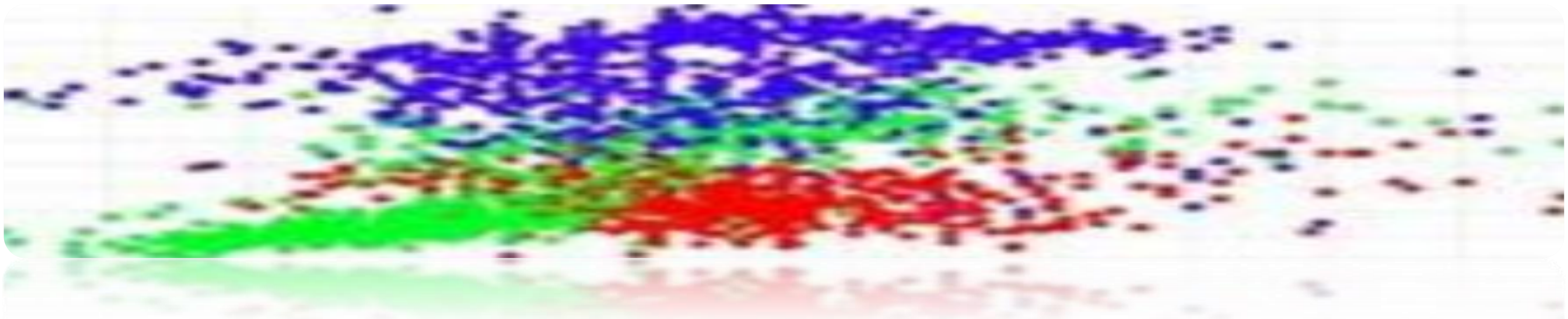


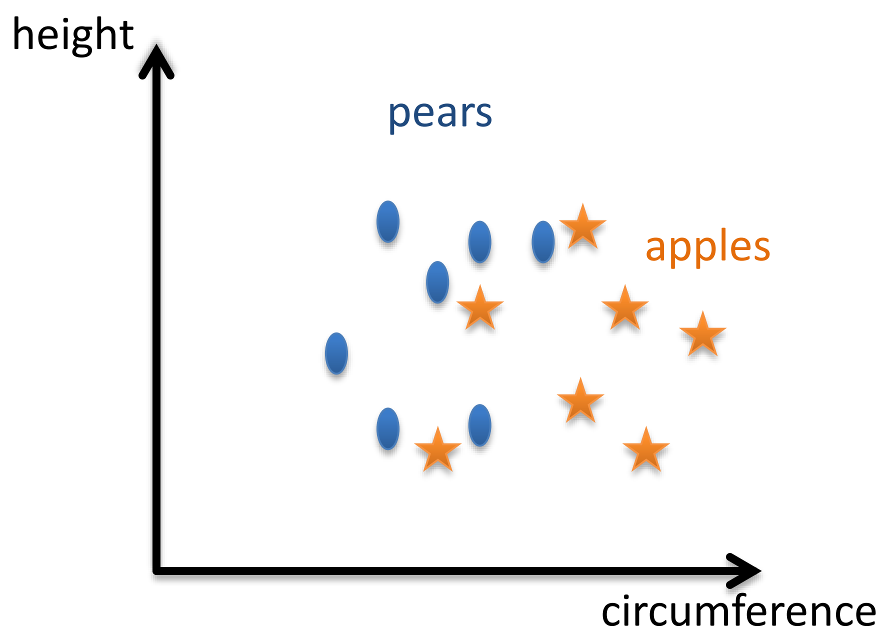
SUPERVISED LEARNING: INTRODUCTION TO CLASSIFICATION

Outline

- Basic terminology
- Features
- Training and validation
- Model selection
  - Error and loss measures
  - Statistical comparison
  - Evaluation measures

Terminology

- **Instance**
  - A real-world object
  - Abstractly described through **feature values**, $\mathbf{x} = (x_1, \dots, x_n)$
- **Class**
  - Set of similar instances
  - Typically described by a **class label**



Features

- Discrete features
  - Categorical features, i.e., features without ordering or scale (e.g., Boolean features, IDs of specific terms contained in a document, …)
  - Ordinal features, i.e., features that can be ordered but do not have a scale (e.g., Running IDs, date of an event, house numbers, seat numbers, …)

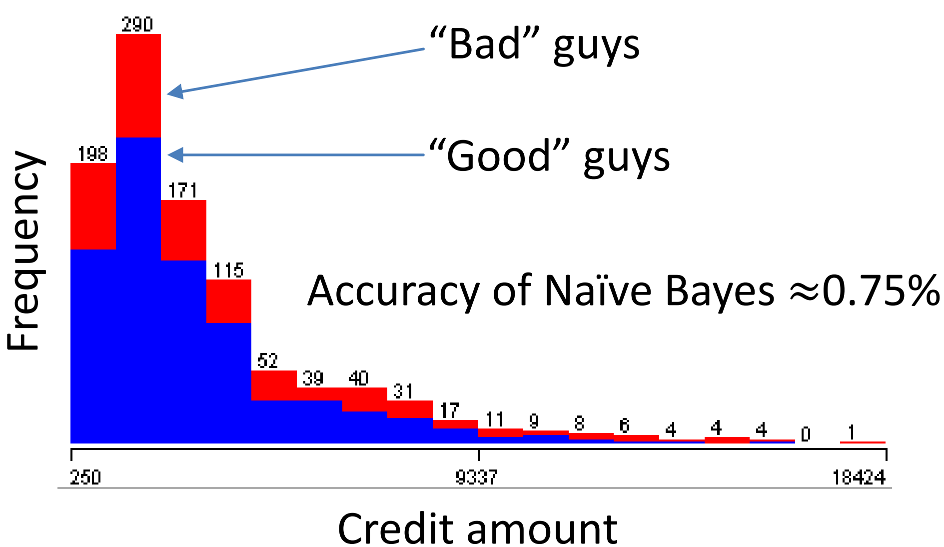
- Continuous features
  - Real-valued features (i.e., with ordering and scale), e.g., height, weight, time, …

