Exercises for Tik-61.181 autumn 1999

Juha Vesanto and Esa Alhoniemi

December 1, 1999

General

This document describes the home exercises and the practical assignement for the course Tik-61.181 autumn 1999: post-graduate seminar on data preparation. The exercises are a mixture of theoretical proves and hand-on experimentation with given data sets. For the former, a right answer/proof (or several of them) exists. For the latter, right answers typically do not exist.

The point of the latter exercises is to get an intuitive idea of how the preparation and data mining techniques work in practice. You are supposed to use Matlab to solve them, but Matlab is not a requirement. If you want to use some other program(s), you can ask the assistents to provide the data sets in ASCII format rather than Matlab binaries.

Files This document can be found from
`http://www.cis.hut.fi/cis/edu/Tik-61.181/s99/exercises.ps`

All associated files can be download from page
`http://www.cis.hut.fi/cis/edu/Tik-61.181/s99/exercises.html`

Or directly from the zipped file
`http://www.cis.hut.fi/cis/edu/Tik-61.181/s99/exercises.zip` which includes the following files:

```
This PostScript document:              exercises.ps
Data binary file for Matlab 5:         ex_data.mat
Data binary file for Matlab 4:         ex_data4.mat
M-files needed in the exercises:       confidence.m, pca_plot.m
ASCII/XLS versions of some of the data: cars.txt, cars.xls
                                       heartT.txt, heartE.txt
Detailed description of heart data:    heart_problem.txt
```

In particular, all needed data are in Matlab binary files (`ex_data.mat` and `ex_data4.mat`). You can load them into Matlab using `load` command. The specific matrices/vectors/structs needed in each exercise are listed in the description of the exercise.

Requirements and deadlines To pass the course, you have to do at least 9 of the exercises. To pass it with distinction, you have to do at least 14 of the exercises (and yes, exercise 15 counts). The deadline for the exercises is January 16th, 2000. You can return the answers to the assistants, to the envelope outside room B313, to the secretary of the laboratory of Computer and Information Science or to Prof. Simula. Remember that there is a post-seminar lecture on the exercises on January 19th, 2000!

Help! If you are having real difficulities with the Matlab exercises, feel free to ask for help from the assistants. We have some ready Matlab scripts that should help you over the initial difficulities.

Home exercise 1

Suppose that we have a fixed set of $n-1$ measurement values of variable f and an nth measurement x that is an outlier. Prove that

$$\lim_{x\to\infty} = \frac{\mathrm{std}_f}{\mathrm{s}_f} = \frac{\sqrt{n}}{2}\frac{n}{n-1} \approx \frac{\sqrt{n}}{2}$$

Here std_f is the standard deviation of f ($\sqrt{\frac{1}{n-1}\sum_i (x_i-\bar{x})^2}$) and s_f is the mean absolute deviation ($\frac{1}{n}\sum_i |x_i-\bar{x}|$).

Home exercise 2

In this exercise, the purpose is to investigate the meaning of different numerical data types in terms of revealing structure from the data. As a case study consider the problem of comparing different companies based on their investment capital. The data is named `ex2` in the `ex_data.mat` file. The variable has investment capitals for 1908 companies (yes, this is true data, but actually it is not of investment capitals).

The task is to divide the data into five groups of similar companies (in terms of their investment capital). Consider the following three options to do this:

1. divide the range into five equally wide bins (using e.g. `hist` function)
2. make a logarithmic transformation, and do the same
3. replace each value with its ordinal number:

   ```
   [dummy,inds] = sort(x);
   [dummy,newx] = sort(inds);
   ```

   and do the same again

(a) Compare the results of the different options. What actually defines the similarities in the five groups in the different cases (intuitively speaking)? What are the different data types (after transformation) in the three cases? Which one of the options is, in your opinion, the best one? Why?

(b) The original values were in finnish marks (FIM). Your customer wishes to understand what the option 2 means, ie. he is somewhat at loss with the meaning of log(FIM). How would you (shortly) describe the meaning of the transformation?

Home exercise 3

A commonly used class of distances between vectors $\mathbf{x}$ and $\mathbf{y}$ are the Holder norms

$$d(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{d} |x_i - y_i|^p \right)^{\frac{1}{p}},$$

where d is the dimension of the vectors and x_i is the ith component of vector $\mathbf{x}$. Sketch the contour of the points which lie at distance 1 from the origin in two dimensions using different values for p. What happens when $p \to \infty$?

