Part II: Web Content Mining
Chapter 5: Classification

- General Setting and Evaluation Techniques
- Nearest Neighbor
- Feature Selection
- Naïve Bayes
- Numeric Approaches
- Relational Learning

Steps of Supervised Learning

- Data collection and preprocessing

  At this step documents are collected, cleaned and properly organized, the terms (features) identified and then vector space representation created. Data are divided into two subsets:

  – Training set, used to build the model (may include a validation subset)

  – Test set, used to evaluate the model

- Building the model

  This is the actual learning (also called *training*) step that includes the use of the learning algorithm. It is usually an iterative and interactive process that includes:

  – Feature Selection

  – Applying the learning algorithm

  – Validating the model (using the validation subset to tune parameters)

- Testing and evaluating the model
- Using the model to classify documents with unknown class labels

Problems with text/web documents

- As text and web documents include thousands of words there is a major disbalance between two basic parameters in learning – the number of features and the number of instances. The terms substantially outnumber the documents, which makes the document space sparsely populated.
- The document features inherit some of the properties of the natural language text they are derived from. Many are irrelevant to the classification task and are often correlated. This violates two basic assumptions that many learning algorithms rely on – the equal importance of all attributes and their statistical independence.
- Documents are of different sizes and thus can be best represented with different number of features. However the learning algorithms need uniform size vectors, which in turn further contributes to the sparsity of the instance space.

Evaluation

Evaluation Criteria

- Accuracy of classification (main approach used in supervised learning)
- Computational efficiency of learning and applying the model (time and space complexity and scalability)
- Model interpretation and simplicity (Occam's razor, MDL)

Benchmark data

- UCI Machine Learning Repository (http://archive.ics.uci.edu/ml/)
- UCI KDD Archive (http://kdd.ics.uci.edu/)
- Reuters Text Categorization Test Collection (http://www.daviddlewis.com/resources/testcollections/reuters21578/)
- TREC data (http://trec.nist.gov/data.html)

Evaluating Accuracy (holdout)

- Holdout

  Reserve a certain amount of data (usually 1/3) for testing and use the remainder (2/3) for training.

- Stratification

  To ensure that each class is represented with approximately equal proportions in both subsets we apply sampling for training and test set within classes.

- Repeated Holdout

  Accuracy/error estimates can be made more reliable by repeating the process with different subsamples. This is achieved by random selection of the training and test sets, repeating the process several times and then averaging the success/error rates.

- Problems with Holdout

  Because of the random selection different test sets may overlap or some documents may never be used for testing. Cross-validation (CV) is an evaluation method that provides a solution to this problem.

Evaluating Accuracy (cross-validation)

- K-fold Cross-Validation (CV)

  Split data into *k* subsets of equal size (usually by random sampling and possibly with stratification). Then use each subset in turn for testing with the remainder for training. Average the error estimates to get an overall error estimate.

- Stratified ten-fold Cross-Validation

  Extensive experiments have shown that this is the best choice to get an accurate estimate. Repeated stratified cross-validation is even better: ten-fold cross-validation is repeated ten times and results are averaged.

- Leave-One-Out Cross-Validation (LOO-CV)

  n-fold cross-validation, where *n* is the number of training documents. LOO-CV makes maximum use of data and avoids any random sampling. However it is very computationally expensive and stratification is not possible.

Nearest Neighbor (basics)

- The Nearest Neighbor (NN) algorithm is a straightforward application of similarity (or distance) for the purposes of classification.
- NN predicts the class of a new document using the class label of the closest document from the training set.
- Because it uses just one instance from the training set this basic version of the algorithm is called *One-Nearest Neighbor* (1-NN).
- The closeness is measured by minimal distance or maximal similarity.
- The most common approach is to use the TFIDF framework to represent both the test and training documents and compute the cosine similarity between the document vectors.

Nearest Neighbor (example)

	history	science	research	offers	students	hall	**Class**	**Similarity to Theatre**	**Rank**
Anthropology	0	0.537	0.477	0	0.673	0.177	A	0.695979	2
Art	0	0	0	0.961	0.195	0.196	B	0.238108	9
Biology	0	0.347	0.924	0	0.111	0.112	A	0.136097	13
Chemistry	0	0.975	0	0	0.155	0.158	A	0.189681	11
Communication	0	0	0	0.780	0.626	0	B	0.605667	3
Computer	0	0.989	0	0	0.130	0.067	A	0.142313	12
Justice	0	0	0	0	1	0	B	0.967075	1
Economics	0	0	1	0	0	0	A	0	18
English	0	0	0	0.980	0	0.199	B	0.050784	16
Geography	0	0.849	0	0	0.528	0	A	0.510589	4
History	0.991	0	0	0.135	0	0	B	0	19
Math	0	0.616	0.549	0.490	0.198	0.201	A	0.242155	8
Languages	0	0	0	0.928	0	0.373	B	0.095021	14
Music	0.970	0	0	0	0.170	0.172	B	0.207746	10
Philosophy	0.741	0	0	0.658	0	0.136	B	0.034530	17
Physics	0	0	0.894	0	0.315	0.318	A	0.385508	6
Political	0	0.933	0.348	0	0.062	0.063	A	0.076221	15
Psychology	0	0	0.852	0.387	0.313	0.162	A	0.343685	7
Sociology	0	0	0.639	0.570	0.459	0.237	A	0.504672	5
Theatre	0	0	0	0	0.967	0.254	? (B)		

Nearest Neighbor (discussion)

- "Justice" is ranked 1, so the class of 'Theatre" is predicted as B (correct prediction).
- However "Justice" has only one non-zero attribute, which in fact produced the prediction. What if this term is irrelevant or noise? Use $k > 1$.
- If $k=3$ then the prediction is B again (majority of {B,A,B}).
- If $k=5$ then the prediction is A (majority of {B, A, B, A, A}).
- How about *k=7, 9, …? Wrong approach* because we are taking votes from documents less and less similar to "Theatre".
- Solutions
  - Use 1-NN (the most popular version of NN)
  - Use distance weighted k-NN