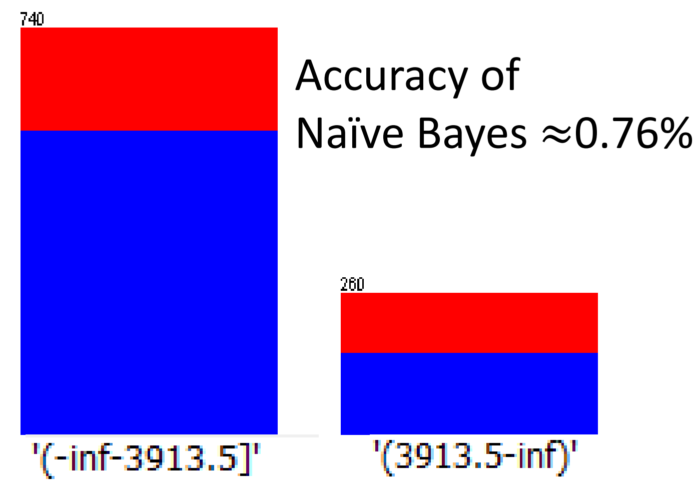
Feature type	Order	Scale	Tendency	Dispersion
Categorical	X	X	mode	n/a
Ordinal	√	X	median	quantiles
Continuous	√	√	mean	Range, variance, standard deviation

Feature transformations

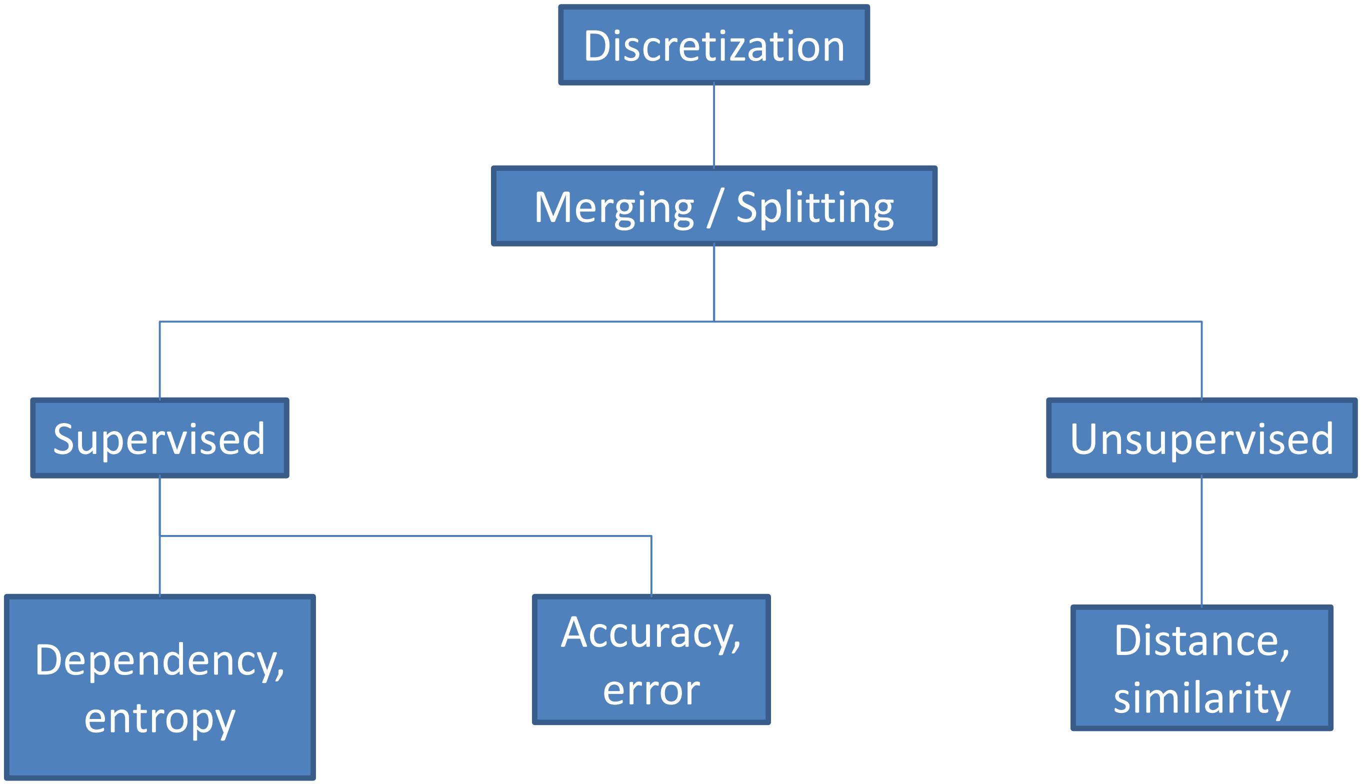
From \ To	Categorical	Ordinal	Continuous
Categorical	*Grouping*	*Ordering*	*Calibration*
Ordinal	*Unordering*	*Ordering*	*Calibration*
Continuous	*Discretization*	*Discretization*	*Normalization*

- Example
  - Prediction of creditworthiness; one of the features is the credit amount





Discretization of continuous features



Groups/bins based on χ^2, mutual information, entropy, MDL, and correlation analysis

Groups/bins based on prediction analysis

Binning strategies
+
cluster, MDL analysis

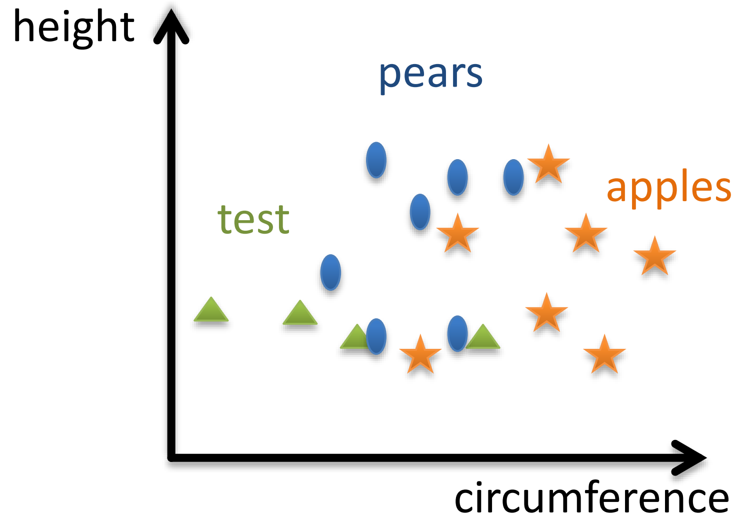
Feature selection

- Statistical process of choosing "good" features for specific classification task (e.g., χ^2-test of independence, mutual information/information gain, correlation analysis, …)

