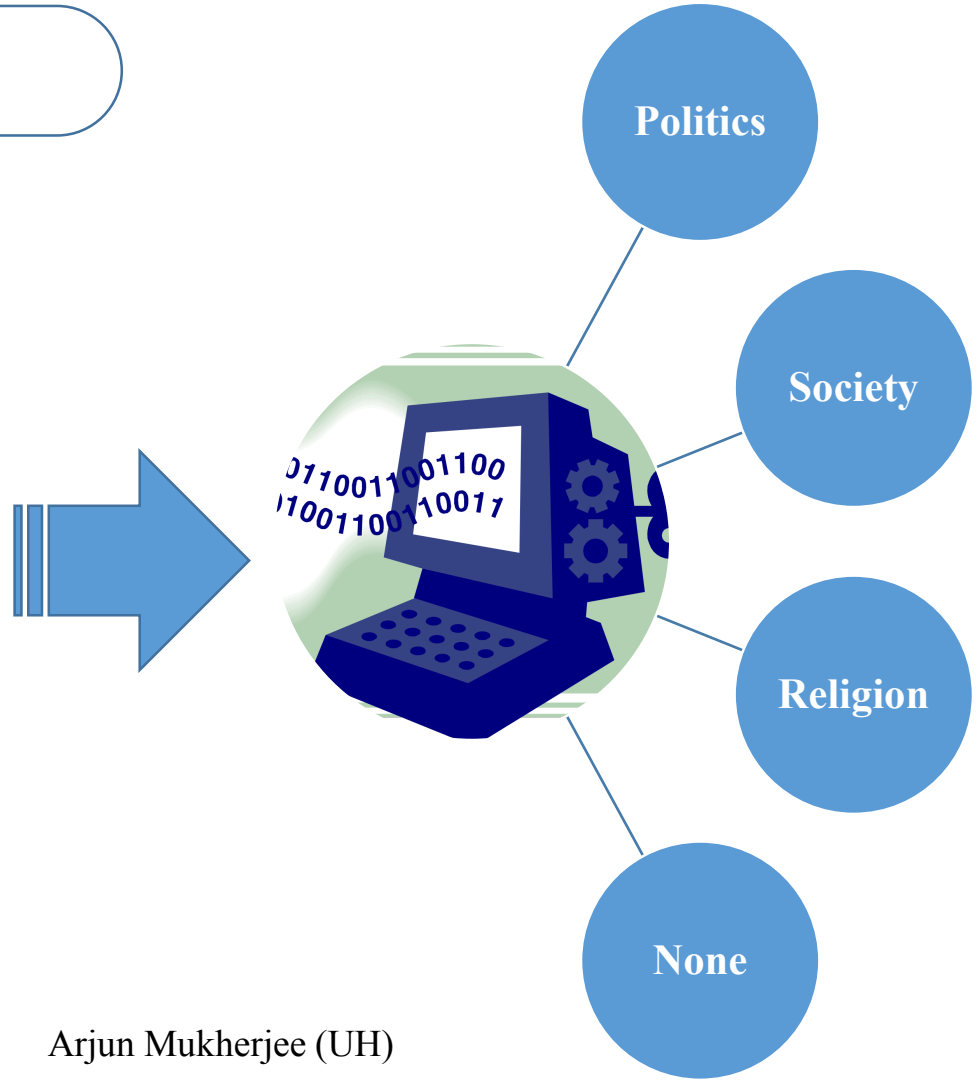
Text Classification

- **Hint:** How would humans categorize?

The New York Times

WEDNESDAY, JULY 11, 2007

A plurality of voters think Barack Obama is the worst president since World War II, a new poll says. Obama's predecessor, former President George W. Bush, came in at second-worst with 28 percent, and Richard Nixon was in third place with 13 percent of the vote. After Jimmy Carter, who 8 percent of voters said was the worst president in the time period, no other president received more than 3 percent.



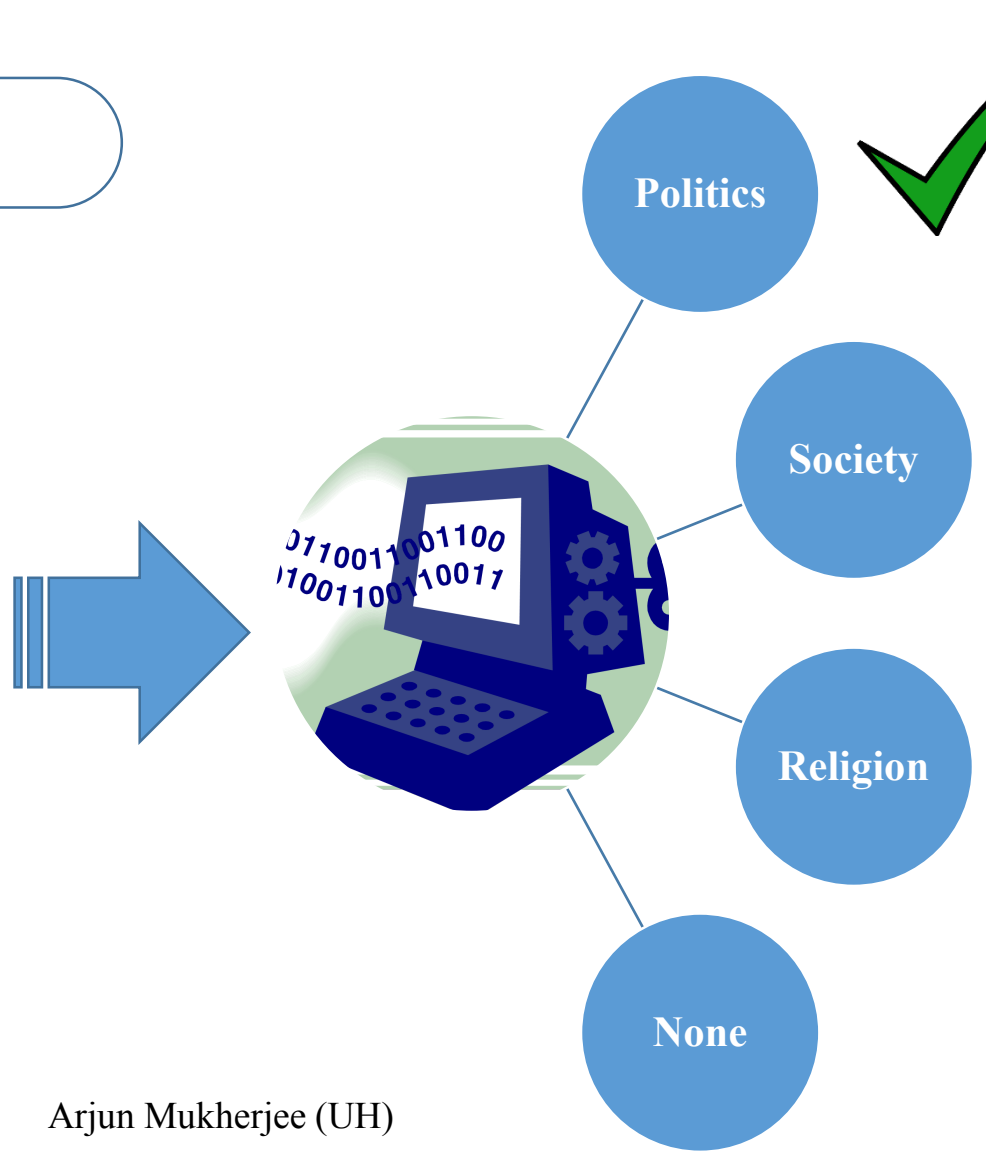
Text Classification

- **Hint:** How would humans categorize?

The New York Times

WEDNESDAY, JULY 11, 2007

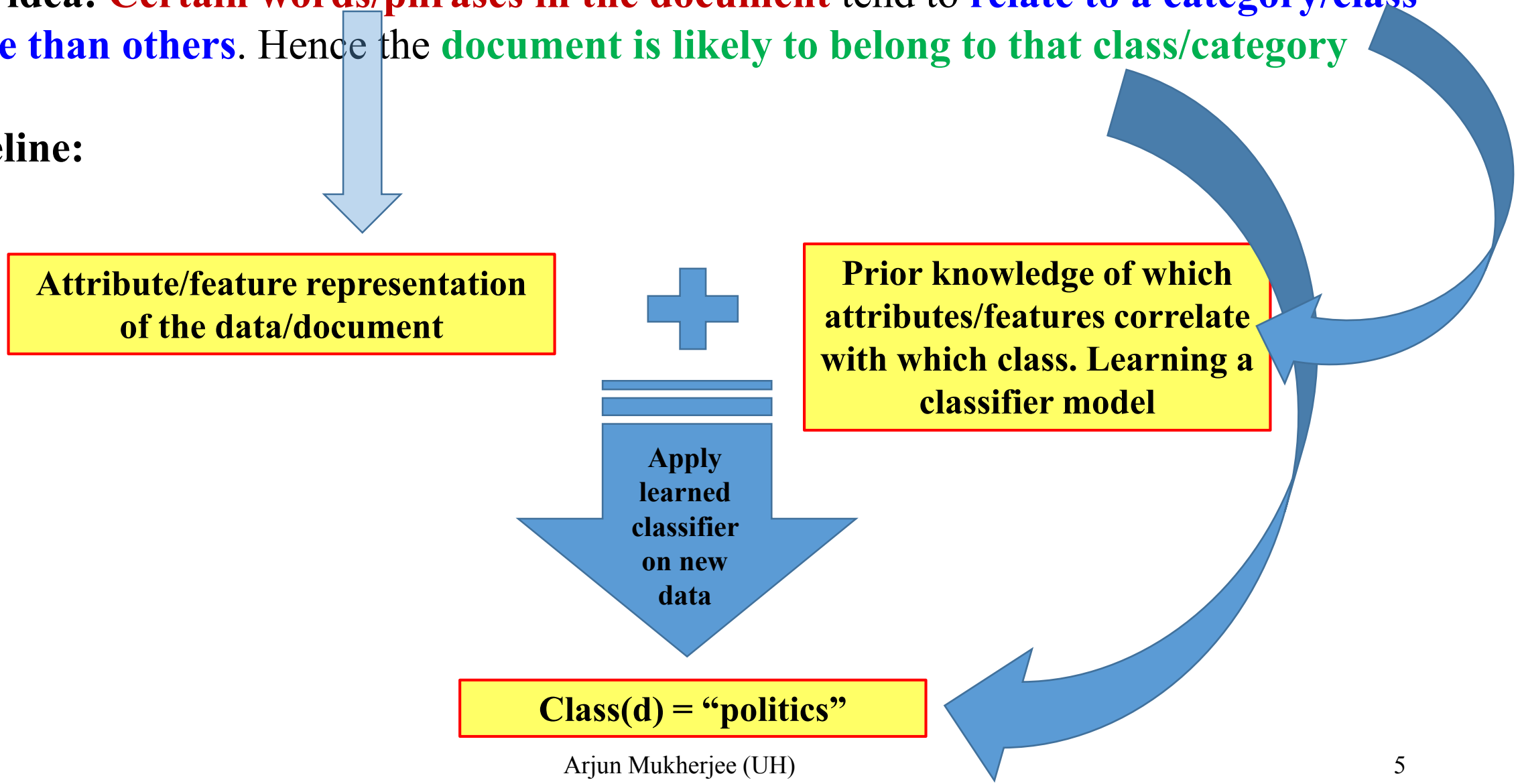
A plurality of **voters** think **Barack Obama** is the worst **president** since World War II, a new **poll** says. Obama's predecessor, former **President George W. Bush**, came in at second-worst with 28 percent, and Richard Nixon was in third place with 13 percent of the **vote**. After Jimmy Carter, who 8 percent of **voters** said was the worst **president** in the time period, no other **president** received more than 3 percent.