Home exercise 4

Assay the Cars data set in a similar manner as the examples in section 4.4. of the book. You don't have to calculate the `'REQ'`, `'VAR'` or `'LIN'` characterizations if you don't want to (ie. if you don't have the algorithms implemented). Instead you might want to look at the distributions of the different variables and say something about them (`'binomial'`, `'gaussian'`, `'exponential'`, `'uniform'` ...), or some other characterizations that you consider important.

The point of this exercise is to go through the data set one variable at a time, collect the essential information and present it in a compact, easily understandable format.

In case you want to use some program other than Matlab (for example Excel, the demo softwares on the CD-ROM, or some other tools) the data set is provided in multiple formats:

- `http://www.cis.hut.fi/cis/edu/Tik-61.181/s99/cars.txt` is the original file from the CD-ROM
- variable `ex4` in file `ex_data.mat` is a Matlab struct containing several fields: `ex4.data` contains the numerical data fields, `ex4.labels` contains the alpha data fields, `ex4.comp_names` contains the field identifiers for numerical variables and `ex4.label_names` contains the field identifiers for alpha variables. For example, `ex4.data(:,1)` contains values of variable `ex4.comp_name(1)` and `ex4.labels(:,1)` contains values of

variable `ex4.label_names(1)`. The values associated with sample `i` are `ex4.data(i,:)` and `ex4.labels(i,:)`. Notice that the three latter fields are all cell arrays (of strings).

If you're using Matlab 4, you have to load `ex_data4.mat`. There, the corresponding variables are `ex4num`, `ex4num_names`, `ex4alpha1`, `ex4alpha2`, `ex4alpha3` and `ex4alpha_names`. The five latter variables are char arrays.

- `http://www.cis.hut.fi/cis/edu/Tik-61.181/s99/cars.xls` is an Excel 97 worksheet

If you're using Matlab, functions `unique`, `isnan` and `isempty` might come in handy. If you're using Matlab 5 and you don't know what cell arrays are, see functions `cell` and `cellstr`, too.

Home exercise 5

In this exercise, we test whether the confidence level estimation procedures proposed in chapter 5 of the book work in practice. The case data (variable `ex5`) contains a data set with seven variables with different kinds of distributions (the values of `i`th variable are `ex5(:,i)`).

Hint: you can extract a random sample (of size `n`) from variable `i` like this:

```
inds = ceil(rand(n,1)*length(ex5));
x = ex5(inds,i);
```

(a) For each variable, calculate the average standard deviation of the variable for different sample sizes. Naturally, variance (second central moment) is just one distribution characteristic. Other characteristics include skewness (see function `skewness`) and kurtosis (see function `kurtosis`). Do you feel that a good measure for confidence on sample size is possible using these characteristics?

(b) For sample size 50, plot the histograms (see `hist` command) of the standard deviations. Does the assumption that they are normally distributed hold?

(c) The confidence level calculation for continuous variables introduced in the book has been implemented in file
`http://www.cis.hut.fi/cis/edu/Tik-61.181/s99/confidence.m`
How does the algorithm work? Do you think you could trust the results?

(d) Apply the function to the variables to calculate the (average) needed sample size to get confidence levels of 75%, 85%, 90% and 95%. Compare the sample sizes with the results from (a) and (b).

(e) Extra question (no need to answer this). Could you propose a better procedure for determining confidence?

Home exercise 6

Monotonic variables have no bounded set from which their values come from. They either increase (or decrease) without bound like time stamps, or their set of potential values is in practice unlimited like names or zipcodes. Monotonic variables are typically easy to identify using background knowledge, but without background knowledge their detecting is hard.

Device an algorithm (or a set of algorithms) to detect monotonicity. Hint: see the end of Chapter 5 from the book. Apply your algorithm(s) to data set `ex6`.

Home exercise 7

The purpose of this exercise is to take a look at the m-of-n coding. The task is to create a coding for colors. In particular look at the following colors:
`'red', 'white', 'black', 'cyan' and 'green'`

(a) One widely used coding for colors is the RGB-coding which gives the proportion of red, green and blue components in the color. The RGB values for the above colors are:

```
'red'    1 0 0
'white'  1 1 1
'black'  0 0 0
'cyan'   0 1 1
'green'  0 1 0
```

Calculate the Euclidian distance matrix between all pairs of colors (e.g. distance between `'cyan'` and `'green'` is 1) and think whether it is sensible or not.

