Mini-Project Overview

Lecture : Prof. Aude Billard (aude.billard@epfl.ch)

Teaching Assistants : 
Guillaume de Chambrier, Nadia Figueroa, 
Denys Lamotte, Nicolas Sommer
Topics for Literature survey and Mini-Projects

http://lasa.epfl.ch/teaching/lectures/ML_MSc_Advanced/miniprojects.html

Topics for survey will entail:

- Data mining methods for crawling mailboxes
- Data mining methods for crawling git-hub
- Ethical issues on data mining
- Method for learning/discovering the features
- Metrics for clustering
- Classification methods for spam/no-spam
- Evaluation Metrics for Recommender Systems
- Data mining methods for financial fraud detection/prevention

Topics for mini-project will entail implementing either of these:

- Manifold learning (Isomap, LLE, SNE)
- Classification (Random Forest)
- Regression (Relevance Vector Regression)
- Non-Parametric Approximations Techniques for Mixture Models
Mini-Projects Requirements

**Coding:**
*Self-contained piece of code in:*
- Matlab
- Python
- C/C++

*Including:*
- Demo scripts
- Datasets
- Systematic assessment.

**Report:**
*Algorithm analysis, including but not limited to:*
- Number/sensitivity to hyper-parameters
- Computational costs train/test
- Growth of computation cost wrt. dataset dimension
- Sensitivity to non-uniformity/non-convexity in data.
- Precision of regression
- Benefits/disadvantages of algorithm wrt. to different types of data/applications.
- ...

Depending on the topic workload can be X% Coding Y% Analysis.

<table>
<thead>
<tr>
<th>Project Name</th>
<th>Coding</th>
<th>Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Isomap and Laplacian Eigenmaps</td>
<td>20%</td>
<td>80%</td>
</tr>
<tr>
<td>LLE, MLLE and HLLE</td>
<td>30%</td>
<td>70%</td>
</tr>
<tr>
<td>SNE and t-SNE</td>
<td>30%</td>
<td>70%</td>
</tr>
<tr>
<td>Random Forest Kernel</td>
<td>70%</td>
<td>30%</td>
</tr>
<tr>
<td>RVR vs SVR</td>
<td>30%</td>
<td>70%</td>
</tr>
<tr>
<td>GMM vs DP-GMM for GMR</td>
<td>50%</td>
<td>50%</td>
</tr>
</tbody>
</table>
## Useful ML Toolboxes

### Matlab

<table>
<thead>
<tr>
<th>Toolbox</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matlab Toolbox for Dimensionality Reduction</td>
<td><a href="https://lvdmaaten.github.io/drtoolbox/">https://lvdmaaten.github.io/drtoolbox/</a></td>
</tr>
<tr>
<td>LIBSVM</td>
<td><a href="http://www.csie.ntu.edu.tw/~cjlin/libsvm/">www.csie.ntu.edu.tw/~cjlin/libsvm/</a></td>
</tr>
<tr>
<td>GMM/GMR v2.0</td>
<td><a href="http://lasa.epfl.ch/sourcecode/?showComments=14#GMM">http://lasa.epfl.ch/sourcecode/?showComments=14#GMM</a></td>
</tr>
<tr>
<td>Gaussian Dirichlet Process Mixture Models (DPMMs)</td>
<td><a href="https://github.com/jacobbeisenstein/DPMM">https://github.com/jacobbeisenstein/DPMM</a></td>
</tr>
<tr>
<td>Dirichlet Process Mixture Modeling</td>
<td><a href="http://www.gatsby.ucl.ac.uk/~fwood/code.html">http://www.gatsby.ucl.ac.uk/~fwood/code.html</a></td>
</tr>
</tbody>
</table>

### Python

<table>
<thead>
<tr>
<th>Toolbox</th>
<th>URL</th>
</tr>
</thead>
</table>
Isomaps and Laplacian Eigenmaps

- **ISOMAP (Isometric Mapping)**: Can be viewed as an extension of multi-dimensional Scaling or Kernel PCA, as it seeks a lower-dimensional embedding which maintains geodesic distances between all points.

