Advanced Machine Learning
Mini-Projects

Professor: Aude Billard
Assistants: Nadia Figueroa, Ilaria Lauzana and Brice Platerrier
E-mails: aude.billard@epfl.ch, nadia.figueroafernandez@epfl.ch
ilaria.lauzana@epfl.ch, brice.platerrier@epfl.ch

Spring Semester 2017

Grade weight of the miniproject

The mini-project accounts for **50%** of the grade and entails implementing an algorithm of choice and one or more variants from the following list, evaluating the algorithm's performance and sensibility to parameter choices.

Team formation and report structure

Mini-project:

- Can form groups of up to 3 students. This means team of 2 and 3 are okay.
- It is also ok if you do the coding project individually.
- Teams can potentially take the same coding project (max 2 teams sharing the same project).

Implementation: It should be handed in as a self-contained piece of **code** (coding language - matlab, python, C/C++) with corresponding demo scripts which analyze the use of the algorithm (selection of a dataset, systematic assessment of strength/weaknesses taking one algorithm as example, e.g. SVM or Boosting).

Report: The team should write a **report** (maximum 8 pages double column format). The results section should discuss important items, such as computation costs at training versus testing, growth of computational costs with dimension of dataset, number of hyper-parameters and how sensitive (if at all) the algorithm is to the choice of these hyper-parameters.

Useful ML Toolboxes

Matlab

Toolbox	URL
Matlab Toolbox for Dimensionality Reduction	`https://lvdmaaten.github.io/drtoolbox/`
Statistics and Machine Learning Toolbox	`http://fr.mathworks.com/help/stats/index.html`
Least Squares - Support Vector Machine	`http://www.esat.kuleuven.be/sista/lssvmlab/`
Matlab Sparse Bayes	`http://www.miketipping.com/sparsebayes.htm`
LIBSVM	`www.csie.ntu.edu.tw/~cjlin/libsvm/`
GMM/GMR v2.0	`http://lasa.epfl.ch/sourcecode/?showComments=14#GMM`
Probabilistic Modeling Toolkit for Matlab/Octave	`https://github.com/probml/pmtk3`
Gaussian Dirichlet Process Mixture Models (DPMMs)	`https://github.com/jacobeisenstein/DPMM`
Dirichlet Process Mixture Modeling	`http://www.gatsby.ucl.ac.uk/~fwood/code.html`

Python

Toolbox	URL
`scikit-learn.` Machine Learning in Python	`http://scikit-learn.org/stable/`
`bnpy.` Bayesian NonParametric Machine Learning for Python	`https://bitbucket.org/michaelchughes/bnpy/`

ML Algorithms

Manifold Learning/Non-linear Dimensionality Reduction

The assessment for these topics should also answer the following questions, with a corresponding theoretical justification and graphical representation (i.e. self-generated datasets):

- How is the expected underlying *manifold geometry* preserved?
- Can the methods handle holes in the dataset? *(i.e. non-convexity)*
- How do these methods behave in datasets with high *curvature*?
- Can the methods handle changes from dense to sparse regions? *(i.e. non-uniform sampling)*
- Does the mapping preserve clusters in the lower dimensional space?
- What type of data are these methods most useful for?
- What is the main algorithmic/theoretical difference between these algorithms?

Isomaps and Laplacian Eigenmaps

The goal of this mini-project is to apply and evaluate Isomaps and Laplacian Eigenmaps on self-generated sets of data. Following are the details of each algorithm (`both implemented in drtoolbox and scikit-learn`):

- ISOMAP (Isometric Mapping): Can be viewed as en extension of Multi-dimensional Scaling or Kernel PCA, as it seeks a lower-dimensional embedding which maintains geodesic distances between all points.
  A global geometric framework for nonlinear dimensionality reduction; Tenenbaum, J.B.; De Silva, V.; & Langford, J.C. Science 290 (5500)

- 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.
  Laplacian Eigenmaps for Dimensionality Reduction and Data Representation; M. Belkin, P. Niyogi, Neural Computation, June 2003; 15 (6):1373-1396

Locally Linear Embedding (LLE) and its variants MLLE and HLLE

The goal of this mini-project is to apply and evaluate LLE and its variants MLLE and HLLE on different sets of data. `all implemented in drtoolbox and scikit-learn)`

- LLE (Locally Linear Embedding): Locally linear embedding (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 Principal Component Analyses which are globally compared to find the best non-linear embedding.
  Nonlinear dimensionality reduction by locally linear embedding; Roweis, S. & Saul, L. Science 290:2323 (2000)

  The LLE algorithm has a major problem when the number of neighbors is greater than the number of input dimensions, as the matrix defining each local neighborhood becomes rank-deficient. LLE solves this by using an arbitrary regularization parameter, which may or may not yield an optimal solution. The following approaches provide variants for addressing the regularization problem.

- MLLE (Modified Locally Linear Embedding): Solves the regularization problem of LLE by using multiple weight vectors in each neighborhood.
  MLLE: Modified Locally Linear Embedding Using Multiple Weights; Zhang, Z. & Wang, J.

