

Contents

Contents	3
Introduction	7
Forward Selection Component Analysis	11
Introduction to Forward Selection Component Analysis	12
The Mathematics and Code Examples	16
Maximizing the Explained Variance	18
Code for the Variance Maximization Criterion	20
Backward Refinement	24
Multi-Threading Backward Refinement	28
Orthogonalizing Ordered Components	36
Putting It All Together	39
Components From a Forward-Only Subset	44
Components From a Backward Refined Subset	46
An Example With Contrived Variables	48
Local Feature Selection	53
Intuitive Overview of the Algorithm	54
What This Algorithm Reports	60
A Brief Detour: the Simplex Algorithm	62
The Linear Programming Problem	63
Interfacing to the Simplex Class	64
A Little More Detail	67
A More Rigorous Approach to LFS	69
Intra-Class and Inter-Class Separation	73
Computing the Weights	77
Maximizing Inter-Class Separation	81
Minimizing Intra-Class Separation	86
Testing a Trial Beta	88
A Quick Note on Threads	93
CUDA Computation of Weights	94
Integrating the CUDA Code Into the Algorithm	95
Initializing the CUDA Hardware	97
Computing Differences from the Current Case	100

Computing the Distance Matrix	102
Computing the Minimum Distances.	104
Computing the Terms for the Weight Equation	112
Transposing the Term Matrix	113
Summing the Terms For the Weights	114
Moving the Weights to the Host	116
An Example of Local Feature Selection.	117
A Note on Run Time.	118
Memory in Time Series Features	119
A Gentle Mathematical Overview	122
The Forward Algorithm	123
The Backward Algorithm.	128
Correct Alpha and Beta, For Those Who Care	131
Some Mundane Computations	136
Means and Covariances	136
Densities	138
The Multivariate Normal Density Function	139
Starting Parameters	141
Outline of the Initialization Algorithm	141
Perturbing Means	142
Perturbing Covariances	143
Perturbing Transition Probabilities.	144
A Note on Random Number Generators	145
The Complete Optimization Algorithm	146
Computing State Probabilities.	147
Updating the Means and Covariances	151
Updating Initial and Transition Probabilities	153
Assessing HMM Memory in a Time Series.	159
Linking Features to a Target	164
Linking HMM States to the Target	173
A Contrived and Inappropriate Example	183
A Sensible and Practical Example	186

Stepwise Selection on Steroids	189
The Feature Evaluation Model	192
Code For the Foundation Model	193
The Cross-Validated Performance Measure	198
The Stepwise Algorithm	201
Finding the First Variable	207
Adding a Variable to an Existing Model	210
Demonstrating the Algorithm Three Ways	214
Nominal-to-Ordinal Conversion	217
Implementation Overview	221
Testing For a Legitimate Relationship	222
An Example From Equity Price Changes	223
Code for Nominal-to-Ordinal Conversion	227
The Constructor	228
Printing the Table of Counts	232
Computing the Mapping Function	234
Monte-Carlo Permutation Tests	237
VarScreen Manual	243
About the VarScreen Program	244
Features of the Program	245
About CUDA Processing	246
Reading a Dataset	248
Univariate Mutual Information	250
Specifying the Test Parameters	253
Examples of Univariate Mutual Information	256
Bivariate Mutual Information / Uncertainty Reduction	260
Specifying the Test Parameters	262
Examples of Bivariate Mutual Information	265
Predictors having Max Relevance, Min Redundancy	270
Specifying the Test Parameters	271
An Example of Relevance Minus Redundancy	273

Hidden Markov Models with Target Correlation	278
Specifying the Test Parameters	279
Operation of This Test.	281
A Contrived and Inappropriate Example of This Test.	282
A Sensible and Practical Example	285
Assessing HMM Memory in a Time Series.	288
Specifying the Test Parameters	289
Stationarity Test for Break in Mean	291
Serial Correlation and Cyclic Permutation	296
FREL: Feature Weighting as Regularized Energy-Based Learning.	300
FREL Operation in VarScreen	307
CUDA Considerations	312
FSCA: Forward Selection Component Analysis	313
LFS: Local Feature Selection	322
What This Algorithm Reports	328
Specifying the Test Parameters	330
An Example of Local Feature Selection	333
A Note on Run Time	334
Enhanced Stepwise Lin-Quad.	335
The Feature Evaluation Model	337
The Cross-Validated Performance Measure	337
Specifying the Test Parameters	339
Demonstrating the Algorithm Three Ways	341
Nominal-to-Ordinal Conversion	344
Implementation Overview	347
Testing For A Legitimate Relationship.	348
An Example From Equity Price Changes.	349
Index	353

Introduction

Serious data miners are often faced with thousands of candidate features for their prediction or classification application, with most of the features being of little or no value. Worse still, many of these features may be useful only in combination with certain other features while being practically worthless alone or in combination with most others. Some features may have enormous predictive power, but only within a small, specialized area of the feature space. The problems that plague modern data miners are endless.

My own work over the last 20+ years in the financial market domain has involved examination of decades of price data from hundreds or thousands of markets, searching for exploitable patterns of price movement. I could not have done it without having a large and up-to-date collection of data mining tools.

Over my career I have accumulated many such tools, and I still keep up to date with scientific journals, exploring the latest algorithms as they appear. In my recent book “Data Mining Algorithms in C++”, published by the Apress division of Springer, I presented many of my favorite algorithms. I now continue this presentation, divulging details and source code of key modules in my *VarScreen* variable screening program that has been made available for free download and frequently updated over the last few years. The topics included in this text are:

Hidden Markov models are chosen and optimized according to their multivariate correlation with a target. The idea is that observed variables are used to deduce the current state of a hidden Markov model, and then this state information is used to estimate the value of an unobservable target variable. This use of memory in a time series discourages whipsawing of decisions and enhances information usage.

Forward Selection Component Analysis uses forward and optional backward refinement of maximum-variance-capture components from a subset of a large group of variables. This hybrid combination of principal components analysis with stepwise selection lets us whittle down enormous feature sets, retaining only those variables that are most important.

Local Feature Selection identifies predictors that are optimal in localized areas of the feature space but may not be globally optimal. Such predictors can be effectively used by nonlinear models but are neglected by many other feature selection algorithms that require global predictive power. Thus, this algorithm can detect vital features that are missed by other feature selection algorithms.

Stepwise selection of predictive features is enhanced in three important ways. First, instead of keeping a single optimal subset of candidates at each step, this algorithm keeps a large collection of high-quality subsets and performs a more exhaustive search of combinations of predictors that have joint but not individual power. Second, cross validation is used to select features, rather than using the traditional in-sample performance. This provides an excellent means of complexity control, resulting in greatly improved out-of-sample performance. Third, a Monte-Carlo permutation test is applied at each addition step, assessing the probability that a good-looking feature set may be not good at all, but rather just lucky in its attainment of a lofty performance criterion.

Nominal-to-ordinal conversion lets us take a potentially valuable nominal variable (a category or class membership) that is unsuitable for input to a prediction model, and assign to each category a sensible numeric value that can be used as a model input.

As is usual in my books, all of these algorithm presentations begin with an intuitive overview, continue with essential mathematics, and culminate in complete, heavily commented source code. In most cases, one or more sample applications are shown to illustrate the technique.

The *VarScreen* program, available as a free download from my website TimothyMasters.info, implements all of these algorithms and many more. This program can serve as an effective variable screening program for data mining applications.