- Features should be
  - Not too general
  - Not too specific
  - Possibly independent of each other
  - Possibly with strong class correlation

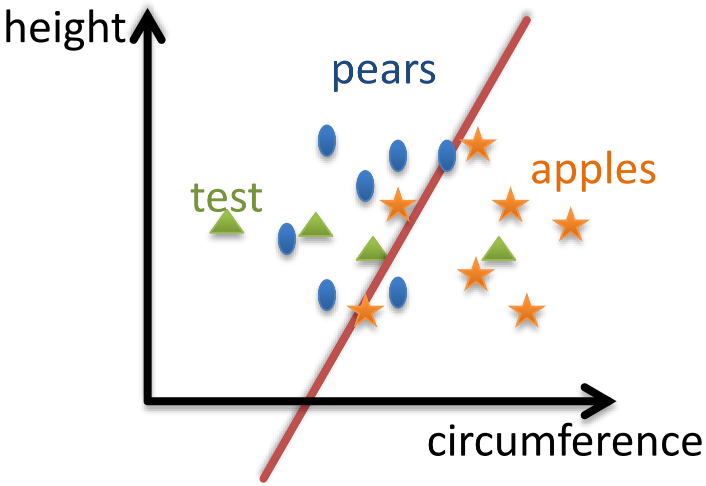
- Feature selection and transformation depends on the **data at hand** and the **underlying classification task**
  - **Rule 1**: Get to know your data
  - **Rule 2**: Make plausible assumptions (e.g. height in a population is normally distributed, income distribution is highly skewed and peaked, …)

Terminology: Training and test set



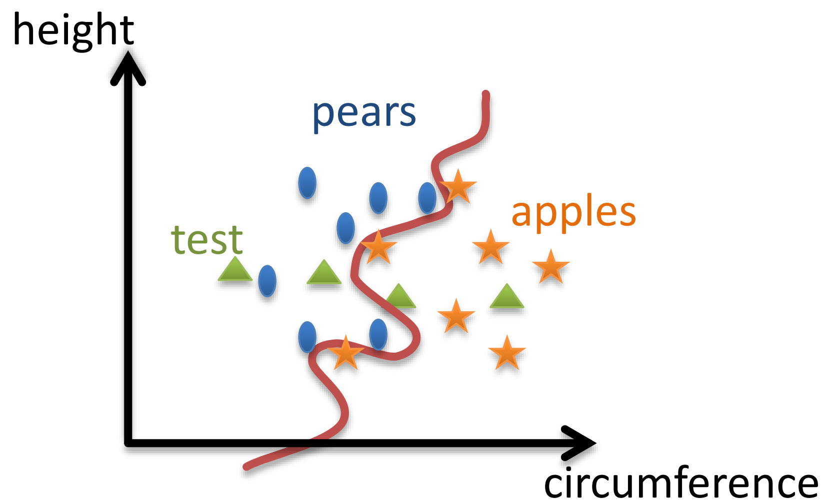
- Training set
  - Pairs $(\mathbf{x}, l(\mathbf{x}))$ of instances and corresponding class labels used to train a classification model
- Test set
  - Instances for which the classes are known but hidden to test the classifier
- Supervised learning
  - Learning with training and test set

Linear classification models



- Naive Bayes
- Perceptron
- Winnow
- Logistic Regression
- Support Vector Machines
- …

Non-linear classification models



- Rule-based classifiers
- K-Nearest Neighbors
- Hierarchical classifiers
- Ensemble classifiers
- Kernel methods
- Neural networks
- …

Multi-class classification

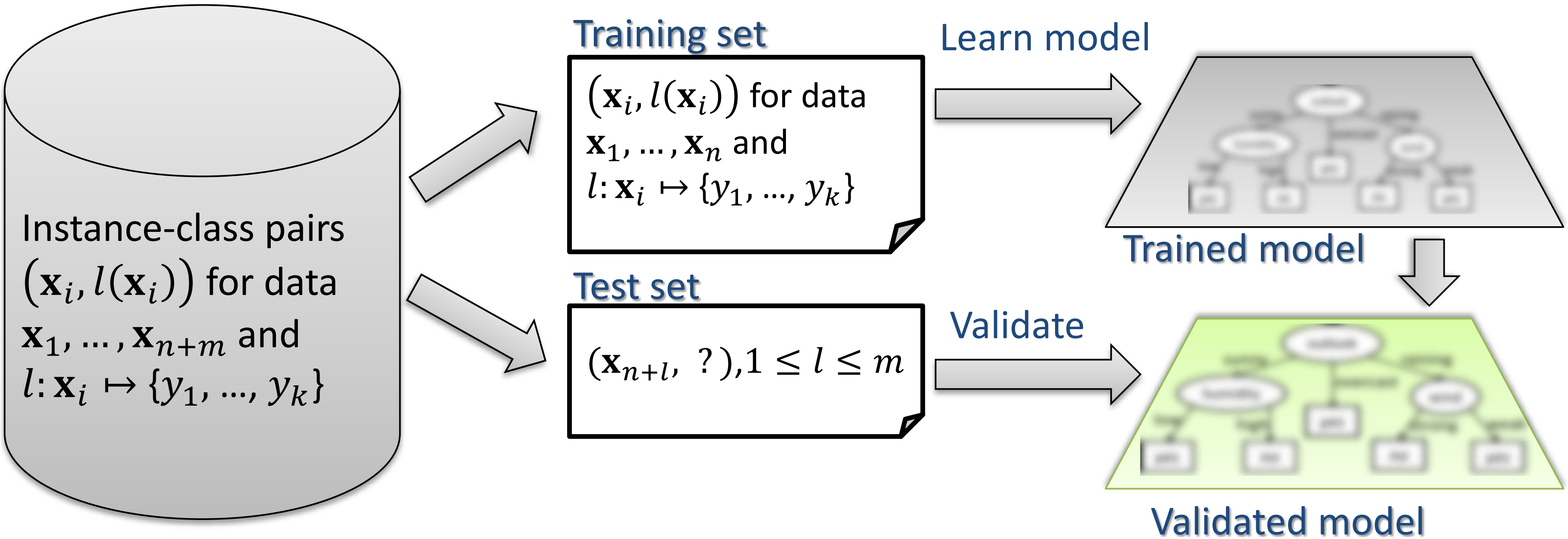
- In binary classification instance can belong either to C or to $\bar{C}$
- For $k > 2$ different classes $C_1, \ldots, C_k$
  - One-of classification: Instance can belong to only one of the k classes
  - Any-of classification: Instance can belong to many or none of the k classes
- Examples of binary classifiers that naturally generalize to multiclass classifiers
  - Naive Bayes
  - Decision trees
  - K-nearest neighbors
  - Neural networks
- In general, geometric models (e.g., Support Vector Machines, Winnow, etc.) do not generalize to multiclass classifiers

