Machine Learning
Exercise 10

Marc Toussaint
Machine Learning & Robotics lab, U Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany

July 2, 2015

1 Boostrapping mini example

Numerically, generate 100 random samples $D = \{x_i\}$, $x_i \sim \mathcal{U}(0,1)$ from the uniform distribution over $[0,1]$. Compute the empirical mean $\widehat{\mu}$ and standard deviation σ_x of those.

Generate 100 bootstrap samples D_j of the data set D. For each D_j compute the mean μ_j. Compute the mean of all means μ_j as well as their standard deviation σ_μ. Does it "match" as expected with σ_x?

2 scikit-learn or Weka or MATLAB Stats Toolbox

We're going to explore some of the "breadth of ML" with existing ML toolkits.

scikit-learn http://scikit-learn.org/ is a ML toolkit in python. It is very powerful, implements a large number of algorithms, has a nice interface, and a good documentation.

Weka http://www.cs.waikato.ac.nz/ml/weka/ is another widely used ML toolkit. It is written in Java and offers a GUI where you can design your ML pipeline.

MATLAB Stats Toolbox http://de.mathworks.com/help/stats/index.html provides some ready-to-use algorithms.

Here are the instuctions for sklearn, but you can also do the same with MATLAB/Weka[1].

a) Install sk-learn. There is an install guide http://scikit-learn.org/stable/install.html and Ubuntu packages exist.

b) Read the tutorial: http://scikit-learn.org/stable/tutorial/basic/tutorial.html

c) Download the "MNIST dataset" via sklearn.
http://scikit-learn.org/stable/datasets/index.html#downloading-datasets-from-the-mldata-org-repository
(For MATLAB you find a description here http://ufldl.stanford.edu/wiki/index.php/Using_the_MNIST_Dataset of how to load the data.)

d) Try to find the best classifier for the data [2]. and compare the performance of different classifiers Try to cover some of the "breadth of ML". Definitely include the following classifiers: LogisticRegression, kNN, SVM, Ensemble methods (DecissionTrees, GradientBoosting, etc.).

d) Try out the same after transforming the data (PCA, KernelPCA, Non-Negative matrix factorization, Locally Linear Embedding, etc.).

- http://scikit-learn.org/stable/modules/classes.html#module-sklearn.decomposition
- http://scikit-learn.org/stable/modules/classes.html#module-sklearn.manifold

e) Many classifiers have parameters such as the regularization parameters or kernel parameters. Use GridSearch to find good parameters. http://scikit-learn.org/stable/modules/classes.html#module-sklearn.grid_search

Here is a little example of loading data, splitting the data into a test and train dataset, training a SVM, and predicting with the SVM in python:

[1] However, in the previous year most students did not enjoy Weka.

[2] The dataset is **big**. If you have problems running your learning algorithms on your machine, just select a subset of the whole data, e.g. 10%

```
from sklearn import svm
from sklearn import datasets
from sklearn.cross_validation import train_test_split

iris = datasets.load_iris()
X, y = iris.data, iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33)
clf = svm.SVC()
clf.fit(X_train, y_train)
clf.predict(X_test)
```

Here is an example in MATLAB: (see http://de.mathworks.com/help/stats/svmclassify.html)

```
load fisheriris
xdata = meas(51:end,3:4);
group = species(51:end);
figure;
svmStruct = svmtrain(xdata,group,'ShowPlot',true);
Xnew = [5 2; 4 1.5];
species = svmclassify(svmStruct,Xnew,'ShowPlot',true)
```