Computational Intelligence, Volume 59, Number 000, 2010

FORECASTING FINANCIAL TIME-SERIES WITH GRAMMAR GUIDED FEATURE GENERATION

ANTHONY MIHIRANA DE SILVA, RICHARD I.A. DAVIS, SYED A. PASHA, PHILIP H.W. LEONG

School of Electrical and Information Engineering, University of Sydney, NSW 2006, Australia

The application of machine learning techniques to forecast financial time-series is not a recent development, yet it continues to attract considerable attention due to the difficulty of the problem which is compounded by the non-linear and non-stationary nature of the time-series. The choice of an appropriate set of features is crucial to improve forecasting accuracy of machine learning techniques. In this paper, we propose a systematic way for generating rich features using context-free grammars. Our proposed methodology identifies potential candidates for new technical indicators that consistently improve forecasts compared to some well-known indicators. The notion of grammar families as a compact representation to generate a rich class of features is exploited and implementation issues are discussed in detail. The proposed methodology is tested on closing price data of major stock market indices and the forecasting performance is compared with some standard techniques. A comparison with the conventional approach using standard technical indicators and naive approaches is shown.

Key words: Context-free grammar; Feature generation; Financial time-series; Machine learning; Time-series forecasting.

1. INTRODUCTION

Machine learning (ML) techniques for time-series analysis and forecasting have been extensively studied in a number of application domains including financial (Hall, 1994; Cheng et al., 1996; Tay and Cao, 2001; Huang et al., 2004) and energy markets (Mohandes, 2002; Liang and Noore, 2004; Espinoza et al., 2005) (see Sapankevych and Sankar (2009); Krollner et al. (2010) for a comprehensive list of applications). The inherent non-linear and non-stationary nature of financial time-series makes ML techniques more suitable for analysis than traditional model-based approaches (Brockwell and Davis, 2002; Franke et al., 2008; Tsay, 2005; Lai and Xing, 2008). The performance of ML techniques depends crucially on a suitable choice of feature vectors to predict the target signal (Cheng et al., 1996; Tay and Cao, 2001). In most cases these are selected by experienced users and domain experts.

The traditional approach is to use standard technical indicators and/or external economic factors as input features to ML algorithms. Technical indicators are formulae developed from models for price and volume that identify patterns and market trends in financial markets (Schabacker, 1930). Kim (2003); Lu et al. (2009); Huang and Tsai (2009) used these indicators as features in the support vector machine (SVM) (Vapnik, 1999; Cortes and Vapnik, 1995) to forecast financial time-series. Technical indicators have also been used as features in neural networks (Kim, 2003; Kamruzzaman and Sarker, 2003; Zapranis, 2006). Lendasse et al. (2000); Ince and Trafalis (2004) applied non-linear dimension reduction to the standard technical indicators and used the projected indicators as features in the SVM and neural net-

works to forecast stock prices. More recent works include Kim et al. (2006); Shen et al. (2011); Ritanjali et al. (2009).

The aforementioned works were empirical studies that identify a subset of the technical indicators which improved forecasts, the selection of the technical indicators and their parameters being ad-hoc. It would therefore be useful to have a framework that can automatically generate useful features by systematically guiding the generation of a pool of candidate features that have a meaningful interpretation.

A general framework for function-based feature generation using context-free grammars (CFG) was first proposed by Markovitch and Rosenstein (2002). Such grammars are used in linguistics to describe sentence structure and words of a natural language and in computer science to describe the structure of programming languages (Sipser, 1997). Unfortunately, the technique is only suitable for problems where the features are apparent from the problem description. Eads et al. (2005) and Pachet and Roy (2009) addressed supervised time-series classification using standard genetic programming to discover a set of fundamental signal processing operations via a grammatical structure. Both these works conclude that conventional classifiers trained using raw data as features can be outperformed by training the same classifiers with grammar generated features. Standard genetic programming was also used by Ritthof et al. (2002) to combine feature generation and feature selection and applied to the interpretation of chromatography time-series. Ritthof et al. used arithmetic operators in their grammar to expand the feature space while Eads et al. extracted time-series information using operators such as the mean, delay, derivative, integral, etc. The genetic programming approach by Guo et al. (2005) used a set of mathematical transformation operators, (e.g. sin, cos, +, -, sqrt, etc) to

produce features of the raw vibration signals from a rotating machine for fault classification. A similar approach has been applied to breast cancer diagnosis (Hong and Nandi, 2005). Both works reported improved classification accuracies. Islamaj et al. (2006) proposed a feature generation algorithm for splice-site prediction in genomics and reported a 6% improvement compared to the state-of-the-art approaches. Krawiec and Bhanu (2005) used a co-evolutionary feature generation approach where a multiple population was evolved simultaneously. Operators applicable to images, e.g. filters, image norms, scalar operators etc. were defined and the best operator sequences, i.e. processing steps embedded in chromosomes, were used for synthetic aperture radar (SAR) image recognition. The approach was shown to be robust under different operating conditions.

In this paper we propose, for the first time, a CFG for forecasting stock market data. Our proposed approach generates interpretable indicators (features) that consistently improve forecasts compared to using standard technical indicators. We also identify potential candidates for new technical indicators. Our approach is flexible in that (i) any ML technique can be used for predictions (ii) users can easily control the number of features generated without requiring any expertise, while experienced users can adjust the grammar to incorporate domain specific knowledge.

We demonstrate our proposed approach using standard ML tools for regression to forecast the closing price of major world stock market indices.

This paper is organized as follows. Section 2 provides some background on standard technical indicators for financial time-series and gives a brief overview of the CFG. Section 3 presents the main part of our work on developing grammar families to generate features. Implementation issues are described in detail. An empirical

study is provided in Section 4. The paper is concluded with some final remarks in Section 5.

Notation and Acronyms The subscript k denotes the current time. O, H, L and C are the opening, highest, lowest and closing prices respectively and V is the traded volume for the day. The typical (average) price M = (H+L+C)/3. Let $\Delta P_k \equiv P_k - P_{k-1}$, then the upward and downward price changes are $\mathrm{U}_k = max(0, \Delta C_k)$ and $\mathrm{D}_k = min(0, \Delta C_k)$ respectively. The money flow $\mathrm{F} = \mathrm{M} \times \mathrm{V}$ and the positive and negative flows are given by $\mathrm{F}_k^+ = max(0, \Delta F_k)$ and $\mathrm{F}_k^- = min(0, \Delta F_k)$ respectively. σ and $\bar{\sigma}$ are the standard and mean deviations of the typical price M. Grammars are parameterised by lag, l, and look-back window size n.

AORD $\equiv$ All Ordinaries, FTSE $\equiv$ FTSE-100 Index, GDAXI $\equiv$ DAX Index, GSPC $\equiv$ S&P-500 Standard and Poor's Index, HSI $\equiv$ Hang Seng Index, NDX $\equiv$ NASDAQ-100, N225 $\equiv$ NIKKEI 225, SSEC $\equiv$ Shanghai Stock Exchange Composite, SSMI $\equiv$ Swiss Market Index, TWII $\equiv$ Taiwan Weighted Index.

2. BACKGROUND

2.1. Technical Indicators

Technical indicators are formulae that identify patterns and market trends in financial markets developed from models for price and volume (Schabacker, 1930).

Technical indicators can be broadly classified as trend, momentum, volatility and volume indicators (Edwards et al., 2007). A trend analysis studies price charts using the moving average filters which give smooth price estimates and identifies overall trend patterns. The exponential moving average (EMA), simple moving average

(SMA) and weighted moving average (WMA) filters are some of the standard trend indicators. Momentum measures the variation of price at a given time. Momentum indicators identify overbought and oversold positions and start of new trends. Rate of convergence (ROC), relative strength index (RSI) and average directional index (ADX) are some commonly used momentum indicators. Volatility indicators like Bollinger bands identify the uncertainty in the market via statistical variance of price movements. Volume indicators identify the volumes of trade that have the potential to cause market movements. The money flow index (MFI) is one such example.

Some standard technical indicators are shown in Tables 1, 2. Table 1 summarizes some well-known trend, volatility and volume indicators with typical parameter values while Table 2 summarizes momentum indicators. Table 3 summarizes 12 studies that use technical indicators as regressors in standard ML tools.

2.2. Context-free Grammar

A context-free grammar (CFG) is a simple mechanism to generate patterns and strings using hierarchically organized production rules. Using the Backus-Naur form (BNF), a formal notation for context-free grammars, a CFG can be described by the tuple $(\mathcal{T}, \mathcal{N}, \mathcal{R}, \mathcal{S})$ where $\mathcal{T}$ is a set of terminal symbols and $\mathcal{N}$ is a set of non-terminal symbols with $\mathcal{N} \cap \mathcal{T} = \varnothing$. The non-terminal symbols in $\mathcal{N}$ and terminal symbols in $\mathcal{T}$ are the lexical elements used in specifying the production rules $\mathcal{R}$ of a CFG. A non-terminal symbol is one that can be replaced by other non-terminal and/or terminal symbols. Terminal symbols are literal values that symbols in $\mathcal{N}$ can take. A terminal symbol cannot be altered by the grammar rules $\mathcal{R}$. $\mathcal{R}$ is a set of relations (also referred to as production rules) in the form of $\mathcal{R} \rightarrow \alpha$ with $\mathcal{R} \in \mathcal{N}$,

$\alpha \in (\mathcal{N} \cup \mathcal{T})$. $\mathcal{S}$ is the start symbol $\mathcal{S} \in \mathcal{N}$. If the grammar rules are defined as $\mathcal{R} = \{x \rightarrow xa, x \rightarrow ax\}$, a is a terminal symbol since no rule exists to change it whereas x is a non-terminal symbol. A language is context-free if all of its elements are generated based on a context-free grammar. It is well-known that most programming languages are modelled using context-free grammars and that compilers are developed based on context-free languages. If $\mathcal{S}$ is the starting symbol and we define a context-free grammar, $\mathcal{T} = \{a, b\}$, $\mathcal{N} = \{\mathcal{S}\}$ and $\mathcal{R} = \{\mathcal{S} \rightarrow a\mathcal{S}b, \mathcal{S} \rightarrow ab\}$, $\mathcal{L} = \{a^n b^n | n \in \mathbb{Z}^+\}$ is a context-free language.