- **LAPLACIAN EIGENMAPS** (also known as Spectral Embedding): It finds a low dimensional representation of the data using a spectral decomposition of the graph Laplacian. The graph generated can be considered as a discrete approximation of the low dimensional manifold in the high dimensional space.
Locally Linear Embedding (LLE) and its Modified (MLLE) and Hessian (HLLE) variants

- **LLE**: LLE seeks a lower-dimensional projection of the data which preserves distances within local neighborhoods. It can be thought of as a series of local PCA which are globally compared to find the best non-linear embedding.

- **MLLE**: Solves the regularization problem of LLE by using multiple weight vectors in each neighborhood.

- **HLLE**: Solves the regularization problem of LLE by using a hessian-based quadratic form in each neighborhood.
Stochastic Neighbor Embedding (SNE) ans its t-distributed (t-SNE) variant

- **SNE:** First, SNE constructs a Gaussian distribution over pairs of high-dimensional objects. Second, SNE defines a similar probability distribution over the points in the low-dimensional map, and it minimizes the Kullback–Leibler divergence (using gradient descent) between the two distributions with respect to the locations of the points in the map.

- **t-SNE:** A variant of SNE, which represents the similarities in the high-dimensional space by Gaussian joint probabilities and the similarities in the embedded space by Student's t-distributions, making it more sensitive to local structure.
Comparison aspects

- Preservation of the geometry
- Handling holes in a dataset (non-convexity)
- Behaviour with high-curvature
- Behaviour with non-uniform sampling
- Preservation of clusters
- Algorithmic/theoretical differences
- Usefullness for different types of datasets
Toolboxes

• Matlab Toolbox :
  – Matlab Toolbox for Dimensionality Reduction

• Python Library :
  – sklearn\_bayes for Python
Random Partition Kernels based on Random Forests

- Random forests are methods consisting in averaging a multitude of tree predictors in order to decrease the variance and avoid the overfitting of traditional regression trees.

- In this method, the Random Forest algorithm is used to create a supervised kernel.

- *The random forest kernel and other kernels for big data from random partitions*; A. Davies and Z. Ghahramani (2014)
Work to be done

1. Understanding the algorithm proposed in the article

2. Coding the kernel (possibly both random forest algorithm and kernel)

3. Choose some interesting datasets

4. Compare the use of this kernel to standard ones (RBF, polynomial) for different machine learning techniques (Kernel PCA, SVM or GP for classification, …)
Toolboxes

- Random Forest Algorithm is implemented in:
  - *The Statistics and Machine Learning Toolbox of Matlab*
  - *Scikit-learn for Python*

- For the comparison:
  - *Matlab Toolbox for Dimensionality Reduction* (Kernel PCA)
  - *Scikit-learn for Python* (SVM, GP, Kernel PCA)
RVR vs SVR

• Relevance Vector Machine (RVM) is a machine learning technique that uses Bayesian inference to obtain solutions for regression and probabilistic classification.

• The RVM has an identical functional form to the support vector machine, but provides probabilistic classification.

• *Sparse Bayesian learning and the relevance vector machine* ; Tipping, M. E. ; *Journal of Machine Learning Research* 1, 211-244 (2001)
Perspectives of comparison for different datasets

- Computational cost for training and testing
- Precision of the regression
- Evolution with the size of the dataset
- Memory cost
- Choice of hyperparameters
- Choice of Kernel
- …
Toolboxes

• Support Vector Machine for regression in:
  – *The Statistics and Machine Learning Toolbox of Matlab*
  – *Scikit-learn for Python*

• Relevance Vector Machine for regression in:
  – *Matlab SparseBayes*
  – *sklearn_bayes for Python*
GMM vs DP-GMM for regression

• Gaussian Mixture Model (GMM) : Parametric approach to learn GMM consists in fitting several models with parametrizations via the EM algorithm and use model selection approaches, like Bayesian Information Criterion, to find the best model.

• Dirichlet Process – GMM : DP is a stochastic process which produces a probability distribution whose domain is itself a probability distribution. It enables to add a prior on the number of models in the mixture.
Perspectives of comparison

• Computational cost for training

• Advantage of automatic determination of parameter vs cross-validation

• Sensitivity to hyper-parameters
Toolboxes

• GMM for regression in:
  – *GMM/GMR v2.0 for Matlab*
  – *Scikit-learn for Python*

• DP-GMM in:
  – *Gaussian Dirichlet Process Mixture Models for Matlab*
  – *bnpy for Python*