Distance Weighted Nearest Neighbor

- Weight votes with the similarity (or distance) of the documents from the training set to the document being classified.
- Instead of adding 1 for each label add a term that is proportional to similarity, for example:
  - similarity itself $sim(X,Y)$
  - or $1/(1-sim(X,Y))$
  - or $1/(1-sim(X,Y))^2$
- Distance weighted 3-NN (with the similarity) predicts B for "Theatre".
  - The weight for A ("Antropology") is $A = 0.695979$
  - The weight for B ("Justice" and "Communication") is $B = 0.967075 + 0.605667 = 1.572742$

LOO-CV with Nearest Neighbor

Attributes	No distance weighting				Distance weighting			
	1-NN	3-NN	5-NN	19-NN	1-NN	3-NN	5-NN	19-NN
6	1.00	1.00	0.90	0.55	1.00	1.00	1.00	0.85
671	0.85	0.85	0.75	0.55	0.85	0.85	0.75	0.55

- Small set of relevant attributes works much better than all attributes.
- Distance weighting does not help much with k=1,3,5.
- Distance weighting provides bigger improvement for k=19.
- Distance weighting does not make any difference with all attributes.
- Overall, 1-NN is a clear winner in all cases.
- In general, improvement may be expected with k >1 and distance weighting only in situations with noise (obviously not present in out data) and not too many irrelevant attributes.

Nearest Neighbor

- Successfully used by statisticians for more than 50 years
- As an ML approach falls in the category of *instance-based learning* because the classification is based on a number of instances from the training data
- Considered as *lazy learning*, because the model creation step does not involve any computation (instances are simply stored)
- All the computation takes place during classification when similarity is calculated for each instance in the training set. This makes the algorithm *computationally expensive*.
- To improve efficiency for large document collections a smaller random sample called *window* is used.
- The most popular similarity-based approach to text and web document classification.

Feature Selection

- Find a subset of attributes that best describe the documents with respect to the classification task (attributes with which the learning algorithm achieves maximal accuracy)
- Simple, but impractical solution: evaluate all subsets (2^n subsets for n attributes) and pick the one that maximizes accuracy
- Strong, but practical assumption: *attributes are independent* so that they can be evaluated and selected individually
- *Ranking:* sort the list of attributes according to their evaluation score (rank) and then pick a number of attributes from the top of the list. Evaluation schemes that we discuss:
  - MDL
  - Error-based
  - Information (entropy)-based
  - Similarity-based

Entropy

- S is a set of document vectors from k classes $C_1, C_2, ..., C_k$
- S_i is the set of vectors belonging to class C_i, $|S| = |S_1| + |S_2| + ... + |S_k|$
- Entropy $H(S)$ is the *average information* needed to predict the class of an arbitrary vector in S.

$$H(S) = -\sum_{i=1}^{k} P(C_i) \log P(C_i)$$

where probability of class C_i is calculated as the proportion of instances in it

$$P(C_i) = \frac{|S_i|}{|S|}$$

Information Gain

- Assume now that attribute *A* has *m* values $v_1, v_2, ..., v_m$
- Then *A* splits the set *S* into *m* subsets $A_1, A_2, ..., A_m$
- *Entropy in the split*, based on attribute *A* is defined as

$$H(A_1, A_2, ..., A_m) = \sum_{i=1}^{m} \frac{|A_i|}{|S|} H(A_i),$$

where $H(A_i)$ is the entropy of the class distribution in set A_i.

- Information gain measures the quality of a split (respectively an attribute) by the decrease of entropy.

$$gain(A) = H(S) - H(A_1, A_2, ..., A_m)$$

Similarity-Based Ranking

- Ranking is based on weights assigned to attributes that reflect their relevance to the classification.
  - For each vector a certain number of *nearest neighbors* from the same and different classes called *near hits* and *near misses* respectively are found.
  - If a near hit has a different value for a certain attribute then that attribute appears to be irrelevant and its weight is decreased.
  - For near misses, the attributes with different values are relevant and their weights are increased.
- The algorithm starts with equal weights for all attributes and repeatedly adjusts them (adding or subtracting small amounts).
- The vectors used to find near hits and misses are usually picked at random.

Near Hits and Near Misses

- Consider the “Theatre” vector
- “Justice” is a near hit of “Theatre”
- “Anthropology” is a near miss of “Theatre”

