Evaluation of Support Vector Machines for Imputation

H. Mallinson, A. Gammerman

Department of Computer Science
Royal Holloway, University of London
Egham, Surrey TW20 0EX
{hugh,alex}@cs.rhul.ac.uk

Contents

1 Introduction **3**
1.1 The Data . . . 3
1.2 The Imputation Method . . . 3
1.3 Hardware . . . 4
1.4 Software . . . 4

2 Support Vector Machines **4**
2.1 Core algorithm . . . 4
2.2 Extending SVM to several missing variables . . . 6
2.3 Model order parameters . . . 6
2.3.1 Kernel functions . . . 7
2.3.2 Kernel parameter: σ . . . 8
2.3.3 Error trade off: C . . . 8
2.3.4 Regression parameter: ϵ . . . 9
2.3.5 Summary . . . 9
2.4 Practical aspects . . . 9
2.4.1 Processing time . . . 9
2.4.2 Data format . . . 9
2.4.3 Algorithm failure . . . 10

3 DLFS: Danish Labour Force Survey **11**
3.1 Overview . . . 11
3.2 Exploratory Experiments . . . 12
3.3 Evaluation Experiments . . . 13
3.3.1 Other training choices . . . 13
3.4 Summary . . . 14

4 ABI: Annual Business Inquiry **16**
4.1 Overview . . . 16
4.2 Evaluation Data Description . . . 16
4.3 Selection of data for training . . . 17
4.4 Preprocessing . . . 18
4.5 Model Order . . . 18
4.6 Results . . . 19
4.7 Summary . . . 19

5 SARS: Sample of Anonymised Records **21**
5.1 Overview . . . 21
5.2 Evaluation Data Description . . . 21
5.3 Training Choices . . . 22
5.4 Parameters to set before training . . . 22
5.5 Training Data for Regression . . . 22
5.6 Training Data for Classification . . . 22
5.7 Overview of Complicating Factors . . . 23
5.8 Results . . . 23
5.9 Summary . . . 25

6 Discussion **26**
6.1 Motivation for investigating SVMs . . . 26
6.2 Assessing performance . . . 26
6.3 Future Work . . . 27

7 Conclusions **28**

1 Introduction

1.1 The Data

In this document we describe experiments conducted for the Euredit Project, Work Package 5.6. Using Support Vector Machines(SVMs), we impute three datasets: Danish Labour Force Survey (DLFS), Sample of Anonymised Records (SARS) and Annual Business Survey (ABI). SARS and ABI both come in two forms; the *Y2* form has missing values but no errors, the *Y3* form has both missing values and errors.

Data-cleaning consists of removing infeasible and implausible values from a dataset (editing) and then supplying plausible estimates for all values that are missing (imputation). We do not attempt to evaluate an editing technique in this work package. We impute the Y3 data sets, but carry out the minimal editing that will allow the correct function of the imputation algorithm[1]. Naturally SVM results on this Y3 data might well be improved with more sophisticated error removal procedures. The purpose of the experiments was to investigate how deleterious the errors were on the imputation of the known missing values. For ABI Y3 we performed no editing. Results in comparison with ABI Y2 clearly show the effect of this.

1.2 The Imputation Method

We apply the support vector machine using the *rbf* kernel. This offers non-linear regression and classification. Exploratory experiments on development datasets showed this kernel to perform well relative to polynomial and sigmoidal kernels on a range of variables. We thus decided to investigate the rbf kernel as a generic model. Unless stated otherwise it can be assumed that discussion of SVMs in this document will imply the use of the rbf kernel.

In the imputation problem several variables may be missing values. We employ a separate SVM model for each variable that lacks values. The SVM is restricted to univariate target variables.

The SVM can be grouped with semi-parametric prediction techniques, such as feedforward neural nets. Both algorithms offer non-linear function estimation. Like the multi-layer perceptron, the SVM is in principle a *universal approximator*; given a large enough training set the algorithm will

[1] we have to implement a basic error removal strategies on SARS Y3: we remove values that were not in the allowed range, e.g. sex takes values 1 (male), 2 (female), so values of 3 were removed

approximate arbitrarily well any functional continuous mapping from one finite-dimensional space to another.

It is argued that semi-parametric techniques offer the flexibility of non-parametric techniques, such as nearest neighbours. The full dataset becomes the model for nearest neighbours however. This may require large storage space. In addition testing datapoints requires comparison with all members of the training set. The SVM can offer a more compact model, requiring less storage space and less time to apply in test phase.

The multi-class SVM algorithm is built from a number of binary classifiers. If there are n classes, a classifier is trained for each class against the rest, resulting in n classifiers.

1.3 Hardware

Our system for experimentation is a described in bullet point form below.

- UNIX mainframe DEC-ALPHA 410
- CPU :466 MHz 3 processors.(System is shared)
- Cache size: 4 Mb Cache
- Operating System: tru64 UNIX 4.0G
- Amount of system physical memory : 1536 Mb
- Virtual memory: 7Gb Swap Space

1.4 Software

The system uses *MATLAB 6.0* for data handling, preprocessing, parameter setting and cross-validation. This platform has some limitations. Our system could be run with chunks of size not greater than 40000 rows by 30 variables. SARS contains more rows and was therefore treated in chunks. The SVM is written in C code.