(b) HSV, for hue, saturation and value, is also widely used for coding colors. Check out function `rgb2hsv` and calculate the corresponding HSV values for the colors above. Calculate distance matrix and compare it with the distance matrix in (a).

(c) For certain purposes, some other characterizations than those used above might be useful. For example, "warm", "dark", "pure" and "casual". Create an m-of-n coding of the above colors using these properties (note that you can use all values between 0 and 1) and compare the distance matrix to those in (a) and (b).

Home exercise 8

Consider a sphere of radius a in d dimensions. The volume of the sphere is given by

$$V_d = \frac{S_d a^d}{d},$$

where S_d is a constant. Show that the fraction of the volume of the sphere which lies at values of the radius between $a - \epsilon$ and a, where $0 < \epsilon < a$, is given by

$$f = 1 - \left(1 - \frac{\epsilon}{a}\right)^d.$$

Hence show that, for any fixed ϵ no matter how small, this fraction tends to 1 as $d \to \infty$. Evaluate the ratio numerically, with $\epsilon/a = 0.01$, for the cases $d = 2$, $d = 10$ and $d = 1000$. Similarly, evaluate the fraction of the volume of the sphere which lies inside the radius $a/2$, again for $d = 2$, $d = 10$ and $d = 1000$. We see that, for points which are uniformly distributed inside a sphere in d dimensions where d is large, almost all of the points are concentrated in a thin shell close to the surface.

Home exercise 9

The Principal Component Analysis (PCA) is probably the most widely used linear projection technique. In the projection, high-dimensional data are projected on a plane whose axes are principal components. The first principal component represents the direction where data deviation is largest, the second one where the deviation is second largest and so on. All the principal components are orthogonal.

(a) In matrix `ex9` there are 6000 six-dimensional data points. Project the points on a plane using all possible principal component combinations. The projection can be carried out using Matlab function `pca_plot.m` available from the course web site. Are you able to distinguish any clusters?

(b) Normalize the data so that the variances of the six variables (i.e., the columns of the data matrix) are equal. Perform the same projection using the normalized data set. How did the normalization affect the projection?

(c) Redraw the PCA plots of (a) and (b) using cluster membership information (in Matlab variable `ex9c`). Were you able to make the correct conclusions about the number, location and shape of clusters without the additional information?

The purpose of this simple exercise was to demonstrate the problem of getting an idea of the shape or cluster structure of the data. As you probably noticed, this is not very easy if the cluster memberships are not known — which is usually the case. In reality the data is also typically corrupted by noise and dimension of the data is higher than six.

Home exercise 10

In this exercise, different normalization operations are tried out in practice, especially with respect to how they operate in case of out-of-range values.

The purpose of variable normalization is to bring the variable values to a suitable range, e.g. $[0, 1]$. This has to be done so that the distribution characteristics are not distorted too much. The problem is that if the final model requires the values to be within a certain range or it is otherwise sensitive to very large (positive or negative) values, out-of-range values from the "live" data may break it down.

There are several options for doing the normalization:

- If minimum and maximum values are known, a simple linear transformation is the best one because it doesn't effect the distribution characteristics at all. If they are not known, one could specify some maximum/minimum allowed values, and replace all out-of-range values with these. However, this is a discontinuous transformation and thus not very nice.

- Another option would be to have a linear transformation for the range of training data, and a non-linear, for example logarithmic, transformation outside that range. Of course, infinitely large values are still possible, but the range is much easier to limit to a certain interval.

$$x' = \begin{cases} -\log(\min(x) - x) & \text{if } x < \min(x), \\ \log(x - \max(x)) & \text{if } x > \max(x), \\ x & \text{otherwise.} \end{cases}$$

- The book proposes, pretty strongly, using the logistic function: $x' = 1/(1+e^{-x})$, where x has already been (linearly) scaled appropriately (see page 274 on the book).

Try the normalization procedures on the test data. There are two data sets with four variables: `ex10` and `ex10o`. The former is the "training" data based on which the normalization parameters should be selected, and the latter has also out-of-range values. How severe are the distortions for different distribution types?

You can also (not obligatory) try selecting the normalization parameters directly based on the `ex10o` data set. This shows how sensitive the normalization procedures are to outliers.