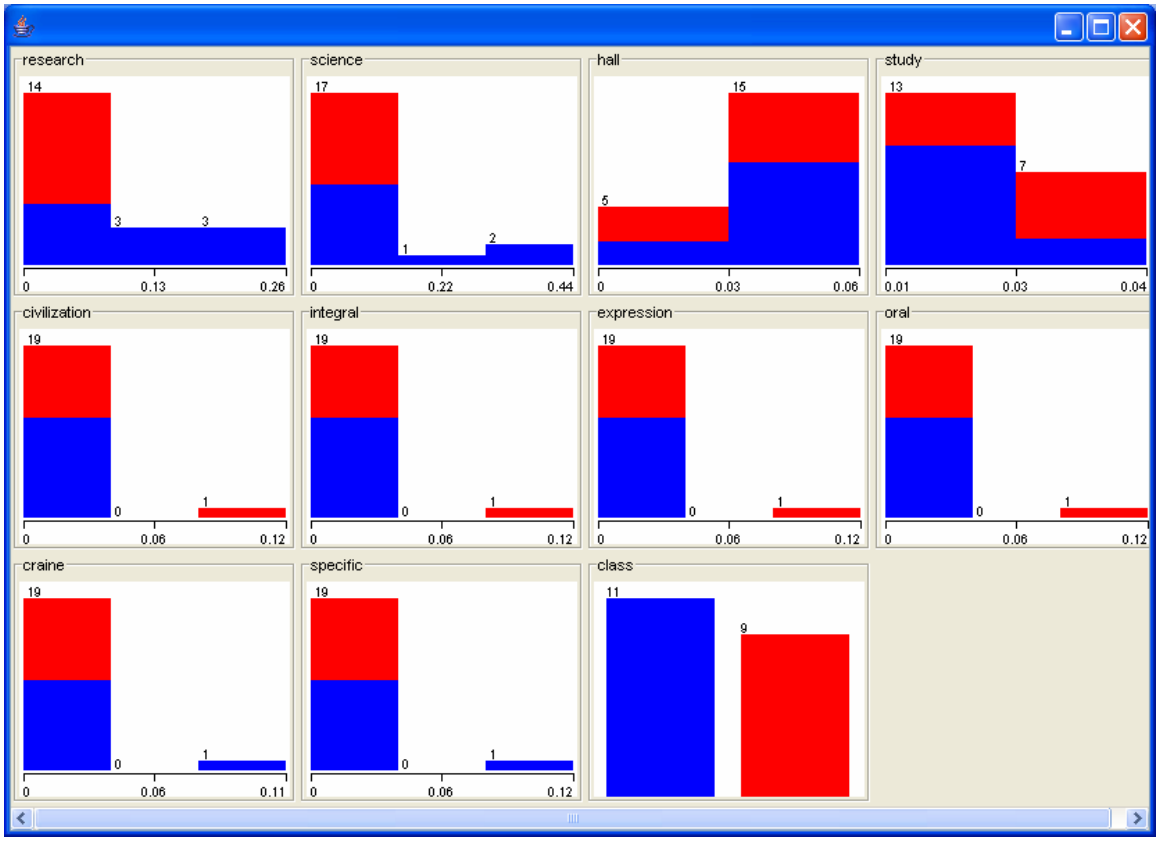
	history	science	research	offers	students	hall	class
Anthropology	0	0.537	0.477	0	0.673	0.177	A
Justice	0	0	0	0	1	0	B
Theatre	0	0	0	0	0.967	0.254	B

- Attributes “history” and “offers” have the same value in “Theatre” and in the near hit and near miss – no change in their weights.
- Attributes “science” and “research” however receive an increase because they have different values in the near miss “Anthropology”.
- Attribute “hall” receives a decrease, because its value is different in the near hit “Justice”.

Attribute Ranking (example)

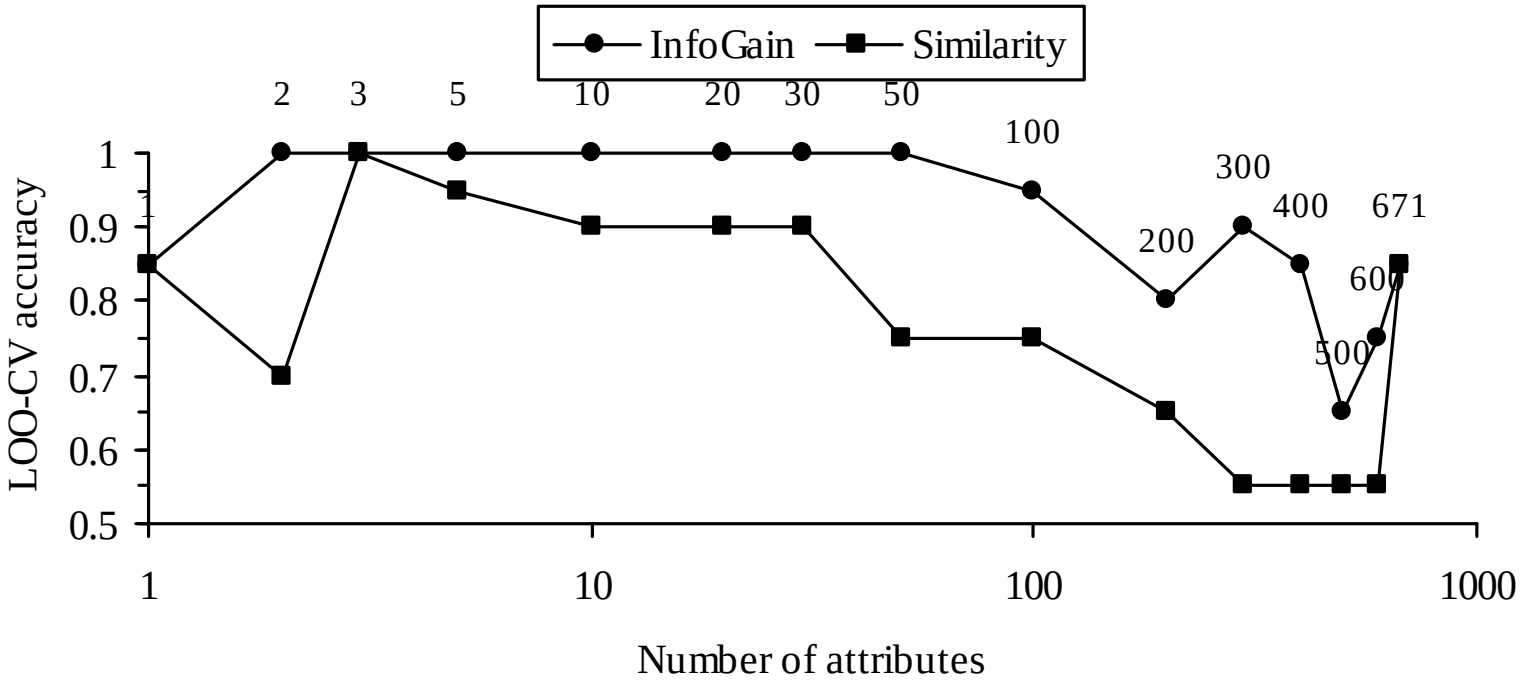
Top 10 attributes selected by two evaluation measures and histograms of their class distributions (after discretization) produced by Weka (www.cs.waikato.ac.nz/ml/weka/)

