Introduction to Machine Learning, Lecture 1: Decision Trees

S. Nõmm

[1]Department of Computer Science, Tallinn University of Technology

05.02.2015

Course organization

- Lectures on Thursdays 14:00-15:30
- Five (5) homeworks: Computational assignment + short write-up.
- Practical on Thursdays 16:00-17:30 sometimes we will explore some examples in detail, sometimes no organized activity (I will be around to help you with your assignments).
- Homeworks give 10% of the final grade each.
- Final exam (Written report on assign topic + short presentation) will give 50% of the final grade.
- Course web-page: `http://courses.cs.ttu.ee/pages/Machine_learning`
- Assignment submission by e-mail sven.nomm@gmail.com

What is Learning?

According to:
`http://www.merriam-webster.com/dictionary/learning`
learning noun
: the activity or process of gaining knowledge or skill by studying, practicing, being taught, or experiencing something : the activity of someone who learns

- the act or experience of one that learns
- knowledge or skill acquired by instruction or study
- modification of a behavioral tendency by experience (as exposure to conditioning)

What is *Machine Learning*?

Arthur Samuel, 1959 has defined Machine learning as: Field of study that gives computers the ability to learn without being explicitly programmed.

Tom Mitchell, 1997 has defined Machine learning as:
A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Examples

- Recognition
  - Handwriting
  - Face
  - Speech
  - Gesture
- Filtering, for example spam filtering
- Translation
- Diagnostics (Medicine)
- etc...

Goals

- Provide an overview about the main methods in modern machine learning
- Give some experience in applying these methods in practice
- Exemplify the usefulness of all the math you've learned so far
- Make you excited about this field. (Hopefully :-))

Contents

- Decision trees
- Nearest neighbours classification
- K-means classification
- Perceptron
- Neural networks
- Linear and polynomial regression
- Logistic regression
- Support vector machines
- Naïve Bayes
- Markov models
- Dimensionality reduction methods
- Reinforcement learning

Methods

- Supervised learning
- Unsupervised learning
- Reinforcement learning
- Deep learning

Terminology

- Classification *versus* regression problem? Labels are discrete (Classes) $\Rightarrow$ classification problem. Labels take continuous values $\Rightarrow$ regression problem.
- Supervised *versus* unsupervised. Labels are given $\Rightarrow$ supervised learning. No labels are given $\Rightarrow$ unsupervised learning.

Terminology

Usually one starts with model selection (some methods like neural networks may require to specify the structure of the model)

- Training - is the process of finding (computing, inferring or learning) model parameters on the basis of training data.
- Development - This step is not always needed, may be compared to a tuning.
- Evaluation - Here, the quality of the model is evaluated. The goal of this step is to describe the goodness of the model and answer the question if it suite our needs or not.

Hyperparameters

In certain cases it is necessary to include parameters which values are not determined by learning. Such parameters are called *hyperparameters*.

- Hyperparameters usually not learned during the training.
- The values of hyperparameters may be set manually or tuned on development data.

Data splitting

- *Training data* Used for training purposes only
- *Development data* If you need to further development
- *Test data* Use it to evaluate your final model. Never use this data for training or development.

There is no universal way to split available data into three parts mentioned above. An alternative is the k-fold cross validation.

Cross-validation

- Divide the training data (after removing test data) randomly into k folds.
- Run k experiments:
  - Compose the training data by concatenating k-1 folds leaving one fold out in turn
  - Train the model on those k-1 folds
  - Test it on the left-out fold
  - Record the result
- Report the average of the k experiments.

There is no universal way to split available data into three parts mentioned above. An alternative is the k-fold cross validation.

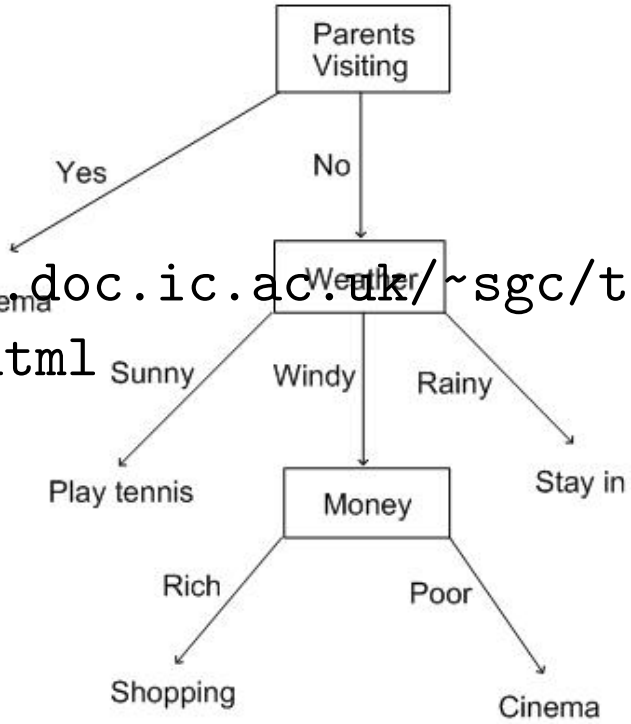
Learning: Underfitting and overfitting

- *Underfitting* the learned function is too simple In the context of human learning: underfitting similar to the case when one learns too little.
- *Overfitting* the learned function is too complex In the context of human learning: overfitting is more similar to memorizing than learning.