One-of vs. any-of classification

- **One-of classification**: Instance can belong to only one of the k classes
  - Typical solution:
    1. Build a classifier for each C_i and its complement $\overline{C}_i$
    2. For each instance, apply each classifier separately
    3. Assign the instance to the class with maximum score (where the score represents how well the instance fits the class)

- **Any-of classification**: Instance can belong to many or none of the k classes
  - Typical solution:
    1. Build a classifier for each C_i and its complement $\overline{C}_i$
    2. The decision of one classifier has no influence on the decisions of the other classifiers

Training and validation

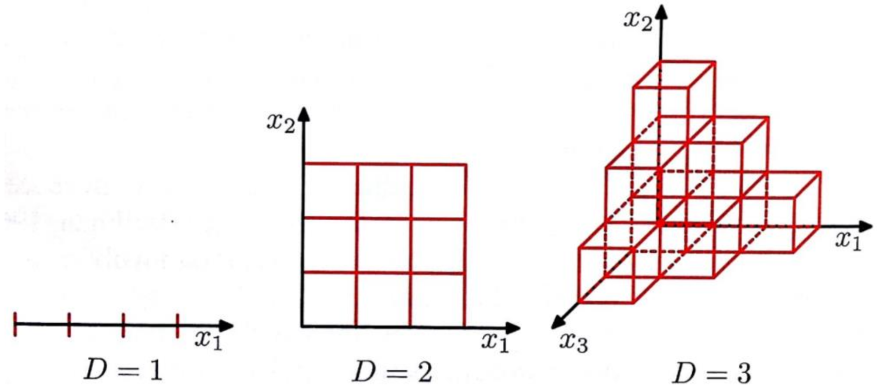


Overfitting

- Refers to cases where an algorithm's performance is much better on the training than on the test set
- May occur when
  - Training set is small
  - Training instances are not representative
  - Parameters are set to the best performing values on the training set
  - Learning is performed for too long (e.g., for neural networks)
  - When the dimensionality of the data is high

Curse of dimensionality

- Instance space grows exponentially with increasing number of dimensions
- Training data becomes quickly non-representative (→ overfitting)



Example from C. Bishop, PRML

- **Distribution of data in high-dimensional space may be counter-intuitive**
- Example 1: Distance-based similarities become non-discriminative (why?)
- Example 2: What is the fraction of the volume lies between 1 and $1 - \varepsilon$ in a sphere of radius $r = 1$?

$$V(r) = \alpha r^D \Rightarrow \frac{V(1) - V(1-\varepsilon)}{V(1)} = \frac{\alpha - \alpha(1-\varepsilon)^D}{\alpha} = 1 - (1-\varepsilon)^D$$

For large *D* most of the volume is concentrated near the surface.

Cross validation

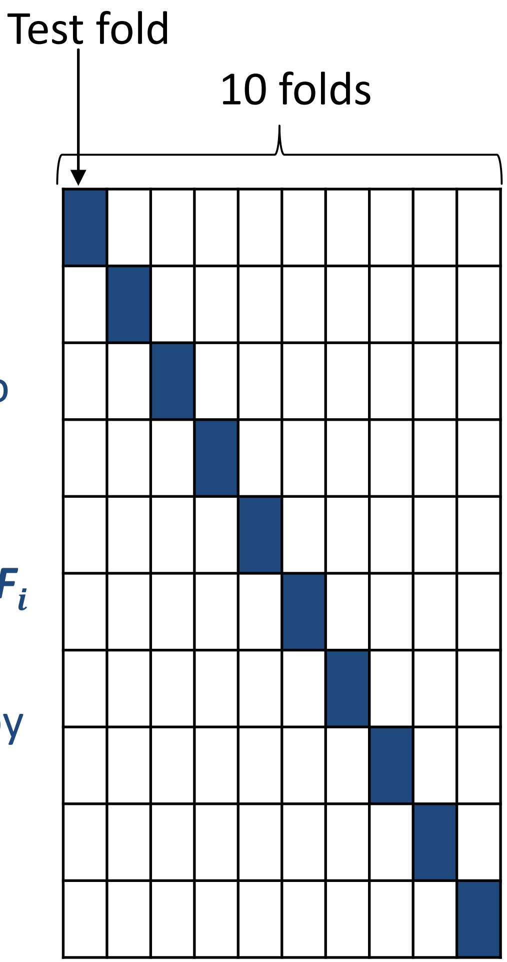
- How to adjust model parameters to good values and avoid overfitting?