Text Classification

- **Key idea: Certain words/phrases in the document** tend to **relate to a category/class more than others**. Hence the **document is likely to belong to that class/category**

- **Pipeline:**



Classification

- Loan application data [Example due to Liu, 2008]

Features of the data

An instance/sample/ data-point

Class/Category: Here Binary (Y/N)

Table 3.1. A loan application data set

ID	Age	Has_job	Own_house	Credit_rating	Class
1	young	false	false	fair	No
2	young	false	false	good	No
3	young	true	false	good	Yes
4	young	true	true	fair	Yes
5	young	false	false	fair	No
6	middle	false	false	fair	No
7	middle	false	false	good	No
8	middle	true	true	good	Yes
9	middle	false	true	excellent	Yes
10	middle	false	true	excellent	Yes
11	old	false	true	excellent	Yes
12	old	false	true	good	Yes
13	old	true	false	good	Yes
14	old	true	false	excellent	Yes
15	old	false	false	fair	No

- **Q: What is the learning goal?**

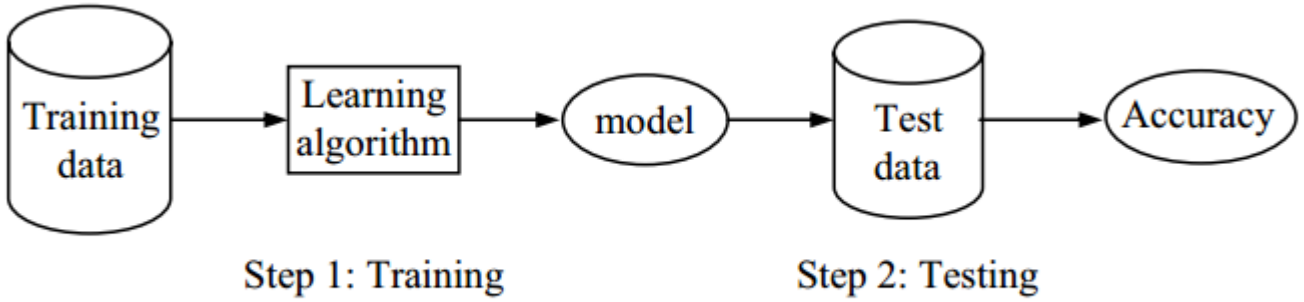
Classification

- Upon learning, the goal is to classify new unseen data (a test case).
- **Q: What is the class of this instance? Whether to give loan or not?**

Age	Has_Job	Own_house	Credit-Rating	Class
young	false	false	good	?

Classification

- Pipeline of Learning process:



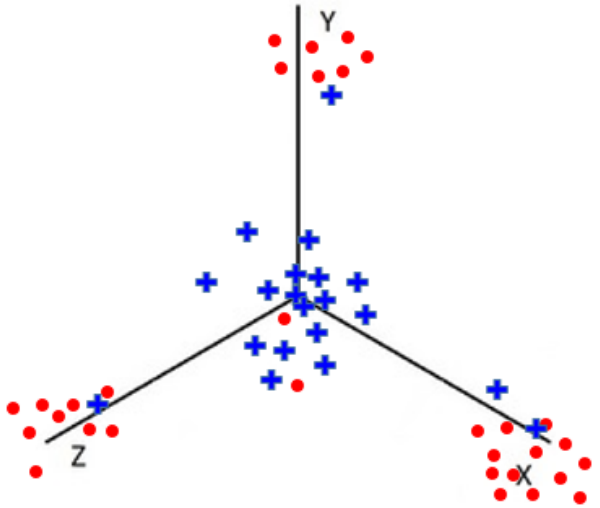
$$Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}},$$

Fig. 3.1. The basic learning process: training and testing

- Training data: Data/instances (features class labels) which are seen during model building.
- Test data: Data/instances whose labels are unknown. Goal is to use the learned model (i.e., feed the features of the test data t the model) and classify/predict the labels of the test data
- Training and Testing datasets are **disjoint**.
- Can often use part of the seen/training data as held-out/development set to tune additional model parameters

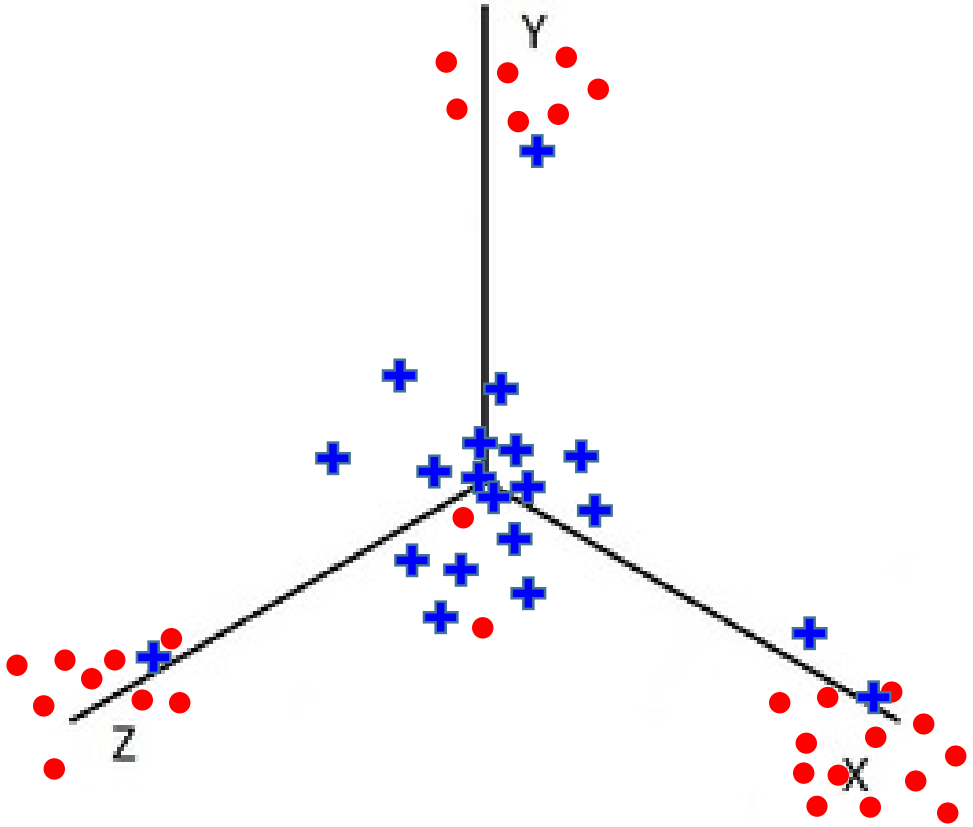
Classification

- **Q: Why/How does it work?**
- **Fundamental assumption of learning:** Distribution of training and test data are the same.
- What do we mean by distribution here? Distribution of the feature space.
- Consider our data has 3 continuous/real features and 2 classes (red/black)

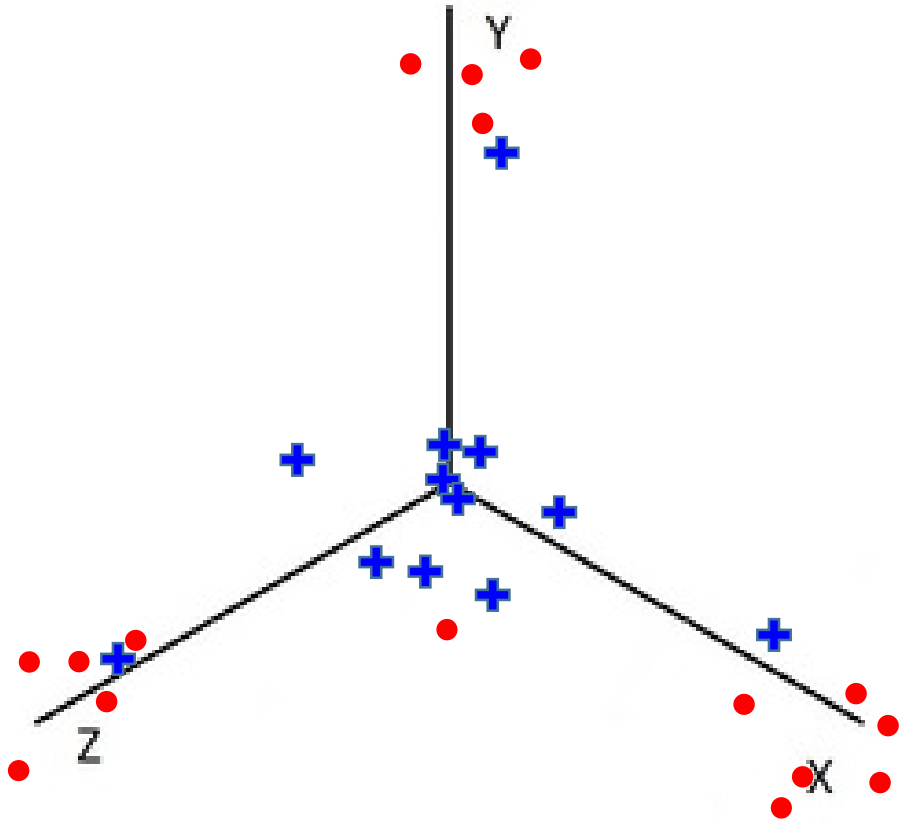


Classification

- Q: Are the training and Test data distributions similar? Why?
- Q:What does the classification model learn? How does that help in prediction?