InfoGain	**Similarity**
research	research
science	offers
hall	science
study	concentrations
civilization	courses
integral	study
expression	studies
oral	based
craine	theoretical
specific	diverse



Attribute Selection (example)

Accuracy of 1-NN with increasing number of attributes ranked by InfoGain and Similarity



Naïve Bayes

- Naïve Bayes is an approach to classification based on a probabilistic setting where documents are represented by sets of *random events* – term occurrences in documents.
- In the *TFIDF* representation, document attributes are considered as normally distributed continuous variables (this approach was discussed in Chapter 3: Clustering)
- In the *Boolean* case, attributes are binary variables indicating the presence or absence of a term in the document (ignoring the number of occurrences) and are modeled by a discrete distribution.
- In the *term-count* representation attributes are random variables taking values from the set of natural numbers (0, 1, 2, …) and are modeled by the multinominal distribution.

Boolean representation

- The class of document x is the one that maximizes $P(C|x)$
- To compute $P(C|x)$ we first apply the *Bayes rule*

$$P(C \mid x) = \frac{P(x \mid C)\, P(C)}{P(x)}$$

- Then make the attribute *independence assumption* that if that x is a vector of n attribute values $x = (x_1, x_2, ..., x_n)$ then

$$P(x \mid C) = P(x_1, x_2, ..., x_n \mid C) = \prod_{i=1}^{n} P(x_i \mid C)$$

- $P(x_i|C)$ is calculated as the proportion of documents from class C that include attribute value x_i.
- $P(C)$ is the probability of sampling for class C , calculated as the proportion of the training documents that fall in class C.
- $P(x)$ is a common denominator not needed for classification.

Boolean representation (data)

	history	science	research	offers	students	hall	**Class**
Anthropology	0	1	1	0	1	1	A
Art	0	0	0	1	1	1	B
Biology	0	1	1	0	1	1	A
Chemistry	0	1	0	0	1	1	A
Communication	0	0	0	1	1	0	B
Computer	0	1	0	0	1	1	A
Justice	0	0	0	0	1	0	B
Economics	0	0	1	0	0	0	A
English	0	0	0	1	0	1	B
Geography	0	1	0	0	1	0	A
History	1	0	0	1	0	0	B
Math	0	1	1	1	1	1	A
Languages	0	0	0	1	0	1	B
Music	1	0	0	0	1	1	B
Philosophy	1	0	0	1	0	1	B
Physics	0	0	1	0	1	1	A
Political	0	1	1	0	1	1	A
Psychology	0	0	1	1	1	1	A
Sociology	0	0	1	1	1	1	A
Theatre	0	0	0	0	1	1	? (B)

Boolean representation (probabilities)

$$P(A \mid Theatre) = \frac{P(Theatre \mid A)\, P(A)}{P(Theatre)}$$

$$P(Theatre \mid A) = P(history = 0 \mid A) \times P(science = 0 \mid A) \times P(research = 0 \mid A) \times$$
$$\times P(offers = 0 \mid A) \times P(students = 1 \mid A) \times P(hall = 1 \mid A)$$

$$P(Theatre \mid A) = \frac{11}{11} \times \frac{4}{11} \times \frac{3}{11} \times \frac{8}{11} \times \frac{10}{11} \times \frac{9}{11} = 0.0536476 \qquad P(A) = 11/19 = 0.578947$$

$$P(Theatre \mid B) = \frac{5}{8} \times \frac{8}{8} \times \frac{8}{8} \times \frac{2}{8} \times \frac{4}{8} \times \frac{5}{8} = 0.0488281 \qquad P(B) = 8/19 = 0.421053$$

$$P(A \mid Theatre) = \frac{0.0536476 \times 0.578947}{P(Theatre)} \approx 0.0310591 \qquad P(A \mid Theatre) = \frac{0.0310591}{0.0310591 + 0.0205592} = 0.601707$$

$$P(B \mid Theatre) = \frac{0.0488281 \times 0.421053}{P(Theatre)} \approx 0.0205592 \qquad P(B \mid Theatre) = \frac{0.0205592}{0.0310591 + 0.0205592} = 0.398293$$

Boolean representation (discussion)

- The prediction made by Naïve Bayes (class *A*) is incorrect.
- Interestingly the same prediction was made by the 19-NN with distance weighting, which uses the whole training set (as Naïve Bayes does).
- The 1-NN prediction (class *B)* is correct, but it is based on a single document.
- Assuming that the original label (*B*) is unknown which classification is more feasible?
- The intuition suggests that the classification that uses more information must be more feasible.
- However although the boolean Naïve Bayes algorithm uses all training documents it ignores the term counts.
- So, our next step is to investigate how a Bayesian model based on term counts will classify our test document.

Multinominal Naïve Bayes

- Assume that there are m terms $t_1, t_2, \ldots, t_m$ and n documents $d_1, d_2, \ldots, d_n$ from class C.
- Assume that term t_i occurs in document d_j n_{ij} times.
- Then term t_i occurs in all documents from class C with probability

$$P(t_i \mid C) = \frac{\sum_{j=1}^{n} n_{ij}}{\sum_{i=1}^{m} \sum_{j=1}^{n} n_{ij}}$$

- According to the *multinominal distribution* the probability of document d_j given class C is

$$P(d_j \mid C) = \left(\sum_{i=1}^{m} n_{ij}\right)! \times \prod_{i=1}^{m} \frac{P(t_i \mid C)^{n_{ij}}}{n_{ij}!}$$

Multinominal Naïve Bayes (cont.)