Home exercise 11

The aim of variable redistribution is to get a balanced distribution (see chapter 7.2). The benefit is that the whole value range (e.g. $[0, 1]$) is used in the modeling instead of only a small part of it and that the effect of outliers is considerably reduced. For example, in the data of exercise 2, more than 60% of the data is in the first 3% of the value range. The precision of the model would be very poor in that area.

(a) Implement some variable redistribution method and apply it to the four variables in `ex10`. Plot the new distributions. One possible redistribution method is the ordinal replacement (see the third option in exercise 2). However, the problem is how to apply the same transform to new data sets, especially out-of-range values.

(b) What possible problems do you see in redistributing variables?

Home exercise 12

In this exercise, the purpose is to take a look at entropy as a way to survey the data. Entropy measures uncertainty of outcome (given some initial conditions). In case of discreet states, it is defined as:

$$H(x) = -\sum_{i=1}^{N} p_{x=i} \log(p_{x=i})$$

Correspondingly, conditional entropy is defined as

$$H(x|y) = -\sum_{j=1}^{M} p_{y=j} \sum_{i=1}^{N} p_{x=i|y=j} log(p_{x=i|y=j})$$

Note that if $p = 0$, $p \log(p) = 0$ by definition.

The minimum value of entropy is 0 and maximum value $\max(H(x)) = \log(N)$. Thus, entropy can be easily normalized between $[0, 1]$, where 1 corresponds to maximum entropy — when each state is equally probable — and 0 to minimum entropy — when only a single state is possible.

(a) Let $N = 5$ be the number of possible states with

$$\text{Case 1:}\, p_1 = p, p_2 = (1-p), p_3 = p_4 = p_5 = 0$$

or

$$\text{Case 2:}\, p_1 = p, p_2 = p_3 = p_4 = p_5 = (1-p)/(N-1)$$

Calculate, and plot, relative entropy $(H(x)/\max(H(x)))$ for different values of p between $[0.5, 0.99]$. Note that the uncertainty must be really low to get a low entropy. Are the results intuitively appealing?

How about if you calculate the uncertainty slightly differently by comparing the probabilities to an even distribution of probabilities:

$$\sum_{i=1}^{N} \min(p_i, 1/N)$$

This calculates the common "area" of the true distribution and an even distribution and has values between $[1/N, 1]$.

(b) Matrix `ex12b` contains two discreet variables: $x =$ `ex12b(:,1)` and $y =$ `ex12b(:,2)` so that the two values $x(i)$ and $y(i)$ correspond to the same data sample. Calculate (relative) entropies $H(x)$, $H(y)$, $H(x|y)$, $H(y|x)$ and $H(x, y)$.

(c) For continuous variables, entropy is defined as

$$H(x) = -\int_a^b p(x) \log(p(x)) dx$$

Variable `ex12c` contains 20 samples from a continuous variable. Since estimating a continuous distribution for a variable is usually not an option, you have to discretize the variable by binning the values. Calculate and plot the entropy using 2–50 equally spaced bins. You can get the bins using Matlab's `hist` function, e.g.:

```
h = hist(x,10);
p = h/sum(h);
```

The bins need not be of equal size, for example: `h = hist(x,c)` uses bin centers listed in vector `c`, which can be defined using some adaptive clustering algorithm, manually e.g. `c = [0 1 2 5 20 50 100]` or even `c = sort(x)`. How is the entropy calculated in this case?

Home exercise 13

Let us assume that our modeling tool requires that all the values are present in a signal. However, the signal available has some missing values. Suggest some technique(s) that could be applied to fill in the gaps in the signal.

Variable `ex13` is a signal with ten empty missing values marked by not-a-numbers (NaNs). Find some estimates for the missing values by applying one of the techiques. Plot the original and the enhanced signal in the same figure. Is the result satisfactory?

Home exercise 14

A measurement signal is used in machine condition monitoring. Normal operation of the machine results as sinusoidal signal with constant frequency, amplitude and level; errors are clear peaks in the signal. There are two problems in the measurements. After the start up (1) the level of the signal tends to drift and (2) the amplitude tends to damp.

In variable `ex14` there is 24 hour measurement history of the variable after start up of the machine with no errors. Your task is to find a compensation of the two distortions so that the signal finally has constant level and amplitude. Plot the resulting signal. What is the sampling frequency of the signal? What is the frequency of the sinusoid?

