Chapter 4 - Data Mining
A Short Introduction
Today's Question

1. Data Mining Overview
2. Association Rule Mining
3. Clustering
4. Classification

Interpretations and their Evaluation

- The "database approach"
  - consult the users in an application
  - develop a conceptual model
  - develop, implement and use the logical model
  - reconsult the user and start over

- The "data mining approach"
  - take a learning dataset
  - build a model from it
  - take a test dataset
  - compute how well the model matches

- The "information retrieval approach"
  - ask human users for the relevance of information for a profile
  - apply the retrieval algorithm to the same problem
  - compute the results: recall, precision
For establishing global models of data collections there exist two different approaches: descriptive and predictive modeling. We illustrate their difference by an example. We assume that a set of data items with two attributes a1 and a2 is given. Assume the global model we are interested in is a classification of the data items.

In descriptive modeling we just know the data items, as indicated by points in the 2-dimensional grid. A descriptive modeling technique, such as clustering, produces classes (or categories), which are not known in advance. For doing this it relies on some criteria that specify when two data items probably belong to the same class. Such a criteria is usually given as a similarity measure.

A predictive modeling technique, such as classification, starts from a given classification of the data items. From that it derives conditions on the properties of the data objects, that allow to predict the membership to a specific class. For example, the prediction could be based on a partitioning of the attribute values along each dimension, as shown in the figure to the right. There, first attribute a1 is partitioned into two intervals, and for each of the intervals a different partitioning of the attribute a2 is used to determine the regions corresponding to classes. Misclassifications may occur as seen in the example.
Both clustering and classification aim at partitioning a dataset into subsets that bear similar characteristics. Different to classification clustering does not assume any prior knowledge, which are the classes/clusters to be searched for. There exist no class label attributes, that would tell which classes exist. Thus clustering serves in particular for exploratory data analysis with little or no prior knowledge.

One important application of clustering we have in fact already introduced in information retrieval. The basic problem of information retrieval, i.e. find a set of documents matching a query, can be interpreted as a clustering problem, where the goal is to find two clusters of documents, namely the cluster of relevant ones and the cluster of non-relevant ones. In the tf-idf scheme in fact the tf-measure served to measure intra-cluster similarity for the two document clusters, whereas the idf-measure served to measure inter-cluster dissimilarity of the document clusters.

Clustering has important applications on the Web in order to extract information from large data collections, both document collections and transactional data. Clustering is also an important tool in scientific data analysis and has, for example, a long tradition in image processing and related areas. Data mining frequently adopts techniques from these areas and extends them to make them applicable for analysing large data sets.
Clustering Problem

- Given: database $D$ with $N$ $d$-dimensional data items
- Find: partitioning into $k$ clusters and noise
- A good clustering method will produce high quality clusters with
  - high \textit{intra-class} similarity
  - low \textit{inter-class} similarity
- The quality of a clustering result depends on both the similarity measure used by the method and its implementation

In its simplest formulation the clustering problem can be described in a way analogous to the vector space retrieval model. Given a database of data items that are represented by $d$-dimensional vectors (feature vectors), then partition the database into $k$ clusters.

Popular similarity measures include Euclidean distance and Manhattan distance.
Criteria for Clustering Methods

- **Quantitative Criteria**
  - Scalability: number of data objects \( N \)
  - High dimensionality

- **Qualitative criteria**
  - Ability to deal with different types of attributes
  - Discovery of clusters with arbitrary shape

- **Robustness**
  - Able to deal with noise and outliers
  - Insensitive to order of input records

- **Usage-oriented criteria**
  - Incorporation of user-specified constraints
  - Interpretability and usability

Clearly, clustering methods have to work efficiently for large datasets. Another scalability problem clustering methods have to deal with is however dimensionality: the problem is that in data sets with high dimensionality (large \( d \)) it becomes increasingly difficult to find clusters, as the occurrence of clusters is highly sensitive on the dimensions that are selected to project the data into a low-dimensional space. Without selecting specific dimensions the data would be too sparse in the high-dimensional space in order to find clusters. The figure illustrates how this problem occurs already in 2 dimensions: Only by choosing the right plane for projecting onto a single dimension we will observe a cluster. If we would project only on the x or y-axis we would not recognize any clustering effect. Thus, when projecting the choice of the subspaces used for projection is crucial. The number of choices for projection dimensions grows combinatorially.