2 Support Vector Machines

2.1 Core algorithm

The SVM algorithm[7] estimates a univariate predictor. It offers a means of estimating the value of one variable, Y, given a number of others $\{X_1, X_2, ...X_n\}$,

jointly denoted by $\mathbf{X}$. Y is the (univariate) target variable, and $X_1, X_2, ..., X_n$ are known as input variables. Scalar Y naturally requires a regression form of the SVM. A detailed tutorial can be found in [18]. If Y takes a finite number of discrete values we employ the classification form of the SVM. The SVM regression algorithm builds a 'model' for the conditional expectation, $E(Y|X_1, X_2, ...X_n)$ of Y. The SVM classifier produces the MAP estimate, $argmax_i(P(Y = i|X_1, X_2, ..., X_n))$, for discrete Y. We underline the point that the output of the SVM is a point estimate, there is no model of the posterior distribution, $P(Y|\mathbf{X})$.

In order to learn the parameters for the SVM, a data sample must be supplied. This *training set* of units, denoted by S, must drawn from the joint distribution $P(\mathbf{X}, Y)$. S $= (\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots (\mathbf{x}_m, y_m)$. The *test set* of units for which we require y values is assumed to be drawn from the marginal distribution, $P(\mathbf{X})$. In other words, train data and test data are assumed to be iid.

The algorithm can be understood as involving a projection of the input data, $\mathbf{x} \rightarrow \phi(\mathbf{x})$ to a much higher dimensional feature space. In this new space a linear model is fitted (either a discrimant or an interpolant). The pre-image of this linear model, in the input-space, can be highly non-linear. In fact the image $\phi(x)$ of each point in the feature space is not calculated explicitly. The linear algorithms used require only the dot-product between data points to train and test. The *kernel function* performs the projection of two data points, and the dot product between them, in one step: $k(x, y) = \langle \phi(x), \phi(y) \rangle$. The high dimensional representation of each point remains implicit.

The trained model is described by a linear expansion of functions;

$$f(z) = \sum_{i=1}^{n} \alpha_i k(x_i, z) + b$$

where n is the size of the training set. However the SVM model can be 'sparse'. This means that many of the α_i are zero.

The SVM training process requires the solution of a convex quadratic programme. Such problems are well understood and a number of commercial toolboxes exist for solving them[17]. Training datasets of dimension 5000 units by 20 variables can be handled in under 60 seconds by our system.

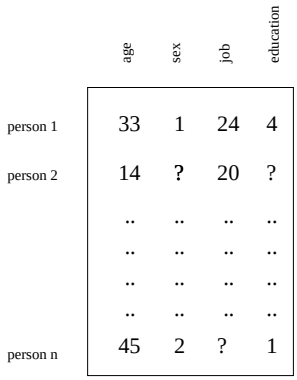

Figure 1: Incomplete data

2.2 Extending SVM to several missing variables

For two of three datasets treated in this report, more than one variable must be imputed in the dataset. A separate SVM is employed for each variable. As one record may lack values on more than one variable, the system needs a strategy for handling missing input variables, in train and test units.

The SVM requires a fully observed training set, and the input variables for a test unit must also be fully observed. In figure 1 we see that two variables are missing simultaneously. If we are attempting to impute the *education* variable for person 2, the input variable *sex* must be dealt with. This complication can be handled in two ways. The model for *education* might be built without the variable, or the variable might be estimated, using a model that does not condition on *education*. We use a crude estimation technique that inserts the mean or mode for input variables. This will be called 'patching'.

2.3 Model order parameters

In order to train the SVM model we must first fix the *family* of models in which the algorithm searches during training. This is also known as 'specifying the model order'. If we believe the data is noisy we will chose a family containing simpler models that tolerate more errors. In the case of a neural network the model order is specified (partly) by the choice of architecture; the number of hidden nodes. The ensuing training procedure finds good val-

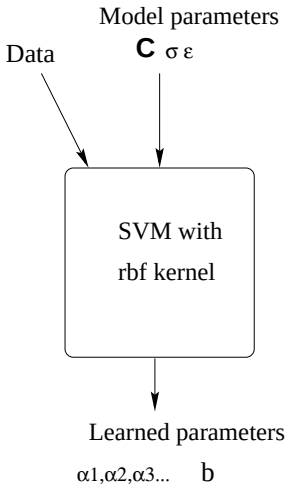

Figure 2: schematic of parameters

ues for the weights, given that architecture. The model parameters together characterise the family.

The model parameters may be set using a priori knowledge or sometimes heuristics exist. In this project, we choose to find them by a process called *cross-validation*. This process selects the values that perform 'best' on a set of data that is labelled, but has not been used for training. We assess the models according to root-mean-square error,(dL2). There is ample training data in all the datasets, so cross-validation is a reasonable approach. Below we describe the various model order parameters for the SVM.

2.3.1 Kernel functions

The kernel function equates a projection of the two data points x and y, to a higher dimensional feature space and their dot product in that space.

$$k(x, y) = \langle \phi(x) \cdot \phi(y) \rangle$$

Three examples of kernel functions are given below;
rbf kernel:

$$k(x, y) = exp(-\frac{\|x - y\|^2}{2\sigma^2})$$

polynomial kernel:

$$k(x, y) = (\langle x \cdot y \rangle + 1)^d$$

linear kernel:

$$k(x, y) = \langle x \cdot y \rangle$$

Most kernels require a small number of further *kernel parameters* to be chosen. An rbf kernel has the *spread* parameter σ, which we describe in more detail below. The experiments described here employ the rbf kernel and also the linear kernel. The former makes a highly non-linear models available. The latter supplies only linear discriminants and regressors.