2.3. Feature Selection

Feature selection (FS) is the process of choosing a subset of features that improves the performance of the ML technique. Formally, for N data samples with M features in each data sample, the FS problem is to find from the M-dimensional observation space R^M, a subspace of m features R^m that best predict the target. This is achieved by removing irrelevant, redundant, and noisy features from the set.

Because the total number of subspaces is 2^M, in general finding an optimal feature subset is intractable (Kohavi and John, 1997). Many problems related to FS have been shown to be NP-hard (Huan and Lei, 2005). FS strategies are essentially twofold; (i) ranking features according to some criterion and (ii) collectively choosing a feature subset (Guyon, 2003). Feature ranking assigns a weight to each feature and feature subset selection evaluates different feature combinations.

A filter model uses intrinsic characteristics of the data for feature ranking. Commonly used critera are the information gain, Pearson's correlation, minimum-redundancy-

maximum-relevance (mRMR) (Ding and Peng, 2003) and the Relief algorithm (Robnik-Šikonja and Kononenko, 2003).

The mRMR method is based on mutual information (Hanchuan et al., 2005). Maximum relevance is assessed where S is a set of M features such that $|S| = M$ and $I(x_i, c)$ is the mutual information between feature x_i and target c. Relevance is represented via the formula

$$D(S, c) = \frac{1}{|S|} \sum_{x_i \in S} I(x_i, c),$$

and redundancy using

$$R(S) = \frac{1}{|S|^2} \sum_{x_i, x_j \in S} I(x_i, x_j).$$

The mRMR method is

$$\max_S \Phi(S),\ \Phi(S) = D(S) - R(S).$$

Incremental search methods are used to find a near-optimal feature subset by maximising $\Phi(\cdot)$.

2.3.1. *Integer Genetic Algorithms.* For large dimensional feature spaces, a filter model can be first applied to reduce the dimensionality in a greedy manner. Subsequently, a wrapper-based feature selection technique which considers interactions between features in detail can be used.

Integer genetic algorithms (GA) operate on n-tuples of integer strings I_i. The n-tuple is termed the population, and the integers represent genes in the chromosomes. Each chromosome is a candidate solution for the formalized problem. The low scoring chromosomes are discarded from the population and replaced with new chromosomes (children). The probability for a population I_i to be selected for recombination

is proportional to the relative fitness score given by, $p_s = \phi(I_i) / \sum_{j=0}^{n} \phi(I_j)$, where $\phi(.)$ is the fitness function to be optimised.

GAs have been used as a wrapper technique thus introducing a search mechanism to avoid enumerating the entire space. A simple approach is to encode features in the chromosome, e.g. the chromosome 01001000 could mean that the 2nd and 5th features are selected (Yang and Honavar, 1998; Il-Seok Oh et al., 2004). Grammar-based GA optimization can be facilitated using our open source gramEvol package (Noorian et al., 2015) which generalises the technique described in this paper.

3. METHODOLOGY

3.1. Feature generation using CFG

We define hierarchical grammar structures with different layers to guide the feature generation flow. The layered organization of operators that we use to generate features is shown in Table 4. At time k, the base layer consists of the m observed variables $x_k^{(1)}, \ldots, x_k^{(m)}$ and the p derived variables $f_k^{(1)}, \ldots, f_k^{(p)}$. In the context of our application O, H, L, C and V are the observed variables. The derived variables are M, D, U, H^+, L^-, $F^{\pm}$ and $i^{\pm}$ which are terms that appear in the formulae for the technical indicators in Tables 1, 2. In the absence of any information other than the target (closing price C) history, the proposed methodology will only involve the derived variables in the base layer.

The transformation layer comprises the base and running operators defined in Table 5. The base operators are simple operators such as the first difference and the absolute value. The running operators compute values using a moving window. For

example, `max(x,n)` will calculate the maximum value of the past n days for each-point in the time-series, i.e. $max(C_k, n) = max(C_k, C_{k-1}, \ldots C_{k-n})$. Similarly, `sd(x,n)` will calculate moving standard deviations for each point of the time-series using a look-back window of size n. The combinatorial operators fuse information across variables to produce more features which are passed to the user defined layers.

Domain transformations (e.g. wavelets, Fourier) can also be useful for constructing better features. For example, the wavelet transformation has been used for multi-resolution analysis of stock data to capture information on different time-scales that is not obvious from the original time-series (Huang and Wu, 2008).

3.2. Generating Technical Indicators using Grammar Families

A CFG can be used to organize production rules to guide feature generation while maintaining variable compatibility. The base layer consists of inhomogeneous quantities such as price, time and volume. By using properly organized CFG families we are able to maintain compatibility by combining them in a meaningful way. The use of grammar families instead of a single grammar can minimize the generation of less informative features. Unlike genetic programming (Guo et al., 2005), CFG facilitates visualization of the feature generation flow which helps to monitor the generated features. The operators defined in Table 5 suffice to generate rich features with the technical indicators summarized in Tables 1, 2 as particular cases. The 7 grammar families used in this work are given in Tables 6-12. The grammars use the notation n for look-back window size and l for lag. For example, the running operator `sd` applied to the closing price with a look-back window size of n is denoted as `sd(C,`n`)`. This operation produces different features for different look-

back window sizes of n. Similarly, the lagged closing price is denoted as `lag(C,`l`)` which lags the closing price by l days. If l=1 this means we use the closing price of the previous day (C_{k-1}) as a feature to predict the closing price of the next (C_{k+1}).

Recall that a CFG can be described by $(\mathcal{T}, \mathcal{N}, \mathcal{R}, \mathcal{S})$. The production rules $\mathcal{R}$ for grammar family 1 are organized into 3 groups. Group 1 has 5 rules (1.a)-(1.e), group 2 has 2 rules (2.a), (2.b) and so on. Each production rule has a head (left hand side), a non-terminal symbol in $\mathcal{N}$, that is assigned by the string of symbols in the body (right hand side). Multiple production rules in the same group are delimited by the pipe "|". For grammar family 1, we have 3 non-terminal symbols denoted in the production rules by $<\cdot>$, namely <*L1*>, <*L2*> and <*L3*>.

Features are generated by invoking the production rules sequentially. The generated features are mapped to the different layers in Table 4. Fig. 1 shows the steps to generate the technical indicator A/D oscillator. The start symbol for grammar family 1 is <*L3*>. By invoking rule (1.b) on <*L3*>, the intermediate non-terminal expression $(<$*L2*$>) \div (<$*L2*$>)$ is produced. Since there are 2 individual non-terminal elements in this intermediate expression, the leftmost non-terminal is always chosen (indicated by the circle around it). With the closing price C lagged by $l = 1$ (C_{k-1}) and the lowest price L lagged by $l = 0$ (L_k) we obtain the formula for the A/D oscillator.

Fig. 2 illustrates the generation of the Disparity indicator using the grammar family 2. With the closing price C lagged by $l = 0$ (C_k) in both terms we obtain the Disparity. Other technical indicators are generated in a similar fashion.

The CFG based framework is flexible in that: (i) the number of grammar families and the organization of the production rules can be adapted. (ii) The user is able to design a sufficiently large grammar to capture as much information as possible with

a manageable feature space. (iii) The user can incorporate domain knowledge by choosing appropriate derived variables and production rules.

All the technical indicators in Tables 1 and 2 can be obtained from the production rules of the 7 grammar families, this being detailed in Table 13.

Pseudocode for enumeration of the symbolic features is given in Algorithm 1.

Algorithm 1 Language Parsing

function LANGUAGE PARSER(nt1, nt2) ▷ inputs are non-terminals of grammar
  $exprList = \{\}$
  $n \leftarrow$ Number of production rules of nt1
  for i in 1:n **do**
    $currExpr \leftarrow RHS(nt1[i])$ ▷ Right hand side of the production rule
    $S = \{RHS(nt2[1]), RHS(nt2[2]), \dots\}$
    $exprList = \{exprList, AllPossibleExpressions(S, currExpr)\}$
  return $exprList$

3.3. Pruning Strategies

Carefully designing the grammar structure using the grammar families helps to keep the number of features generated within a manageable size. Some of the features are parametrized by a window of size n and lag l. Clearly, the number of different features explode with increasing the number of values that n and l can take. Simple pruning procedures can ensure the feature generation is tractable and only informative features are generated in a systematic way. These are outlined below.

(i) Limiting the number of production rules at each step avoids explosion of the feature space. Grammar families were designed such that each family generates features from a class of indicators.

(ii) Separate production rules for numerator and denominator terms in fractional

forms can be used. This helps to reduce all possible permutations using a single production rule, e.g. in grammar families 3 and 4.

(iii) Avoiding invalid operations in a production rule avoids generating meaningless features, e.g. adding price to volume.

(iv) For feature expressions with many possible combinations, only the N best numerical features can be selected based on an appropriate criterion e.g. mRMR.

(v) Features that exhibit little variation can be dropped.

Pseudocode for generation of numerical features is given in Algorithm 2.