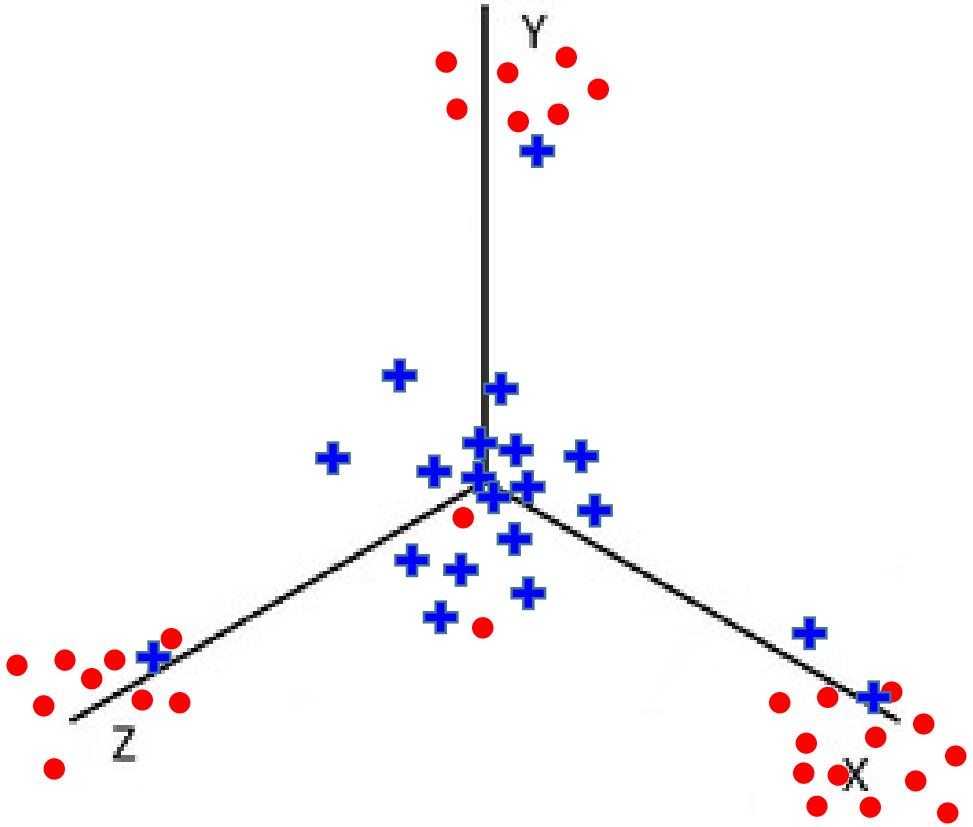
Feature space of Training Data



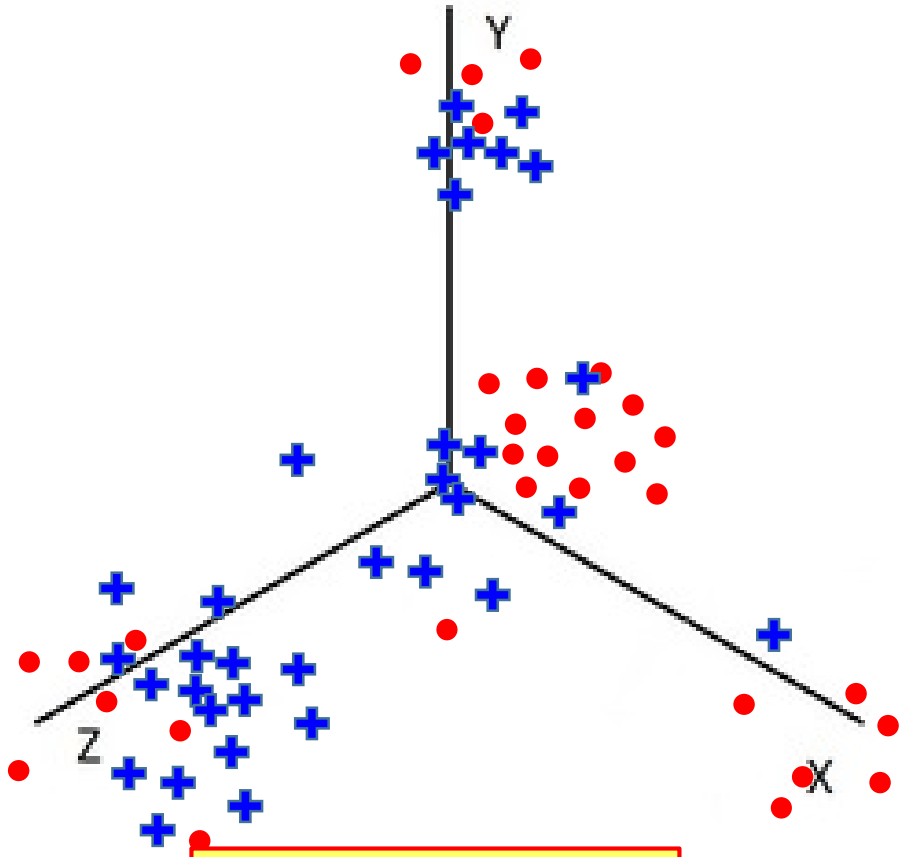
Feature space of Test Data

Classification

- Q: Are the Training and Test data distributions similar? Why or why not?
- Q:What does the classification model learn? Would it do a good job in prediction?



Feature space of Training Data



Feature space of Test Data

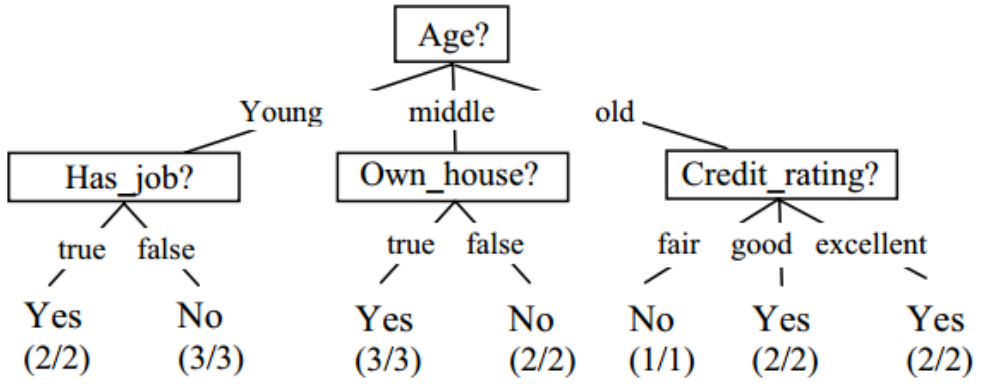
Decision Tree

- A tree based classification model.
- Very efficient and offers competitive classification accuracy
- A decision tree (with decision and leaf) nodes for the loan dataset

Table 3.1. A loan application data set

ID	Age	Has_job	Own_house	Credit_rating	Class
1	young	false	false	fair	**No**
2	young	false	false	good	**No**
3	young	true	false	good	**Yes**
4	young	true	true	fair	**Yes**
5	young	false	false	fair	**No**
6	middle	false	false	fair	**No**
7	middle	false	false	good	**No**
8	middle	true	true	good	**Yes**
9	middle	false	true	excellent	**Yes**
10	middle	false	true	excellent	**Yes**
11	old	false	true	excellent	**Yes**
12	old	false	true	good	**Yes**
13	old	true	false	good	**Yes**
14	old	true	false	excellent	**Yes**
15	old	false	false	fair	**No**

Example 2: Figure 3.2 shows a possible decision tree learnt from the data in Table 3.1. The tree has two types of nodes, **decision nodes** (which are internal nodes) and **leaf nodes**. A decision node specifies some test (i.e., asks a question) on a single attribute. A leaf node indicates a class.