Qualitative criteria address the ability of dealing with continuous as well as categorical attributes, and the type of clusters that can be found. Many clustering methods can detect only very simple geometrical shapes, like spheres, hyperplanes etc.

Clustering methods can be sensitive both to noisy data and the order of how the records are processed. In both cases it would be undesirable to have a dependency of the clustering result on these aspects which are unrelated to the nature of data in question.

Finally, an important criterion is the ability of how well a clustering method can incorporate user requirements both in terms of information that is provided from the user to the clustering method (in terms of constraints), which can guide the clustering process, and in terms of what information is provided from the method to the user.
Partitioning methods are a basic approach to clustering. Partitioning methods attempt to partition the data set into a given number \( k \) of clusters optimizing intra-cluster similarity and inter-cluster dissimilarity. Since an exhaustive enumeration for finding the optimal partitioning is not practical various heuristic methods have been proposed.
The k-Means Partitioning Method

• Assume objects are characterized by a d-dimensional vector

• Given $k$, the k-means algorithm is implemented in 4 steps
  - Step 1: Partition objects into $k$ nonempty subsets
  - Step 2: Compute seed points as the centroids of the clusters of the current partition. The centroid is the center (mean point) of the cluster
  - Step 3: Assign each object to the cluster with the nearest seed point
  - Step 4: Stop when no new assignment occurs, otherwise go back to Step 2

In k-Means, the centroids are computed as the arithmetic mean of the cluster all points of a cluster. The distances are computed according to a given distance measure, e.g. Euclidean distance.
In this example the k-means algorithm terminates after two iterations (the colors indicate the current clusters, the red points are the current centroids).
Properties of k-Means

- **Strengths**
  - Relatively efficient: $O(tkn)$, where $n$ is # objects, $k$ is # clusters, and $t$ is # iterations. Normally, $k, t << n$.
  - Often terminates at a local optimum, depending on seed point
  - The global optimum may be found using techniques such as: deterministic annealing and genetic algorithms

- **Weaknesses**
  - Applicable only when mean is defined, therefore not applicable to categorical data
  - Need to specify $k$, the number of clusters, in advance
  - Unable to handle noisy data and outliers
  - Not suitable to discover clusters with non-convex shapes

This assessment follows the list of criteria for evaluating clustering methods that we have introduced earlier.
4. Classification

- Data: tuples with multiple categorical and quantitative attributes and at least one categorical attribute (the class label attribute)

- Classification
  - Predicts categorical class labels
  - Classifies data (constructs a model) based on a training set and the values (class labels) in a class label attribute
  - Uses the model in classifying new data

- Prediction/Regression
  - Models continuous-valued functions, i.e., predicts unknown or missing values

- Typical Applications
  - Credit approval, target marketing, medical diagnosis, treatment effectiveness analysis

Classification creates a GLOBAL model, that is used for PREDICTING the class label of unknown data. The predicted class label is a CATEGORICAL attribute. Classification is clearly useful in many decision problems, where for a given data item a decision is to be made (which depends on the class to which the data item belongs).
Classification Process

- **Model:** describing a set of predetermined classes
  - Each tuple/sampling is assumed to belong to a predefined class based on its attribute values
  - The class is determined by the class label attribute
  - The set of tuples used for model construction: *training set*
  - The model is represented as classification rules, decision trees, or mathematical formulae

- **Model usage:** for classifying future or unknown data
  - Estimate accuracy of the model using a *test set*
  - Test set is independent of training set, otherwise over-fitting will occur
  - The known label of the test set sample is compared with the classified result from the model
  - *Accuracy rate* is the percentage of test set samples that are correctly classified by the model

In order to build a global model for classification a training set is needed from which the model can be derived. There exist many possible models for classification, which can be expressed as rules, decision trees or mathematical formulae. Once the model is built, unknown data can be classified. In order to test the quality of the model its accuracy can be tested by using a test set. If a certain set of data is available for building a classifier, normally one splits this set into a larger set, which is the training set, and a smaller set which is the test set.
In classification the classes are known and given by so-called class label attributes. For the given data collection TENURED would be the class label attribute. The goal of classification is to determine rules on the other attributes that allow to predict the class label attribute, as the one shown right on the bottom.
In order to determine the quality of the rules derived from the training set, the test set is used. We see that the classifier that has been found is correct in 75% of the cases. If rules are of sufficient quality they are used in order to classify data that has not been seen before. Since the reliability of the rule has been evaluated as 75% by testing it against the test set and assuming that the test set is a representative sample of all data, then the reliability of the rule applied to unseen data should be the same.
Criteria for Classification Methods