Algorithm 2 Feature Expression Generation

function BUILD FEATURE EXPRESSIONS(n[], l[]) ▷ n = window-sizes, l = lags
  $N \leftarrow 7$ ▷ Number of grammar families
  for i in 1:N **do**
    $h \leftarrow height(grammar[i])$ ▷ h = number of levels in a grammar family
    $exprList = \{RHS(grammar[i].level[1])\}$
    for $gramLvl$ in 2:h **do**
      $count1 = Count(exprList)$
      $count2 = RuleCount(grammar[i].level[gramLvl])$
      $exprListTemp = \{\}$
      for $gramRuleBottom$ in 1:$count1$ **do**
        $nt1 = exprList[gramRuleBottom]$
        $topExprList = RHS(grammar[i].level[gramLvl])$
        for $gramRuleTop$ in 1:$count2$ **do**
          $nt2 = topExprList[ruleCountTop]$
          $exprListTemp$ = $\{exprListTemp,$ LANGUAGE PARSER($nt1$, $nt2$) $\}$
        $exprList = exprListTemp$
  $exprlist$ = $RemoveDuplicates(exprlist)$
  for k in 1:$exprlist$ **do** ▷ k is a feature expression
    if k contains n, l **then**
      Generate features $\forall\ n, l$ combinations of $k(n, l)$
  $[V]_k = mRMR(eval(exprlist))$ ▷ Expressions to numeric features
  $[V]_{1000} = sort([V]_k)[1:1000]$ ▷ Highest 1000 mRMR ranked features
  return $[V]_{1000}$

3.4. Feature Selection

The proposed CFG based framework generates a large number of features parametrized by the window size n and lag l. Some features have no parameters while others have one or both. We used $n = 5, 15, 30$ and $l = 0, 1, \ldots, 6$. This means that multiple features can be generated from an expression, i.e. the expression `sd(lag(C,l),n)` leads to 21 different features considering all $n \times l$ combinations. It is important to select the appropriate (n, l) for each feature to obtain the most informative features.

A range of FS techniques were explored to compare the performance. The filter FS techniques used were information gain, mRMR, correlation and Relief. The individual feature goodness for each feature was assessed against the target variable, e.g. the closing price of stock index time-series. Once the features were ranked, an appropriate number of features were used for the prediction task. Dimensionality reduction using the principal component analysis (PCA) was also considered. mRMR followed by integer GAs was found to give the best results.

Integer GA mutation was done according to the criterion $x(1 + u)$, where x was the current gene and $u \sim \mathcal{U}(-.05, .05)$. The crossover was performed on 2 chromosomes selected using the roulette-wheel criterion (also called steady-state selection). In roulette-wheel selection, the probability of selecting the ith chromosome, denoted with b_i, follows a Bernoulli distribution by $p = \phi(b_i)/\sum_{j=0}^{n} \phi(b_j)$, where $\phi(\cdot)$ is the cost function. Each gene of the chromosome b_i represents a feature number hence a chromosome represents a feature subset. This feature subset was used to train and validate a SVM and the Root-Mean-Squared-Error (RMSE) of the feature subset was considered as the cost of the chromosome $\phi(b_i)$. Single-point crossover was used. A

crossover point in the integer string was chosen at random and the string sections of the two parents were exchanged to produce a child chromosome.

3.5. Predicting Financial Time-series

In a typical application of ML to time-series prediction, it is common practice to divide the time-series into training, validation, and testing (out-of-sample) sets. The training set was used to construct a model. The validation set was used to evaluate the generalization ability of the trained model. The model parameters were tuned such that the model performs satisfactorily on the validation set.

The sliding or moving window (also known as walk-forward testing) is a form of online training where the model is frequently retrained. The number of samples in the testing set determines the retraining frequency of the model. For one-step ahead predictions, this means that the model can be retrained after every prediction. The data is divided into a series of overlapping training-validation-testing sets. The typical training-validation-testing concept is still present, but now only the most recent observations are used to construct models. We have found this to be more robust.

An SVM with radial basis function (RBF) kernel $K(x, y) = exp(-\gamma\|x - y\|^2)$, where γ is a user-selected parameter, is used for prediction (Vapnik, 1999; Cortes and Vapnik, 1995). This kernel maps input samples to a high-dimensional feature space and is a universal approximator widely used in machine learning. Cross-validation via parallel grid-search, genetic algorithms, random search, heuristics search and inference of model parameters within the Bayesian evidence framework are some

parameter search techniques. The performance of different parameter combinations is assessed by the learner performance, e.g. mean squared error.

K-fold cross validation, 2-fold cross validation, leave-one-out cross validation and repeated random sampling cross validation are some popular cross validation techniques. Time-series cross validation is slightly different because the data are not independent and leaving an observation out does not remove all the associated information due to the correlations with other observations. We used extensive parameter combination assessment via a parallel grid-search on the validation data. Parameter tuning using the validation data prevents the over-fitting problem. The final performance of the learner was evaluated using the best parameters in the validation phase. Time-series cross validation (Hyndman, 2010) was done by first fitting the model to the data $y_1, \ldots, y_t$, and then forecasting $\hat{y}_{t+1}$. The root mean square error (RMSE), $e_t^* = \sqrt{(\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2)}$, is then computed and this repeated for $t = m, \ldots, n-1$ where m is the minimum number of observations needed to fit the model. Finally, the average RMSE is computed as a figure of merit.

The proposed feature selection procedure hence begins with the generation of a large set of numerical features which are pruned to the top 1000 using mRMR as described in Algorithm 2. A GA and SVM is then used to perform further prune to N features using the wrapper-based feature selection approach described in Algorithm 3.

In order to quickly discover better feature subsets, specific chromosomes were placed in the initial population ($suggestions$ in Algorithm 3) which were known to work well in general, e.g. standard technical indicators. This is possible since the rule sequence to generate a specific technical indicator (or a feature in general) is known.

This ensures that the initial population is healthy and encourages the generation of high performing feature subsets. The rest of the population consists of randomly generated individuals.

Algorithm 3 Feature Selection using Integer Genetic Algorithm, $trainData$ is the training set and $vData$ the validation set.

function FEATURE SELECTION($[V]_{1000}, N$) ▷ N = number of desired features
  $chroms \leftarrow \{\}$ ▷ chromosomes, initially empty
  for s in 1:$|suggestions|$ **do**
    $chroms = \{chroms, suggestions[s]\}$
  for r in $suggestions$:$popSize$ **do** ▷ $popSize = 100$
    $chroms = \{chroms, GenerateRandomChrom()\}$
  $bestChroms \leftarrow \{\}$
  $error[1 : popSize] = 0$
  for i in 1:$iterations$ **do** ▷ $iterations = 50$
    $chroms = bestChroms$
    for j in 1:$popSize$ **do** ▷ $popSize = |chroms|$
      $feats = MappedFeatures(chrom_j)$
      $trained = TrainSVM(trainData, eval(feats))$
      $tuned_{C,\gamma} = TuneSVMModel(vData, trained)$
      $error_j = RMSE(vData, tuned_{C,\gamma})$ ▷ RMSE of j^{th} chrom
    $bestChroms = RouletteWheelSelection(chroms, error)$
    $bestChroms = Crossover(bestChroms, 0.3)$ ▷ Probability 0.3
    $bestChroms = Mutate(bestChroms, 0.01)$ ▷ Probability 0.01
  $rankedErrors = Rank(bestChroms)$
  $bestChrom = IndexOf(rankedErrors[1])$
  $bestFeats = MappedFeatures(bestChrom)$
return $bestFeats$

4. DATA ANALYSIS

This section explores the effectiveness of using standard technical indicators as features and attempts to discover better numeric features, i.e. new technical indicator type formulae, that can give better predictions for a particular ML algorithm.

We tested the performance of our proposed methodology for forecasting the daily

closing prices of major stock market indices. The data comprise daily recordings between 23 October 1998 and 15 May 2006. Of the 1900 trading days, data from 1500 days were used for training, 200 for validation and the remaining 200 for prediction. The `quantmod` package (Jeffrey, 2013) in R was used to extract the data as well as the time-series of the technical indicators.

The daily closing price of stock indices were forecasted using a support vector machine (SVM) with Gaussian kernel. The grammar generated features were used as inputs and a comparison with standard technical indicators made. For a fair comparison the number of features used in both cases was 25. Feature subset selection using the GA was repeated 10 times with different initialization. The results were averaged over the 10 runs.

In the case of the SVM, the parameters were selected by a grid search in the region $C = \{1, 10i, i = 1, 2, \ldots, 10\}$ for the regularization parameter and $\gamma = \{2^{1-2i}, i = 1, 2, \ldots, 8\}$ for the kernel parameter. The search was parallelized on multiple cores using the `snowfall` package in R (Knaus, 2010). The SVM was implemented using the `e1071` package (Meyer et al., 2012).

4.1. Results and Discussion

For a given stock index, feature selection using the GA method was repeated 10 times with different initialization and a histogram was constructed. The results for 4 indices are shown in Table 14, with the standard technical indicators (TIs) indicated in bold. We find for each stock index only a very small number of standard TIs appear in the grammar generated features. For example, for GSPC only Disparity

was found to be significant. For the SSMI no standard TIs were selected and for the FTSE only Bias and Disparity were found to be moderately significant.

Table 15 is a list of grammar generated features arranged in descending order of the frequency of selection, obtained by aggregating the results for the 10 indices considered in Table 16. We find that only 7 of the 25 standard TIs appear in the list, the other 57 are grammar generated features. The other frequently selected TIs were K (9), ROC (8), OSCP (8), Slow K (8), SMA (8), ATR (6), R (6), ADO (4) and Chaikin volatility (4).

One pattern apparent from Table 15 is `X-(sd(lag(Y,k),n)` where `X,Y` $\in \{C, H, L, M\}$. This occurs 5 times in the top 10 most frequent features. Similarly, `X-(Y-lag(Z,k)/n)` where `X,Y,Z` $\in \{C, H, L, M\}$, occurs 6 times in the top 20. These are two examples of a new and salient indicators discovered by our technique.

