Machine Learning for NLP
Uppsala University
Department of Linguistics and Philology

About the Course
- Introduction to machine learning
- Focus on methods used in NLP
  - Decision trees and nearest neighbor methods
  - Linear models for classification and structured prediction
  - Ensemble methods
  - Unsupervised learning (clustering)
- Builds on Statistical Methods in NLP
  - Mostly discriminative methods
  - Generative probability models covered in first course

Reading List
- Main textbook:
  - Ethem Alpaydìn, Introduction to Machine Learning (2nd ed)
- Additional material:
  - Hal Daumé III, A Course in Machine Learning (draft)
    - Do not distribute chapters!
    - Do submit bug reports!
  - Papers on specific methods not covered in the book

Assignments and Examination
- Assignments:
  - Decision trees and nearest neighbor
  - Perceptron learning
  - Clustering
- Examination:
  - Written report submitted for each assignment
  - All three assignments necessary to pass the course
  - Grade determined by majority grade on assignments
Practical Organization
- Lectures in Adobe Connect:
  - Raise hand to ask questions
  - Recordings of lectures available on course home page
- Lecturers:
  - Joakim Nivre (1–4)
  - Oscar Täckström (5)
  - Magnus Rosell (6)
- Assignments:
  - No lab sessions, supervision by email
  - Reports submitted to joakim.nivre@lingfil.uu.se

Machine Learning
- Machine learning is programming computers to optimize a performance criterion for some task using example data or past experience
- Why learning?
  - No known exact method – speech recognition
  - Exact method too expensive – statistical physics
  - Task evolves over time – network routing
- Compare:
  - No need to use machine learning for computing payroll

Elements of Machine Learning
- Generalization:
  - Generalize from specific examples
  - Based on statistical inference
- Data:
  - Training data: specific examples to learn from
  - Test data: (new) specific examples to assess performance
- Models:
  - Theoretical assumptions about the task/domain
  - Parameters that can be inferred from data
- Algorithms:
  - Learning algorithm: infer model (parameters) from data
  - Inference algorithm: infer predictions from model
Types of Machine Learning

- Association
- Supervised Learning
  - Classification
  - Regression
- Unsupervised Learning
- Reinforcement Learning

Learning Associations

- Basket analysis:
  \[ P(Y | X) \] probability that somebody who buys X also buys Y where X and Y are products/services

  Example: \( P(\text{chips} | \text{beer}) = 0.7 \)

Classification

- Example: Credit scoring
- Differentiating between low-risk and high-risk customers from their income and savings

  Discriminant: IF income > \( \theta_1 \) AND savings > \( \theta_2 \)
  THEN low-risk ELSE high-risk

Classification in NLP

- Binary classification:
  - Spam filtering (spam vs. ham)
  - Spelling error detection (error vs. no error)

- Multiclass classification:
  - Text categorization (news, economy, culture, sport, ...)
  - Named entity classification (person, location, organization, ...)

- Structured prediction:
  - Part-of-speech tagging (classes = tag sequences)
  - Syntactic parsing (classes = parse trees)
Regression

- Example: Price of used car
- $x$: car attributes
  - $y$: price
  - $y = g(x | \theta)$
  - $g()$ model,
  - $\theta$ parameters

Uses of Supervised Learning

- Prediction of future cases:
  - Use the rule to predict the output for future inputs
- Knowledge extraction:
  - The rule is easy to understand
- Compression:
  - The rule is simpler than the data it explains
- Outlier detection:
  - Exceptions that are not covered by the rule, e.g., fraud

Unsupervised Learning

- Finding regularities in data
- No mapping to outputs
- Clustering:
  - Grouping similar instances
- Example applications:
  - Customer segmentation in CRM
  - Image compression: Color quantization
  - NLP: Unsupervised text categorization

Reinforcement Learning

- Learning a policy = sequence of outputs/actions
- No supervised output but delayed reward
- Example applications:
  - Game playing
  - Robot in a maze
  - NLP: Dialogue systems
Back to Classification

- Learning the class $C$ of a “family car” from examples
- Prediction: Is car $x$ a family car?
- Knowledge extraction: What do people expect from a family car?
- Output (labels): Positive (+) and negative (−) examples
- Input representation (features):
  $x_1$: price, $x_2$: engine power

Training set $X$

$X = \{x^t, r^t\}_{t=1}^N$

$r^t = \begin{cases} 
1 & \text{if } x \text{ is positive} \\
0 & \text{if } x \text{ is negative}
\end{cases}$

Empirical (training) error

$E(h|X) = \sum_{t=1}^N \mathbb{1}[h(x^t) \neq r^t]$
S, G, and the Version Space

Most specific hypothesis, S
Most general hypothesis, G

\( h \in \mathcal{H} \), between S and G is consistent \( \{E(h \mid X) = 0\} \) and make up the version space.