In fig.2 a schematic representation of the parameters for SVM learning is given. Before the training begins, the model order is specified by setting parameters C and σ. In the case of regression a third parameter ϵ is also set. The training procedure then discovers the parameters $\alpha_1, \alpha_2, \alpha_m$ and b which characterise the best fitting model. m is the number of training points. Using these parameters new points can be predicted.

2.3.2 Kernel parameter: σ

The rbf kernel has one parameter[2], σ. This parameter controls the smoothness of the model, by describing the region of influence of each point in the training set. As sigma shrinks, each point has a smaller, more local influence. This model will resemble a nearest-neighbours model. As sigma grows, the model becomes more global. We cross-validate values for σ from between $\frac{d}{5}$ and d, where d is the dimension of the input vector.

For DLFS and SARS, the σ values were about $\frac{2d}{3}$. In the ABI dataset many variables seemed to have almost perfect linear correlations. If an rbf kernel was chosen the σ value chosen by cross-validation was 500, a much larger value than was expected. The model approximates a linear kernel with this size of σ.

2.3.3 Error trade off: C

During training, parameterisations for the prediction model are sought that make few mistakes on the training data. However the training procedure also penalises choices of parameter that lead to more complex models. The parameter C, controls the trade off between model complexity and errors.

[2]The rbf kernel can have a separate σ_i for each input dimension, but we choose a simpler option

The larger C, the greater the penalty for mistakes on the training data in relation to the penalty for using complex models. The algorithm will seek more complex models to reduce this penalty.

2.3.4 Regression parameter: ϵ

In the case of the SVM for regression only, a further model parameter is used. The ϵ tube represents a region around the regressor within which errors are not counted. As ϵ grows the regressor becomes smoother.

2.3.5 Summary

We compare a range of settings for the model order parameters on a hold-out, or validation set. Usually 5 settings of each suffices. This corresponds to $5^3 = 125$ different settings for a regressor, and 25 for a classifier (ϵ not required).

2.4 Practical aspects

2.4.1 Processing time

The SVM implemention used is research software. It has no gui, and requires experience to use correctly. It is a lengthy and cumbersome process to specificy cross-validation ranges etc. Experimental set up takes 5 - 10minutes. Set up also involves naming of some output and log files, the specification of the number of columns to tackle and so on.

We now consider computational time. Each variable exhibiting missingness requires the separate training and validation of an SVM. For one variable, we perform 4-fold cross validation, for each cross-validated setting. The same size training data can lead to varying processing time, depending upon noise levels, and the complexity of the dependency.

On DLFS, each of the 125 models compared took 10 seconds to evaluate. The dataset only required one variable to be imputed, and the task was complete in 1.5 hours. Each ABI variable took 2-3 times longer to cross-validate, and there were a total of 10 variables to impute. The full data set took over 24 hours to process.

2.4.2 Data format

The algorithm presently accepts data in ascii text files, and does not interface with Windows or other standard database software.

2.4.3 Algorithm failure

If too small values for σ are chosen, the algorithm may fail to terminate. Very noisy data may lead to slow convergence, or non termination.

3 DLFS: Danish Labour Force Survey

3.1 Overview

- experiments on DLFSY2: RS2001, RS2002, RS2003
- 14 variables; describing job type, age, marital status of respondents.
- 1 scalar variable missing values: *income*
- input variables: 1 scalar (*age*), 12 discrete: including *age, marital status etc.*
- platform: MATLAB 6.0 and C code.
- exploratory analysis: comparisons on complete portion of evaluation data using artificial missing data mechanism. Comparing with linear techniques and MLP.
- preprocessing: normalisation of all variables. variable deletion
- training-data size: 5000 units
- processing time:
  set-up time: 5 minutes
  cross-validation: 125 settings * 4-fold * 10 seconds = 1.3 hours
  final training time = 1 minute
  testing time = 0.1 minutes
  total = 1.4 hours
- model order: C=20 $\sigma = 6$ $\epsilon = 0.1$

This data, collated in Denmark 1996, consists of population register records for individuals selected for interview for a labour force survey. The total number of records is 15579, measured on 14 variables[3]. *income* is the only variable missing values. 4175 records must be imputed. The SVM in regression form is used; *income* takes values in the range [0-1,000,000]. It is measure in Danish Krone. The missing data pattern is genuine.

See the table in Appendix A for a description of each variable and the values it takes. *age* is integer valued, we treat it as a continuous variable. All other input variables are nominal, although education could be treated as ordinal. DLFS has missing values but no errors.

A 2-dimensional plot (fig.5) shows *age* (rounded to the nearest decade) on the x axis plotted against *income*. The dots show the average income for people of the given age. The small circles show 1 standard deviation. *age* is non-linearly correlated with *income*. As *age* increases the average income

[3]in fact there are 13 informative variables *response* just indicates if the *income* value is missing

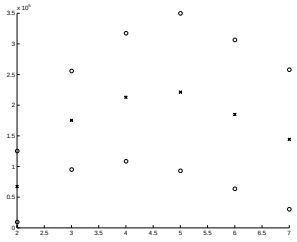

Figure 3: Age vs. Income

increases. A peak is reached at 50 years of age, after which average income declines with age. The Pearson correlation coefficient gives a value of 0.3.

Under the assumption of MCAR missingness, a mean imputation would give an error of approximately 115000 DK, the standard deviation of the observed values.

3.2 Exploratory Experiments

Exploratory experiments were performed by introducing an artificial missingness pattern into the 11404 complete records. An MCAR pattern was used. We repeated the experiment 20 times. We imputed values using a linear SVM, a neural net (MLP) and the group-mean algorithm. Group-mean variables were (3) age, (6) business type, (2) sex,(5) education. The variables were chosen incrementally. The single best variable was covariate was age (discretised into 5 subgroups).