To gauge the performance of the ML techniques, a comparison was made with the traditional model-based approaches such as the AR(1), EMA with window size $p = 5, 10, 15$, exponential time-series smoothing (ETS) and the ARIMA. The parameters for the ETS and the ARIMA models were chosen using the `forecast` package (Hyndman et al., 2013). Table 16 summarizes the out-sample RMSE for the 10 indices, and compares them with an SVM using standard technical indicators, and SVM using the proposed feature generation technique. From Table 16, we make the following observations:

(1) With the exception of NDX, forecasts for all major stock indices were more ac-

curate using the machine learning approaches compared to traditional model-based approaches.

(2) With the exception of SSEC, TWII and GDAXI where the SVM with standard TIs as features produced minimum error, for all other major indices including NDX, the SVM with grammar based features performed better. For SSEC and TWII the difference in the performance was only marginal, .51 and .28 respectively.

(3) The advantage of the proposed approach is most pronounced for FTSE where an improvement in performance of 7.63 was recorded. Using standard TIs as features led to a performance below that of model-based methods except EMA.

(4) For FTSE, the performance of ETS comes closest to our proposed approach. However, we find that for all other indices, the difference in the performance is more significant.

(5) For NDX, the model-based methods give a higher accuracy. This is most likely due to some structure in the time-series where there is dependence on the history. This is supported by the RMSE which is considerably smaller compared to other indices which seems to suggest that such models provide a reasonable fit to the data. However, we observe that the performance of the model-based methods is only marginally better than our technique.

5. CONCLUSION AND FUTURE WORK

The forecast accuracy of ML techniques is highly dependent on the choice of suitable input feature vectors. We applied context-free grammars to automatically generate a large pool of informative features in the form of customized technical

indicators, and introduced the notion of grammar families as a compact representation to generate a rich class of features. The proposed approach is flexible in that the number of grammar families and the organization of the production rules can be adapted. Users can tune the grammar and incorporate domain specific knowledge by choosing appropriate derived variables and production rules. We discussed in detail the practical issues and the implementation procedure. Empirical results using major stock market indices showed improvements over the standard approach of picking a set of standard technical indicators as input features.

The grammar developed in this work was designed to generate only a limited set of technical indicators. Advanced technical indicators can be generated by combining probabilistic context-free grammar and genetic algorithms. The grammar can also be applied to transformations such as the wavelets of the time-series. This will be pursued in future work. In addition, robust feature selection for non-stationary time-series using ensemble approaches and non-linear dimension reduction techniques will also be investigated further.

In this study, feature subset selection was done on the validation data. It is assumed that the subset is optimal for the forecast interval. However, due to the dynamics of the time-series it is likely that such a subset will not be optimal over the entire forecasting interval. This is evident from GDAXI, SSEC and TWII where the grammar based SVM does not perform better than the TI-based SVM. It is expected that by regularly repeating the feature subset selection the accuracy of the ML techniques using the grammar generated features can be further improved. The details of how the subset selection can be done dynamically and the choice of the interval over which selection needs to be repeated will be investigated elsewhere.

Improved genetic algorithms for feature selection tailored to this technique will assist in making the technique more efficient on more complex problems with more features. In combination with increasing computing power, the range of potential applications of this technique is likely to increase rapidly.

The proposed approach is general and is not limited to financial time-series. Other time-series can be explored and appropriate grammars can be developed. By using the proposed approach, an insight into features that work well can be obtained hence experts can use the proposed feature generation framework in any application to supplement their own set of manually selected features.

[Table 1 about here.]

[Table 2 about here.]

[Table 3 about here.]

[Table 4 about here.]

[Table 5 about here.]

[Table 6 about here.]

[Table 7 about here.]

[Table 8 about here.]

[Table 9 about here.]

[Table 10 about here.]

[Table 11 about here.]

[Table 12 about here.]

[Table 13 about here.]

[Table 14 about here.]

[Table 15 about here.]

[Table 16 about here.]

[Figure 1 about here.]

[Figure 2 about here.]

ACKNOWLEDGMENT

This research was supported by the Australian Research Council's Linkage Projects funding scheme (project number LP110200413); funding from the Faculty of Engineering & Information Technologies, The University of Sydney, under the Faculty Research Cluster Program; and Westpac Banking Corporation.

REFERENCES

BROCKWELL, P.J., and R.A. DAVIS. 2002. Introduction to Time-series and Forecasting. Springer.

CHENG, W., W. WAGNER, and C.H. LIN. 1996. Forecasting the 30 Year US Treasury Bond with a System of Neural Networks. Journal of Computational Intelligence in Finance, **4**:10–16.

CORTES, C., and V. VAPNIK. 1995. Support-vector Networks. Machine Learning, **20**(3):273–297.

DING, CHRIS, and HANCHUAN PENG. 2003. Minimum redundancy feature selection from microarray gene expression data. *In* Proceedings of the IEEE Computer Society Conference on Bioinformatics, CSB '03, IEEE Computer Society, Washington, DC, USA. ISBN 0-7695-2000-6. pp. 523–. http://dl.acm.org/citation.cfm?id=937976.938050.

EADS, D., K. GLOCER, S. PERKINS, and J. THEILER. 2005. Grammar-guided Feature Extraction for Time-series Classification. *In* Proceedings of the 9th Annual Conference on Neural Information Processing Systems (NIPS).

EDWARDS, R.D., J. MAGEE, and W.H.C. BASSETTI. 2007. Technical Analysis of Stock Trends. CRC Press.

ESPINOZA, M., J. SUYKENS, and B. DE MOOR. 2005. Load Forecasting Using Fixed-size Least Squares Support Vector Machines. *In* Computational Intelligence and Bioinspired Systems. Springer, pp. 1018–1026.

FRANKE, J., W. HÄRDLE, and C.M. HAFNER. 2008. Statistics of Financial Markets: An Introduction. Springer.

GUO, H., L.B. JACK, and A.K. NANDI. 2005. Feature Generation using Genetic Programming with Application to Fault Classification. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, **35**(1):89–99.

GUYON, ISSABELLE. 2003. An Introduction to Variable and Feature Selection. Journal of Machine Learning Research, **3**:1157–1182.

HALL, J.W. 1994. Adaptive Selection of US Stocks with Neural Nets. *In* Trading on The Edge: Neural, Genetic, and Fuzzy Systems for Chaotic Financial Markets. Wiley, pp. 45–65.

HANCHUAN, P., L. FUHUI, and D. CHRIS. 2005. Feature Selection based on Mutual Information Criteria of Max-dependency, Max-relevance, and Min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence, **27**(8):1226–1238.

HONG, G., and A.K. NANDI. 2005. Breast Cancer Diagnosis Using Genetic Programming Generated Feature. *In* IEEE Workshop on Machine Learning for Signal Processing, pp. 215–220.

HUAN, L., and Y. LEI. 2005. Toward Integrating Feature Selection Algorithms for Classification and Clustering. IEEE Transactions on Knowledge and Data Engineering, **17**(4):491 – 502.

HUANG, CHENG-LUNG, and CHENG-YI TSAI. 2009. A Hybrid SOFM-SVR with a Filter-based Feature Selection for Stock Market Forecasting. Expert Systems with Applications, **36**(2):1529–1539.

HUANG, SHIAN-CHANG, and TUNG-KUANG WU. 2008. Integrating GA-based Time-scale Feature Extractions with SVMs for Stock Index Forecasting. Expert Systems with Applications, **35**(4):2080–2088.

HUANG, Z., H. CHEN, C.J. HSU, W.H. CHEN, and S. WU. 2004. Credit Rating Analysis with Support Vector Machines and Neural Networks: A Market Comparative Study. Decision Support Systems, **37**(4):543–558.

HYNDMAN, R.J. 2010. Why Every Statistician Should Know About Cross-validation. http://robjhyndman.com/hyndsight/crossvalidation/.

HYNDMAN, R.J., G. ATHANASOPOULOS, S. RAZBASH, D. SCHMIDT, Z. ZHOU, Y. KHAN, and C. BERGMEIR. 2013. forecast: Forecasting Functions for Time-series and Linear Models. http://CRAN.R-project.org/package=forecast. R package version 4.06.

IL-SEOK OH, JIN-SEON LEE, and BYUNG-RO MOON. 2004. Hybrid Genetic Algorithms for Feature Selection.

IEEE Transactions on Pattern Analysis and Machine Intelligence, **26**(11):1424 –1437.

INCE, H., and T.B. TRAFALIS. 2004. Kernel Principal Component Analysis and Support Vector Machines for Stock Price Prediction. *In* In Proceedings of IEEE International Joint Conference on Neural Networks, Volume 3, pp. 2053–2058.

ISLAMAJ, R., L. GETOOR, and W.J. WILBUR. 2006. A Feature Generation Algorithm for Sequences with Application to Splice-site Prediction. *In* Knowledge Discovery in Databases: PKDD 2006. Springer, pp. 553–560.

JEFFREY, A. R. 2013. quantmod: Quantitative Financial Modelling Framework. http://CRAN.R-project.org/package=quantmod. R package version 0.4-0.

KAMRUZZAMAN, J., and R.A. SARKER. 2003. Forecasting of Currency Exchange Rates using ANN: A Case Study. *In* Proceedings of the International Conference on Neural Networks and Signal Processing, Volume 1, IEEE, pp. 793–797.

KIM, KYOUNG-JAE. 2003. Financial Time-series Forecasting using Support Vector Machines. Neurocomputing, **55**(1):307–319.

KIM, M., S. MIN, and H. INGOO. 2006. An Evolutionary Approach to the Combination of Multiple Classifiers to Predict a Stock Price Index. Expert Systems with Applications, **31**(2):241–247.

KNAUS, J. 2010. snowfall: Easier Cluster Computing (Based on snow). http://CRAN.R-project.org/package=snowfall. R package version 1.84.

KOHAVI, R., and G.H. JOHN. 1997. Wrappers for Feature Subset Selection. Artificial Intelligence, **97**(1):273–324.