- This formula for $P(d_j|C)$ looks similar to the Naïve Bayes assumption.
- The basic differences are the factorials, which are added to account for the bag of words model where the ordering of words is ignored. Thus

$n_{ij}!$ accounts for all possible orderings of each word.

$\left(\sum_{i=1}^{m} n_{ij}\right)!$ accounts for all possible orderings of all words in the document.

Term-counts representation (data)

	history	science	research	offers	students	hall	**Class**
Anthropology	0	1	1	0	4	1	A
Art	0	0	0	2	1	1	B
Biology	0	1	3	0	1	1	A
Chemistry	0	2	0	0	1	1	A
Communication	0	0	0	1	2	0	B
Computer	0	5	0	0	2	1	A
Justice	0	0	0	0	1	0	B
Economics	0	0	1	0	0	0	A
English	0	0	0	2	0	1	B
Geography	0	1	0	0	2	0	A
History	7	0	0	2	0	0	B
Math	0	1	1	1	1	1	A
Languages	0	0	0	1	0	1	B
Music	1	0	0	0	1	1	B
Philosophy	1	0	0	2	0	1	B
Physics	0	0	1	0	1	1	A
Political	0	5	2	0	1	1	A
Psychology	0	0	2	1	2	1	A
Sociology	0	0	1	1	2	1	A
Theatre	0	0	0	0	4	1	? (B)

Term-counts representation (probabilities)

- First compute $P(t_i|C)$ by taking the sum of the counts in column t_i within class C and divide it by the total count for all columns.
- To avoid zero probabilities use the Laplace estimator. For example, $P(history|A)=(0+1)/(57+2)=0.017$
- Then plug $P(t_i|C)$ into the formula and obtain the document probabilities given the class.

$$P(Theatre \mid A) = 5! \times \frac{0.017^0}{0!} \times \frac{0.288^0}{0!} \times \frac{0.22^0}{0!} \times \frac{0.068^0}{0!} \times \frac{0.305^4}{4!} \times \frac{0.017^1}{1!}$$

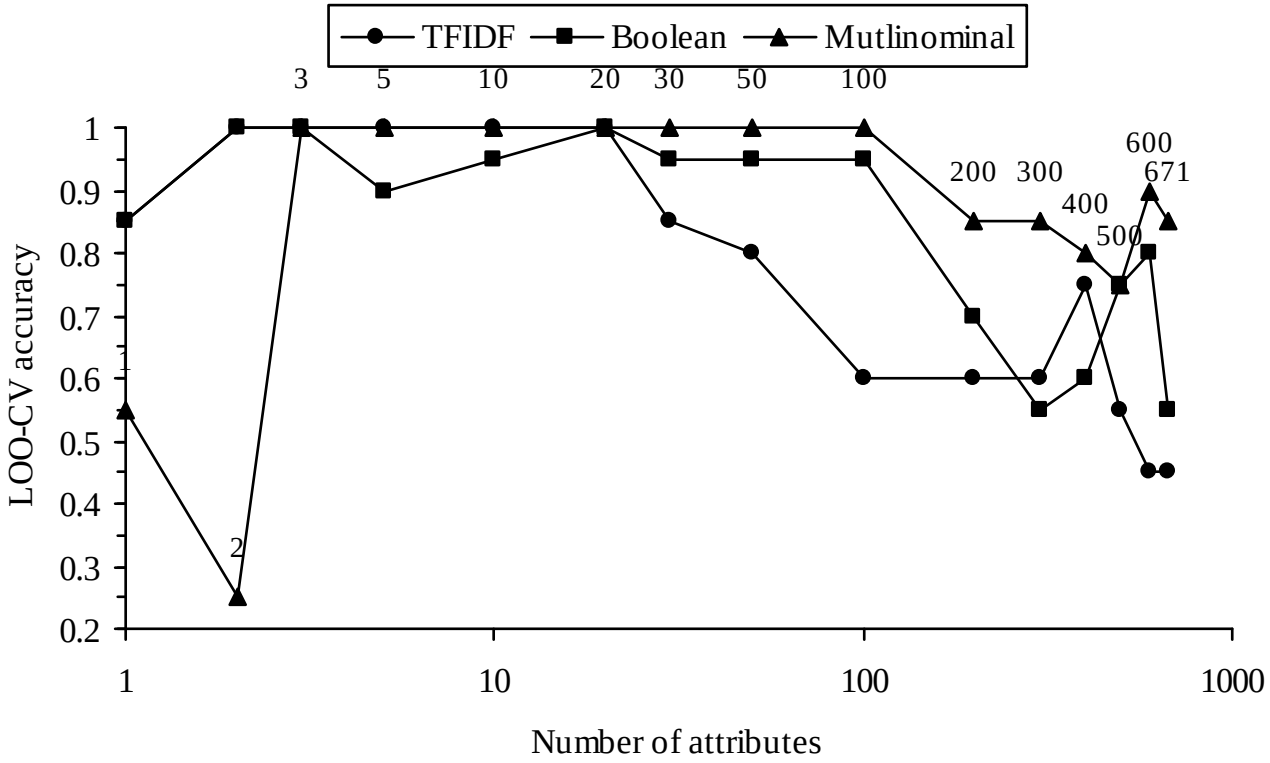
$$P(Theatre \mid B) = 5! \times \frac{0.323^0}{0!} \times \frac{0.0323^0}{0!} \times \frac{0.0323^0}{0!} \times \frac{0.355^0}{0!} \times \frac{0.194^4}{4!} \times \frac{0.194^1}{1!}$$

- Finally compute $P(A|Theatre)$ and $P(B|Theatre)$ by using only the numerator of the Bayes rule and then normalize the likelihoods.

$$P(A \mid Theatre) \approx 0.0000354208 \qquad P(A \mid Theatre) = 0.88$$

$$P(B \mid Theatre) \approx 0.00000476511 \qquad P(B \mid Theatre) = 0.12$$

Naïve Bayes accuracy with three different data representations



The drop of the multinominal curve at two attributes ("research" and "science") illustrates a conceptual difference with the Boolean case – the way term count 0 is used to compute the attribute probability $P(t_i|C)$.