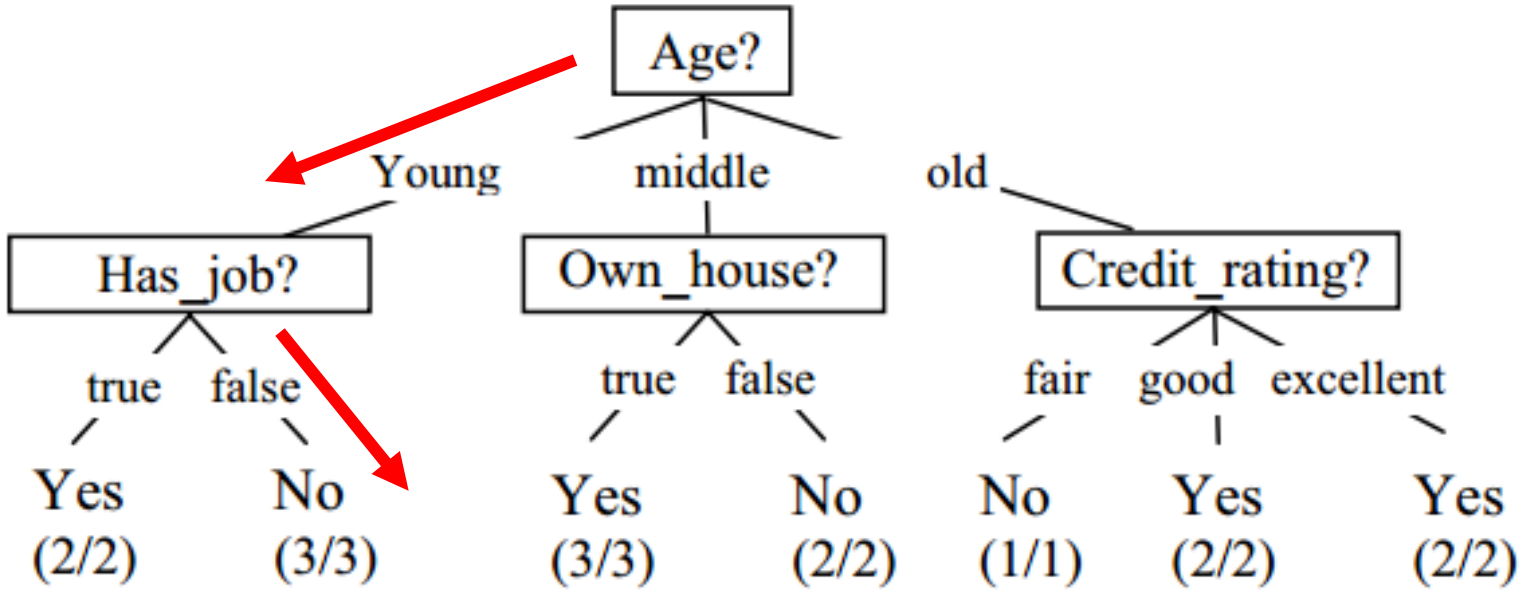


same meaning. Each leaf node gives a class value (Yes or No). (x/y) below each class means that x out of y training examples that reach this leaf node have the class of the leaf. For instance, the class of the left most leaf node is Yes. Two training examples (examples 3 and 4 in Table 3.1) reach here and both of them are of class Yes. ■

Decision Tree

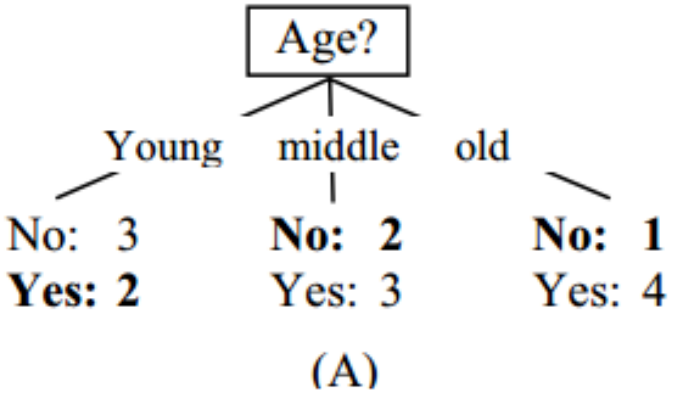
- **Q:How do we classify a new test instance using this tree?**

Age	Has_Job	Own_house	Credit-Rating	Class
young	false	false	good	? No

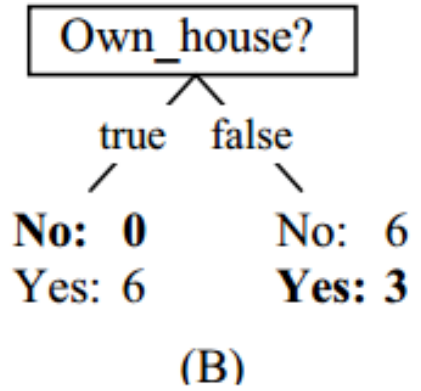


Decision Tree

- **Q:Which out of two possible roots are better? Why?**



(A)



(B)

- **Fig (B) is better. Why?**
- **A: as it makes fewer mistakes using majority classification.**
- **Q:What is majority classification?**
- **A: Assigning majority class label seen in training for every test instance having that attribute.**

Decision Tree: Entropy

- Entropy: An information theoretic measure of impurity or disorder

$$entropy(D) = -\sum_{j=1}^{|C|} \Pr(c_j) \log_2 \Pr(c_j)$$

$$\sum_{j=1}^{|C|} \Pr(c_j) = 1,$$

- $\Pr(c_j)$ is the probability of class c_j in data set D

Decision Trees: Entropy

- Q: How does entropy relate to class distribution and (im)purity in the data?

Example 6: Assume we have a data set D with only two classes, positive and negative. Let us see the entropy values for three different compositions of positive and negative examples:

1. The data set D has 50% positive examples (Pr(*positive*) = 0.5) and 50% negative examples (Pr(*negative*) = 0.5).

$$entropy(D) = -0.5 \times \log_2 0.5 - 0.5 \times \log_2 0.5 = 1.$$

2. The data set D has 20% positive examples (Pr(*positive*) = 0.2) and 80% negative examples (Pr(*negative*) = 0.8).

$$entropy(D) = -0.2 \times \log_2 0.2 - 0.8 \times \log_2 0.8 = 0.722.$$

3. The data set D has 100% positive examples (Pr(*positive*) = 1) and no negative examples, (Pr(*negative*) = 0).

$$entropy(D) = -1 \times \log_2 1 - 0 \times \log_2 0 = 0.$$

We can see a trend: When the data becomes purer and purer, the entropy value becomes smaller and smaller. In fact, it can be shown that for this binary case (two classes), when Pr(*positive*) = 0.5 and Pr(*negative*) = 0.5 the entropy has the maximum value, i.e., 1 bit. When all the data in D belong to one class the entropy has the minimum value, 0 bit. ■

- As data get purer, entropy lowers.
- Key idea employed in decision tree.

Decision Trees: Information Gain

- If we make attribute A_i, with v values, the root of the current tree, this will partition D into v subsets D_1, D_2 …, D_v . The expected entropy if A_i is used as the current root:

$$entropy_{A_i}(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times entropy(D_j)$$

- Information gained by selecting attribute A_i to branch or to partition the data is

$$gain(D, A_i) = entropy(D) - entropy_{A_i}(D)$$

- We choose the attribute with the highest gain to branch/split the current tree.

Decision Trees: Building using Info. Gain

- Using the definition of *gain(D, A_i)*, we have: $entropy_{A_i}(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times entropy(D_j)$

$$entropy(D) = -\frac{6}{15} \times \log_2 \frac{6}{15} - \frac{9}{15} \times \log_2 \frac{9}{15} = 0.971$$

$$gain(D, A_i) = entropy(D) - entropy_{A_i}(D)$$

$$entropy_{Own\_house}(D) = -\frac{6}{15} \times entropy(D_1) - \frac{9}{15} \times entropy(D_2)$$
$$= \frac{6}{15} \times 0 + \frac{9}{15} \times 0.918$$
$$= 0.551$$

$$entropy_{Age}(D) = -\frac{5}{15} \times entropy(D_1) - \frac{5}{15} \times entropy(D_2) - \frac{5}{15} \times entropy(D_3)$$
$$= \frac{5}{15} \times 0.971 + \frac{5}{15} \times 0.971 + \frac{5}{15} \times 0.722$$
$$= 0.888$$

Table 3.1. A loan application data set

ID	Age	Has_job	Own_house	Credit_rating	Class
1	young	false	false	fair	**No**
2	young	false	false	good	**No**
3	young	true	false	good	**Yes**
4	young	true	true	fair	**Yes**
5	young	false	false	fair	**No**
6	middle	false	false	fair	**No**
7	middle	false	false	good	**No**
8	middle	true	true	good	**Yes**
9	middle	false	true	excellent	**Yes**
10	middle	false	true	excellent	**Yes**
11	old	false	true	excellent	**Yes**
12	old	false	true	good	**Yes**
13	old	true	false	good	**Yes**
14	old	true	false	excellent	**Yes**
15	old	false	false	fair	**No**

Age	Yes	No	entropy(Di)
young	2	3	0.971
middle	3	2	0.971
old	4	1	0.722

gain(*D*, Age) = 0.971 – 0.888 = 0.083
gain(*D*, Own_house) = 0.971 – 0.551 = 0.420
gain(*D*, Has_job) = 0.971 – 0.647 = 0.324
gain(*D*, Credit_rating) = 0.971 – 0.608 = 0.363.

Decision Trees: Building using Info. Gain

- Q:Which node is the best root?
- A: Node having least gain. Own_house.

$$entropy(D) = -\frac{6}{15} \times \log_2 \frac{6}{15} - \frac{9}{15} \times \log_2 \frac{9}{15} = 0.971$$

$$entropy_{Own\_house}(D) = -\frac{6}{15} \times entropy(D_1) - \frac{9}{15} \times entropy(D_2)$$

$$= \frac{6}{15} \times 0 + \frac{9}{15} \times 0.918$$

$$= 0.551$$

$$entropy_{Age}(D) = -\frac{5}{15} \times entropy(D_1) - \frac{5}{15} \times entropy(D_2) - \frac{5}{15} \times entropy(D_3)$$

$$= \frac{5}{15} \times 0.971 + \frac{5}{15} \times 0.971 + \frac{5}{15} \times 0.722$$

$$= 0.888$$

Age	Yes	No	entropy(Di)
young	2	3	0.971
middle	3	2	0.971
old	4	1	0.722

$$entropy_{A_i}(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times entropy(D_j)$$

$$gain(D, A_i) = entropy(D) - entropy_{A_i}(D)$$

Table 3.1. A loan application data set

ID	Age	Has_job	Own_house	Credit_rating	Class
1	young	false	false	fair	**No**
2	young	false	false	good	**No**
3	young	true	false	good	**Yes**
4	young	true	true	fair	**Yes**
5	young	false	false	fair	**No**
6	middle	false	false	fair	**No**
7	middle	false	false	good	**No**
8	middle	true	true	good	**Yes**
9	middle	false	true	excellent	**Yes**
10	middle	false	true	excellent	**Yes**
11	old	false	true	excellent	**Yes**
12	old	false	true	good	**Yes**
13	old	true	false	good	**Yes**
14	old	true	false	excellent	**Yes**
15	old	false	false	fair	**No**

$gain(D, \text{Age}) = 0.971 - 0.888 = 0.083$
$gain(D, \text{Own\_house}) = 0.971 - 0.551 = 0.420$
$gain(D, \text{Has\_job}) = 0.971 - 0.647 = 0.324$
$gain(D, \text{Credit\_rating}) = 0.971 - 0.608 = 0.363.$

Decision Tree

- **Adding successive nodes:** After selecting the root node, we can again use IG on the partitioned data recursively.