- **N-fold cross validation**
  - Partition the dataset randomly in N folds (ideally, the frequency of each class in a fold should be proportional to its frequency in the full dataset).
  - Repeat for each fold $\boldsymbol{F_i}$

    Train the model on the remaining N-1 folds and test on $\boldsymbol{F_i}$

    Estimate the error err_i on $\boldsymbol{F_i}$
  - Compute the weighted average of the parameter values by taking errors into account

- **Leave-one-out cross validation**
  - N-fold cross validation where each fold is a single instance (N is the number of instances)



10-fold cross validation

Bootstrapping

- Alternative to cross validation
  - Sample uniformly from a dataset D with n elements n times with replacement
  - Use the n sampled elements as training set, and the elements from D that were not sampled as test set
  - Repeat the above two steps m times

- What is the expected size of the training and test set?
  - Expected fraction of instances in the test set is
$$\left(1-\frac{1}{n}\right)^n \approx \frac{1}{e} = 0.368$$
  - Expected fraction of instances in the training set is 0.632
  - Compute weighted average of the parameter values by taking the error from each of the m runs into account

Model selection through validation

- Given different prediction algorithms for the same data, which one should we select?

- There are different possibilities

  - Choose the algorithm with the lowest average error (or highest average prediction accuracy) from the N runs of cross validation

  - Choose the algorithm that minimizes the following bootstrapping error

$$\overline{err} = \frac{1}{m}\sum_{i=1}^{m} 0.632\ err_{i,test} + 0.368\ err_{i,training}$$

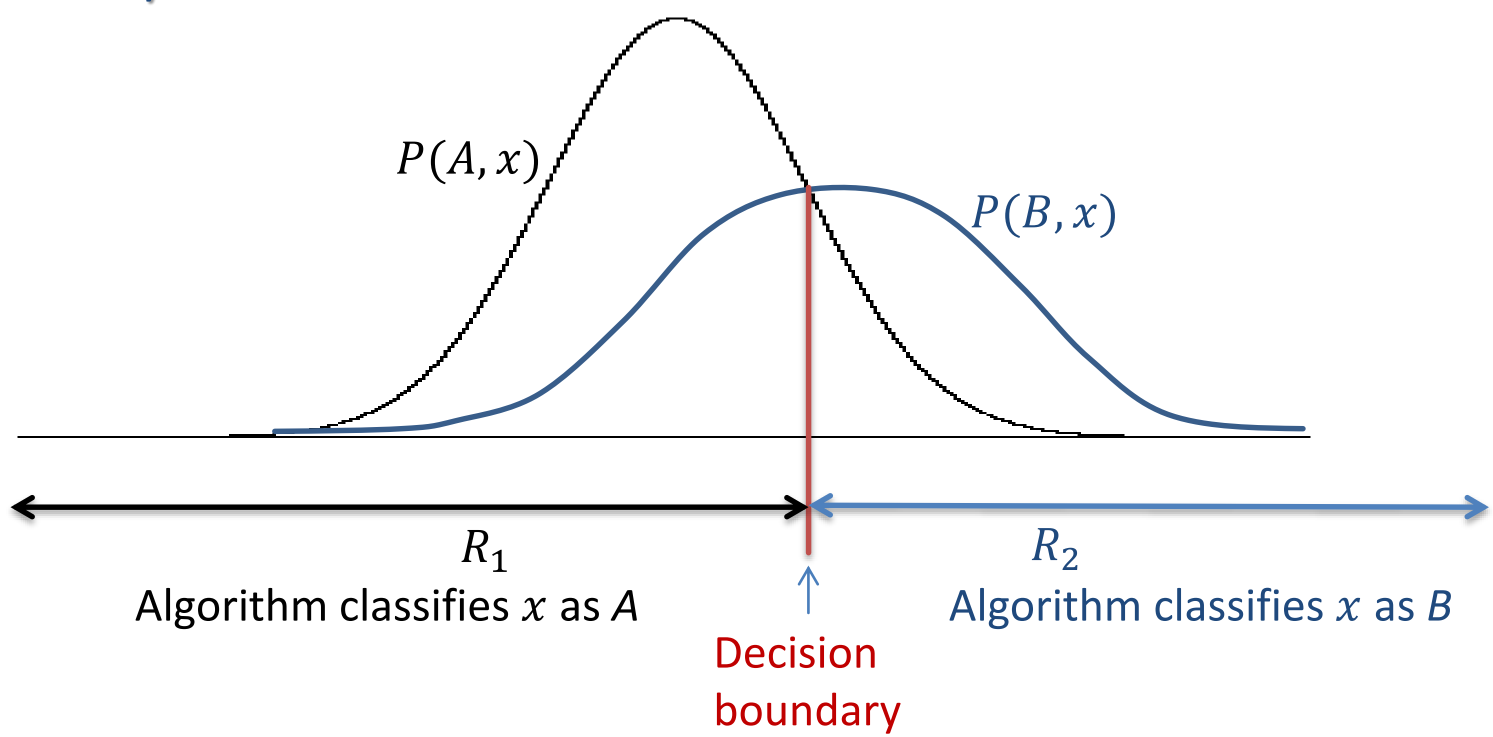
Error types

- Classification error
  - Training error: Fraction of misclassifications in training set
  - Test error: Fraction of misclassifications in the test set
- Generalized error
  - For data $\mathbf{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ and classification algorithm h the generalized error is

$$P\big(h(\mathbf{x}) \neq l(\mathbf{x})\big) = E(test\ error) = err(h)$$

Bayes error

- Probability of misclassification



$$err_{Bayes} = \int_{x \in R_1} P(B|x)P(x)\ dx + \int_{x \in R_2} P(A|x)P(x)\ dx$$

A classifier that minimizes the Bayes error is called a **Bayes optimal classifier.**

Classification loss (1)

- Consider a model for predicting whether someone has the disease or not

predicted		disease	ok
ground truth	disease	0	1000
	ok	10	0

$l_{i,j}$: loss matrix

- **Probabilistic loss function (weighted Bayes error)**
  - Let $\boldsymbol{c} = (c_1, \dots, c_k)$ be the target classes to which an instance $\mathbf{x}$ can belong with some probability
  - Expected loss is given by the weighted Bayes error:

$$\bar{L}_{Bayes} = \sum_i \sum_j \int_{\mathbf{x} \in R_j} l_{i,j} P(c_i|\mathbf{x}) P(\mathbf{x}) \; d\mathbf{x}$$

is minimized when each $\mathbf{x}$ is assigned to the class *j*

that minimizes $\sum_i l_{i,j} P(c_i|\mathbf{x})$

Classification loss (2)

- Loss functions

For instance $\mathbf{x}$, let y be the predicted and t the true class.