KRAWIEC, K., and B. BHANU. 2005. Visual Learning by Coevolutionary Feature Synthesis. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, **35**(3):409–425.

KROLLNER, B., B. VANSTONE, and G. FINNIE. 2010. Financial Time-series Forecasting with Machine Learning Techniques: A Survey. *In* European Symp. ANN: Computational and Machine Learning, pp. 25–30.

LAI, T.L., and H. XING. 2008. Statistical Models and Methods for Financial Markets. Springer.

LENDASSE, A., E. DE BODT, V. WERTZ, and M. VERLEYSEN. 2000. Non-linear Financial Time-series Forecasting - Application to the BEL 20 Stock Market Index. European Journal of Economic and Social Systems, **14**(1):81–91.

LIANG, T., and A. NOORE. 2004. A Novel Approach for Short-term Load Forecasting using Support Vector Machines. International Journal of Neural Systems, **14**(05):329–335.

LU, C., T. LEE, and C. CHIU. 2009. Financial Time-series Forecasting using Independent Component Analysis and Support Vector Regression. Decision Support Systems, **47**(2):115–125.

MARKOVITCH, S., and D. ROSENSTEIN. 2002. Feature Generation using General Constructor Functions. *In* Machine Learning, Volume 49, MIT Press, pp. 59–98.

MEYER, D., E. DIMITRIADOU, K. HORNIK, A. WEINGESSEL, and F. LEISCH. 2012. e1071: Misc Functions of the Department of Statistics (e1071), TU Wien. http://CRAN.R-project.org/package=e1071. R package version 1.6-1.

MOHANDES, M. 2002. Support Vector Machines for Short-term Electrical Load Forecasting. International Journal of Energy Research, **26**(4):335–345.

NOORIAN, FARZAD, ANTHONY M. DE SILVA, and PHILIP H.W. LEONG. 2015. gramEvol: Grammatical Evolution in R. *In* Journal of Statistical Software. To appear.

PACHET, F., and P. ROY. 2009. Analytical Features: A Knowledge-based Approach to Audio Feature Generation. EURASIP Journal on Audio, Speech, and Music Processing, **2009**(1):1–23.

RITANJALI, M., G. PANDA, B. MAJHI, and G. SAHOO. 2009. Efficient Prediction of Stock Market Indices using Adaptive Bacterial Foraging Optimization (ABFO) and BFO Based Techniques. Expert Systems with Applications, **36**(6):10097 – 10104.

RITTHOF, O., R. KLINKENBERG, S. FISCHER, and I. MIERSWA. 2002. A Hybrid Approach to Feature Selection and Generation using an Evolutionary Algorithm. *In* U.K. Workshop on Computational Intelligence, pp. 147–154.

R.K., LAI, CHIN-YUAN FAN, WEI-HSIU HUANG, and PEI-CHANN CHANG. 2009. Evolving and Clustering Fuzzy Decision Tree for Financial Time-series Data Forecasting. Expert Systems with Applications, **36**(2):3761–3773.

ROBNIK-ŠIKONJA, MARKO, and IGOR KONONENKO. 2003. Theoretical and empirical analysis of relieff and rrelieff. Mach. Learn., **53**(1-2):23–69. ISSN 0885-6125. . http://dx.doi.org/10.1023/A:1025667309714.

SAPANKEVYCH, N., and R. SANKAR. 2009. Time-Series Prediction using Support Vector Machines: A Survey. IEEE Computational Intelligence Magazine, **4**(2):24–38.

SCHABACKER, R.W. 1930. Stock Market Theory and Practice. BC Forbes Publishing Company.

SHEN, W., X. GUO, C. WU, and D. WU. 2011. Forecasting Stock Indices using Radial Basis Function Neural Networks Optimized by Artificial Fish Swarm Algorithm. Knowledge-Based Systems, **24**(3):378 – 385.

SIPSER, M. 1997. Context-Free Grammars. *In* Introduction to the Theory of Computation. PWS Publishing, pp. 91–122.

TAY, F.E.H., and L. CAO. 2001. Application of Support Vector Machines in Financial Time-series Forecasting. Omega, **29**(4):309–317.

TSAY, R.S. 2005. Analysis of Financial Time-series, Volume 543. Wiley-Interscience.

VAPNIK, V. 1999. The Nature of Statistical Learning Theory. Springer.

YANG, J., and V. HONAVAR. 1998. Feature Subset Selection using a Genetic Algorithm. IEEE Intelligent Systems and Their Applications, **13**(2):44–49.

YU, L., S. WANG, and LAI KIN-KEUNG. 2005. Mining Stock Market Tendency Using GA-based Support Vector Machines. *In* Lecture Notes in Computer Science 3828, pp. 336–345.

ZAPRANIS, A. 2006. Testing the Random Walk Hypothesis with Neural Networks. *In* Artificial Neural Networks ICANN, Volume 4132 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pp. 664–671.

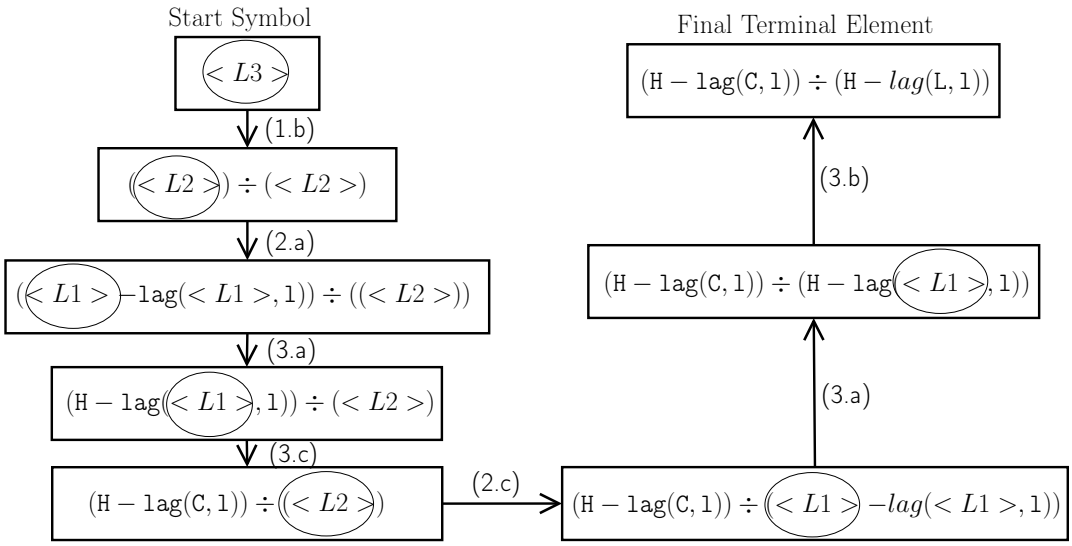

FIGURE 1: Step-wise generation of A/D oscillator indicator.

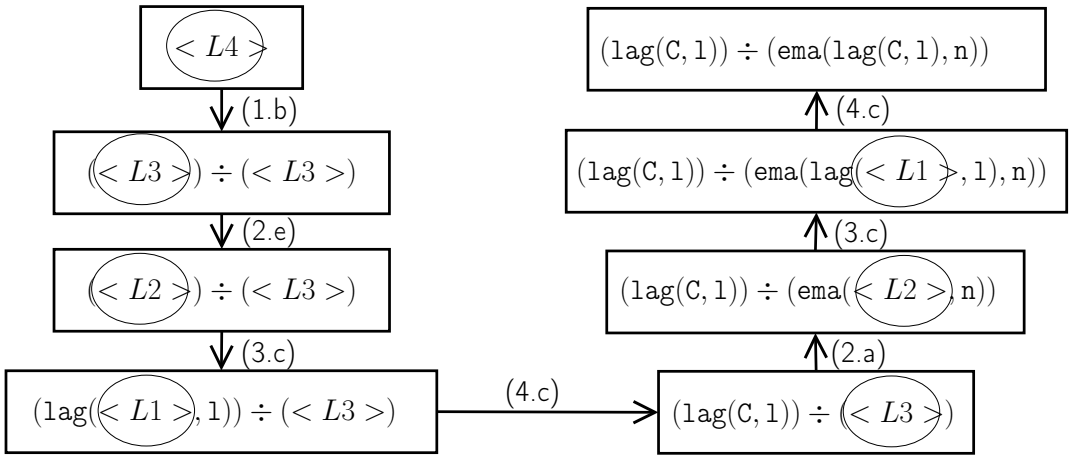

FIGURE 2: Step-wise generation of the disparity indicator.

Indicator name	Acronym	Formula	Parameters
Simple moving average	SMA	$\frac{1}{n}\sum_{i=0}^{n-1} P_{k-i}$	$n = 5, 15, 30$
Weighted moving average	WMA	$\frac{1}{n}\sum_{i=0}^{n-1} (n-i)P_{k-i}$	$n = 15, 5, 30$
Exponential moving average	EMA	$\sum_{i=0}^{n-1} \alpha(1-\alpha)^i P_{k-i}$	$n = 5, 15, 30$ $\alpha = 2/(n+1)$
Disparity	DIS	$\mathrm{P_k}/\mathrm{ema}(\mathrm{P}_k, n)$	$n = 5, 10$
Bias	BIAS	$(\mathrm{P_k} - \mathrm{sma}(\mathrm{P_k}, n))/n$	$n = 5, 10$
Bollinger bands	BB	$\mathrm{sma}(\mathrm{P}_k, n) \pm 2\sigma$	$n = 15$
Chaikin volatility	-	$\mathrm{sma}(\mathrm{H}_k - \mathrm{L}_k, n) \pm 2\sigma$	$n = 15$
Average true range	ATR	$max(\mathrm{H}_k - \mathrm{L}_k, \lvert\mathrm{H}_k - \mathrm{C}_{k-n}\rvert, \lvert\mathrm{L}_k - \mathrm{C}_{k-1}\rvert)$	-
Money flow index	MFI	$(1 + \mathrm{R})/\mathrm{R}$	$n = 14$
Aroon indicator (Up, Down)	-	$(n - i^+)/n\ ,\ (n - i^-)/n$	$n = 6, 12, 24$

Table 1: Standard trend, volatility and volume indicators.