- **Final decision tree:**

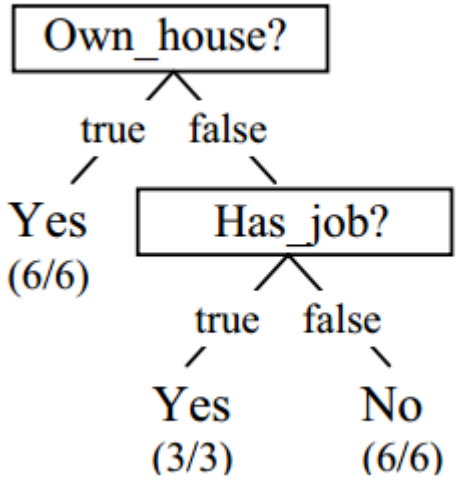


Table 3.1. A loan application data set

ID	Age	Has_job	Own_house	Credit_rating	Class
1	young	false	false	fair	**No**
2	young	false	false	good	**No**
3	young	true	false	good	**Yes**
4	young	true	true	fair	**Yes**
5	young	false	false	fair	**No**
6	middle	false	false	fair	**No**
7	middle	false	false	good	**No**
8	middle	true	true	good	**Yes**
9	middle	false	true	excellent	**Yes**
10	middle	false	true	excellent	**Yes**
11	old	false	true	excellent	**Yes**
12	old	false	true	good	**Yes**
13	old	true	false	good	**Yes**
14	old	true	false	excellent	**Yes**
15	old	false	false	fair	**No**

- **Stopping criteria:** Remaining training data in the (upon partitioning) form a single class or every attribute has been used along the path.

Naïve Bayes

Naive Bayes Classifiers

Connectionist and Statistical Language Processing

Frank Keller

keller@coli.uni-sb.de

Computerlinguistik

Universität des Saarlandes

Classifier Evaluation

- Consider the following partitioning of the data.

Training/Development Set for model building $[D_{Train}]$	GAP D_{Param}	Hold Out/Test set for evaluation $[D_{Test}]$

- Recall that $$Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}},$$
- We must always ensure $\mathbf{D_{Train} \cap D_{test} = \varphi}$ **Why?**
- Usually if the data is large, we can use randomly sample/shuffle the contents of our entire data ad use 80% for $\mathbf{[D_{Train}]}$ and 20%for $\mathbf{[D_{Test}]}$.
- **GAP $\approx$ $[D_{Parma}]$ is optional for tuning additional parameters of the model.**

Classifier Evaluation

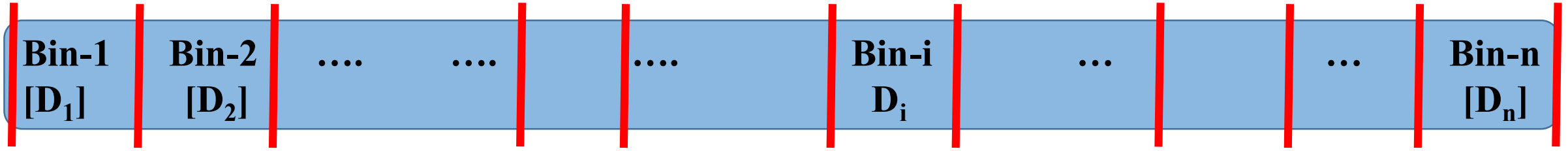
- Consider the following partitioning of the data.

Training/Development Set for model building $[D_{Train}]$	GAP D_{Param}	Hold Out/Test set for evaluation $[D_{Test}]$

- Recall that $$Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}},$$
- We must always ensure $\mathbf{D_{Train} \cap D_{test} = \varphi}$ **Why?**
- Usually if the data is large, we can use randomly sample/shuffle the contents of our entire data ad use 80% for $\mathbf{[D_{Train}]}$ and 20%for $\mathbf{[D_{Test}]}$.
- **GAP $\approx$ $[D_{Parma}]$ is optional for tuning additional parameters of the model.**

Cross Validation

- A single rain-test result is often not reliable, we need n-fold cross validation.



- CV is a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set.
- (1) Randomly shuffle/sample the dataset and partition into n disjoint subsets.
- (3) Use each subset/bin as the test set and combine the rest *n*-1 subsets/bins as the training set to learn a classifier.
- (4) The procedure is run n times, which give *n* accuracies.
- (5) The final estimated accuracy of learning is the average of the *n* accuracies.
- 10-fold and 5-fold cross-validations are commonly used.

Cross Validation

- Two important applications of cross validation.
- (1) Parameter estimation: Suppose our model/classifier has some parameters (e.g., thresholds, # of leaves, branches, depth) we want to estimate. We can try different values of our parameters (i.e., different decision trees for our data) and use cross validation accuracy to find the best parameters.

- (2) Comparing two classifiers/models.
- Obtain classification accuracy for each model for each fold
- Feed the values of two groups (Acc of C1 vs. Acc. Of C2) to t-test for estimating statistical significance.

$$t = \frac{\overline{X}_1 - \overline{X}_2}{s_{\overline{X}_1 - \overline{X}_2}} \quad s_{\overline{X}_1 - \overline{X}_2} = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}.$$

- Can use free online tools (e.g., Graphpad, or MS Excel)!

Fold	Acc. C1	Acc. C2
Fold 1	87%	80%
…	…	…
Fold n	90%	81%

X_1 X_2

Cross Validation

- **Leave-one-out cross-validation**: This method is used when the data set is very small.
- It is a special case of cross-validation
- Each fold of the cross validation has only a single test example and all the rest of the data is used in training.
- If the original data has *m* examples, this is *m*-fold cross-validation
- A method rarely used in text classification/mining problems.

Classification Measures

- Accuracy is only one measure (error = 1-accuracy).
- Accuracy is not suitable in some applications. **Why?**
- In text mining, we may only be interested in the documents of a particular topic, which are only a small portion of a big document collection.
- In classification involving skewed or highly imbalanced data, e.g., network intrusion and financial fraud detections, we are interested only in the minority class.
  - High accuracy does not mean any intrusion is detected.
  - E.g., 1% intrusion. Achieve 99% accuracy by doing nothing.
- The class of interest is commonly called the **positive class**, and the rest **negative classes**.

Precision and Recall

- Precision: Fraction of discovered positive/relevant instances which are actually positive/relevant

- Recall: Fraction of relevant/positive instances that could be retrieved/discovered by the model.

- Based on the following confusion matrix, we define:

	Classified positive	Classified negative
Actual positive	TP	FN
Actual negative	FP	TN

where

TP: the number of correct classifications of the positive examples (**true positive**)
FN: the number of incorrect classifications of positive examples (**false negative**)
FP: the number of incorrect classifications of negative examples (**false positive**)
TN: the number of correct classifications of negative examples (**true negative**)

Precision and Recall

- Precision *p* is the number of correctly classified positive examples divided by the total number of examples that are classified as positive.

- Recall *r* is the number of correctly classified positive examples divided by the total number of actual positive examples in the test set.

- Based on the following confusion matrix, we define: $p = \frac{TP}{TP + FP}$. $r = \frac{TP}{TP + FN}$.

	Classified positive	Classified negative
Actual positive	TP	FN
Actual negative	FP	TN

where

TP: the number of correct classifications of the positive examples (**true positive**)
FN: the number of incorrect classifications of positive examples (**false negative**)
FP: the number of incorrect classifications of negative examples (**false positive**)
TN: the number of correct classifications of negative examples (**true negative**)

Precision and Recall

- Precision p is the number of correctly classified positive examples divided by the total number of examples that are classified as positive.

- Recall r is the number of correctly classified positive examples divided by the total number of actual positive examples in the test set.

- Based on the following confusion matrix, we define:
  Note: precision and recall only measure classification performance on the positive class. **Q: Why?**

$$p = \frac{TP}{TP + FP}. \qquad r = \frac{TP}{TP + FN}.$$

	Classified positive	Classified negative
Actual positive	TP	FN
Actual negative	FP	TN

where

TP: the number of correct classifications of the positive examples (**true positive**)
FN: the number of incorrect classifications of positive examples (**false negative**)
FP: the number of incorrect classifications of negative examples (**false positive**)
TN: the number of correct classifications of negative examples (**true negative**)

Precision and Recall

- Does 100% precision mean the classifier is good? Example due to [Liu, 2008]

Example 11: A test data set has 100 positive examples and 1000 negative examples. After classification using a classifier, we have the following confusion matrix (Table 3.3),

Table 3.3. Confusion matrix of a classifier

	Classified positive	Classified negative
Actual positive	1	99
Actual negative	0	1000

This confusion matrix gives the precision $p = 100\%$ and the recall $r = 1\%$ because we only classified one positive example correctly and classified no negative examples wrongly. ■

- **Q: What is the accuracy here?**
- **Q:Can we just rely on precision or recall or accuracy? What are the extremes?**
- Need combined measure
- If data is balanced, equal proportion of positive/negative in test set, then accuracy is a good metric. **Q: Why?**

F_1-Score or F_1 measure

- It is hard to compare two classifiers using two measures. F_1 score combines precision and recall into one measure

$$F = \frac{2}{\frac{1}{p} + \frac{1}{r}} \qquad F = \frac{2pr}{p + r}$$

- The harmonic mean of two numbers tends to be closer to the smaller of the two.
- For F_1-value to be large, both p and r much be large. Example plots [Mukherjee et al., 13]