- 0-1 loss: $l(y,t) = [\![y \neq x]\!] = \begin{cases} 1, & y \neq t \\ 0, & y = t \end{cases}$ (indicator function)

- Absolute loss: $l(y,t) = |y - t|$

- Information loss: $l(t,\mathbf{x}) = \begin{cases} -\log P(t|\mathbf{x}), & t\ is\ true\ class\ of\ \mathbf{x} \\ -\log(1 - P(t|\mathbf{x})), & otherwise \end{cases}$

- Quadratic loss: $l(y,t) = c(y-t)^2$, for constant c

- Exponential loss: $l(y,\mathbf{x}) = \frac{1}{n}\sum_i e^{-y\ sign(\mathbf{x})}$

Model selection through hypothesis testing

- Let h and h' be two models and $err_1, \dots, err_k$ and $err_1', \dots, err_k'$ the respective error values derived from k-fold cross validation
- Are the error means any different?
- Fact: $\overline{err}$ and $\overline{err'}$ are approximately normally distributed
- $\bar{d} = \overline{err} - \overline{err'}$ is t-distributed, with k-1 degrees of freedom

$$P\left(-t_{k-1,1-\alpha/2} \leq \frac{(\bar{d}-0)\sqrt{k}}{s_d} \leq t_{k-1,1-\alpha/2}\right) = 1 - \alpha$$

- If $t_{k-1,1-\alpha/2} < \left|\frac{(\bar{d}-0)\sqrt{k}}{s_d}\right|$ reject H_0 (i.e., $\mu = \mu'$) otherwise retain it

- What if H_0 is $\bar{d} = \bar{e} - \bar{e}' \geq 0$ ($\Rightarrow H_1$ is $\bar{d} = \bar{e} - \bar{e}' < 0$) ?

→ Left-sided test: $P\left(-t_{k-1,1-\alpha} \leq \frac{(\bar{d}-0)\sqrt{k}}{s_d}\right) = 1 - \alpha$

- If $\frac{(\bar{d}-\mu_d)\sqrt{k}}{s_d} < -t_{k-1,1-\alpha}$ reject H_0 otherwise retain it

Evaluation measures: Accuracy

- Let h be a prediction model for class C

		Actual values	
		C	$\bar{C}$
Values predicted by h	C	# True Positives (TP)	# False Positives (FP)
	$\bar{C}$	# False Negatives (FN)	# True Negatives (TN)

Confusion matrix

Decision threshold

$$Accuracy(h) = \frac{TP + TN}{TP + FN + FP + TN}$$

- What does accuracy capture?
  - It captures the success rate, but does not say anything about prediction power!
- Is an accuracy of 99% good?

Evaluation measures: Precision and Recall

- Let h be a prediction model for class C

		Actual values	
		C	$\bar{C}$
Values predicted by h	C	# True Positives (TP)	# False Positives (FP)
	$\bar{C}$	# False Negatives (FN)	# True Negatives (TN)

Reflects decision threshold

$$Precision(h) = \frac{TP}{TP + FP}; \qquad Recall(h) = \frac{TP}{TP + FN}$$

- What does Precision represent?
  - Probability that h is correct whenever it predicts C

- What does Recall represent?
  - Probability that h recognizes an instance from C

Evaluation measures: Sensitivity and Specificity

- Let h be a prediction model for class C

		Actual values	
		C	$\bar{C}$
Values predicted by h	C	# True Positives (TP)	# False Positives (FP)
	$\bar{C}$	# False Negatives (FN)	# True Negatives (TN)

$$Sensitivity(h) = Recall(h) = \frac{TP}{TP + FN}; \quad Specificity(h) = \frac{TN}{TN + FP}$$

- What does Specificity represent?
  - Probability that h recognizes an instance from $\bar{C}$

HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam

Evaluation measures: F1-measure

- Let h be a prediction model for class C

		Actual values	
		C	$\bar{C}$
Values predicted by h	C	# True Positives (TP)	# False Positives (FP)
	$\bar{C}$	# False Negatives (FN)	# True Negatives (TN)

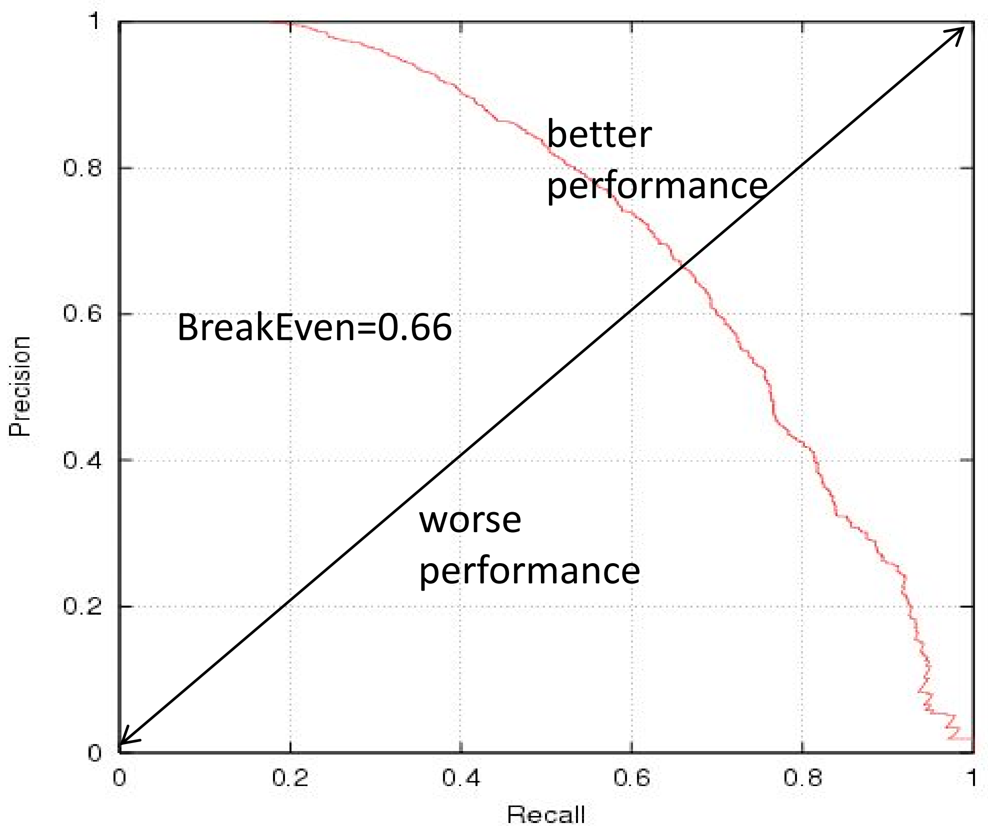
$$F_\alpha(h) = \frac{(1+\alpha^2) \cdot Precision(h) \cdot Recall(h)}{\alpha^2 \cdot Precision(h) + Recall(h)}$$