Naïve Bayes (discussion)

- All three representations use only the number of occurrences of words in documents (*bag of words* approach).
- Multinominal Naïve Bayes performs best because it incorporates this information directly into the classification mechanism.
- The next best algorithm, boolean Naïve Bayes uses only a part of this information – whether or not a word occurs in the document.
- The TFIDF scheme transforms the types of attributes (counts into continuous values) and also assumes that they are normally distributed (which may not be true).
- The TFIDF framework is originally designed to represent documents as points in a *metric space* with the assumption that similar documents are represented by close points in space.
- Therefore the TFIDF representation is more suitable for similarity-based and metric-based approaches than probabilistic ones.

Naïve Bayes (discussion cont.)

- Multinominal Naïve Bayes is suitable for domains with large vocabularies such as the Web because it works relatively well without selecting a small subset of relevant attributes.
- Although it is based on the independence assumption, which is almost never present in real data, the Naïve Bayes algorithm works surprisingly well in practice. A possible explanation is that classification doesn't require accurate probability estimates as long as maximum probability is assigned to the correct class.
- Naïve Bayes is more efficient than Nearest Neighbor, because the probabilities are computed during learning and used at the classification step, which is straightforward.

Numeric Approaches

- Documents are considered as *points in a metric space* and similarity is defined by the *Euclidean distance*.
- Classification is expressed as a *separation problem*:
  - Given a set of points in Euclidean space
  - Find a surface that divides the space in two parts, so that the points that fall in each part belong to a single class.
- The most popular approach to solve this problem, called *linear regression* uses a *hyperplane* as a separating surface.
  - The predicted class value C is computed as a *linear combination* of the attribute values x_i.

    $$C = w_0 + w_1 x_1 + ... + w_m x_m$$

  - The learning objective is to find the coefficients w_i given a number of training instances with their class values.

Linear Regression for binary classification

- Substitute class labels with the values -1 and 1.
- The predicted class is determined by the sign of the linear combination.

	history	science	research	offers	students	hall	**Class**	**Numeric Class**
Anthropology	0	0.537	0.477	0	0.673	0.177	A	-1
Art	0	0	0	0.961	0.195	0.196	B	1
Biology	0	0.347	0.924	0	0.111	0.112	A	-1
Chemistry	0	0.975	0	0	0.155	0.158	A	-1
Communication	0	0	0	0.780	0.626	0	B	1
Computer	0	0.989	0	0	0.130	0.067	A	-1
Justice	0	0	0	0	1	0	B	1
Economics	0	0	1	0	0	0	A	-1
English	0	0	0	0.980	0	0.199	B	1
Geography	0	0.849	0	0	0.528	0	A	-1
History	0.991	0	0	0.135	0	0	B	1
Math	0	0.616	0.549	0.490	0.198	0.201	A	-1
Languages	0	0	0	0.928	0	0.373	B	1
Music	0.970	0	0	0	0.170	0.172	B	1
Philosophy	0.741	0	0	0.658	0	0.136	B	1
Physics	0	0	0.894	0	0.315	0.318	A	-1
Political	0	0.933	0.348	0	0.062	0.063	A	-1
Psychology	0	0	0.852	0.387	0.313	0.162	A	-1
Sociology	0	0	0.639	0.570	0.459	0.237	A	-1
Theatre	0	0	0	0	0.967	0.254	? (B)	$C>0$?

Linear Regression Problem

- Find coefficients $w_0, w_1, ..., w_6$ that satisfy a system of 19 linear equations:

$$-1 = w_0 + w_1 \times 0 + w_2 \times 0.537 + w_3 \times 0.477 + w_4 \times 0 + w_5 \times 0.673 + w_6 \times 0.177$$
$$1 = w_0 + w_1 \times 0 + w_2 \times 0 + w_3 \times 0 + w_4 \times 0.961 + w_5 \times 0.195 + w_6 \times 0.196$$
$$\ldots$$
$$-1 = w_0 + w_1 \times 0 + w_2 \times 0 + w_3 \times 0.639 + w_4 \times 0.570 + w_5 \times 0.459 + w_6 \times 0.237$$

- The solution for the vector of coefficients $(w_0, w_1, ..., w_6)$ is:
  (1.1397, -0.0549, -1.8396, -1.9736, -0.0175, -0.3912, -0.623)
- The predicted class value of "Theatre" is then:

$$1.1397 + 0 + 0 + 0 + 0 + -0.3912 \times 0.967 + -0.623 \times 0.254 = 0.603168$$

- The result is positive and thus the predicted class label of "Theatre" is *B*.