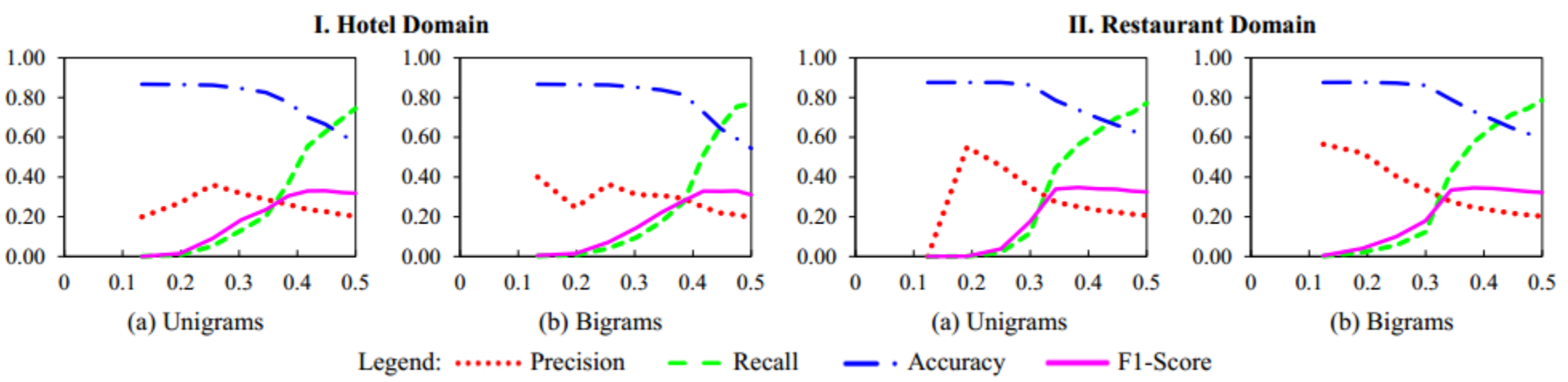


Figure 1: SVM 5-fold CV metrics by varying the proportion of fake reviews in the training data. For testing, natural distribution is used.

Support Vector Machine (SVM)

- A **linear classifier** for very **high dimensional** data. **Q: What does this mean?**
- Consider a set of training examples:

$$\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\},$$

where $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ir})$ is a r-dimensional **input vector** in a real-valued space $X \subseteq \Re^r$, y_i is its **class label** (output value) and $y_i \in \{1, -1\}$. 1 denotes the positive class and -1 denotes the negative class. Note that we use slightly different notations in this section. For instance, we use y instead of c to represent a class because y is commonly used to represent classes in the SVM literature. Similarly, each data instance is called an **input vector** and denoted by a bold face letter. In the following, we use bold face letters for all vectors.

- An SVM classifier finds a linear function of the form

$$f(\mathbf{x}) = \langle \mathbf{w} \cdot \mathbf{x} \rangle + b \qquad y_i = \begin{cases} 1 & \text{if } \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 0 \\ -1 & \text{if } \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b < 0 \end{cases}$$

$$f(x_1, x_2, \ldots, x_r) = w_1 x_1 + w_2 x_2 + \ldots + w_r x_r + b,$$

For text classification, $\boldsymbol{x_i}$ would be documents, represented by the feature vector of R dimensional words/vocabulary, y are two classes that we would want to classify, e.g., spam vs. non-spam emails.

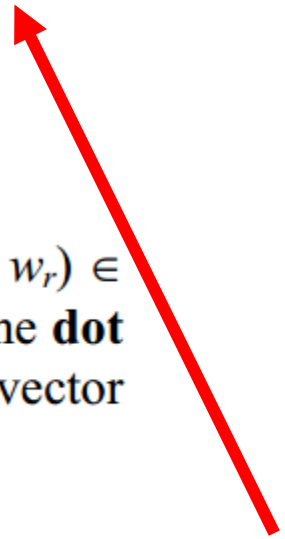
Support Vector Machine (SVM)

- A **linear classifier** for very **high dimensional** data.
- Given the training examples: $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$,
- An SVM classifier finds a linear function of the form

$$f(\mathbf{x}) = \langle \mathbf{w} \cdot \mathbf{x} \rangle + b \qquad y_i = \begin{cases} 1 & \text{if } \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 0 \\ -1 & \text{if } \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b < 0 \end{cases}$$

$$f(x_1, x_2, \ldots, x_r) = w_1 x_1 + w_2 x_2 + \ldots + w_r x_r + b,$$

Hence, $f(\mathbf{x})$ is a real-valued function $f: X \subseteq \mathcal{R}^r \rightarrow \mathcal{R}$. $\mathbf{w} = (w_1, w_2, \ldots, w_r) \in \mathcal{R}^r$ is called the **weight vector**. $b \in \mathcal{R}$ is called the **bias**. $\langle \mathbf{w} \cdot \mathbf{x} \rangle$ is the **dot product** of **w** and **x** (or **Euclidean inner product**). Without using vector notation, Equation (31) can be written as:



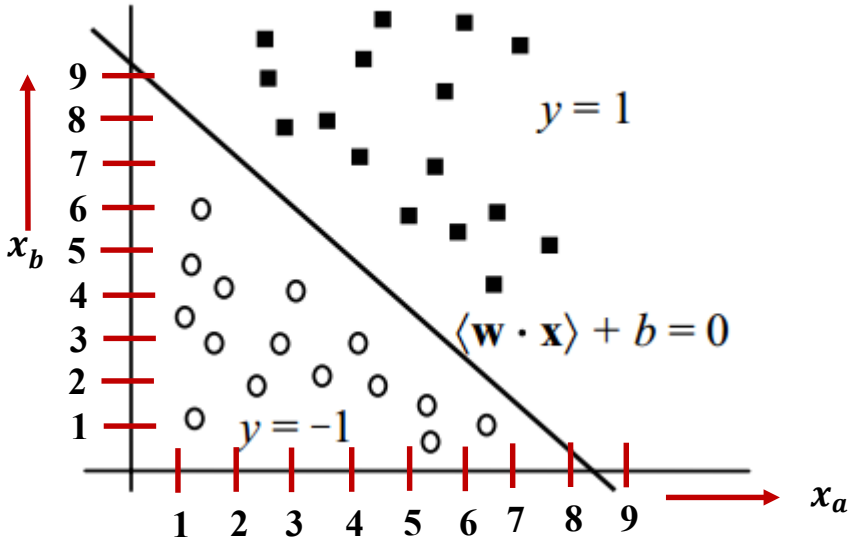
- **Q: What is the learning goal?** Estimate $\vec{w}$ or $\boldsymbol{w}$ and b from training examples.
- **Q: How to classify new unseen/test data?** Plug in the values and output class $y_t = f(\boldsymbol{x_t}) = < \boldsymbol{w}.\boldsymbol{x_t} > + b$

Support Vector Machine (SVM)

- SVM finds the hyperplane that separates the positive and negative training instances.

$$\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$$

- This is the decision boundary or the decision surface
- In 2-D (i.e, $\mathbb{R}^2$) the hyperplane is a line, in 3-D (i.e., $\mathbb{R}^3$), it is a plane
- Consider the following examples in 2-D.



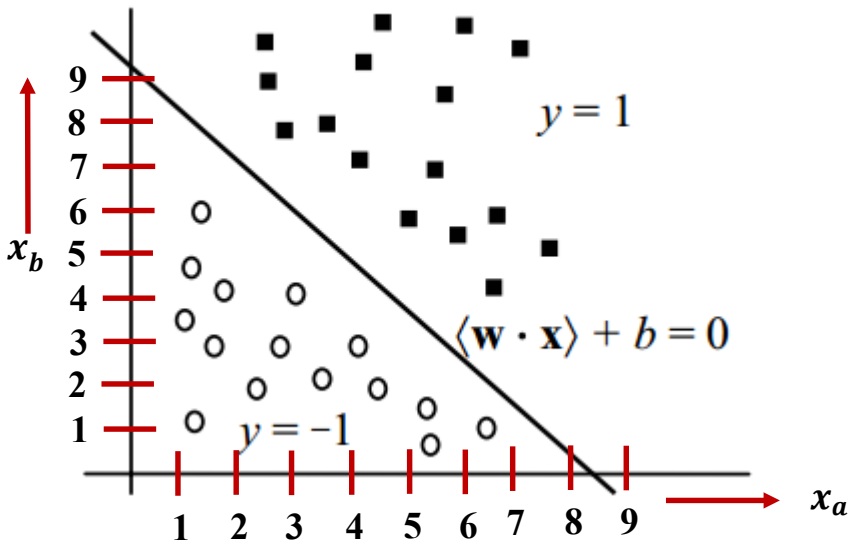
- **Q: What is the equation of the hyperplane?** $\frac{1}{8.5} x_a + \frac{1}{9.5} x_b - 1 = 0$
- **Q: What is w, b ? What is the slope of the line? How many features does this problem has?**

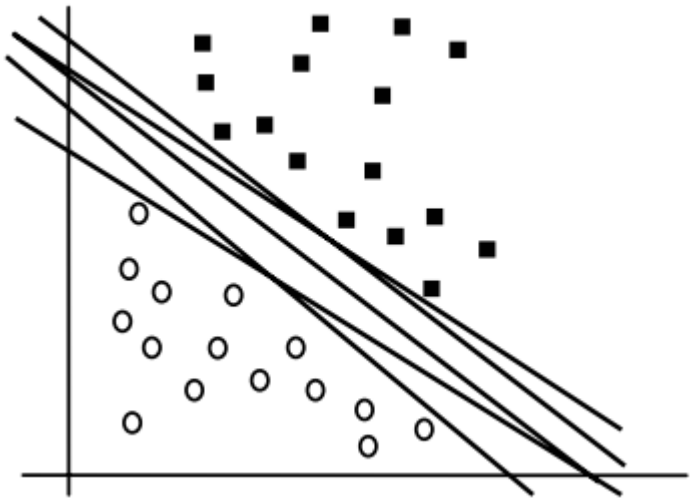
Support Vector Machine (SVM)

- SVM finds the hyperplane that separates the positive and negative training instances.

$$\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$$

- This is the decision boundary or the decision surface
- In 2-D (i.e, $\mathbb{R}^2$) the hyperplane is a line, in 3-D (i.e., $\mathbb{R}^3$), it is a plane
- Consider the following examples in 2-D.