$$F_1(h) = \frac{2 \cdot (Precision(h) \cdot \ Recall(h))}{Precision(h) + Recall(h)}$$

- The F1-score is the harmonic mean between Precision and Recall

Evaluation measures: Precision-recall curve

- Let h be a prediction model for class C



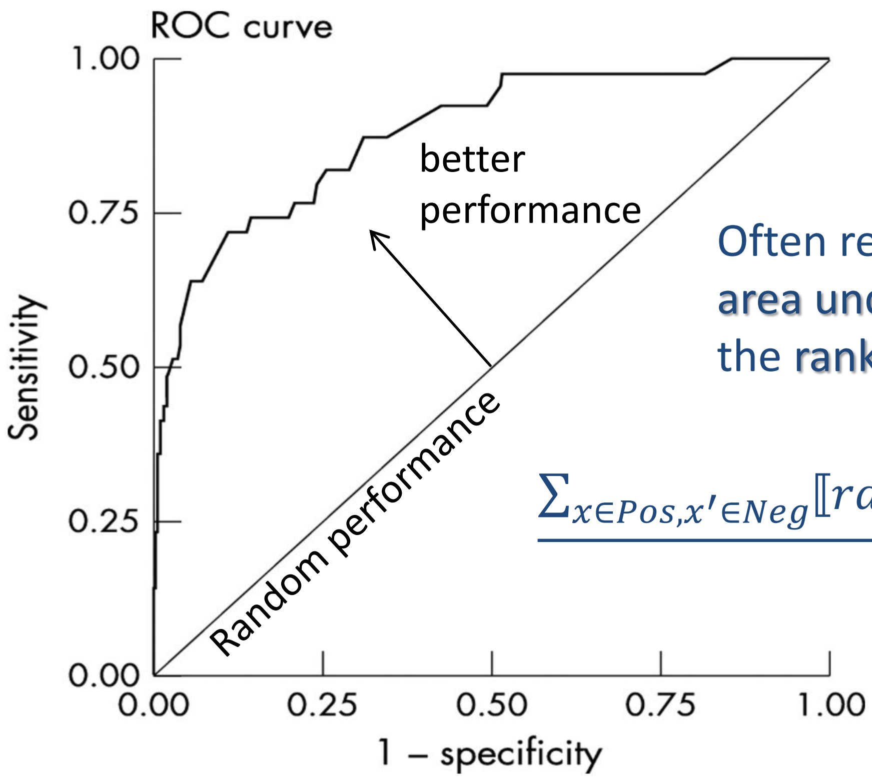
- With decreasing decision threshold, plot precision recall values
  - $BreakEven = v$ such that $Precision(h) = v = Recall(h)$

Evaluation measures: Receiver-Operating Characteristic curve

- Let h be a prediction model for class C; for decreasing decision threshold compute…

$$Sensitivity = Recall = True\ Positive\ Rate = \frac{TP}{TP + FN}$$

$$1 - Specificity = False\ Postitive\ Rate = \frac{FP}{FP + TN}$$



Often reported as performance measure: **area under the curve (AUC)**, which represents the **ranking accuracy**

$$\frac{\sum_{x \in Pos, x' \in Neg} [\![rank(x) > rank(x')]\!] + \frac{1}{2} [\![rank(x) = rank(x')]\!]}{Pos \cdot Neg}$$