Margin

- Choose \( h \) with largest margin

Noise

Unwanted anomaly in data
- Imprecision in input attributes
- Errors in labeling data points
- Hidden attributes (relative to \( \mathcal{H} \))

Consequence:
- No \( h \) in \( \mathcal{H} \) may be consistent!

Noise and Model Complexity

Arguments for simpler model
- Easier to make predictions
- Easier to train (fewer parameters)
- Easier to understand
- Generalizes better (if data is noisy)
Inductive Bias

- Learning is an ill-posed problem
  - Training data is never sufficient to find a unique solution
  - There are always infinitely many consistent hypotheses
- We need an inductive bias:
  - Assumptions that entail a unique \( h \) for a training set \( X \)
    - Hypothesis class \( \mathcal{H} \) – axis-aligned rectangles
    - Learning algorithm – find consistent hypothesis with max-margin
    - Hyperparameters – trade-off between training error and margin

Generalization

- Generalization – how well a model performs on new data
  - Overfitting: \( \mathcal{H} \) more complex than \( C \)
  - Underfitting: \( \mathcal{H} \) less complex than \( C \)
  - Trade-off between three factors:
    - Complexity of \( \mathcal{H} \), \( c(\mathcal{H}) \)
    - Training set size \( N \)
    - Generalization error \( E \) on new data
- Dependencies:
  - As \( N \uparrow \), \( E \downarrow \)
  - As \( c(\mathcal{H}) \uparrow \), first \( E \downarrow \) and then \( E \uparrow \)

Model Selection

- To estimate generalization error, we need data unseen during training:
  \[
  \hat{E} = E(h \mid \mathcal{V}) = \sum_{t=1}^{N} I(h(x') \neq r') \\
  \mathcal{V} = \{x', r'\}_{t=1}^{M} \neq X
  \]
- Given models (hypotheses) \( h_1, \ldots, h_k \) induced from the training set \( X \), we can use \( E(h_i \mid \mathcal{V}) \) to select the model \( h_i \) with the smallest generalization error

Model Assessment

- To estimate the generalization error of the best model \( h_i \), we need data unseen during training and model selection
- Standard setup:
  - Training set \( X \) (50–80%)
  - Validation (development) set \( \mathcal{V} \) (10–25%)
  - Test (publication) set \( \mathcal{T} \) (10–25%)
- Note:
  - Validation data can be added to training set before testing
  - Resampling methods can be used if data is limited
Cross-Validation

- **K-fold cross-validation**: Divide \( X \) into \( X_1, \ldots, X_K \)

\[
\begin{align*}
\mathcal{V}^1 &= X_1 \\
\mathcal{V}^2 &= X_1 \cup X_2 \\
&\vdots \\
\mathcal{V}^K &= X_K \cup X_1 \cup \cdots \cup X_{K-1}
\end{align*}
\]

- **Note**:
  - Generalization error estimated by means across \( K \) folds
  - Training sets for different folds share \( K-2 \) parts
  - Separate test set must be maintained for model assessment

Bootstrapping

- Generate new training sets of size \( N \) from \( X \) by random sampling with replacement
- Use original training set as validation set ( \( \mathcal{V} = X \) )
- Probability that we do not pick an instance after \( N \) draws

\[
\left(1 - \frac{1}{N}\right)^N = e^{-1} = 0.368
\]

that is, only 36.8% of instances are new!

Measuring Error

<table>
<thead>
<tr>
<th>True Class</th>
<th>Predicted class</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>TP: True Positive</td>
</tr>
<tr>
<td>No</td>
<td>FP: False Positive</td>
</tr>
</tbody>
</table>

- Error rate = \# of errors / \# of instances = (FP+FN) / \( N \)
- Accuracy = \# of correct / \# of instances = (TP+TN) / \( N \)
- Recall = \# of found positives / \# of positives = TP / (TP+FN)
- Precision = \# of found positives / \# of found = TP / (TP+FP)

Statistical Inference

- Interval estimation to quantify the precision of our measurements

\[
m \pm 1.96 \frac{\sigma}{\sqrt{N}}
\]

- Hypothesis testing to assess whether differences between models are statistically significant

\[
\frac{(p_{11} - p_{01} - 1)^2}{\epsilon_{01} + \epsilon_{10}} \sim \chi^2
\]
Supervised Learning – Summary

- Training data + learner → hypothesis
- Learner incorporates inductive bias
- Test data + hypothesis → estimated generalization
- Test data must be unseen
- Next three lectures:
  - Different learners with different inductive biases

Anatomy of a Supervised Learner

- Model: \( g(x; \theta) \)
- Loss function: \( E(\theta | X) = \sum_{i} L(y_i, g(x_i; \theta)) \)
- Optimization procedure: \( \theta^* = \arg \min_{\theta} E(\theta | X) \)