	K-S	mae	rmse	worst case
SVM rbf	0.07	53000	83000	780000
MLP	0.12	58000	87000	790000
Group-mean	0.13	61000	90000	810000
Linear SVM	0.10	61000	92000	790000

Below we show the performance improvement for the group-mean algorithm as each variable is added.

group-mean variables	K-S	mae	rmse	worst case
(3 6 2 5)	0.092	57398	90943	723737
(3 6 2)	0.131	61635	95742	746162
(3 6)	0.213	65199	101534	788992
(3)	0.369	72862	115218	854249
SVM linear	0.126	62409	94746	715649

These experiments showed us which variables are most strongly correlated with age. The non-linear SVM is outperforms the linear SVM on rmse. The worst case error is large, nearly 80% of the range. The rmse estimates had a variance of 3000DK.

3.3 Evaluation Experiments

We applied the SVM with an rbf kernel to the DLFS data, normalising all variables, including the target, to zero mean and unit variance. There are 11404 records available as training data. We use only 5000 units, as larger training showed no improvement in cross-validated error, and took a long time to process. As the target variable is scalar, we cross-validate three variables; σ, C and ϵ, each initially having five settings. This results in $5^3 = 125$ different model orders being compared. We used 4-fold cross-validation, comparing rmse. Evaluation of each setting took 1 minute approximately. The experiment took 2 hours to complete.

parameter	cross-validated settings	best setting found
σ	3 6 9 12 15	6
C	2 5 10 20 100	10
ϵ	0 0.01 0.05 0.1 0.2	0.1

3.3.1 Other training choices

Two refinements to the algorithm, variable selection and stratification, were investigated. *Variable selection* involves the selection of the most informative variables in a step-wise additive process. The single best variable is selected first (based on a validation set, comparing rmse). This variable is

then combined with all others and the best pair is chosen, then the best three and so on. The best variables, in decreasing order, were *age,business type, sex, marital status* and *employment status*. These were the same as those chosen by the group-mean algorithm. This feature reduction should improve generalisation by removing noisy variables uncorrelated with the target variable.

The results below for the third algorithm show the SVM trained separately (stratified) for male and female respondents. This split showed some improvement in cross-validation. We investigated all variables apart from age as stratification variables. Only stratifying on *sex* improved cross-validated error. The model order that was found by cross-validation was similar for all three setups; $\sigma = 5$, C=10 and $\epsilon = 0.1$.

algorithm	training size	slope	dL1	dL2	KS	MSE
SVM rbf	5000	0.93	46000	80000	0.102	1600000
SVM rbf vs	5000	0.94	45000	80000	0.099	1100000
SVM rbf strat	5000 each class	0.94	46000	80000	0.095	1100000

There is little difference between the results. The dL2 (root-mean-square-error) rounded to two significant figures, is identical for all three approaches. This figure represents approximately half of the mean value for *income* and two thirds of the inter-quartile range which is 116000DK. Clearly only a relatively weak dependency has been found.

The Kolmogorov-Smirnoff is a measure of preservation of distribution. It gives the maximum percentage difference between the cumulative distribution functions of the true and imputed values. The SVM scores 10%. The results are better than those estimated from exploratory experiments.

3.4 Summary

This problem involved prediction of the *income* variable. This variable has mean 175000DK, standard deviation 115000DK. We have a rmse of 80000DK. This represents an improvement over mean imputation by 30%. The Kolmogorov-Smirnoff measure shows a 10% disparity between the cumulative distribution functions. The algorithm could achieve a lower result if we chose to add noise (residuals) to the predictions, but this would naturally worsen the dL1 performance. The SVM seems to not require stratifi-

cation, or feature reduction. The SVM trained successfully and in a timely fashion. Results for the SVM were amongst the best in the Euredit project for the measures of preservation of true values. Preservation of distribution was expected to be less good, as imputing the conditional expectation will compress the distribution and reduce the second moment.

4 ABI: Annual Business Inquiry

4.1 Overview

- experiments on ABI Y2: RA2001, RA2002, RA2004,RA2005,RA2008
- experiments on ABI Y3 Experiments: RA3002, RA3007
- total 31 variables: 16 used: 15 scalar, 1 discrete
- 10 imputation variables: *turnover, emptotc, puresale, purtot, taxtot, stockbeg, stockend, assacq, assdisp, employ*
- platform: MATLAB 6.0 and C code on unix.
- exploration: Pearson correlation and 2-D plots
- preprocessing: normalisation of all variables
- trainingdata size: 3000 units.
- editing of Y3: None performed.
- processing time for each variable
  set-up time: 5 minutes
  cross-validation (model selection):
  125 settings * 4-fold * 30 seconds = 250 minutes
  training time = 1 minute
  testing time = 0.5 minutes
  total = 4.5 hours
- Model order: For all variables: linear settings.
- $\sigma = 500$, C= 200, $\epsilon = 0$.

4.2 Evaluation Data Description

This dataset collated in 1998, contains 6233 records measured on 31 variables. Each record contains responses to selected questions from the UK Annual Business Inquiry for sector 1. There are 2 questionnaires with one (the 'short' version) only asking for summary information. Variable values for questions not on the short form are set to (-9) for businesses answering the short form. There are four 'reference variables' acquired independently from the dataset.

The dataset exists in two forms, Y2 which is missing values, and Y3 which is missing values and also contains perturbed or erroneous values. We performed experiments on both datasets. We do not attempt to clean the Y3 data, and we expect results to be less good.

