Dennis Carnelossi Furlaneto

An analysis of Ensemble Empirical Mode Decomposition applied to Trend Prediction on Financial Time Series

Thesis presented as a requirement to obtain a Masters degree in Computer Science from the graduation program at Universidade Federal do Paraná.

Field: *Computer Science*.

Advisor: Luiz Eduardo S. Oliveira.

Co-advisor: David Menotti.

Curitiba PR
2017

Abstract

Financial time series are notoriously difficult to analyse and predict, given their non-stationary, highly oscillatory nature. In this thesis, the effectiveness of the Ensemble Empirical Mode Decomposition (EEMD) is evaluated at generating a representation for market indexes and exchange rates that improves short-term trend prediction for these financial instruments.

The results obtained in two different financial datasets suggest that the promising results reported using EEMD on financial time series in other studies were obtained by inadvertently adding look-ahead bias to the testing protocol via pre-processing the entire series with EEMD, which do affect the predictive results. In contrast to conclusions found in the literature, our results indicate that the application of EEMD with the objective of generating a better representation for financial time series is not sufficient, by itself, to substantially improve the accuracy and cumulative return obtained by the same models using the raw data.

Keywords: Trend prediction, Machine Learning, Finance.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Time series have been a target of studies for decades, some of these studies dating from before the 50's [Wiener, 1949]. Problems in this area are many and varied, ranging from detecting irregularities in ECG signals [Polat and Güneş, 2007] to predicting solar power generation from weather events [Sharma et al., 2011] and forecasting price movements in financial instruments [Hassan and Nath, 2005][Guresen et al., 2011].

These problems are governed by variables that are most of the time not entirely apparent or not measurable. These variables could be a numerical measure that express the emotional state of a person during a ECG test, the concentration of chemicals in the atmosphere or financial indicators of companies that have stocks in the market. Not considering storage space and processing power, if the set of all possible variables that affect an event could be mapped, than the problem could be solved using an analytic equation that makes use of those variables to compute a precise result. Unfortunately, knowing and measuring all quantitative variables that influence a certain event is usually not possible.

In the absence of the complete set of variables, the rules that govern the evolution of an event, or signal in the case of this study, must be inferred from regularities in the past

[Gershenfeld and Weigend, 1994] using an incomplete set of variables. In this case, more complex and robust modelling processes are used in favor of an analytic equation.

Financial markets are notoriously difficult to analyze and predict, which is one of the reason why prediction in the finance industry is among the most popular problems when studying time series [Sapankevych and Sankar, 2009]. Usually these problems revolve around predicting the closing price or trend of a specific quantity associated to an asset using features extracted from the same or correlated time series. Historically, the number of studies focusing on regression problems, or predicting what will be the closing price of a financial asset, outweighs the number of studies with the focus on trend classification [Kumar and Thenmozhi, 2006]. This is counter-intuitive as systems with small regression errors (e.g. root mean squared error (RMSE) and mean absolute error (MAE)) could still lead to incorrect decision making. Small errors would be especially damaging when working with prediction on more mature markets, where the volatility and strength of stocks and market indexes movements are less pronounced than in emerging markets. Besides the obvious monetary gain that could be obtained by developing accurate models, these problems are also interesting from a research standpoint given how difficult it is to forecast values on the non-stationary, highly oscilatory time series [Mikosch and Stărică, 2004].

A number of authors have attacked this problem using models such as Auto Regressive Moving Average (ARMA) and its variations [Atsalakis and Valavanis, 2009], Generalized AutoRegressive Conditional Heteroskedasticity (GARCH), Linear Regression, Neural Networks [Kaastra and Boyd, 1995][Guresen et al., 2011]. More recently SVMs [Sapankevych and Sankar, 2009] and ensembles have also been used to improve classification and regression results [Cheng et al., 2012].