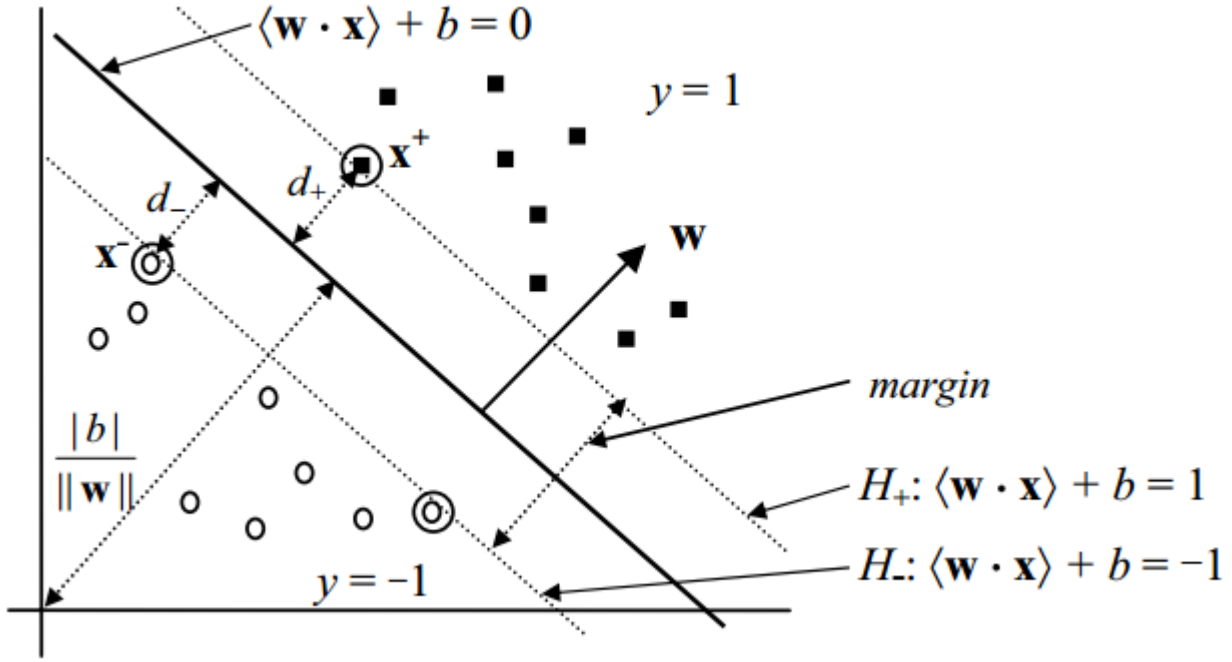
- **Q: There exists infinitely many planes which can separate +/- examples. Which one to choose?**

Support Vector Machine (SVM)

- SVM finds the **maximal margin hyperplane** that separates the positive and negative training instances.

$$\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$$

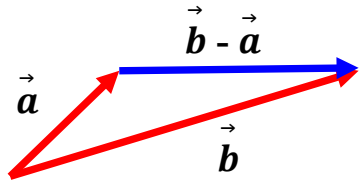
- Machine learning theory shows that this hyperplane minimizes the classification errors. (Intuitive isn't it?)
- Let ($\mathbf{x}^+$, 1) and ($\mathbf{x}^-$, -1) be the closest point to the (ideal) hyperplane



- **Q: How to express the margin as a function of w? Recall our goal is to find the plane (i.e., estimate w, b) that maximizes the separation margin**

Support Vector Machine (SVM)

- Linear Algebra recap.
- A vector, $\vec{x}$ with head ($\vec{b}$), tail ($\vec{a}$), is expressed as $\vec{x} = \vec{b} - \vec{a}$ (head vector – tail vector).



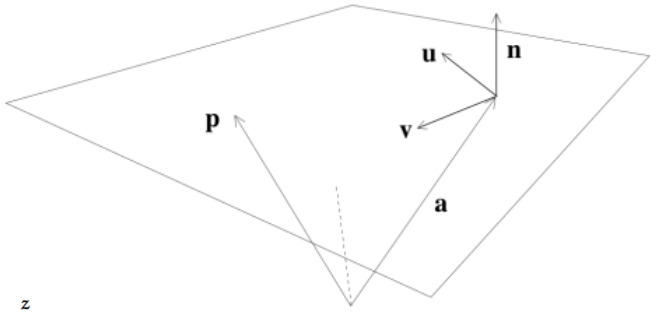
- A plane can be specified by its normal vector.

$$\mathbf{p} = \mathbf{a} + s\mathbf{u} + t\mathbf{v}, \qquad (s,t) \in \mathcal{R}.$$

Vector from origin to a point in the plane

Two non-parallel directions in the plane

- In our case, the hyperplane $\langle \mathbf{w}.\mathbf{x} \rangle + b = 0$ has $\mathbf{w}$ as its normal vector.



Alternatively, it can be specified as:

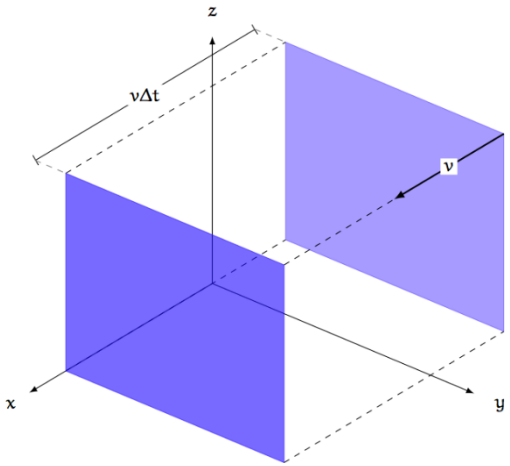
$$(\mathbf{p} - \mathbf{a}) \cdot \mathbf{n} = 0 \Leftrightarrow \mathbf{p} \cdot \mathbf{n} = \mathbf{a} \cdot \mathbf{n}$$

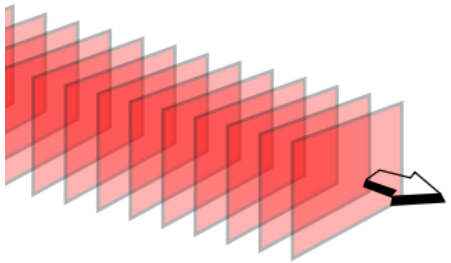
Normal vector
(we will call this w)

Only need to specify this dot product, a scalar (we will call this the offset, b)

- Unit vector in the direction of the normal vector:
- $\boldsymbol{u} = \frac{\vec{w}}{||w||}$ where $||w|| = \sqrt{(\sum w_i^2)}$

- **Scaling/scalar multiplication of plane:**
  A plane moves parallel to itself as long as the normal vector remain parallel to itself.





† Plane normal picture courtesy of D.Sontag's ML course

Support Vector Machine (SVM)

- For ease of learning, we set the scale by requiring that.

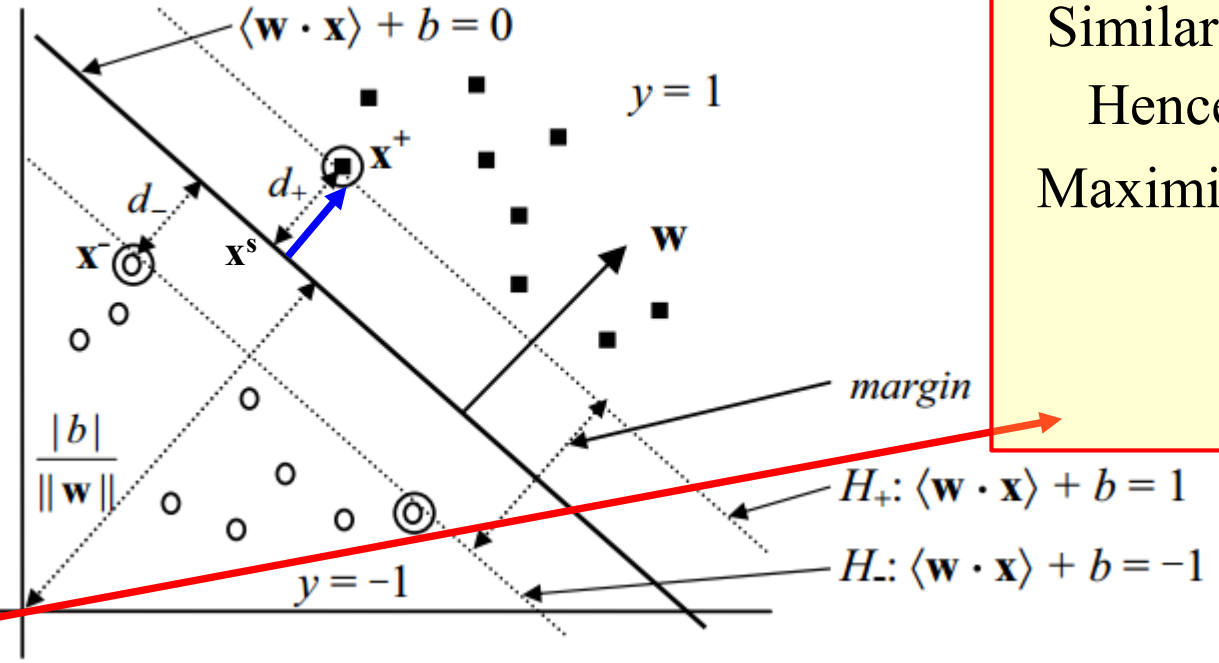
$$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 1 \quad \text{if } y_i = 1$$
$$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \leq -1 \quad \text{if } y_i = -1$$

- The corresponding parallel hyperplanes passing through $\mathbf{x}^+$ and $\mathbf{x}^-$

$$H_+: \quad \langle \mathbf{w} \cdot \mathbf{x}^+ \rangle + b = 1$$
$$H_-: \quad \langle \mathbf{w} \cdot \mathbf{x}^- \rangle + b = -1$$

- Consider the point $\mathbf{x}^s$ at d_+ distance on the hyperplane $\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$.