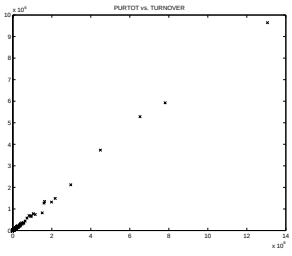

Figure 4: PURTOT vs. TURNOVER

We reduce the dimensionality of this dataset to 16 variables. The fully observed reference variables are *ref*, *class* and *weight*. Other reference variables are *turnreg, empreg and formtype*. The first two describe the registered turnover and the size of the workforce respectively. The last identifies short and long form questionnaires.

All the variables used in the experiments are common to both long and short forms of the questionnaire, and are described in detail in the metadata file. All of the imputation variables are non-negative scalar valued, representing sums of money, with the exception of *employ* which encodes the number of employees.

There are a few records far removed from the others in the input space. These can be called 'representative outliers'; correctly measured units with some variables taking values much larger than all the others. Below we show a histogram of log(turnover+1) partitioned in 10 bins. The skewness of the marginal distribution of turnover is apparent. We calculate the third moment; $m_3 = \frac{1}{n}\sum_{i=1}^{n}(x_i - mean(x))^3$ to get a measure of the skewness. For turnover, $m_3 = 7\text{e}{+17}$.

4.3 Selection of data for training

The data in sector 1 contains 6233 records. Of these over 4000 are completely observed. In the table below is shown the number of units with no variables missing, one variable missing and so on. Most variables have just one missing. We select 3000 units from the 4932 that are fully observed.

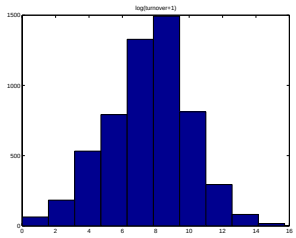

Figure 5: log(turnover)

num. missing in row	0	1	2	3	4	5	6
num. of units	4932	1052	135	27	67	18	2

4.4 Preprocessing

We applied the SVM with rbf kernel to this dataset, using the common variables listed above plus the reference variables (not acquired through the questionnaire). We deleted all other variables. All variables were normalised. No editing was performed.

4.5 Model Order

The model order was found by first cross-validating the settings below.

parameter	cross-validated settings
σ	3 6 9 12 15
C	2 5 10 20 100
ϵ	0 0.01 0.05 0.1 0.2 1

Models were compared with respect to dL1, the root mean square error. When a setting was chosen that was highest or lowest of the range, cross-validation was repeated with the range extended accordingly. We found that σ was chosen to be the largest value, (15) after an initial round. The cross-validation phase was then repeated with larger σ values.

Ultimately, highly linear models were chosen, with σ more than 200 for all variables apart from *assacq*. C was 500; also relatively high. ϵ was 0.01,

or 0 for most variables. Due to the skewness of the data, the normal range of σ was not effective. This kernel parameter normally takes values less than the dimension of the input space. As σ grows the global influence of each data-point gets larger.

4.6 Results

We imputed the ten variables common to the short and long forms that were missing values. Appendix B and C contain tables for SVM linear and SVM rbf. Slope values for *turnover* indicate that the variable was strongly correlated with input variables, the Pearson correlation coefficient between *turnover* and *purtot* is 0.92.

We can see by comparing the dL2 of each variable with its variance and mean, and observing that R^2 is over 0.9, that most variables have been imputed accurately. However *stockbeg* and *stockend* are less well preserved.

This poor result has been diagnosed as due to the crude 'patching' heuristic. A third of the units are missing *stockbeg* and *stockend* simultaneously.

4.7 Summary

The data concerns the finances of a range of businesses, many small but a few multi-nationals also. The distributions are skewed. The performance of the linear algorithm relative to the non-linear indicates that no strong non-linearity was found. Observation of the Pearson weighted moments confirmed this. Below we give each imputation variable, the most highly correlated covariate (m.h.c.c) and the value of the weighted Pearson correlation coefficient (Pearson). All variables are correlated with at least one other to degree 0.9, with the exception of *assacq*.

variable	TURNOVER	EMPTOTC	PURESALE	PURTOT	TAXTOT
Pearson	0.9984	0.9904	0.9993	0.9993	0.9067
m.h.c.c.	PURTOT	TURNOVER	PURTOT	PURESALE	STOCKEND

variable	STOCKBEG	STOCKEND	ASSACQ	ASSDISP	EMPLOY
Pearson	0.9657	0.9657	0.9686	0.2821	0.9822
m.h.c.c	STOCKEND	STOCKBEG	TURNREG	EMPLOY	TURNOVER

The SVM took over 4 hours for each variable, if one trains with the non-linear kernel. The results show that the non-linear SVM was able to capture the linear dependency. However a poor patching heuristic lead to weaker results for some variables, e.g.*stockbeg* and *stockend*. On this dataset it would have made more sense to train a separate model for each missing data pattern in the test set. We did not take weights into account in the training of the model. The correlation statistics show that all variables apart from *assdisp* are well modelled by a linear regressor. The non-linearity of the SVM seems not to be required.

