# CS273a Project Description

Intro to Machine Learning: Fall 2013

**Due: Friday December 13th, 2012**

For your course project, you will explore data mining and prediction "in the wild", in a real life data set and compared against the performance of teams from around the world.

**Data:** We will use a data set from a past Knowledge Discovery in Data (KDD) Cup, a yearly competition in machine learning and data mining associated with the KDD conference. In particular, we will use the 2004 Competition's Particle Physics data set.

The challenge is described in full on the webpage:
`http://osmot.cs.cornell.edu/kddcup/`

From the webpage:
*The goal in this task is to learn a classification rule that differentiates between two types of particles generated in high energy collider experiments. It is a binary classification problem with 78 attributes. The training set has 50,000 examples, and the test set is of size 100,000.*

Although the original challenge was to compete with respect to four metrics (accuracy, ROC, cross-entropy, and "q-score"), you may pick any one (or more than one) to compete with.

## Step 1: Form teams.

Please form teams of 2-3 students who share your interests and with whom you will directly collaborate. Piazza can help with locating other students in need of joining. Please send me your team grouping and name no later than Tuesday, November 19th.

There is no barrier to collaboration on this project, so feel free to talk to other teams about what they are doing, how they are doing it, and how well it works (and you can see this last point on the leaderboard as well). You are also free to use online code, or other sources. However, please make sure that your own entries are developed by your team members – the purpose of this project is to give you practical experience, so it will not be helpful if you just follow instructions from a friend.

## Step 2: Download the data.

To download the data, you must be registered on the site; see:
`http://osmot.cs.cornell.edu/kddcup/datasets.html`
To load the data directly into Matlab, you will have to replace the '?' in the test set files with some numeric value.

Note that some data have missing features, denoted with a 999 or 9999 value (see webpage and below). You may remove these features, ignore them, set them to zero, set them to the mean or median of the non-missing values, or impute them in any other way that you choose.

## Step 3: Build your learners

Use the techniques we have developed so far in class to construct predictive models for your target(s). You may use the Matlab code provided from class; online code packages such as Weka, PMTK, or others; or write your own.

I suggest that you try several different models, such as nearest neighbor methods, decision trees, linear classifiers (logistic regression, support vector machines, etc.), naive Bayes classifiers, and/or boosted classifiers (decision stumps, etc.). Each member of your team may try one or two models, and can explore fitting them to the data and assessing their performance using validation or cross-validation.

After learning several models, you may also want to combine them using bagging or by learning a predictor from the classifier outputs ("stacking").

**Some areas for possible exploration:**

(a) ROC scoring: One official scoring works by computing an ROC curve. While any binary predictor can be converted into a trivial ROC curve, this may not be the most advantageous approach. "Soft" scores (for example, unthresholded predictor values) will be converted into an ROC in a more fine-grain and possibly better way.

(b) Q-scoring: Q-scoring is some non-standard score that the physics domain experts care about. You could redesign an algorithm to explicitly attempt to optimize Q-scoring.

(c) Missing data: some features have missing values. These can be dealt with by simply ignoring those features; setting their values to zero; setting their values to the median or mean of the data with non-missing values; building an unsupervised model to impute their values; or many other approaches. (My code `imputeMissing.m` may be helpful; it replaces `nan` entries with a mean, median, or a simple Gaussian model imputation.)

(d) Feature selection or creation: the data have about 78 features. You may want to use feature selection to determine which features are useful, and reduce the number of features to avoid overfitting; on the other hand, you may want to generate new features if you feel that a model is actually under-fitting. See my functions like `fProject` and `fKitchenSink` for ideas.

## Step 4: Evaluate; go to Step 3.

Output your predictions to a set of files as described in the instructions page:
`http://osmot.cs.cornell.edu/kddcup/submission.html`

You can check the leaderboard to see your "test" performance:
`http://osmot.cs.cornell.edu/cgi-bin/newtable.pl?prob=phy`

Please limit yourself to at most a few (1–2) submissions per day (unless debugging the submission process itself) to avoid over-loading their servers and to enforce good practices.

## Step 5: Write it up

Your team will produce a single write-up document, approximately **6 pages** long, describing the problem you chose to tackle (which loss, etc.) and the methods you used to address it, including which model(s) you tried, how you trained them, how you selected any parameters they might require, and how they performed in on the test data. Consider including tables of performance of different approaches, or plots of performance used to perform model selection (i.e., parameters that control complexity).

Within your document, please try to describe to the best of your ability who was responsible for which aspects (which learners, etc.), and how the team as a whole put the ideas together.

You are free to collaborate with other teams, *including* sharing ideas and even code, but please document where your predictions came from. (This also relaxes the proscription from posting code or asking for code help on Piazza, at least for project purposes.) For example, for any code you use, please say in your report who wrote the code and how it was applied (who determined the parameter settings and how, etc.) Collaboration is particularly true for learning ensembles of predictors: your teams may each supply a set of predictors, and then collaborate to learn an ensemble from the set.

## Requirements

I am looking for several elements to be present in any good project. These are:

(a) Exploration of at least one or two techniques that we did not spend significant time on in class. For example, using neural networks, support vector machines, or random forests are great ideas; if you do this, explore in some depth the various options available to you for parameterizing the model, controlling complexity, etc. Other options might include feature design, or optimizing your models to provide good ROC behavior. Your report should describe what aspects you chose to focus on.

(b) Performance validation. You should practice good form and use validation or cross-validation to assess your models' performance, do model selection, combine models, etc. You should *not* simply upload hundreds of different predictors to the website to see how they do. Think of the website as "test" performance – in practice, you would only be able to measure this once you go live.

(c) Adaptation to under- and over-fitting. Machine learning is not very "one size fits all" – it is impossible to know for sure what model to choose, what features to give it, or how to set the parameters until you see how it does on the data. Therefore, much of machine learning revolves around assessing performance (e.g., is my poor performance due to underfitting, or overfitting?) and deciding how to modify your techniques in response. Your report should describe how, during your process, you decided how to adapt your models and why.