Money ratio R = $\Sigma_0^{n-1}\mathrm{F}_i^+/\mathrm{F}_i^-$. Given price P, $i^+ = \arg\max_i \mathrm{P}_{k-i}$ and $i^- = \arg\min_i \mathrm{P}_{k-i}$, $i = 0, 1, \ldots, n-1$.

Indicator name	Acronym	Formula	Parameters
Rate of convergence	ROC	$(\mathrm{C}_k - \mathrm{C}_{k-n})/\mathrm{C}_{k-n}$	$n = 1$
Commodity channel index	CCI	$(\mathrm{M}_k - \mathrm{sma}(\mathrm{M}_k, n))/0.015\bar{\sigma}$	$n = 15$
Relative strength index	RSI	$\mathrm{RS}/(1 + \mathrm{RS})$	$n = 14$
Moving average convergence divergence	MACD	$\mathrm{ema}(\mathrm{C}_k, n_1) - \mathrm{ema}(\mathrm{C}_k, n_2)$, $n_1 > n_2$	$n_1 = 26, n_2 = 12$
Momentum	MOM	$\mathrm{C}_k - \mathrm{C}_{k-n}$	$n = 4$
William's indicator	R	$(\mathrm{H}^+_{k-n} - \mathrm{C}_k)/(\mathrm{H}^+_{k-n} - \mathrm{L}^-_{k-n})$	$n = 15$
Stochastic oscillator	K	$(\mathrm{C}_{k-n} - \mathrm{L}^-_{k-n})/(\mathrm{H}^+_{k-n} - \mathrm{L}^-_{k-n})$	$n = 14$
Stochastic indicator	D	$\mathrm{sma}(\mathrm{K}(n_1), n_2)$	$n_1 = 15, n_2 = 5$
Slow stochastic indicator	Slow D	$\mathrm{sma}(\mathrm{D}(n_1, n_2), n_3)$	$n_1 = 15, n_2 = 3, n_3 = 3$
Close location value	CLV	$((\mathrm{C}_k - \mathrm{L}_k) - (\mathrm{H}_k - \mathrm{C}_k))/(\mathrm{H}_k - \mathrm{L}_k)$	-
Price Oscillator	OSCP(n_1, n_2)	$(\mathrm{sma}(\mathrm{P}_k, n_1) - \mathrm{sma}(\mathrm{P}_k, n_2))/\mathrm{sma}(\mathrm{P}_k, n_1)$	$n_1 = 5, n_2 = 10$
Accumulation/Distribution Oscillator	ADO	$(\mathrm{H}_k - \mathrm{C}_{k-1})/(\mathrm{H}_k - \mathrm{L}_k)$	-

Table 2: Standard momentum indicators.

$\mathrm{RS} = \mathrm{ema}(\mathrm{U}, n)/\mathrm{ema}(\mathrm{D}, n)$. $\mathrm{H}^+_{k-n} = max(H_{k-i})^{n-1}_{i=0}$ and $\mathrm{L}^-_{k-n} = min(L_{k-i})^{n-1}_{i=0}$.

Technical indicators used as features	# of TIs	Data	Method	Reference
K, D, Slow D, MOM, ROC, R, ADO, DIS, OSCP, RSI, CCI	13	KOSPI	SVM	Kim (2003)
O, H, L, C, V, RSI, WMA	7	N225, TAIEX	ICA + SVM	Lu et al. (2009)
SMA, RSI, PSY, MOM, D, VR, OBV, DIS, ROC	9	KOSPI	GA	Kim et al. (2006)
C, K, D, Slow D, ROC, MOM, SMA, σ, σ ratio, EMA, MACD, ADO, DIS, OSCP, CCI, RSI	18	S&P 500	SVM + GA	Yu et al. (2005)
SMA, BIAS, RSI, K, D, MACD, PSY, V	8	TSEC Stocks	K-means + FDT + GA	R.K. et al. (2009)
SMA, EMA, Projection oscillator, MACD, Qstick, TRIX, etc.	54	DJI, S&P 500	ANN	Zapranis (2006)
SMA, RSI, K, D, MACD, R, PSY, $\Gamma^{\pm}$, BIAS, VR, A ratio, B ratio	13	FITX	SOM + SVM	Huang and Tsai (2009)
Returns, differences of returns, oscillators, SMA, EMA	42	BEL 20	KPCA + RBF-NN	Lendasse et al. (2000)
EMA, ADO, K, RSI, ROC, Price accelerations	10	Stocks	ABFO + BFO	Ritanjali et al. (2009)
ΔC, BB, RSI, K, M, CLV, MFI, R	10	Stocks	KPCA + SVM	Ince and Trafalis (2004)
OBV, SMA, BIAS, PSY, Average stock yield, C	12	Stocks	AFSA + NN	Shen et al. (2011)
SMA, C	6	Forex	ARIMA + ANN	Kamruzzaman and Sarker (2003)

Table 3: Works using standard technical indicators as features.

OBV = On-balance volume, VR = Volume ratio, TRIX = % change of the triple smoothed moving average of the closing price, QStick oscillator and PSY = Psychological stability of investors. $\Gamma^{\pm}$ = directional movement indicators.

User defined layers		e.g. ema($c_k^{(i)}$, n)
Transformation layer elements $c_k^{(i)}$ are passed to the higher layers		
Transformation layer	Fractional combinations	e.g. $(b_k^{(i)} - b_k^{(j)})/b_k^{(j)}$
	Additive combinations	e.g $b_k^{(i)} \pm b_k^{(j)}$
	Base operators	e.g. $log(b_k^{(i)})$
	Running operators	e.g. $func(b_k^{(i)}$, n) (see Table 5)
Base layer elements $b_k^{(i)}$ are passed to the transformation layer		
Base layer	Derived variables	$f_k^{(1)}, \ldots, f_k^{(l)}$
	Observed variables	$x_k^{(1)}, \ldots, x_k^{(m)}$

Table 4: Layered organization of operators for feature generation.

Base operators		**Running operators** (window size n)	
diff(x, n)	$x_k - x_{k-n}$	sma(x,n)	simple moving average
log(x)	natural log	wilder(x,n)	Wilder exponential moving average
delt(x)	$(x_k - x_{k-1})/x_k$	ema(x,n)	exponential moving average
abs(x)	$\|x_k\|$	wma(x,n)	weighted moving average
lag(x, n)	x_{k-n}	max(x,n)	maximum value
		min(x,n)	minimum value
		sd(x,n)	standard deviation
		sum(x,n)	summation
		meandev(x,n)	mean deviation
		skewness(x,n)	skewness
		kurtosis(x,n)	kurtosis
		median(x,n)	median

Table 5: Base operators and running operators.

Family 1

$\mathcal{N}$ = {L1, L2, L3}
$\mathcal{T}$ = {$-$, $\div$, `lag`, `sma`, `meandev`, `sum`, `H`, `L`, `C`, `M`, `n`, `l`, `N`, (,) }
$\mathcal{S}$ = {L3}

Production rules : $\mathcal{R}$

$\langle L3 \rangle$	::=	$(\langle L2 \rangle) \div (\texttt{lag}(\langle L2 \rangle, \texttt{l}))$ | $(\langle L2 \rangle) \div (\langle L2 \rangle)$	(1.a), (1.b), (1.c)
	|	$((\langle L2 \rangle) - (\langle L2 \rangle)) \div \texttt{N}$ | $\langle L2 \rangle$	(1.d), (1.e)
$\langle L2 \rangle$	::=	$\langle L1 \rangle - \texttt{lag}(\langle L1 \rangle, \texttt{l})$ | $\langle L1 \rangle - \texttt{sma}(\langle L1 \rangle, \texttt{n})$	(2.a), (2.b)
	|	$\texttt{meandev}(\langle L1 \rangle, \texttt{n})$ | $\texttt{sum}(\langle L1 \rangle, \texttt{n})$ | $\langle L1 \rangle$	(2.c), (2.d), (2.e)
$\langle L1 \rangle$	::=	`H` | `L` | `C` | `M`	(3.a), (3.b), (3.c), (3.d)

Table 6: Grammar family 1.

Family 2

$\mathcal{N}$ = {L1, L2, L3, L4}
$\mathcal{T}$ = {− , ÷, `lag`, `sma`, `ema`, `wma`, `H`, `L`, `C`, `M`, `delt`, `diff`, `n`, `l`, (,) }
$\mathcal{S}$ = {L4}

Production rules : $\mathcal{R}$

⟨*L4*⟩	::=	(⟨*L3*⟩) ÷ (⟨*L3*⟩) | (⟨*L3*⟩ − ⟨*L3*⟩) | ⟨*L3*⟩	(1.a), (1.b), (1.c)
⟨*L3*⟩	::=	`ema`(⟨*L2*⟩, `n`) | `sma`(⟨*L2*⟩, `n`) | `wma`(⟨*L2*⟩, `n`)	(2.a), (2.b), (2.c)
	|	`sma`(`ema`(⟨*L2*⟩, `n`), `n`) | ⟨*L2*⟩	(2.d), (2.e)
⟨*L2*⟩	::=	`diff`(⟨*L1*⟩) | `delt`(⟨*L1*⟩) | `lag`(⟨*L1*⟩, `l`)	(3.a), (3.b), (3.c)
⟨*L1*⟩	::=	`H` | `L` | `C` | `M`	(4.a), (4.b), (4.c), (4.d)

Table 7: Grammar family 2.