Decision Trees

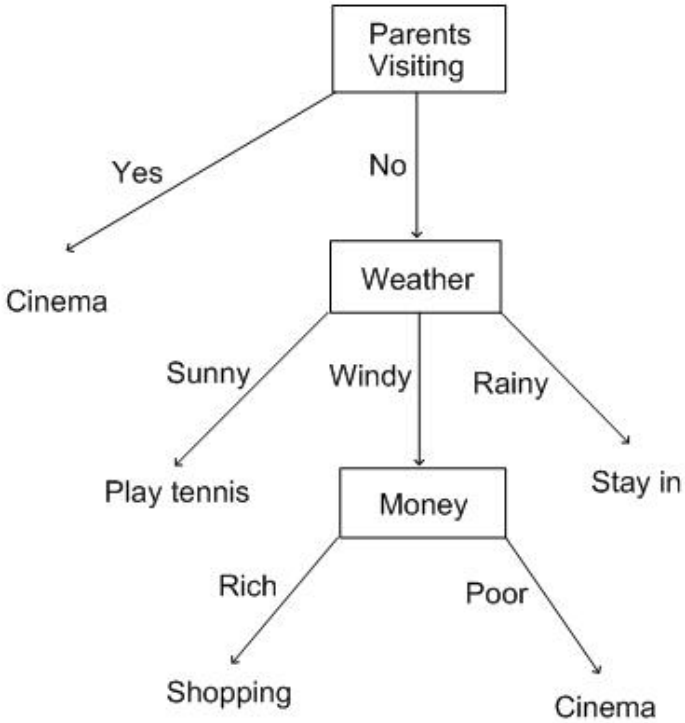
Decision Trees is a non-parametric supervised learning technique used for classification and regression. uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes and resource costs.

Example ;-)



http://www.doc.ic.ac.uk/~sgc/teaching/pre2012/v231/lecture11.html

Example



http://www.doc.ic.ac.uk/~sgc/teaching/pre2012/v231/lecture11.html

Example

- Internal tree nodes are questions.
- Leaves represent the answers - *labels* or *classes*.
- Data is usually given in tabulated form.
- Each row represents one data point or item.
- Questions in the rows are called *features* or *attributes*.
- Specific answers to each question are called *feature values*.

Parents	Weather	Money	Activity
yes	sunny	yes	cinema
no	windy	yes	shopping
no	sunny	no	tennis
yes	windy	yes	cinema
no	rainy	no	stay at home
...	...	...	...

Case of **supervised** learning (We are provided with labels). This is **classification** problem.

Decision Tree Learning

We cannot try out all possible trees most of the times.

- What questions to ask?
- In what order to ask?
- Which label to predict?

Let us assume a **binary** decision tree:

- Each question has two answers (yes or no).
- With m features, how many questions can we ask?
- How many possible orderings there are?
- What is the best ordering?
- Do we have to ask all the questions?

How?

- Start with *labelled* training data (data items with their correct answers).
- Based on *labelled* training data construct questions and the ordering.

Greedy heuristic: find the most useful feature for guessing the answer. In other words "If I could ask only one question then what would it be?"

How to greedily select the feature?

- With each feature repeat the procedure:
- Divide the data into sets based on feature values.
- Construct the histogram of answers for each set.
- Compute a score of how useful is this feature in predicting answers.

For computing the score:

- Assign to each set it's majority label based on the histogram.
- Use this assignment to label each item in the training set.
- Compute the accuracy of the labelled data.
- Choose the feature with the highest score.

Choosing features

- The first chosen feature is the root of the decision tree.
- Continue by using divide and conquer strategy: for each set obtained by choosing the feature choose the next feature from the remaining features.
- How long should we continue? When should we stop?
  - No more features left.
  - When some feature at some point in the tree classifies the data 100accurately.
  - When some other stopping criterion is met.

More complicated example: When to play tennis

Outlook	Temperature	Humidity	Wind	Play
sunny	warm	high	weak	no
sunny	warm	high	strong	no
rain	warm	high	weak	yes
rain	cool	normal	weak	yes
rain	cool	normal	strong	no
sunny	cool	normal	strong	yes
sunny	warm	high	weak	no
sunny	cool	normal	weak	yes
rain	warm	normal	weak	yes
sunny	warm	normal	strong	yes
rain	warm	high	strong	yes
sunny	warm	normal	weak	yes
rain	warm	high	strong	no

Cost function

- *Cost function* or *loss function* tells us how badly the model is working - the worse the model the higher the loss or cost.
- Learning should make the cost smaller.
- The form of the cost function depends on the model and on the problem.
- For binary classification the cost can be 1 or 0, depending whether the item was classified correctly or not.
- Previously we maximized the accuracy of the decision made by a specific feature, this is equivalent to minimizing misclassification rate.