5 SARS: Sample of Anonymised Records

5.1 Overview

- Experiments on SARSY2: RS20001, RS2002, RS2004
- Experiments on SARSY3: RS3001,RS3005, RS3006
- 31 variables: 2 index, 9 household, 20 personal
- 25 missing values
  types of missing variable: 2 scalar, 2 binary, 21 multiclass
- Platform: MATLAB 6.0 and C code.
- Exploratory analysis: comparisons on development data (sector 2 SARS) with linear techniques, MLP.
- Preprocessing: normalisation of all variables, removal of rarest classes. Deletion of index variables.
- Training-data size
  multiclass: 2000 units in each class
  regression: 3000 units total
- Editing of Y3: Values removed that had no meaning, e.g. Sex = 3;
- Processing time for each variable:
  Set up time: 5 minutes
  Cross-Validation (model selection) for categorical variable with 5 classes; 2 parameters are set, C and σ:
  25 settings * 4-fold * 5 classes * 10 seconds = 1.4 hours
  Training time = 1 minute
  Testing time = 0.5 minutes
  Total = 1.5 hours
  Total for 25 variables = approx. 2 days.
- Model order: C $\in$ [5-80], $\sigma \in$ [7-20] depending on the variable
- Appendix contains tabulated results

5.2 Evaluation Data Description

The data are a 1% sample of household records from the 1991 UK census. This amounts to nearly half a million records. There are 31 variables which represents a selection from the total number. Two of the variables are handled as scalar (regression) variables; *age* and *hours* worked. All other variables are multi-class discrete valued, with the exception of *sex* and *long-term-ill*, which are binary. We present a table of the variables in Appendix

C, that gives details of the variables and their rates of missingness. The dataset was presented in two forms; Y2 with missing values, and Y3 with errors and missing values. We attempted imputation of both.

5.3 Training Choices

We apply the SVM with an rbf kernel to the data. Our aim is to evaluate this form of SVM as a generic model. This kernel offers non-linear modelling. Preprocessing involved deletion of the index variables and normalisation of all others. Training data was randomly selected from the clean, fully observed data. Size was 3000 units for regression, 2000 units per class for classification. In the case of multi-classification, there is a binary SVM trained for each class versus all the others. The classification with the largest positive margin wins.

5.4 Parameters to set before training

SARS data contains a mixture of discrete and scalar variables. C and σ must be set for both types. For scalar variables, ϵ is also set.

parameter settings	cross-validated
σ	8 12 16 20 24
C	5 20 80 300
ϵ	0 0.01 0.05 0.1 0.2

5.5 Training Data for Regression

At least 30% of the data is complete. MATLAB can hold only 65000 records in memory at a time, so we dealt with the SARS data in chunks. A chunk of 30000 records is loaded and a randomly selected subset is chosen for training. For regression we take 3000 of the observed units.

5.6 Training Data for Classification

Multi-classification and classification were more complicated tasks. This was due to the very unbalanced class distributions. For example, the 4th variable bath indicates whether the household has exclusive use of a bathroom (1) , shared use (2) or none at all (3).The relative frequencies of these values are;

100 : 0.7 : 0.01. Less than 1% of people share a bathroom. Less than one in ten thousand have no bathroom at all. It is likely that there is no region of the input space where value 1 is not the mode. The SVM minimises root-mean-square error or mean-absolute-error on the training set. This will be achieved by classifying all units to the modal class. Rare classes are therefore 'swamped'. No imputations are made to them.

5.7 Overview of Complicating Factors

It was more complicated to handle for the following reasons:

1. The data set is relatively large, resulting in long processing times.
2. The data set contains variables with some very rare classes. In some cases these classes had to be ignored. Indeed in one case the variable was not imputed at all as the extremely unbalanced distribution meant no imputations were made to the non- modal classes.
3. -9 values are not to be imputed. This was stated in the experimental scenario.
4. Some variables are derived. This means that the absence of one always implies the absence of the other.
5. The data set is hierarchical. Each unit contains information concerning a person, and also information concerning their household. This information is duplicated for every member of the household. The i.i.d. assumption does not hold as a result of this hierarchical structure.

Each of these issues complicated the imputation process, and required extra-code to be developed. The main role of the extra code was in the selection and preparation of suitable training data for each variable.

5.8 Results

Below we present results for selected discrete-valued variables from SARS Y2. *hhsptype* takes values in {1-14}. There is an ordering in these values, but the variable is treated here as categorical. 20% of units take value *detached*, 40% are *semidetached*, 30% are *terraced*, and 6% are *residential flats*. All others categories together make up the remaining 4% of units. This unbalanced distribution is common to many of the SARS variables.

All Data		hhsptype	ltill	mstatus	residsta	econprim	isco1
SVM	D	0.347	0.053	0.156	0.068	0.211	0.614

For the evaluation data we have no results to make comparisons with at this stage. As a rough marker, we present results below for a modal imputation on data for region 2. We know the true values for this dataset.

Let us assume that the missingness is MCAR. We would then expect a strategy of imputing to the modal class to result in an error rate of 0.6. The result, tabulated below, calculated from region 2 data is close to this. On the evaluation data, the SVM achieves an error rate of about half this value; 0.35.

region 2		hhsptype	ltill	mstatus	residsta	econprim	isco1
modal	D	0.605	0.141	0.590	0.051	0.637	0.841

workplace describes the location of a respondents workplace. For this variable 86% of the units occupy the modal class. The modal imputation therefore achieves an error rate of 0.14. The SVM achieves 0.15 training set error, and 0.21 test set error. It seems that the SVM was able to find no functional relationship for this variable.

var	params	Slope	R^2	dL1	dL2	dLinf
age	C=20 $\sigma = 9$ $\epsilon = 0.2$	1.010	0.938	3.354	5.703	67.000
		K-S	m_1	m_2	MSE	
		0.024	0.238	58.016	0.001	
var	params	Slope	R^2	dL1	dL2	dLinf
hours	C=50 $\sigma = 12$ $\epsilon = 0.05$	0.963	0.346	6.364	8.544	22.000
		K-S	m_1	m_2	MSE	
		0.455	1.182	54.000	0.001	

5.9 Summary

The census is a hierarchical dataset. Within a household the distribution of each member is highly dependent on the other members. For example, the age of a child will be strongly related to the age of a parent. The SVM system applied here does not exploit this structure.