Home exercises 15

(If you want to, you can return your answer to this on a separate sheet with no name on it. Anyhow, we guarantee the answers will not effect your grade. Actually, we assistants thrive on critique.)

Give feedback of the exercises:

1. The purpose of home exercises was to act as an eye-opener about how the techniques are used, and what difficulties exist. Did they succeed in that? Were they useful? Were they interesting?

2. How about the practical assignement?

3. What was most difficult about the exercises?

   - using Matlab
   - understanding the questions / assignements
   - motivating yourself to do the exercises
   - something else ... what?

4. In hindsight, do you feel the course was worth the trouble? Do you see any use for the new knowledge in the near future?

5. Any special messages to assistants or Mr. Pyle (delivered anonymously, of course)?

Practical assignement

The assignement is simply as follows: design and implement a preparation procedure for the data set given in matrix `heartT` (or in ASCII file `heartT.txt`), and use linear regression to predict the last variable based on the others. After this, use the preparation procedure and the model to predict the execute data `heartE` (ASCII file `heartE.txt`). Note that although all variables are given as numerals, some of them are actually nominal. A more detailed description of the problem and the variables is given in file `heart_problem.txt`.

In the procedure, pay attention to things like out-of-range problems, missing values and remapping nominal variables. Notice also that you can boost the performance of the linear model if you prepare the data appropriately (and vice versa).

Here's a sample skeleton for the assignement. You simply have to implement the functions `prepare_input` and `prepare_output`.

```
InputT  = prepare_input(heartT(:,1:13)); % prepare input variables
OutputT = prepare_output(heartT(:,14));  % prepare output variables
Model   = InputT \ OutputT;              % make the model
PredT   = InputT*Model;                  % prediction
r       = max(OutputT)-min(OutputT);     % range of results
errT    = mean((OutputT-PredT).^2)/r;    % error for training set
plot(OutputT,PredT,'r.');                % plot the result
title(sprintf('Training set %5.2f',errT));

InputE  = prepare_input(heartE(:,1:13)); % prepare input variables
OutputE = prepare_output(heartE(:,14));  % prepare output variables
PredE   = InputE*Model;                  % prediction
errE    = mean((OutputE-PredE).^2)/r;    % error for testing set
plot(OutputE,PredE,'r.');                % plot the result
title(sprintf('Training set %5.2f',errE));
```

Deliverables:

- description of preparation procedure (code is also welcome, but is not sufficient alone)
- results for both training and execute data

As a reference, here is perhaps the simplest possible solution. Don't imitate it. The results are in Fig. 1. In Fig. 2 is the result after some more complex preprocessing.

```
function D = prepare_input(Dorig)

% replace all out-of-bounds values with minimum/maximum values
% replace all missing values with the center value
maxvals = [77 1 4 200 564 1 2 194 1    5 3 3 7 4];
```

```
minvals = [28 0 1  80   0 0 0  60 0 -2.6 1 0 3 0];
meavals = (maxvals-minvals)/2;
[len dim] = size(Dorig);
for i=1:dim,
  ind = find(Dorig(:,i)>maxvals(i));
  if ~isempty(ind), Dorig(ind,i) = maxvals(i); end
  ind = find(Dorig(:,i)<minvals(i));
  if ~isempty(ind), Dorig(ind,i) = minvals(i); end
  ind = find(isnan(Dorig(:,i)));
  if ~isempty(ind), Dorig(ind,i) = meavals(i); end
end
D = Dorig;

function x = prepare_output(xo)

% replace all out-of-bounds values with minimum/maximum values
% replace all missing values with the center value
mival = 0; maval = 4; meval = 2;
ind = find(xo>maval);  if ~isempty(ind), xo(ind)=maval; end
ind = find(xo<mival);  if ~isempty(ind), xo(ind)=mival; end
ind = find(isnan(xo)); if ~isempty(ind), xo(ind)=meval; end
x = xo;
```