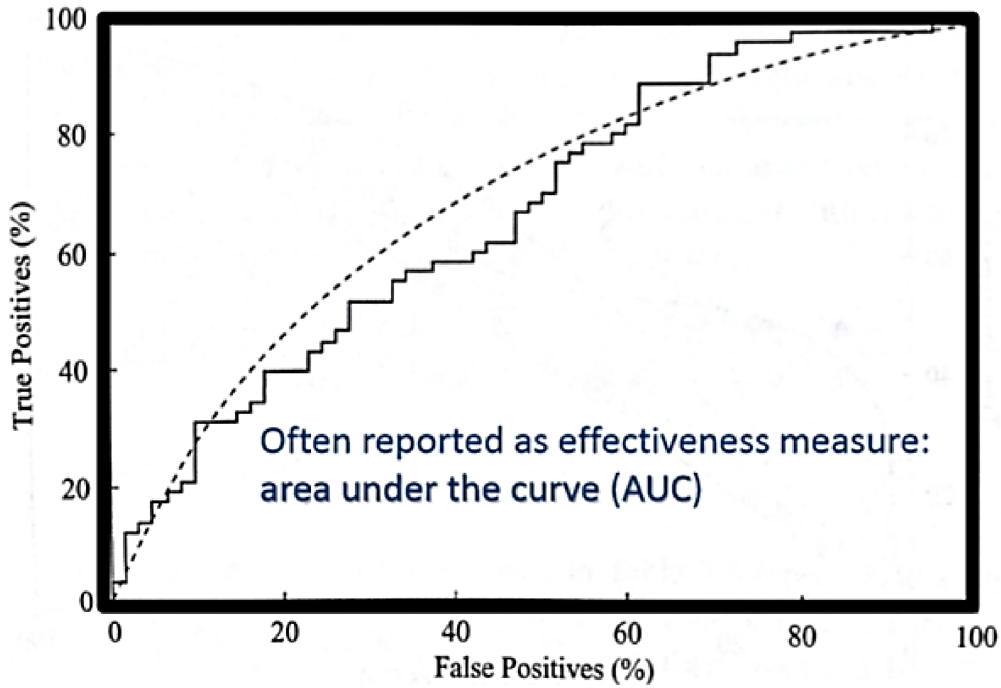
Example of ROC curve construction

- For decreasing decision threshold with respect to prediction probabilities

Test set items

Rank	Predicted	Actual Class
1	0.95	*yes*
2	0.93	*yes*
3	0.93	*no*
4	0.88	*yes*
5	0.86	*yes*
6	0.85	*yes*
7	0.82	*yes*
8	0.80	*yes*
9	0.80	*no*
10	0.79	*yes*
11	0.77	*no*
12	0.76	*yes*
13	0.73	*yes*
14	0.65	*no*
15	0.63	*yes*
16	0.58	*no*
17	0.56	*yes*
18	0.49	*no*
19	0.48	*yes*
…	…	…

Source: I.Witten, E. Frank, M. Hall: Data Mining – Practical Machine Learning Tools and Techniques



Often reported as effectiveness measure: area under the curve (AUC)

$$Sensitivity = Recall = True\ Positive\ Rate = \frac{TP}{TP + FN}$$

$$1 - Specificity = False\ Postitive\ Rate = \frac{FP}{FP + TN}$$

ROC-based calibration of decision thresholds

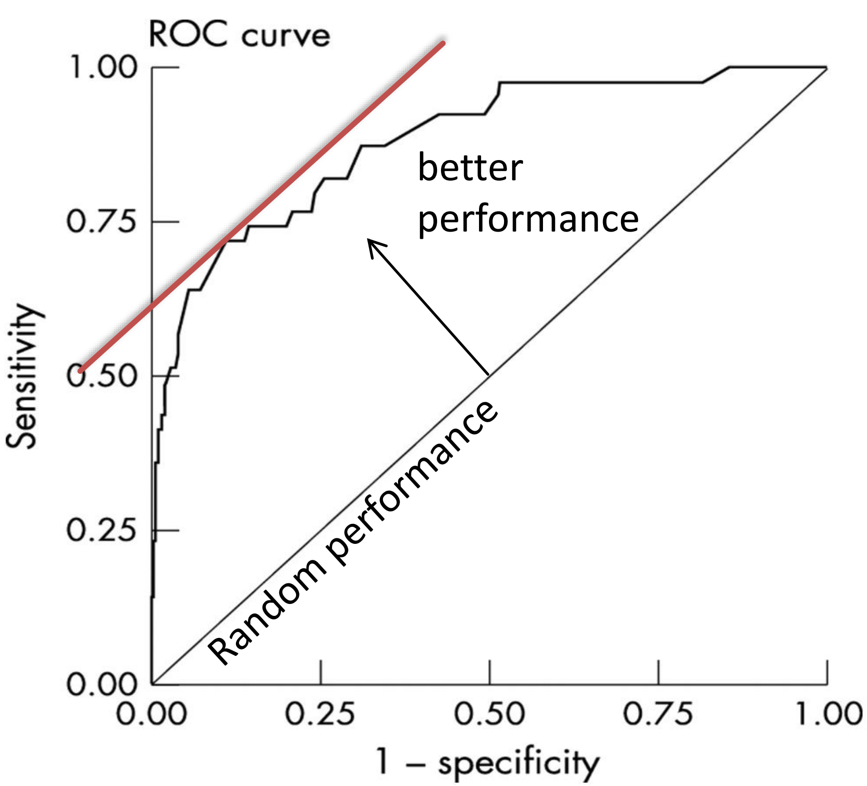
- Let h be a prediction model for class C; for decreasing decision threshold compute…

$$Sensitivity = Recall = True\ Positive\ Rate = \frac{TP}{TP + FN}$$

$$1 - Specificity = False\ Postitive\ Rate = \frac{FP}{FP + TN}$$

Uppermost point of ROC curve that touches line of slope 1 corresponds to highest average Recognition probability of positive and negative class (i.e., highest avg. Recall for both classes)

For maximum accuracy the slope has to be $\frac{|Neg|}{|Pos|}$



Evaluation measures for multi-class classification (1)

- Let h be a prediction model for classes $C_1, \ldots., C_n$

		Actual values	
		C_i	$\overline{C}_i$
Values predicted by h	C_i	# True Positives (TP_i)	# False Positives (FP_i)
	$\overline{C}_i$	# False Negatives (FN_i)	# True Negatives (TN_i)

$$Prec_{micro}(h) = \frac{\sum_i TP_i}{\sum_i TP_i + FP_i};$$

$$Prec_{macro}(h) = \frac{1}{n}\sum_i \frac{TP_i}{TP_i + FP_i}$$

$$Rec_{micro}(h) = \frac{\sum_i TP_i}{\sum_i TP_i + FN_i};$$

$$Rec_{macro}(h) = \frac{1}{n}\sum_i \frac{TP_i}{TP_i + FN_i}$$

$$F_{1micro}(h) = \frac{2 \cdot Prec_{micro} \cdot Rec_{micro}}{Prec_{micro} + Rec_{micro}};$$

$$F_{1macro}(h) = \frac{1}{n}\sum_i F_1(C_i)$$

Evaluation measures for multi-class classification (2)

- In a one-against-all fashion (i.e., class of interest is the positive class and all the other classes together are the negative class) one could analyze
  - ROC curve of for each class
  - Precision-Recall behavior of the classifier for each class
  - Accuracy evaluation for each class
- Overall performance could be reported as the weighted average of precision, recall, accuracy (i.e., weighted by the proportion of instances in each class)

Which measure to use

- For deeper analysis of thresholds or other parameters
  - ROC curves
  - Precision-recall curves
  - Error/loss curves
  - Statistical analysis

- If threshold or parameter analysis is not an issue
  - Precision
  - Recall
  - Accuracy
  - Specificity
  - Prediction error