Hand-engineered features crafted by experts are generally used in conjunction with these models, and are also featured in many studies. Many finance-specific features such as %K,

%D, Momentum, Williams %R, A/D Oscillator, Fast and Slow moving averages, High Price (day), Low Price (day), Volume, Closing Price and Open Price [Kim, 2003] have been proposed as predictors to improve classification and regression performance, but given the current state-of-the-art and the results obtained by researchers without the use of these features [Hsu et al., 2016], it is unclear if the machine learning models perform better using these features instead of the raw daily return values.

Time-series decomposition techniques such as the Ensemble Empirical Model Decomposition (EEMD) are used to deconstruct signals in various simpler, quasi-periodic signals. If the components generated by this method could be effectively used as an input to regression or classification models, using the decomposed components would advantageous since most of the human bias from feature generation would be removed from the equation.

EMD/EEMD was successfully applied in other fields [An et al., 2013, Guo et al., 2012, Wang et al., 2015, Lei et al., 2009] and is specially useful for non-stationary series, but very few studies have applied it to predictive tasks in the field of finance. The studies that did apply EEMD on financial time series [Al-Hnaity and Abbod, 2015, Fenghua et al., 2014a, Xiong et al., 2011, Yu et al., 2008] reported good predictive results, but they did so by using evaluation protocols containing look-ahead bias, which can be defined as the inadvertent use of information that is not available until a later date. This shortcoming might affect the predictive results, causing the accuracy and other metrics to have considerably different values than what a predictive system would be able to obtain in a real scenario.

The objective of this work is to analyse the applicability of EEMD in generating a different representation of financial time series that could be used for predicting market movements, and study the impact of the look-ahead bias in this scenario.

1.1 Motivation

Trend and Price prediction of financial time series are two of the most important problems in time series analysis and finance. Devising a technique to accurately predict the pseudo-chaotic, non-stationary price time series of financial instruments would change how financial markets are studied by invalidating the Efficient Market Hypothesis (EMH), dating from 1970 and still one of the most influential hypothesis on the field. The EMH states that security markets are extremely efficient in reflecting information, to the point of the price of securities account for all available information on their current prices. In this scenario, financial time series can't be used to gain an edge in predicting the future because all present information is already accounted for in the current price - there is no lag between new information and price movements [Malkiel, 2003]. In addition, by acquiring intuition on how to apply EEMD in this problem, the study would also provide insights on how to deal with other problems of similar nature.

In addition, despite the importance of this problem and the effort of many researchers, predicting financial time series is still an open problem. Compared to other fields, where classification accuracy for the state-of-the art result stands at over 95% [Graham, 2014, Wan et al., 2013, Lee et al., 2015] , the accuracy obtained on financial time series problems has a lot of room for improvement. The accuracy for trend classification tasks on financial instruments vary greatly depending on the techniques and on the instrument, but normally is in between 60-70% [Wang and Choi, 2013]. For price prediction, the results are measured as a function of the RMSE (root mean squared error) and mean absolute error (MAE) and mean squared error (MSE), but are used only as a way to compare the techniques since all techniques achieve high values in these error metrics.

1.2 Challenges

The difficulties associated to this research falls under three categories: feature extraction and selection, designing the testing protocol and the lack of public available data.

- Designing a new methodology

  Spectral decomposition techniques such as Ensemble Empirical Mode Decomposition (EEMD) can be used to decompose the original signal into multiple signals, which have been shown to improve [Fenghua et al., 2014b][Al-Hnaity e Abbod (2015)], [Jothimani et al. (2016))], [Yu et al (2008)] predictive performance when used as input to regression and classification models instead of the original signal. These studies however present two main flaws: look-ahead bias is inadvertently added during the pre-processing stage, and there is very little information about how the EEMD parameters were chosen before the decomposition.

- Lack of public, consolidated datasets

  The assumption of easily available data might be the reason why researchers that work with financial time-series don't make their datasets available, but this ultimately makes comparing results between studies very hard since a complete match of the data used among studies is a very rare occurrence.

  Despite the existence of free financial data providers such as Yahoo! Finance, Google Finance and FRED, it would be ideal to have a readily available, extensive and clean dataset which researchers could use for their work with the certainty of using the exact same dataset as others. A good example of such dataset is ImageNet [Deng et al., 2009].