Family 3

$\mathcal{N}$ = {L1, L2, L3}
$\mathcal{T}$ = {$-$, $\div$, `lag`, `sma`, `meandev`, `sum`, H_h, L_l, `C`, `n`, `k`, (,) }
$\mathcal{S}$ = {L3}

Production rules : $\mathcal{R}$

$\langle L3 \rangle$	::=	$(\langle L2 \rangle) \div (\langle L2 \rangle)$ | `sma`($\langle L2 \rangle$, `n`) | $\langle L2 \rangle$	(1.a), (1.b), (1.c)
$\langle L2 \rangle$	::=	$\langle L1 \rangle - $ `lag`($\langle L1 \rangle$, `k`) | `sma`($\langle L1 \rangle$, `n`)	(2.a), (2.b)
	|	`meandev`($\langle L1 \rangle$, `n`) | `sum`($\langle L1 \rangle$, `n`) | $\langle L1 \rangle$	(2.c), (2.d), (2.e)
$\langle L1 \rangle$	::=	H^+ | L^- | `C`	(3.a), (3.b), (3.c)

Table 8: Grammar family 3.

Family 4

$\mathcal{N}$ = {L1, L2, L3, L4, L5}
$\mathcal{T}$ = {$-$, $\div$, `lag`, `ema`, `sma`, `meandev`, `sum`, `H`, `L`, `C`, `n`, `k`, i^+, i^-, (,) }
$\mathcal{S}$ = {L5}

Production rules : $\mathcal{R}$

$\langle L5 \rangle$	::=	$(\langle L3 \rangle \div \langle L4 \rangle) \mid (\langle L3 \rangle \div$ `N`$) \mid \langle L4 \rangle$	(1.a), (1.b), (1.c)
$\langle L4 \rangle$	::=	`ema`$(\langle L1 \rangle$, `n`$) \mid$ `sum`$(\langle L1 \rangle$, `n`$) \mid$ `max`$(\langle L1 \rangle$, `n`$)$	(2.a), (2.b)
	$\mid$	`min`$(\langle L1 \rangle$, `n`$) \mid (\langle L1 \rangle) \div$ `N` $\mid \langle L1 \rangle$	(2.c), (2.b), (2.c)
$\langle L3 \rangle$	::=	$\langle L2 \rangle -$ `ema`$(\langle L2 \rangle$, `n`$) \mid$ `ema`$(\langle L2 \rangle$, `n`$) \mid$ `meandev`$(\langle L2 \rangle$, `n`$)$	(3.a), (3.b)
	$\mid$	`sum`$(\langle L2 \rangle$, `n`$) \mid$ `max`$(\langle L2 \rangle$, `n`$) \mid$ `min`$(\langle L2 \rangle$, `n`$)$	(3.c), (3.d), (3.e)
$\langle L2 \rangle$	::=	`H` $\mid$ `L` $\mid$ `C`	(4.a), (4.b), (4.c)
$\langle L1 \rangle$	::=	$i^+ \mid i^-$	(5.a), (5.b)

Table 9: Grammar family 4.

Family 5

$\mathcal{N}$ = {L1, L2, L3}
$\mathcal{T}$ = {$-$, $\div$, lag, ema, sma, meandev, sum, H, L, C, n, k, i^+, i^-, (,) }
$\mathcal{S}$ = {expr}

Production rules : $\mathcal{R}$

$\langle L5 \rangle$	::=	$\langle L3 \rangle \div (\langle L3 \rangle + \langle L4 \rangle) \mid \langle L3 \rangle \div (\langle L3 \rangle - \langle L4 \rangle)$	(1.a), (1.b), (1.c)
$\langle L4 \rangle$	::=	ema($\langle L1 \rangle$, n) | sum($\langle L1 \rangle$, n) | meandev($\langle L1 \rangle$, n) | max($\langle L1 \rangle$, n)	(2.a), (2.b)
	|	min($\langle L1 \rangle$, n) | delt($\langle L1 \rangle$)	(2.c), (2.d), (2.e)
$\langle L3 \rangle$	::=	ema($\langle L1 \rangle$, n) | sum($\langle L1 \rangle$, n) | meandev($\langle L1 \rangle$, n) | max($\langle L1 \rangle$, n)	(3.a), (3.b)
	|	min($\langle L1 \rangle$, n) | delt($\langle L1 \rangle$)	(3.c), (3.d), (3.e)
$\langle L2 \rangle$	::=	F^- | D	(4.a), (4.b)
$\langle L1 \rangle$	::=	F^+ | U	(5.a), (5.b)

Table 10: Grammar family 5.

Family 6

$\mathcal{N}$ = {L1, L2, L3}
$\mathcal{T}$ = {$-$, $\div$, `lag`, `ema`, `sma`, `meandev`, `sum`, `H`, `L`, `C`, `n`, `k`, i^+, i^-, (,) }
$\mathcal{S}$ = {L4}

Production rules : $\mathcal{R}$

Production rules : $\mathcal{R}$

$\langle L4 \rangle$	::=	$(\langle L1 \rangle - \langle L3 \rangle) \div (\langle L2 \rangle - \langle L3 \rangle) \mid \langle L2 \rangle \mid \langle L3 \rangle$	(1.a), (1.b), (1.c)
$\langle L3 \rangle$	::=	`sma`($\langle L1 \rangle$, `n`) - 2$\times$ `sd`($\langle L1 \rangle$, `n`)	(2.a)
$\langle L2 \rangle$	::=	`sma`($\langle L1 \rangle$, `n`) + 2$\times$ `sd`($\langle L1 \rangle$, `n`)	(3.a)
$\langle L1 \rangle$	::=	`H` | `L` | `C` | `H-L` | `H-C` | `C-L`	(4.a), (4.b), (4.c), (4.d), (4.e), (4.f)

Table 11: Grammar family 6.

Family 7

$\mathcal{N}$ = {L1, L2, L3}
$\mathcal{T}$ = {− , ÷, `lag`, `sma`, `H`, `L`, `C`, `M`, `n`, `k`, (,) }
$\mathcal{S}$ = {expr}

Production rules : $\mathcal{R}$

⟨*L3*⟩	::=	(⟨*L2*⟩) ÷ (⟨*L2*⟩) | (⟨*L2*⟩ − ⟨*L2*⟩) | ⟨*L2*⟩	(1.a), (1.b), (1.c)
⟨*L2*⟩	::=	`ema`(⟨*L1*⟩, `n`) | `sma`(⟨*L1*⟩, `n`) | `wma`(⟨*L1*⟩, `n`)	(2.a), (2.b), (2.c)
	|	`sma`(`ema`(⟨*L1*⟩, `n`), `n`) | ⟨*L1*⟩	(2.d), (2.e)
⟨*L1*⟩	::=	`lag`(`V`, `l`)	(3.a), (3.b)

Table 12: Grammar family 7.

Family #	Standard technical indicators generated
1	CLV, CCI, ROC, A/D Oscillator, Bias, Lagged prices
2	EMA, SMA, WMA, Lagged prices, Disparity, MACD, SD, Price Oscillator
3	R, K, D, Slow D
4	Aroon
5	RSI, MFI
6	BB, Chakin volatility
7	Volume related indicators

Table 13: Standard technical indicators generated by each grammar family.

Feature	Parameters	Freq.
$\sigma(\Delta H_k, 15)/\sigma(\delta L_k, 15)$		10
Disparity	$n = 5$	8
$(M_k - \Delta H_k)/15$		7
$\text{SMA}(L_k, n) + 2 \times \sigma(L_k, 30)$	$n = 5, 15, 30$	6
$(C_k - (M_k - H_{k-2}))/n$	$n = 5, 15$	5
$(H_k - (L_k - H_{k-2}))/n$	$n = 5, 15$	5
$C_k - \text{SMA}(\Delta H_k, 30)$		5
$H_k - \text{SMA}(\text{EMA}(\Delta C_k, 30))$		5
$\text{SMA}(C_k, 15) + 2 \times \sigma(C_k, n)$	$n = 5, 15, 30$	5
$\text{SMA}(H_k - \Delta C_k, n)$	$n = 15, 30$	5
$\text{SMA}(H_k, n) + 2 \times \sigma(H_k, 15)$	$n = 5, 15, 30$	5
$H_k - \sigma(M_{k-i}, 5)$	$i = 0, 2, 5, 6$	4
$H_{k-1} - \text{SMA}(\Delta H, 30)$		4
$L_k - \text{SMA}(\text{EMA}(\Delta L, n_1), n_2)$	$n_1 = 15, 30, n_2 = 5, 15, 30$	4
$\Sigma(L_k, n)/\max(i^+, 30)$	$n = 5, 15$	4
$C_k - \text{SMA}(\text{EMA}(\Delta L, n_1), n_2)$	$n_1 = 5, 15, n_2 = 15, 30$	4
H_{k-1}		3
$H_k - \text{WMA}(\Delta H_k, n_1)$	$n_1 = 5, 15$	3
$(L_k - (H_k - C_{k-i}))/n$	$n = 5, 15, i = 1, 2$	3
$(M_k - (L_k - C_{k-2}))/n$	$n = 5, 15$	3