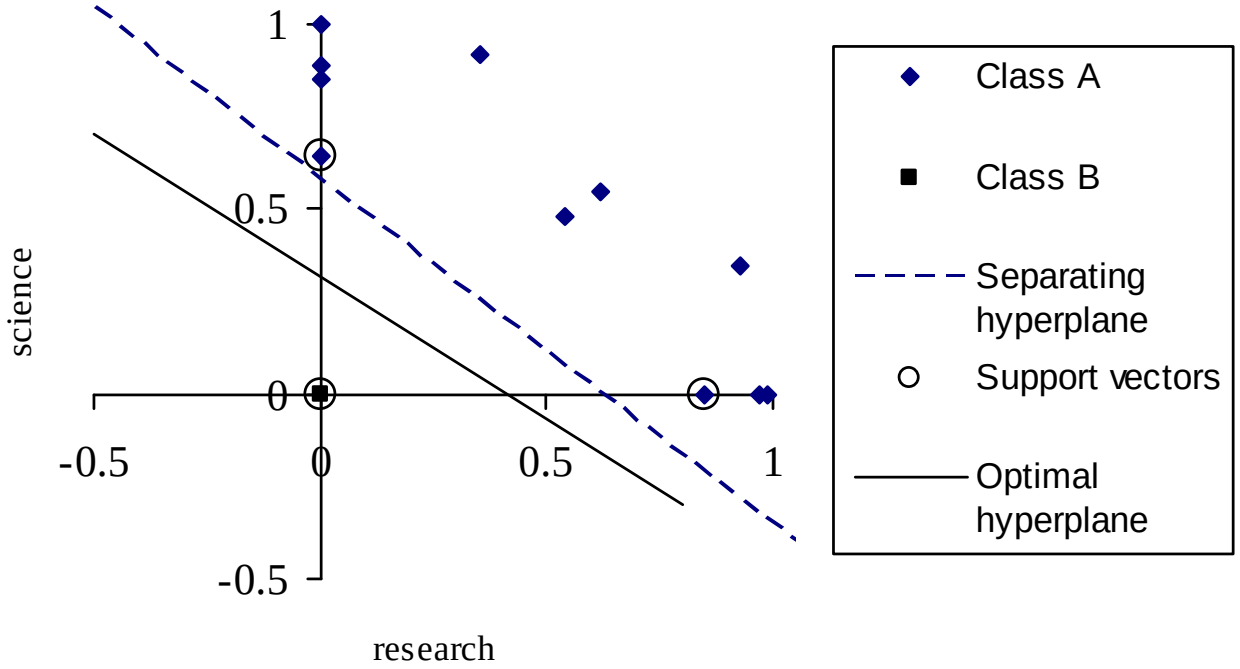
Solving Linear Regression Problem

- There exist standard numerical analysis techniques to solve the linear regression problem.
- The solution *minimizes the squared error*, which is the sum of the squared difference between the predicted and the actual class values through all training instances.
- Generally a solution exists if there are *more instances than attributes*.
- There are situations when classes are not *linearly separable*, that is, cannot be separated by a hyperplane (or line in two dimensions).
- The simplest example of a non-linearly separable classes is the XOR problem, defined with two attributes x and y and a class value equal to the boolean function $x \oplus y$ ($0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, $1 \oplus 1 = 0$).

Linear Regression Extensions

- *Maximum Margin Hyperplane* – an optimal hyperplane with the maximum distance from the nearest points, called support vectors
- Separating hyperplane

$$1.1397 - 0.0549x_1 - 1.8396x_2 - 1.9736x_3 - 0.0175x_4 - 0.3912x_5 - 0.623x_6 = 0$$



Support Vector Machines (SVM)

- The SVM approach addresses the biggest disadvantage of linear regression – its inability to represent non-linear boundaries between classes.
- The basic idea is to use a non-linear mapping (called *kernel*), which transforms the original vector space with non-linear class boundaries into a new (usually higher dimensional) space, where the class boundaries are linear.
- SVM are also combined with the maxmum margin hyperplane approach, which makes them the most accurate classifier especially suitable for non-linear cases.
- SVM are considered the most accurate text document classifier.
- The training algorithms for SVM are computationally expensive.

Relational Learning

The first step in Relational Learning is transforming the vector space representation to a *relational representation*:

- Each cell in the document-term matrix defines an instance of a binary relation, *contains*, which has a truth value.
- If term *T* occurs in document *D* then *contains(D, T)* is true, otherwise it is false.
- To represent the document class we may use:
  - a binary relation *class(C,D)*, where *C* takes the values of 'A' or 'B',
  - or the unary relations *class_A(D)* and *class_B(D)*.
- The collection of instances of these relations is called *background knowledge*.

Background Knowledge

Prolog clauses (facts)

```
class_A('Anthropology').
class_A('Biology').
class_A('Chemistry').
class_A('Computer').
class_A('Economics').
class_A('Geography').
class_A('Math').
class_A('Physics').
class_A('Political').
class_A('Psychology').
class_A('Sociology').
```

```
class_B('Art').
class_B('Communication').
class_B('Justice').
class_B('English').
class_B('History').
class_B('Languages').
class_B('Music').
class_B('Philosophy').
class_B('Theatre').
```

```
contains('Anthropology', science).
contains('Anthropology', research).
contains('Anthropology', students).
contains('Anthropology', hall).
contains('Art', offers).
contains('Art', students).
contains('Art', hall).
contains('Biology', science).
…
contains('Theatre', students).
contains('Theatre', hall).
```

Relational Learning Task

- Given the *background knowledge* and *target relations*,
- Find clauses (also called rules) that define the target relations in terms of the background relations.
- For example, if the target relation is *class(C,D)*, the system may generate the following clauses:

```
class('A',D) :- contains(D,research).
class('A',D) :- contains(D,science).
class('B',D) :- not contains(D,science), not contains(D,research).
```

- Or, if the target relations are *class_A* and *class_B*, the system may generate the following clauses:

```
class_A(D) :- contains(D, research).
class_A(D) :- contains(D, science).
class_B(D) :- not contains(D, science), not contains(D, research).
```

Relational Learning Notation

The notation used here is based on Prolog syntax, where:

- ":-" means implication ()
- "," – logical conjunction
- "not" – logical negation
- Strings beginning with an upper case letter are variables and those beginning with a lower case letter or in single quotes are constants.

```
class_A(D) :- contains(D, research).
class_A(D) :- contains(D, science).
class_B(D) :- not contains(D, science), not contains(D, research).
```

These clauses read:

- If a document contains "science" its class is A.
- If a document contains "research" its class is A.
- If a document does not contain "science" and does not contain "research" its class is B.