The household variables will also be dependent upon the value of each of the members of that household. For example, the number of cars will be dependent upon the ages of all of the members of the household, and their employment status. This dependency is also ignored by the SVM.

The present implementation is therefore best judged as applied to a subset of the data. It would make sense to compare the SVM with other techniques on single-person households only. For this subset the data can be assumed to be 'flat', and the units iid.

The algorithm failed to discover strong dependencies for the household variables, apart from *hhspace*. The classes are not well separated, and often massively dominated by the modal class. Scalar variables are more successfully modelled. This is possibly due to scalar noise 'cancelling out'.

Tables are given in the Appendix for SARS Y3. Results were comparable for with SARS Y2. The perturbations did not seem to strongly affect the performance of the algorith.

6 Discussion

6.1 Motivation for investigating SVMs

SVMs offer highly flexible non-linear interpolation and discrimination models. When modelling a univariate target variable conditional upon a number of fully observed covariates, the SVM algorithm is able to make efficient use of the available data and find non-linear dependencies in high dimensional data where they exist. They have produced good results on some well-known classification and regression problems[10]. Moreover they are theoretically well founded, offering means of bounding the generalisation error. The training and testing process requires the solution of a convex quadratic programme. Methods exist which scale to millions of data points. A training data set of dimension 5000x 20 units requires 1minutes on a 450Mhz PC. The number of parameters is low allowing quick cross-validation.

Imputation is one approach to the 'Missing Data Problem'. The ultimate goal is to be able to extract statistics accurately and efficiently from a dataset missing values on some or all variables. Imputation is only one way of solving the problem; by inserting 'surrogate' values into those variables, which are unobserved. The goal of the imputation ask is to preserve the true statistics and to do this the imputations should approximate as accurately as possible draws from the joint probability density. If multiple imputation is to be attempted more than one draw is made for each missing value. Our approach is to model each variable that is missing values separately. I.e. we train a separate SVM for each variable, each giving us the conditional expectation of the target variable in the case of regression, or the maximum aposteriori class in the case of classification. Our assumption must be therefore that each conditional distribution is unimodal, as we are assuming a functional relationship to underlie the data. Our model selection procedure examines performance on cross-validation subsets of the data, picking the model that gives best rmse or best mae. If a functional relationship relates some variables, the SVM should find it.

6.2 Assessing performance

These datasets are of varying size and complexity. DLFS required considerably less time to tackle than ABI and SARS. It had only one variable missing values the scalar income variable. Preprocessing for scalar variable imputation is less onerous. SARS and ABI required more complex data selection and preprocessing routines. SARS is a hierarchical dataset. SVMs

are designed for i.i.d. data, and SARS clearly does not satisfy this.

The success of an imputation approach is highly dependent on how data-handling tasks are carried out. There is no easy way diagnose whether poor performance is due to poor preprocessing choices, although benchmarking with simpler techniques that do not require the same order of preprocessing is helpful.

To measure imputation performance requires several criteria. There is no clearly defined measure of success for all missing data problems. In general terms we expect the SVM to perform well when measured according to preservation of true values (e.g. root-mean-square-error). We expect the SVM to produce imputations that underestimate noise, and so give less good preservation of distribution

6.3 Future Work

Investigation of the poor ABI results on *stockbeg* and *stockend* indicate that the strong correlation between these two variables leads to poor results on many units because they are simultaneously absent. The patching heuristic is clearly too naive. As an improvement the nearest-neighbours approach will be investigated. As future work we also intend to investigate noise estimation, and preserving the variance. Our approach assumes a sharply peaked unimodal conditional probability distribution for each target variable. We do not address ways of measuring the strength of these assumptions. If $P(Y|X)$ is unimodal but very broad, we might develop a means of estimating the noise in the conditional probability distribution and adding it to the imputations. We do not have an answer to the problem of conditional distributions that are multi-modal. We achieved large values for m_2, in the evaluation experiments, showing poor preservation of the second moment. Clearly if preservation of variance was paramount, the present SVM setup would not be the optimal technique. We see these experiments as focused on the estimation of true values: d1error, d2error and slope.

Generally we do not expect the missing data mechanism to be MCAR. I.e. the test units are not iid with the training units. Hence training error estimates are often poor indicators of the accuracy of the imputation action itself. It seems important that diagnostics are developed; techniques derived for estimating the quality of an imputation action, and rating it in terms of confidence and credibility.

7 Conclusions

Many experiments were conducted on DLFS development data comparing group-mean, neural net and linear algorithms with SVM. Three experiments were conducted on the evaluation data. Stratification and variable selection gave additional information about the ability of the algorithm to handle the data. The results for the imputation of *income* indicate that the non-linear correlation has been well captured by the SVM. Results using much simpler non-linear approaches presented by CBS indicate however that the SVM's great flexibility is not fully exploited. A piecewise linear model with an additional interaction terms and age^2 achieved similar results. This kind of modelling is more time-consuming and could be said to require more skill than the SVM approach. It was encouraging to note that the presence of 'noise' variables, did not adversely affect the performance.