• Predictive accuracy

• Speed and scalability
  - time to construct the model
  - time to use the model
  - efficiency in disk-resident databases

• Robustness
  - handling noise and missing values

• Interpretability
  - understanding and insight provided by the model

• Goodness of rules
  - decision tree size
  - compactness of classification rules
In the following we will introduce a method to construct a specific kind of classification models, namely decision trees. A decision tree splits at each node the data set into smaller partitions, based on a test predicate that is applied to one of the attributes in the tuples. Each leaf of the decision tree is then associated with one specific class label.

Generally a decision tree is first constructed in a top-down manner by recursively splitting the training set using conditions on the attributes. How these conditions are found is one of the key issues of decision tree induction. After the tree construction it usually is the case that at the leaf level the granularity is too fine, i.e. many leaves represent some kind of exceptional data. Thus in a second phase such leaves are identified and eliminated.

Using the decision tree classifier is straightforward: the attribute values of an unknown sample are tested against the conditions in the tree nodes, and the class is derived from the class of the leaf node at which the sample arrives.
A standard approach to represent the classification rules is by a decision tree. In a decision tree at each level one of the existing attributes is used to partition the data set based on the attribute value. At the leaf level of the classification tree then the values of the class label attribute are found. Thus, for a given data item with unknown class label attribute, by traversing the tree from the root to the leaf its class can be determined. Note that in different branches of the tree, different attributes may be used for classification. The key problem of finding classification rules is thus to determine the attributes that are used to partition the data set at each level of the decision tree.
Algorithm for Decision Tree Construction

- **Basic algorithm for categorical attributes (greedy)**
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training samples are at the root
  - Examples are partitioned recursively based on test attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)

- **Conditions for stopping partitioning**
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning - majority voting is employed for classifying the leaf
  - There are no samples left

- **Attribute Selection Measure**
  - Information Gain (ID3/C4.5)

The basic algorithm for decision tree induction proceeds in a greedy manner. First all samples are at the root. Among the attributes one is chosen to partition the set. The criterion that is applied to select the attribute is based on measuring the information gain that can be achieved, or how much uncertainty on the classification of the samples is removed by the partitioning. Three conditions can occur such that no further splits can be performed:

1. all samples are in the same class, therefore further splitting makes no sense,
2. no attributes are left which can be used to split. Still samples from different classes can be in the leaf, then majority voting is applied.
3. no samples are left.
Which Attribute to Split?

Maximize Information Gain

Class P: buys_computer = "yes"
Class N: buys_computer = "no"

\[ I(p, n) = I(9, 5) = 0.940 \]

<table>
<thead>
<tr>
<th>age</th>
<th>p</th>
<th>n</th>
<th>I(p, n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;30</td>
<td>2</td>
<td>3</td>
<td>0.971</td>
</tr>
<tr>
<td>30-40</td>
<td>4</td>
<td>0</td>
<td>0.000</td>
</tr>
<tr>
<td>&gt;40</td>
<td>3</td>
<td>2</td>
<td>0.971</td>
</tr>
</tbody>
</table>

\[ E(age) = \frac{5}{14} I(2, 3) + \frac{5}{14} I(3, 2) = 0.69 \]

Gain(age) = I(p, n) − E(age) = 0.250
Gain(income) = 0.029
Gain(student) = 0.151
Gain(credit_rating) = 0.048

The amount of information needed to decide if an arbitrary sample in \( S \) belongs to P or N

\[ I(p, n) = \frac{p}{p+n} \log_2 \frac{p}{p+n} + \frac{n}{p+n} \log_2 \frac{n}{p+n} \]

Attribute \( A \) partitions \( S \) into \( \{S_1, S_2, ..., S_v\} \)