- The vector $\mathbf{x}^+ - \mathbf{x}^s = d_+\left(\frac{\vec{w}}{||w||}\right)$
- We also know that H+ and H pass through $\mathbf{x}^+$ and $\mathbf{x}^s$



$\mathbf{w}.\mathbf{x}^+ + b = 1$

\- $\mathbf{w}.\mathbf{x}^s + b = 0$

Subtract

$\mathbf{w}.(\mathbf{x}^+ - \mathbf{x}^s) = 1$

$\mathbf{w}.\left(d_+\left(\frac{\vec{w}}{||w||}\right)\right) = 1$

Plug in $\mathbf{x}^+ - \mathbf{x}^s$

$d_+ = \frac{1}{||w||}$

Thus, $d_+ = \frac{1}{||w||}$

Similarly, one can show, $d_- = \frac{1}{||w||}$

Hence margin = $d_+ + d_- = \frac{2}{||w||}$

Maximizing the margin is same as minimizing $||w||^2$

Support Vector Machine (SVM)

- **Linear SVM: separable case**
- Given a set of linearly separable training examples,

$$D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_r, y_r)\}$$

- Learning is to solve the following constrained minimization problem,

$$\text{Minimize}: \frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2}$$

$$\text{Subject to}: y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1, \quad i = 1, 2, ..., r$$

- Notice that $y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 1, \quad i = 1, 2, ..., r$ summarizes

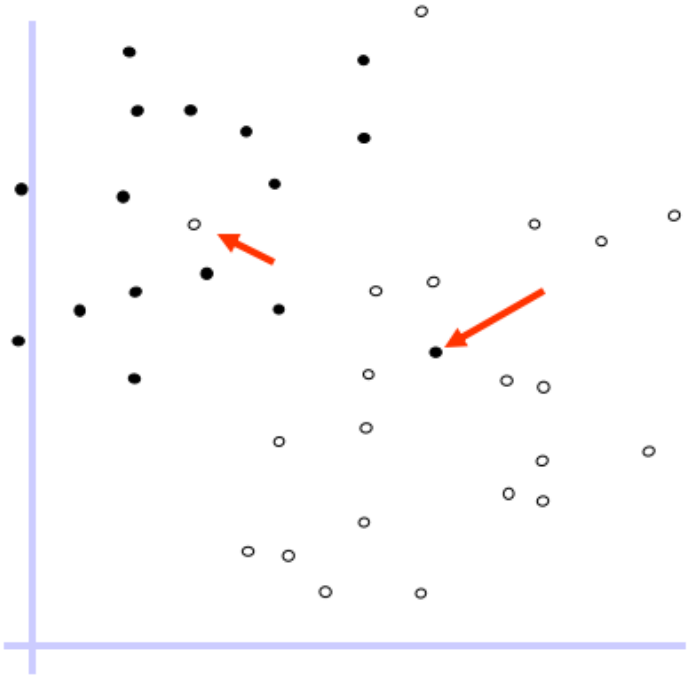
$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 1 \quad$ for $y_i = 1$

$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \leq -1 \quad$ for $y_i = -1$

❑Solving this uses Quadratic programming (which is beyond the scope of this course). We can treat this as a black box!

Support Vector Machine (SVM)

- What if the data is not linearly separable?



- Can we insist that $y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1, \quad i = 1, 2, ..., r$?

Support Vector Machine (SVM)

- Linear separable case is the ideal situation.
- Real-life data may have noise or errors.
  - Class label incorrect or randomness in the application domain.
- Recall in the separable case, the problem was

$$\text{Minimize}: \frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2}$$

$$\text{Subject to}: y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1, \quad i = 1, 2, ..., r$$

- With noisy data, the constraints may not be satisfied.
- It is possible to have no solution!
- Because no plane actually demarcates all positive and negative instances

Support Vector Machine (SVM)

- We need to relax the constraints
- Allow errors by relaxing the margin constraints
- Introducing **slack** variables, ξ_i (≥ 0) as follows:

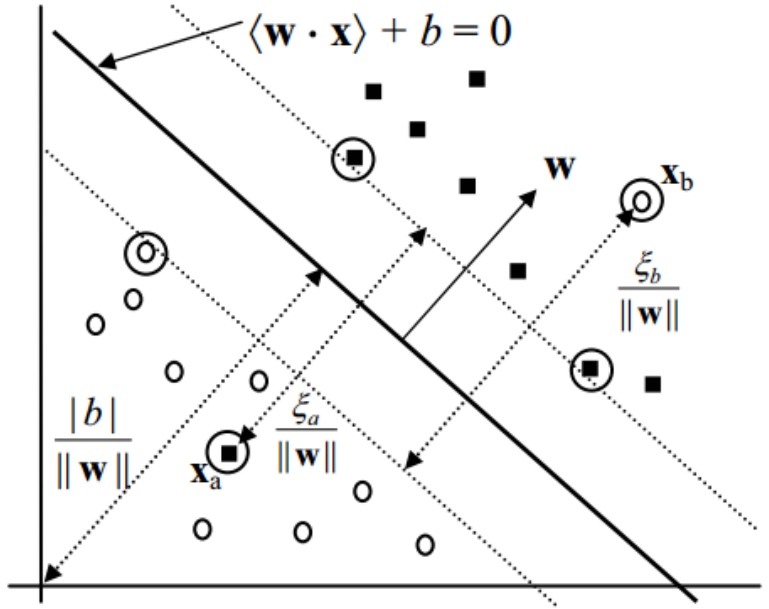
$$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 1 - \xi_i \quad \text{for } y_i = 1$$

$$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \leq -1 + \xi_i \quad \text{for } y_i = -1.$$

- The new constraints:

  Subject to: $y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1 - \xi_i$, $i = 1, \ldots, r$,

$$\xi_i \geq 0, \quad i = 1, 2, \ldots, r.$$



The non-separable case: $\mathbf{x}_a$ and $\mathbf{x}_b$ are error data points

- Example: Two error data points $\mathbf{x}_a$ and $\mathbf{x}_b$ (circled) in wrong regions

Support Vector Machine (SVM)

- **Q:How to address the situation?**
- Need to penalize the errors in the objective function.
- **Q: Why? What does it actually do?**
- Finds optimal plane which has the least errors (from the training data)

- A natural way of doing it is to assign an extra cost for errors to change the objective function to

$$\text{Minimize}: \frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2} + C\left(\sum_{i=1}^{n} \xi_i\right)^k$$

- $k = 1$ is commonly used.
- The parameter C is important. Must be tuned to ensure we get a good model.
  Q:Why?
- Higher C tends to be strict, lower values of C tend to allow more errors in the learned model. C > 0.

Support Vector Machine (SVM)

- Adding the slack variable, we have the new optimization problem.

$$\text{Minimize}: \frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2} + C\sum_{i=1}^{r} \xi_i$$

$$\text{Subject to}: y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, \quad i = 1, 2, ..., r$$

$$\xi_i \geq 0, \quad i = 1, 2, ..., r$$

- This is often referred as soft-margin SVM
- Are we on the same page?

- **Q: Given a set of training examples $\{(x_i, y_i)\}$, which of the variables $(w, \xi, b, C, r, x_i, y_i)$ are known/unknown in the above optimization problem?**

Support Vector Machine (SVM)

- **Q: What you should know to get going?**
- The intuition, where to find software (use SVMLight)
- Vector, line, length
- Margin
- QP with linear constraints
- How to handle non-separable data (advanced!)
  - ❑ Slack variables
  - ❑Kernels to map to new feature space (Section 3.8.3 WDM)

- More math detail (for nerds!): Paper by C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition

Feature Selection

- In most (text) classification problems, words, n-grams serve as features.
- However, not all are relevant, useful to model building. Some are also redundant.

- **Q: Why is this not good for model building/learning a classifier?**

- How to solve this?

- Instead of using all features, select features which can separate the classes well, i.e., features having good discriminative strengths.

- Measures of discriminative strength: Information Gain (IG), Mutual Information, Chi-Squared Statistic (χ^2)

Feature Selection

- Let C = {c_1, c_2, ..c_m} and F = {f_1, f_2, …f_n} denote the set of classes and features

- Three commonly used metrics:.

- (1) Information Gain: Expected change in entropy (of data, T) from a prior state to a state that takes some information (here, feature, a) as given.

$$IG(T,a) = H(T) - H(T|a)$$

	c	$\bar{c}$
f	W	X
$\bar{f}$	Y	Z

$$IG(f) = -\sum_{i=1}^{m} P(c_i)\log P(c_i) + \sum_{f,\bar{f}} P(f)\sum_{i=1}^{m} P(c_i \mid f)\log P(c_i \mid f)$$

- **Q: What does this mean? If IG(f_1) > IG(f_2), what can we say about f_1, f_2?**

- **Q: How do we compute the probabilities? Assume two classes c1, c2 and features are words.**

Feature Selection

- Let C = {c_1, c_2, ..c_m} and F = {f_1, f_2, …f_n} denote the set of classes and features
- Three commonly used metrics:.
- (2) Mutual Information: Expected measure of association of two random variables, F, C.

$$MI(f,c) = \sum_{f,\bar{f}} \sum_{c,\bar{c}} P(f,c) \log \frac{P(f,c)}{P(f)P(c)}$$

Point wise mutual information (PMI)

$$\mathrm{MI}(f) = \max_i \{\mathrm{MI}(f, c_i)\}$$

- **Compute probabilities using 2x2 contingency table.**

	c	$\bar{c}$
f	W	X
$\bar{f}$	Y	Z

Feature Selection

- Let C = {c_1, c_2, ..c_m} and F = {f_1, f_2, …f_n} denote the set of classes and features

- Three commonly used metrics:.

- (3) Chi-Squared (χ^2) Statistic: A measure for lack of independence between a feature f and class c.

$$\chi^2(f,c) = \frac{N(WZ - YX)^2}{(W+Y)(X+Z)(W+X)(Y+Z)}$$

$$N = W + X + Y + Z \qquad \chi^2(f) = \sum_{i=1}^{m} P(c_i)\chi^2(f,c_i)$$

	c	$\bar{c}$
f	W	X
$\bar{f}$	Y	Z

- Compute probabilities using 2x2 contingency table.
- **Q: If ($\chi^2(f_1) > \chi^2(f_2)$), which is a better feature? f_1 or f_2?**

- Usually Chi-squared statistic and Information Gain (IG) have good discriminative strengths than MI.