Cost function

The goal is to choose a feature j^* for subset S of data among the set of features F:

$$j^*(S) = \arg\min \text{cost}(\{x_i, y_i : x_i \in S, x_{i,j} = c_k\}) + \text{cost}(\{x_i, y_i : x_i \in S, x_{i,j} \neq c_k\})$$

Probability that item x_i with label y_i belong to class c:

$$\hat{\pi}_c = \frac{1}{\mid S \mid} \sum_{x_i \in S} \mathbf{1}\{y_i = c\}$$

Most probable class label: $\hat{y} = \arg\max_c \hat{\pi}_c$ Misclassification rate (one possible cost function):

$$\frac{1}{\mid S \mid} \sum_{x_i \in S} \mathbf{1}\{y_i \neq \hat{y}\} = 1 - \hat{\pi}_{\hat{y}}$$

Entropy

- Entropy is an information-theoretic measure that measures uncertainty.
- The bigger the entropy the more uncertain we are.
- When we know for sure then the entropy is zero.
- When we are completely ambivalent (uniform distribution over classes) then the entropy has maximal value.
- Entropy is computed over probability distribution.
- We want to minimize entropy in order to minimize uncertainty in our decisions.

$$H(\hat{\pi}) = -\sum_{c=1}^{C} \hat{\pi}_c \log_2 \hat{\pi}_c$$

Popular cost functions

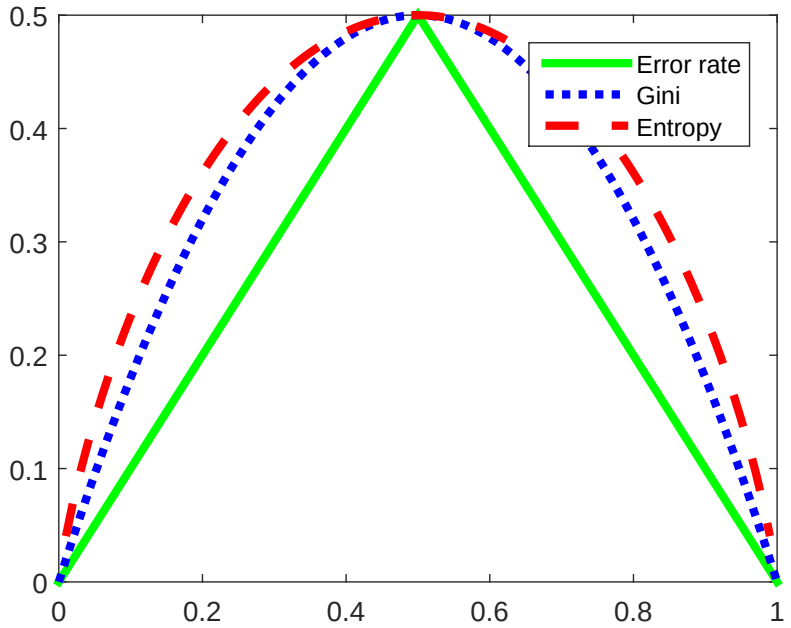
- Minimizing entropy is equivalent to maximizing information gain between test $X_j = k$ and class label Y.

$$\text{infoGain}(X_j = k, Y) = H(Y) - H(Y \mid X_j)$$

- *Gini index* is the expected error rate:

$$\sum_{c=1}^{C} \hat{\pi}_c (1 - \hat{\pi}_c) = \sum_{c=1}^{C} \hat{\pi}_c - \sum_{c=1}^{C} \hat{\pi}_c^2 = 1 - \sum_{c=1}^{C} \hat{\pi}_c^2$$

Two class case: cost functions



Pruning

- To prevent overfitting stop growing the tree when the decrease in error is below some threshold.
- Often none of the splits produces a significant reduction in error, but after several splits a substantial error reduction is found.
- Thus a more common approach is to grow a big tree:
  - full tree,
  - use a stopping criterion based on number of data points in leaves
- Then prune the tree by eliminating the branches giving the least increase in the error.

What is learning

- If a student learns the solutions of the exercises solved in class but fails to solve slightly different but analogous problems in exam, did he really learn anything?
- If the model classifies all the training examples correctly but does terribly on test data, did it really learn anything?
- Memorizing *versus* learning. **NB!!! I do expect you to learn!**
- Learning enables to generalize to new data, simply memorizing does not.
- When we overfit then we rather memorize than learn.
- When we underfit then we learn too little.

Advantages of decision trees

- Decision trees are self-explanatory, they can be converted to a set of rules also understandable to non-professionals.
- Decision trees can handle both numeric and nominal inputs.
- Decision trees make no assumptions about space distribution and classifier structure.
- Decision trees can handle missing data (some attribute values are missing).
- The representation is rich enough to represent any discrete-value classifier.

Disadvantages of decision trees

- Decision trees can be quite unstable - small variations in the data can result in a completely different tree.
- As optimal learning is intractable then heuristics (for example greedy) must be used. This means that at each node locally optimal decisions are made and this does not guarantee globally optimal result.
- There are concepts that are difficult to learn, leading to overly complex trees.