- HLLE (Hessian Locally Linear Embedding): Solves the regularization problem of LLE by using a hessian-based quadratic form in each neighborhood.
  Hessian Eigenmaps: Locally linear embedding techniques for high-dimensional data; Donoho, D. & Grimes, C. Proc Natl Acad Sci USA. 100:5591 (2003)

Stochastic Neighbor Embedding (SNE) and variant t-SNE

The goal of this mini-project is to apply and evaluate SNE and t-SNE on self-generated sets of data. Following are the details of each algorithm (`both implemented in drtoolbox and scikit-learn`):

- SNE (Stochastic Neighbor Embedding): SNE is a method for mapping high-dimensional points into a low-dimensional space based on stochastic selection of similar neighbors (using similarities or affinities between points). 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 KullbackLeibler divergence (using gradient descent) between the two distributions with respect to the locations of the points in the map. The aim of the embedding is to approximate this distribution as well as possible when the same operation is performed on the low-dimensional images of the objects.
  Stochastic Neighbor Embedding; Hinton, G. E. and Roweis, S. T.; Advances in Neural Information Processing Systems 15 (2003)
- t-SNE (t-distributed Stochastic Neighbor Embedding): A variant of SNE, which represents the similarities in the high-dimensional space by Gaussian joint probabilities and the similarites in the embedded space by Student's t-distributions, making it more sensitive to local structure.
  Visualizing High-Dimensional Data Using t-SNE; van der Maaten, L.J.P.; Hinton, G. Journal of Machine Learning Research (2008)

In addition to answering the general assessment questions for these topics the team should generate or test high-dimensional datasets such as the ones in `Examples:https://lvdmaaten.github.io/tsne/#implementations` and apply standard clustering or classification algorithms of their choosing and evaluate their performance with F-measure, BIC, AIC, Precision, Recall, etc.

Non-Linear Regression

Relevance Vector Machine for regression (RVR)

The goal of this mini-project is to apply RVR and to compare it to SVR with some datasets you will have chosen.
`(both implemented in Matlab SparseBayes (RVR) and LIBSVM (SVR))`
The analysis of the results can focus on these following points and more :

- Computational cost for training and testing
- Precision of both regression

- Evolution with the size of the dataset
- Use of memory
- Choice of hyper-parameters
- ...

Additional information about RVR and algorithm to implement can be found in these papers :

- *Sparse Bayesian learning and the relevance vector machine ; Tipping, M. E. ; Journal of Machine Learning Research 1, 211-244 (2001)*
- *Fast marginal likelihood maximisation for sparse Bayesian models ; Tipping, M. E. and A. C. Faul ; In C. M. Bishop and B. J. Frey (Eds.), Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, Key West, FL, Jan 3-6 (2003)*

Parametric vs Non-Parametric Mixture Model Learning for Regression

The parametric approach to learning Gaussian Mixture Models is to fit several models with different parametrizations (i.e. number of clusters) via the EM algorithm and use model selection approaches (such as evaluating with BIC) to find the model with the best fit. An alternative approach is to use Bayesian Non-parametrics, which impose a prior distribution on clusterings with an infinite, unbounded, number of partitions, such as the Dirichlet Process - Gaussian Mixture Model (DP-GMM). Learning for this model can be achieved through variational inference or sampling methods like Markov Chain Monte Carlo. Both algorithms for learning are implemented in `bnpy` and `scikit-learn`. `GMM/GMR` learned through EM is implemented in Matlab and some not-so-well documented implementations of the DP-GMM can be found as well.

The goal of this project is to apply both algorithms for learning and implement and compare their performance on a regression task on different datasets. The assessment should answer the following questions:

- Other than the advantage that DP-GMR allows for automatic determination of the proper model size (while existing methods require cross-validation), what are the benefits of this Bayesian approach?
- Is the difference in computational cost significant?
- Sensitivity to hyper-parameters?

Additional information about GMR and DP-GMR can be found in the following papers:

- *Gaussian mixture models and the EM algorithm*
  `http://people.csail.mit.edu/rameshvs/content/gmm-em.pdf`
- *The Infinite Gaussian Mixture Model; Rasmussen, C.E; in Advances in Neural Information Processing Systems 12*
- *A nonparametric Bayesian approach toward robot learning by demonstration; Sotirios P. Chatzis, Dimitrios Korkinof, Yiannis Demiris, Robotics and Autonomous Systems, Volume 60, Issue 6, June 2012, Pages 789-802*

Mini-Project Work Load Decomposition

In the following table we provide a decomposition of the expected workload for each mini-project depending on the complexity of the algorithms and the available implementations. Please take this into account before choosing your mini-project.

Project Name	Coding	Analysis
Isomap and Laplacian Eigenmaps	20%	80%
LLE, MLLE and HLLE	30%	70%
SNE and t-SNE	30%	70%
RVR vs SVR	30%	70%
GMM vs DP-GMM for GMR	50%	50%