First-Order Inductive Learning (FOIL)

- The algorithm FOIL (First-Order Inductive Learning), proposed by Quinlan (1990), is one of the most successful relational learning algorithms.
- It is simple and efficient and uses an entropy-based technique (similar to InfoGain) to guide the search for clauses.
- It is freely available from http://www.rulequest.com/Personal/foil6.sh.
- For information on FOIL and examples of its use for web document classification see the document:

  Zdravko Markov and Ingrid Russell. Relational Learning for Web Document Classification, Machine Learning Experiences in Artificial Intelligence: A Multi-Institutional Project, NSF funded grant, July 2007 (http://uhaweb.hartford.edu/compsci/ccli/rl.htm)

Relational Learning with FOIL

Generating clauses for the *class_A* target relation:

- The instances of *class_A* (shown in Slide 39) are called *positive examples*
- Given the positive examples the algorithm automatically generates a set of *negative examples* by using the Closed World Assumption (CWA). Under CWA all instances of relations not explicitly specified as being true are assumed to be false.
- Thus the negative examples are instances of the same relation, where its argument takes all values from the contrasting class B ('Art', 'Communication', 'Justice' etc.).
- The objective is to find clauses that *cover* all positives and do not cover any negatives (the latter condition may be relaxed in the presence of noise, i.e. some percentage of negative examples may be covered too).
- A clause *covers* an example if there are substitutions for its variables, such that when applied the clause head coincides with the example and each of its body literals occurs in (or follows from) the background knowledge.

FOIL algorithm (inner loop)

- The algorithm works in a general to specific fashion starting from the most general hypothesis, *class_A(D)*, where *D* is a free variable.
- *class_A(D)* covers all positives, however all negatives too. Therefore it should be specialized by adding body literals that include variable *D* (why?)
- The potential candidate literals along with the number of positive and negative examples covered by the clause after adding the literal:

Candidate literal	**# of covered positives**	**# of covered negatives**
contains(D, history).	0	3
contains(D, science).	7	0
contains(D, research).	8	0
contains(D, offers).	3	6
contains(D, students).	10	5
contains(D, hall).	9	6

- Best clause selected using *information gain*:

```
class_A(D) :- contains(D, research).
```

- The clause does not cover any negative examples. Otherwise, it has to be extended with more literals until it covers no negatives. This ends the inner loop.

FOIL algorithm (outer loop)

- If the clause found by the inner loop covers all positive examples, the algorithm stops.
- However it covers only 8. Therefore more clauses are needed to cover the rest of the positive examples.
- The 8 already covered examples are excluded from the initial set of positives and the algorithm enters the inner loop again with the remaining ones:

```
class_A('Chemistry').
class_A('Computer').
class_A('Geography').
```

- Now it finds the clause that covers all remaining examples and thus the outer loop ends.

```
class_A(D) :- contains(D, science)
```

- Negated literals such as “not contains(D,science)” are also considered as candidates and evaluated in the same way as the non-negated ones. This is how the clause for relation *class_B* has been found.

Relational Classification

- After being generated the clauses for the target relation can be used to classify new documents.
- The new document is first represented as a set of relations "contains" with the terms that it includes.
- Then a deductive system (e.g. Prolog) checks which one of the relations "class_A" or "class_B" with the document specified as an argument is true.
- The definition of the target relation is also an *explanation* of the class of documents in terms of relations from the background knowledge. This is essentially the *knowledge discovery* step (turning data into knowledge) in the process of web mining.

Relational Representation of the Web (example)

Target Relation:

```
chairperson(X,D) :-
      department_page(D), contains(D,X),
      link_to(D1,D), link_to(D1,D2),
      directory_page(D2), contains(D1,X).
```

Background Knowledge:

```
directory_page('Departments').
directory_page('Dept. Directory').
department_page('Theatre').
…
faculty('Johnson').
staff('Perrotta').
…
chairperson(johnson, 'Theatre').

contains('Theatre', 'Johnson').
contains('Theatre', students).
contains('Theatre', hall).
…
contains('Dept. Directory', 'Johnson').
contains('Dept. Directory', 'Perrotta').
…
link_to('Departments', 'Anthropology').
link_to('Departments', 'Biology').
…
link_to('Departments', 'Theatre').
…
link_to('Departments', 'Dept. Directory').
```

Web Page Similarity (example)

Background Knowledge:

```
similar(computer,math).
similar(computer,physics).
similar(math,computer).
similar(math,physics).
similar(physics,computer).
similar(physics,math).
similar(anthropology,political).
similar(anthropology,psychology).
similar(anthropology,sociology).
...

link_to(computer,scholarship).
link_to(math,scholarship).
link_to(physics,scholarship).

contains(anthropology,research).
contains(anthropology,science).
contains(anthropology,social).
contains(anthropology,ba).
contains(political,social).

...
```

Target Relation inferred by FOIL:

```
similar(A,B) :- link_to(A,C),link_to(B,C),A\=B.
similar(A,B) :- contains(B,social),contains(A,social),A\=B.
```

Deductive inference:

```
?- similar(computer,X).
X=math;
X=physics

?- similar(political,X).
X=anthropology

?- similar(X,Y),
   writeln(X-Y),fail.
computer-math
computer-physics
math-computer
math-physics
physics-computer
physics-math
political-anthropology
psychology-anthropology
sociology-anthropology
```

Adding new pages and links:

```
contains(polish_studies,social).
link_to(art,exhibition).
link_to(theatre,exhibition).
```

Inferring new similarities:

```
?- similar(art,X).
X = theatre

?- similar(polish_studies,X).
X = anthropology ;
X = political ;
X = psychology ;
X = sociology
```