(a) GSPC

Feature	Parameters	Freq.
$(C_k - (M_k - L_{k-i}))/n$	$n = 5, 15, i = 1, 2, 3, 5$	6
$(C_k - \bar{\sigma}(H_k, n_1))/n_2$	$n_1 = 5, 15, 30, n_2 = 5, 15$	6
$(C_k - (L_k - C_{k-i}))/n$	$n = 5, 15, i = 0, 1, 2, 3$	5
$(C_k - \sigma(M_{k-3}, n))$	$n = 5$	5
$\sigma(\Delta C_k, n_1)/\sigma(\delta M_k, n_2)$	$n_1 = 5, 15, 30, n_2 = 5, 15, 30$	5
$(C_k - (H_k - H_{k-i}))/n$	$n = 5, 15, i = 2, 3$	4
$C_k - \sigma(C_{k-i}, 5)$	$i = 3, 4, 5$	4
$\sigma(\Delta C_k, n_1)/\sigma(\delta C_k, n_2)$	$n_1 = 5, 15, n_2 = 5, 15$	4
$\Sigma(H_h(6), 5)/C_k$		4
$(C_k - (H_k - L_{k-i}))/n$	$n = 5, 15, i = 1, 3$	3
$(C_k - (H_k - \text{SMA}(L_k, n_1)))/n_2$	$n_1 = 5, 30, n_2 = 5, 15$	3
$(C_k - (L_k - H_{k-2}))/5$		3
$(C_k - (L_k - L_{k-i}))/n$	$n = 5, 15, i = 1, 3$	3
$(C_k - (L_k - M_{k-i}))/5$	$i = 2, 3$	3
$(C_k - (M_k - L_{k-2}))/15$		3
$(C_k - (M_k - \text{SMA}(H_k, n_1)))/n_2$	$n_1 = 5, 15, n_2 = 5, 15$	3
$(C_k - \bar{\sigma}(L_k, 30))/15$		3
$(C_k - \bar{\sigma}(M_k, n_1))/n_2$	$n_1 = 5, 30, n_2 = 5, 15$	3
$(H_k - (L_k, H_{k-i}))/15$	$i = 0, 2, 3$	3
$(H_k - \bar{\sigma}(C_k, n))/5$	$n = 5, 30$	3

(b) SSMI

Feature	Parameters	Freq.
$L_k - \sigma(H_{k-i}, n)$	$n = 5, 15, i = 1, 2, 4, 5$	6
$L_k - \sigma(M_{k-i}, n)$	$n = 5, 15, i = 0, 1, 3, 5$	6
$C_k - \sigma(L_{k-i}, 5)$	$i = 3, 5$	5
$C_k - \sigma(M_{k-i}, n)$	$n = 5, 15, i = 0, 2, 3, 6$	5
$M_k - \sigma(C_{k-i}, n)$	$n = 5, 15, i = 0, 1, 3, 6$	5
$C_k - (L_k - \text{SMA}(L_k, 30))/5$		4
$C_{k-i} - \Delta H_k$	$i = 0, 1, 2$	4
$C_k - \sigma(C_{k-i}, n)$	$n = 5, 15, i = 0, 3, 5$	4
$L_k - \sigma(C_{k-i}, n)$	$n = 5, 15, i = 3, 5, 6$	4
$M_k - \text{SMA}(\text{EMA}(\Delta L, n_1), n_2)$	$n_1 = 5, 15, n_2 = 5, 15, 30$	4
$(M_k - (L_k - H_{k-i}))/n$	$n = 5, 15, i = 1, 2, 3$	4
Bias		3
$(C_k - (H_k - H_{k-i}))/n$	$n = 5, 15, i = 1, 3$	3
$(C_k - (L_k - C_{k-i}))/15$	$i = 0, 2, 3$	3
$(C_k - \bar{\sigma}(C_{k-i}, n_1))/5$	$n_1 = 15, 30$	3
Disparity		3
$C_{k-i} - \sigma(\Delta H, 5)$	$i = 2, 3$	3
$C_k - \sigma(H_{k-i}, n)$	$n = 5, 15, i = 1, 3, 6$	3
$M_k - \sigma(L_{k-i}, 5)$	$i = 3, 4, 5$	3
$(L_k - (H_k - \text{SMA}(L_k, 5)))/n$	$n = 5, 15$	3

(c) FTSE

Feature	Parameters	Freq.
$C_k - \sigma(M_{k-i}, n)$	$n = 5, 15, i = 2, 6$	4
$L_k - \sigma(H_{k-i}, n)$	$n = 5, 15, i = 1, 3, 6$	4
$(L_k - (H_k - L_{k-i}))/n$	$n = 5, 15, i = 0, 2, 3$	4
ROC		4
$\text{SMA}(M_k, n_1) + 2 \times \sigma(M_k, n_2)$	$n_1 = 5, 15, n_2 = 5, 15, 30$	4
Aroon		3
$(C_k - (M_k - H_{k-1}))/n$	$n = 5, 15$	3
$(C_k - \bar{\sigma}(H_k, 5))/n$	$n = 5, 15$	3
Disparity		3
$(H_k - \bar{\sigma}(L_k, n))/15$	$n = 5, 15$	3
$C_k - \sigma(\Delta M_k, 5)$		3
$C_k - \text{SMA}(\Delta C_k, n)$	$n = 5, 30$	3
$C_k - \text{SMA}(\Delta H_k, n)$	$n = 5, 30$	3
$H_k - \sigma(\Delta L_k, n)$	$n = 5, 15, 30$	3
L_{k-6}		3
$L_k - \sigma(L_{k-i}, 5)$	$i = 1, 2, 6$	3
$L_k - \sigma(L_{k-i}, n)$	$n = 5, 15, i = 1, 5, 6$	3
$M_k - \sigma(H_{k-i}, n)$	$n = 5, 15, i = 1$	3
$M_k - \sigma(L_{k-i}, n)$	$n = 5, 15, i = 1, 2, 6$	3
$M_k - \sigma(M_{k-i}, n)$	$n = 5, 15, i = 1, 2$	3

(d) HSI

Table 14: Feature frequency for 4 selected indices when using wrapper based GA for feature selection.

	Feature	Freq.
1	**Disparity**	32
2	C-(sd(lag(C, l), n))	24
3	C-(sd(lag(M, l), n))	23
4	C-(sd(lag(L, l), n))	22
5	H-(sd(lag(C, l), n))	21
6	L-(sd(lag(L, l), n))	21
7	sd(diff(H), n))/(sd(delt(L), n))	21
8	(H-(M-lag(C, l)))/n	20
9	(C-(L-lag(L, l)))/n	19
10	**Bias**	18
11	L-(sd(lag(H, l), n))	18
12	(M-(H-lag(H, l)))/n	18
13	(C-(M-lag(H, l)))/n	17
14	H-(sd(lag(L, l), n))	17
15	L-(sd(lag(M, l), n))	17
16	(L-(H-lag(L, l)))/n	17
17	(sd(diff(H), n))/(sd(delt(M), n))	17
18	**Lower Bollinger Band**	17
19	(C-(meandev(M, n)))/n	16
20	(H-(L-lag(H, l)))/n	16
21	M-(sd(lag(M, l), n))	16
22	M-(sma(ema(diff(L), n), n))	16
23	(C-(H-lag(C, l)))/n	15
24	C-(sd(diff(C), n))	15
25	C-(sma(ema(diff(L), n), n))	15
26	L-(sd(diff(C), n))	15
27	sma(L, n) - 2*(sd(L, n))	15
28	(C-(H-lag(H, l)))/n	14
29	(C-(L-lag(C, l)))/n	14
30	(C-(L-lag(H, l)))/n	14
31	(C-(L-sma(L, n)))/n	14
32	**CLV**	14

Feature	Freq.	
33	(lag(M, l))-(sd(lag(H, l), n))	14
34	M-(sma(ema(diff(M), n), n))	14
35	sma(L, n) + 2*(sd(L, n))	14
36	C-(H-lag(L, l)))/n	13
37	C-(H-sma(L, n)))/n	13
38	C-(sd(lag(H, l), n))	13
39	H-(sd(lag(M, l), n))	13
40	(M-(L-lag(H, l)))/n	13
41	(sd(diff(C), n))/(sd(delt(C), n))	13
42	(sd(diff(C), n))/(sd(delt(M), n))	13
43	(sd(diff(L), n))/(sd(delt(M), n))	13
44	sma(M, n) + 2*(sd(M, n))	13
45	C-(sma(diff(H), n))	12
46	C-(sma(ema(diff(M), n), n))	12
47	H-(sma(diff(H), n))	12
48	L-(sd(lag(C, l), n))	12
49	M-(sd(lag(L, l), n))	12
50	(L-(H-lag(C, l)))/n	12
51	(M-(L-lag(C, l)))/n	12
52	(sd(diff(M), n))/(sd(delt(L), n))	12
53	**Upper Bollinger Band**	12
54	**Aroon**	11
55	(C-(H-lag(M, l)))/n	11
56	(C-(M-lag(L, l)))/n	11
57	(H-(meandev(L, n)))/n	11
58	**Lagged closing price**	11
59	H-(sd(diff(L), n))	11
60	H-(sd(lag(H, l), n))	11
61	M-(sd(lag(C, l), n))	11
62	M-(sma(ema(diff(H), n), n))	11
63	(sum(L, n))/(max(i^+, n))	11
64	sma(H, n) + 2*(sd(H, n))	11

Table 15: Technical indicators and selected grammar feature frequency.

Method	FTSE	N225	NDX	HSI	SSMI
ARIMA	34.33	207.07	14.35	138.95	47.96
ETS	33.54	207.13	14.05	139.95	47.14
AR(1)	33.71	207.21	**13.98**	140.24	47.60
EMA	42.63	275.20	14.05	183.51	61.42
SVM (TIs)	40.27	203.99	15.15	136.70	46.26
SVM (Grammar)	**32.64**	**203.30**	14.37	**135.11**	**46.17**

Method	SSEC	TWII	AORD	GDAXI	GSPC
ARIMA	19.78	69.52	29.97	44.19	7.58
ETS	19.78	69.46	29.80	44.08	7.60
AR(1)	19.71	69.41	29.97	44.18	7.61
EMA	28.16	96.55	38.81	58.26	9.39
SVM (TIs)	**18.03**	**66.85**	29.23	**43.70**	7.54
SVM (Grammar)	18.54	67.13	**28.61**	44.49	**7.49**

Table 16: RMSE for test data for major stock indices using the ARIMA, ETS, AR(1), EMA ($p = 5$) and SVM using technical indicators (TIs) and grammar features.