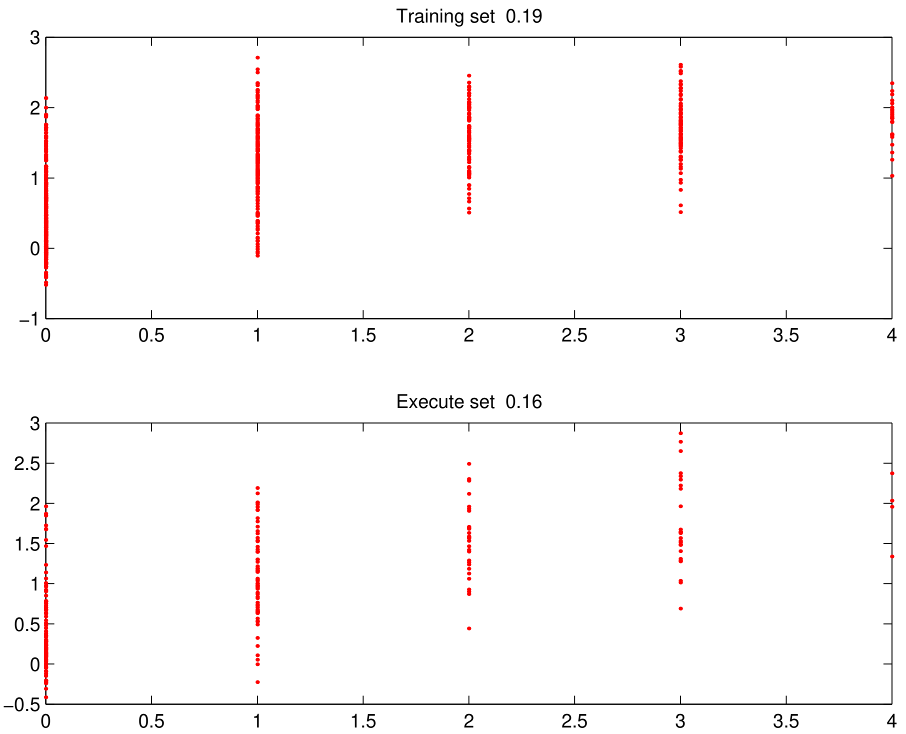

Figure 1: Results of linear model after applying a very simple preparation procedure to the heart data set. The correct result is on the x-axis and the predicted result on the y-axis. Training set results are on top figure and execute set results on the bottom figure.

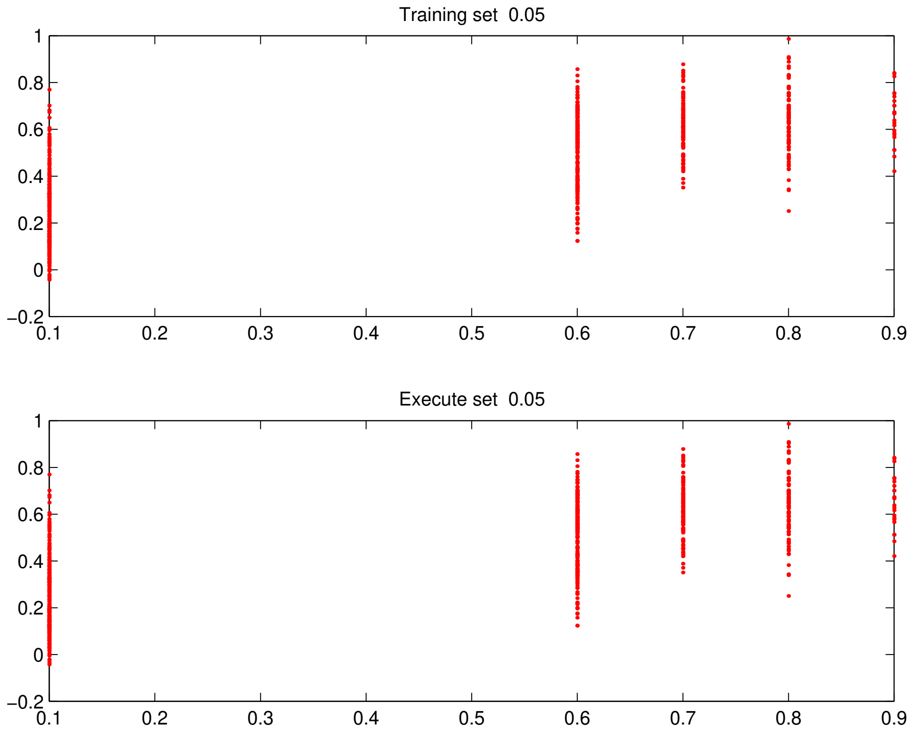

Figure 2: Results of linear model after a more complex preparation procedure to the heart data set. The correct result is on the x-axis and the predicted result on the y-axis. Training set results are on top figure and execute set results on the bottom figure.