If \( S_j \) contains \( p_j \) examples of P and \( n_j \) examples of N, the expected information needed to classify objects in all subtrees \( S_j \) is

\[ E(A) = \sum_{i=1}^{v} \frac{p_j + n_j}{p + n} I(p_j, n_j) \]

The encoding information that would be gained by branching on \( A \)

\[ Gain(A) = I(p, n) - E(A) \]

Here we summarize the basic idea of how split attributes are found during the construction of a decision tree. It is based on an information-theoretic argument. Assuming that we have a binary category, i.e. two classes P and N into which a data collection \( S \) needs to be classified, we can compute the amount of information required to determine the class, by \( I(p, n) \), the standard entropy measure, where \( p \) and \( n \) denote the cardinalities of P and N. Given an attribute \( A \) that can be used for partitioning further the data collection in the decision tree, we can calculate the amount of information needed to classify the data after the split according to attribute \( A \) has been performed. This value is obtained by calculating \( I(p, n) \) for each of the partitions and weighting these values by the probability that a data item belongs to the respective partition. The information gained by a split then can be determined as the difference of the amount of information needed for correct classification before and after the split. Thus we calculate the reduction in uncertainty that is obtained by splitting according to attribute \( A \) and select among all possible attributes the one that leads to the highest reduction. On the left hand side we illustrate these calculations for our example.
Pruning

- Classification reflects "noise" in the data
  - Remove subtrees that are overclassifying

- Apply Principle of Minimum Description Length (MDL)
  - Find tree that encodes the training set with minimal cost
  - Total encoding cost: \( \text{cost}(M, D) \)
  - Cost of encoding data \( D \) given a model \( M \): \( \text{cost}(D | M) \)
  - Cost of encoding model \( M \): \( \text{cost}(M) \)
    \[
    \text{cost}(M, D) = \text{cost}(D | M) + \text{cost}(M)
    \]

- Measuring cost
  - For data: count misclassifications
  - For model: assume an appropriate encoding of the tree

It is important to recognize that for a test dataset a classifier may overspecialize and capture noise in the data rather than general properties. One possibility to limit overspecialization would be to stop the partitioning of tree nodes when some criteria is met (e.g. number of samples assigned to the leaf node). However, in general it is difficult to find a suitable criterion. Another alternative is to first build the fully grown classification tree, and then in a second phase prune those subtrees that do not contribute to an efficient classification scheme. Efficiency can be measured in that case as follows: if the effort in order to specify a class (the implicit description of the class extension) exceeds the effort to enumerate all class members (the explicit description of the class extension), then the subtree is overclassifying and non-optimal. This is called the principle of minimum description length. To measure the description cost a suitable metrics for the encoding cost, both for trees and data sets is required. For trees this can be done by suitably counting the various structural elements needed to encode the tree (nodes, test predicates), whereas for explicit classification, it is sufficient to count the number of misclassifications that occur in a tree node.
Extracting Classification Rules from Trees

- Represent the knowledge in the form of IF-THEN rules
  - One rule is created for each path from the root to a leaf
  - Each attribute-value pair along a path forms a conjunction
  - The leaf node holds the class prediction

- Rules are easier for humans to understand

- Example
  
  \[
  \begin{align*}
  \text{IF } \text{age} &= "\leq 30" \text{ AND } \text{student} = "\text{no}" & \text{THEN } \text{buys\_computer} = "\text{no}" \\
  \text{IF } \text{age} &= "\leq 30" \text{ AND } \text{student} = "\text{yes}" & \text{THEN } \text{buys\_computer} = "\text{yes}" \\
  \text{IF } \text{age} &= "31\ldots40" & \text{THEN } \text{buys\_computer} = "\text{yes}" \\
  \text{IF } \text{age} &= ">40" \text{ AND } \text{credit\_rating} = "\text{excellent}" & \text{THEN } \text{buys\_computer} = "\text{yes}" \\
  \text{IF } \text{age} &= ">40" \text{ AND } \text{credit\_rating} = "\text{fair}" & \text{THEN } \text{buys\_computer} = "\text{no}" \\
  \end{align*}
  \]

A decision tree can also be seen as an implicit description of classification rules. Classification rules represent the classification knowledge as IF-THEN rules and are easier to understand for human users. They can be easily extracted from the classification tree as described.
If continuous attributes occur the decision tree can be constructed as a binary decision tree, by finding an attribute value that splits the samples into 2 partitions. Consequently also categorical attributes are treated that way. In order to determine suitable split points for continuous attributes the samples need first to be sorted. On the left we see a sample database with the class label attribute Risk and a continuous attribute Age and a categorical attribute Car Type used for classification.
This example illustrates the principle of how splitting is performed both for continuous and categorical attributes.