ABI experiments were less conclusive. The skewness of the data lead to surprisingly large values being chosen for the kernel parameter σ by cross-validation. The low dL1 values and good R^2 values for many variables seem to show that the linear relationship has been found. However the crude patching method combined with the strong correlations has lead to some surprisingly bad results for *stockbeg* and *stockend*. A nearest neighbours approach to patching is one avenue of investigation. ABI Y3 data have not been discussed. The effect of the perurbations was huge, and the performance statistics showed the model had produced highly unreliable imputations.

The SARS experiments showed good results for the scalar variables, *age* and *hours worked*. Some categorical variables gave reasonable error rates. But a large number performed worse than the modal imputation. Many variables are dominated by the modal class, and there may have been little structure to find.

A more informative experiment would compare performance on single person households, where the hierarchical structure could be ignored, and household based imputation would not be an option.

As expected, distributions were not well preserved. Large values of m_2 were observed and sometimes K-S, the Kolmogorov Smirnoff measure showed a value of 0.5 (50% difference between true and estimated cumulative distribution functions). The SVM imputes at the conditional expectation, or the MAP. Unless residuals are added, the distribution of imputations will be expected to be much more compressed than the distribution of true values.

References

[1] C. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, Oxford, 1995.

[2] R. E. Bellman. *Adaptive Control Processes.* Princeton UNiversity Press, 1961.

[3] M. Seeger. Pac-bayesian generalization error bounds for gaussian process classification. Technical Report EDI-INF-RR-0094, Division of Informatics, Edinburgh University, 2002.

[4] Bankier. Nearest neighbour imputation methodology. Technical report, Canadian Office of Statistics, 1998.

[5] R. Herbrich, T. Graepel, and C. Campbell. Bayes point machines: Estimating the bayes point in kernel space, 1999.

[6] J. L. Schafer. *Analysis of Incomplete Multivariate Data.* Number 72 in Monographs on Statistics and Applied Probability. Chapman and Hall, London, 1997. ISBN: 0412040611.

[7] V.N. Vapnik. *The Nature of Statistical Learning Theory.* Springer, New York, 1995.

[8] V. Vovk A. Gammerman and V. N. Vapnik. Learning by transduction. In G. Cooper and S. Moral, editors, *Uncertainty in Artificial Intelligence Proceedings of the 14th Conference*, pages 148–155, 1998.

[9] L. Bottou Y. LeCun, L. D. Jackel and A. Brunot. Comparison of learning algorithms for handwritten digit recognition. In F. Fogelman-Soulie and P. Gallinari, editors, *Proceedings ICANN'95 - International Conference on Artificial Neural Networks*, pages 53–60, 1995.

[10] T. Joachims. Text categorization with support vector machines. In *Proceedings of European Conference on Machine Learning (ECML)*, 1998.

[11] I. M. Guyon B. E. Boser and V. N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152, 1992.

[12] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall, 1997.

[13] N. Cristianini and Shawe-Taylor J. *An Introduction to Support Vector Machines*. CUP, 2000.

[14] Volker Tresp, Subutai Ahmad, and Ralph Neuneier. Training neural networks with deficient data. In Jack D. Cowan, Gerald Tesauro, and Joshua Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 128–135. Morgan Kaufmann Publishers, Inc., 1994.

[15] Zoubin Ghahramani and Michael I. Jordan. Learning from incomplete data. Technical Report AIM-1509, MIT Center for Biological and Computational Learning, 1994.

[16] R. J. Vanderbei. Loqo: An interior point code for quadratic programming. Technical Report SOR-94-15, Statistics and Perations Research, Princeton University, 1994.

[17] B. A. Murtagh and M. A. Saunders. Minos 5.1 user's guide. Technical Report SOL-83-20R, Stanford University, CA, USA, 1983.

[18] Alex J. Smola and Bernhard Schölkpf. A tutorial on support vector regression. Technical Report NC2-TR-1998-030, GMD, 1998.

[19] Y. Bengio and F. Gingras. Recurrent neural networks for missing or asynchronous data. *Advances in Neural Information Processing Systems*, 1996.

[20] C. Burges B. Schölkopf and V. Vapnik. Extracting the support data for a given task. *Proceedings First International Conference on Knowledge Discovery and Data Mining*, 1995.

[21] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics 21*, 1953.

[22] A. Smola B. Schölkopf and K. Müller. Kernel principal component analysis. *Advances in Kernel Methods - Support Vector Learning*, 1999.

[23] A. Gammerman C. Saunders and V. Vovk. Ridge regression learning algorithm in dual variables. *Machine Learning: Proceeding of the Fifteenth International Conference*, 1999.

[24] J. Platt. Fast training of support vector machines using sequential minimal optimzation. *Advances in Kernel Methods - Support Vector Learning*, 1996.

[25] A. N. Tikhonov. On solving ill-posed problem and method fo regualrization. *Doklady Akademii Nauk USSR*, 1963.

[26] T. Poggio and F. Girosi. Regularization algorithms for learning that are equivalent to multilayer networks. *Science*, 317, 1990.

[27] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society B*, 39:1 – 38, 1977.

[28] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Knowlege Discovery and Data Mining*, pages 100 – 130, 1998.

[29] G. E. Box and G. C. Tiao. *Bayesian Inference in Statistical Analysis*. J. Wiley and Sons, New York, wiley classics library edition edition, 1992.

[30] D. B. Rubin. http://www.statsol.ie. SOLAS 2.0.

[31] D. B. Rubin. *Multiple Imputation for Nonresponse in Surveys*. Wiley Series in Probability and Mathematical Statistics. Wiley, New York, 1987. ISSN: 0271-6232.

[32] R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. Wiley, New York, 1987.