# Machine Learning 2013: Project 1 - Regression Report

altrif@student.ethz.ch
bvancea@student.ethz.ch
draganm@student.ethz.ch

November 1, 2013

## Experimental Protocol

## 1 Tools

For the project we used to following tools:

- The Python programming language
- The Scikit-learn Python library. We used this library for preprocessing data and prediction.
- The Pandas Python library. We used this library for parsing csv files and data management.
- The py-earth Python libary. We used this library for a open-source MARS implementation.

## 2 Algorithm

We tried several models for the regression task, however the model that performed the best was a MARS (Multivariate Adaptive Regression Splines) model. This type of model was not described in the lectures and we will briefly describe the approach used. The algorithm constructs a piecewise-linear model of the form:

$$\hat{f}(x) = \sum_{i=1}^{k} c_i B_i(x) \tag{1}$$

In this model, $c_i$ represent constants and $B_i$ represent Hinge functions of the form $max(0, x_i - c)$ or $max(c - x_i, 0)$. The pair of functions presented is called a *reflected pair*. It is possible to model a basis function as a product of two basis functions and therefore increasing the *degree of interaction* in the model. The MARS algorithm consistens of two steps:

**The forward pass** Starting from a single constant function, the algorithm iteratively adds the pair of Hinge functions that decreases the sum-of-squares error.

**Backward pass** Due to the fact that the **forward pass** uses a *greedy* procedure, this pass prunes the model of the least-effective terms (according to the GCV (Generalized Cross-Validation) criterion).

Because of the nature of the algorithm used, MARS provides both regularization and feature selection.

## 3 Features

Our methodology for contructing new features is the following:

1. We constructed the correlation matrix for the training set. This includes both the feature vectors X and the vector of labels y.

2. We applied several transformations to those features which have the strongest correlation with the y vector. More precisely, for all the features for which the value of the correlation was greater than an $\epsilon$, we added new features consiting of $sin(x)$, $log(x)$, $x^2$ etc.

3. We also took in consideration that features could influence one another. Because of this, we also added new features corresponding to combinations of two highly correlated features. Therefore, for each pair of strongly corelated features $x_1$, $x_2$, we added a new corresponding features: $sin(x_1) + sin(x_2)$.

## 4 Parameters

For the MARS model, the parameter we needed to choose was the maximum degree of interaction between the basis functions generated by the model. We obtained the best degree through k-Fold cross-validation over a range of possible degrees. We repeated the procedures for several values of k: 2,3,5,10 and picked the best performing model. In our experiments, a MARS model with the maximum degree of 2 performed the best in all the cases.

## 5 Lessons Learned

We also tried to train several other regression models:

- A simple least-squares linear regressor. This model performed the worst of all the models we tried. The main reason for this is that, even if we added new features, we did not obtain a linear relationship between features and labels.

- A $L_2$ regularized least-squares regression (Ridge). This model performed slighly better than the least-squares model, however it still did not go over the hard baseline. We tried several polynomial degrees and several values of $\alpha$ using cross-validation, but we think that even with the regularization, we would not obtain a linear relationship.

- A $L_1$ regularized least-square regression (Lasso). Our main assumption was that we might not obtain a robust model due to the fact that we may have added more features than necessary. We attempted to regularize using the $L_1$ norm, hoping to obtain a sparser model containing only the most relevant features.