For reasons we will discuss later we construct separate attribute lists for each attribute, that is used for classification. The attribute list contains the attribute for which it is constructed (i.e. Age and Car Type), the class label attribute and the transaction identifier tid. The attribute list is sorted for continuous attributes.

Now let us see how a split point is found for the continuous attribute Age. The distribution of the class attribute for the whole data set (4 High and 2 Low) is stored in variables C_above and a pointer is positioned on top of the attribute list. Then the pointer is moved downwards. Whenever the Age value changes the values C_below and C_above are updated (such that they always keep the distribution of H and L values above and below the pointer). Also, when the Age value changes the information gain is computed if the split were performed at that point (in the same way as done for categorical attributes before). After passing through the attribute list the optimal split value for the Age attribute is known.

For the categorical attribute we have to establish a statistics of the distribution of the classes for each of the possible attribute values and store it in a matrix. Then we check the information gain that can be obtained for each of the possible subsets of attribute values and thus determine the optimal "split" for the categorical attribute.

Finally, the attribute is chosen that results in the best (binary) split.
Scalability

- Naive implementation
  - At each step the data set is split and associated with its tree node

- Problem with naive implementation
  - For evaluating which attribute to split data needs to be sorted according to these attributes
  - Becomes dominating cost

- Idea: Presorting of data and maintaining order throughout tree construction
  - Requires separate sorted attribute tables for each attribute
  - Attribute selected for split: splitting attribute table straightforward
  - Build Hash Table associating TIDs of selected data items with partitions
  - Select data from other attribute tables by scanning and probing the hash table

If we would associate the complete table of samples with the nodes of the classification tree during its induction, we would have to re-sort the table constantly, when searching for split points for different continuous attributes. This would become for large databases the dominating cost in the algorithm. For that reason the attribute values are stored in separate tables (as we have already seen in the example before) which are sorted once at the beginning. After a split the order in the attribute tables can be maintained:

1. For the attribute table on which the split occurs the table needs just to be cut into two pieces.

2. For the other tables a scan is performed, after building a hash table of the TIDs is constructed for associating the TIDs with their partition. During the scan the hash table is probed in order to redirect the tuples to their proper partition. The order of the attribute table is maintained during that process.

Remark: this is a similar idea as was employed in the construction of inverted files!
This illustration demonstrates a split of attribute tables.
Evaluation of Classification Methods

- Models, like decision trees, extracted from data are used for prediction
  - We would like to know how good the precision is
- Different methods exist to extract models from data
  - We would like to compare them
- Basic idea
  - Separate available data into training set and test set
  - Test the model derived from training set using the test set

<table>
<thead>
<tr>
<th>NAME</th>
<th>RANK</th>
<th>YEARS</th>
<th>TENURED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tom</td>
<td>Assistant Prof</td>
<td>2</td>
<td>no</td>
</tr>
<tr>
<td>Merlisa</td>
<td>Associate Prof</td>
<td>7</td>
<td>no</td>
</tr>
<tr>
<td>George</td>
<td>Professor</td>
<td>5</td>
<td>yes</td>
</tr>
<tr>
<td>Joseph</td>
<td>Associate Prof</td>
<td>7</td>
<td>yes</td>
</tr>
</tbody>
</table>

Unseen Data: (Jeff, Professor, 4)

Tenured? YES

75% reliability!
Problems in Evaluation

- Availability of sufficient data
  - E.g. fraud diagnosis: only few frauds exist
  - Training data that needs to be manually classified
- Reliability of error estimates
  - Test data is a statistical sample
- Cost of misclassification
  - E.g. fraud diagnosis: predicting no fraud is very reliable, since frauds are rare, but not what we are interested in

- Fundamentally
  - Data mining is related to the problem of building “theory” of the data
  - Old scientific and philosophical issue
Cross-validation

- Model of splitting into training and test set too simplistic in practice
  - Test set could not be representative
  - Whole classes could be missed by the test set
- Stratification
  - Select test set as random sample, but such that each class is proportionally represented
  - "stratification holdout"
- Cross-validation
  - Split available data into k partitions
  - Each of the k partitions is once used as test set, the union of the remaining k-1 as training set
  - Average error rate over all k steps gives overall error
  - Typical k=10
  - Can be repeated for different splits
  - Partitions can be stratified
Determining the Error Rate

- Testing instances using the classifier is a Bernoulli process
  - Correct classification with probability $p$
  - Success rate $f = S/N$ random distributed variable with mean $p$ and variance $p(1-p)/N$ for $N$ trials with $S$ successful predictions
  - For large $N$ approaching normal distribution

- Determine $P[-z \leq X \leq z] = c$ for a normally distributed random variable

\[ X = \frac{f - p}{\sqrt{p(1-p)/N}} \]

- Confidence $c$, confidence range $z$
- Equating $X = z$ and $X = -z$ and resolving for $p$ gives lower and upper bounds for $p$ given observed $f$ for $N$ samples, and confidence $c$


**Considering Cost**

- **The “cow case”**
  - Predict when cows are “in heat”
  - Important for fertilization
  - Observe different body parameters
  - Initial result: never (correct 97% of the time!)
- **Solution: analyze different types of errors separately and the associate (financial) cost and benefits**

<table>
<thead>
<tr>
<th>Predicted class</th>
<th>Actual class</th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>yes</td>
<td>True positive: 30</td>
</tr>
<tr>
<td>no</td>
<td>False positive: -1</td>
</tr>
</tbody>
</table>
ARCS - Clustering Association Rules for Classification

- Association Rules, Clustering and Classification are examples of “pure” data mining techniques
- ARCS is an example that separations are not always clear-cut
- Goal: use association rules for classification
  - General form: \( A_{q1} \land A_{q2} \Rightarrow A_c \)
  - Example: \( \text{age}(X, "40..41") \land \text{income}(X, "40K .. .60K") \Rightarrow \text{buys}(X, "HDTV") \)
  - Head consists of quantitative attributes, body categorical
  - Therefore can be seen as classification rule

- Approach
  - Analyze quantitative attributes globally for possible association rules
  - Cluster the resulting rules for simplification
Binning and Rule Detection

- Define bins for quantitative attributes, e.g. equidistant
- Count number of occurrences in bins: support
- If support above threshold, test whether a rule holds: confidence
Clustering Association Rules

- Typically multiple adjacent bins lead to the same rule
  - age(X, "40..40") ∧ income(X, "40K .. .50K") ⇒ buys(X, "HDTV") etc.
- Would like to aggregate this into a single rule
  - Find (rectangular) clusters in the 2D grid: greedy algorithm
    - Step 1: enumerate all potential clusters
    - Step 2: extract clusters starting from the largest
  - Use "low pass filters" to fill holes (smoothing)
Summary

• What is the difference between clustering and classification?

• What is the difference between model construction, model test and model usage in classification?

• Which criterion is used to select an optimal attribute for partitioning a node in the decision tree?

• How are clusters characterized?

• When is the k-Means algorithm terminating?
References

- **Textbook**
  - Jiawei Han, *Data Mining: concepts and techniques*, Morgan Kaufman, 2000, ISBN 1-55860-489-8

- **Some relevant research literature**
Looking back

- **Part 1: Semi-structured data management**
  - How to model and process structured data from the Web
  - Model known, processing centralized

- **Part 2: Distributed data management**
  - How to process structured data in a physically distributed system
  - Model known, processing distributed

- **Part 3: Information retrieval and data mining**
  - How to extract models from unstructured content
  - Model unknown, processing centralized

- **Part 4? (see Web 2.0)**
The Exam

- Date and Place: **To be Announced**

- Two midterm exams and one final exam (written)
  - midterms contribute 25% each to final grade, if improvement

- Conceptual questions and practical problems
  - will assume you attended the lecture
  - will assume you did the programming exercises
  - examples from earlier years (exercises, exams) provided for preparation

- Support: Lecture Slides + Exercises + Handwritten Notes
... and finally

... thanks for the attention,
... for all questions and observations,

and good luck for the future!