  In this work, this shortcoming is addressed by using two datasets: one of the few consolidated financial datasets found in the literature, the "Istanbul Stock Exchange"

[Akbilgic et al., 2014], and another dataset composed of 10 years of daily data from different market indexes, exchange rates and commodities, built during this study. This dataset can be used for regression and classification tasks, and will be made available to other researchers.

1.3 Objectives and Contributions

The main objective of this research is to analyse how effective EEMD (Ensemble Empirical Mode Decomposition) is at generating features from noisy, non-stationary financial time-series, to be used for 1-day trend prediction.

The main contribution of this work is the design of a testing protocol that does not add look-ahead bias through the usage of EEMD, a time series decomposition technique, and the proper evaluation of this technique using the protocol. In addition, the reasoning behind the selection of every parameter used in EEMD is also made explicit throughout the evaluation process. The testing protocol formulated during this study and the results obtained were published on the Expert Systems with Applications Journal [Furlaneto et al., 2017].

A secondary contribution of this work is the construction of a financial dataset that other researchers will be able to use on their research. This dataset is composed of 10 years of daily data from different market indexes, exchange rates and commodities, and despite being used for a classification problem in this study, it was built in such a way that it can also be used for regression tasks.

Chapter 2

Theoretical Background

This section reviews the main concepts of the techniques that lay the foundation for this research project. The spectral decomposition techniques SSA and EEMD receive special attention as they are directly associated to the hypothesis of this thesis. The metrics used to evaluate each prediction model are just as important, so the Area under the ROC curve (AUC) and the Accuracy will also be reviewed. The ensemble techniques that will be used in this work, Bagging and Stacking, will be briefly presented.

Although a number of classifiers such as SVMs, Regression Trees, Logistic Regression and Random Forests will be used as base classifiers for the ensemble techniques, they won't be covered in this document as it would become unnecessarily long. This work assumes the reader is sufficiently familiar with these base classifiers and how they are used to predict a binary target variable given a much-dimensional dataset.

2.1 Financial Time Series

The analysis of data that have been observed at different points in time leads to new and unique problems. The correlation introduced by the sampling of adjacent points in time can

restrict the applicability of many conventional statistical methods, specially the methods dependent on the assumption that adjacent observations are independent and identically distributed [Shumway and Stoffer, 2013].

These series of data, or Time Series, have been a target of studies for decades, with some of the studies dating from before the 50s [Wiener, 1949]. Problems in this area are many and varied, ranging from detecting irregularities in ECG signals [Polat and Güneş, 2007] to predicting solar power generation from weather events [Sharma et al., 2011] and forecasting price movements in financial instruments [Hassan and Nath, 2005][Guresen et al., 2011].

Time Series can be defined as a sequence of values of a quantity, arranged in temporal order, and often spaced by a fixed time interval. The time domain approach is motivated by the assumption that the correlation between adjacent points in time is best explained in terms of a dependence of the current value on past values. The time domain approach focuses on modeling some future value of a time series as a parametric function of the current and past values. In this scenario, we begin with regression models of the present value of a time series on its own past values, and on the past values of other series. Therefore, this model can be used as a forecasting tool for classification and regression tasks, and is particularly popular with economists for this reason [Shumway and Stoffer, 2013], who focus themselves on the analysis of Financial Time Series.

Financial Time Series analysis, as stated by Ruey Tsay in his book [Tsay, 2005], is concerned with the theory and practice of financial assets valuation over time. These financial assets are classified as intangible assets, in contrast to tangible assets whose value depends on physical properties such as buildings, land and machinery. The typical expected benefit of intangible assets is a claim to future cash [Fabozzi, 2008], and assets such as stocks and commodity contracts are negotiated in financial exchanges. The series originated from these

assets are composed of floating-point numbers or integers, and normally refer to the value of the asset and the volume of negotiations associated to that asset in a given point in time. A large number of different financial assets exist and explaining all of them would go much beyond the scope of this work, but a brief explanation about market indexes, commodities, currency rates must be added to this work given their presence in the datasets of this study, as well as their importance in financial markets.

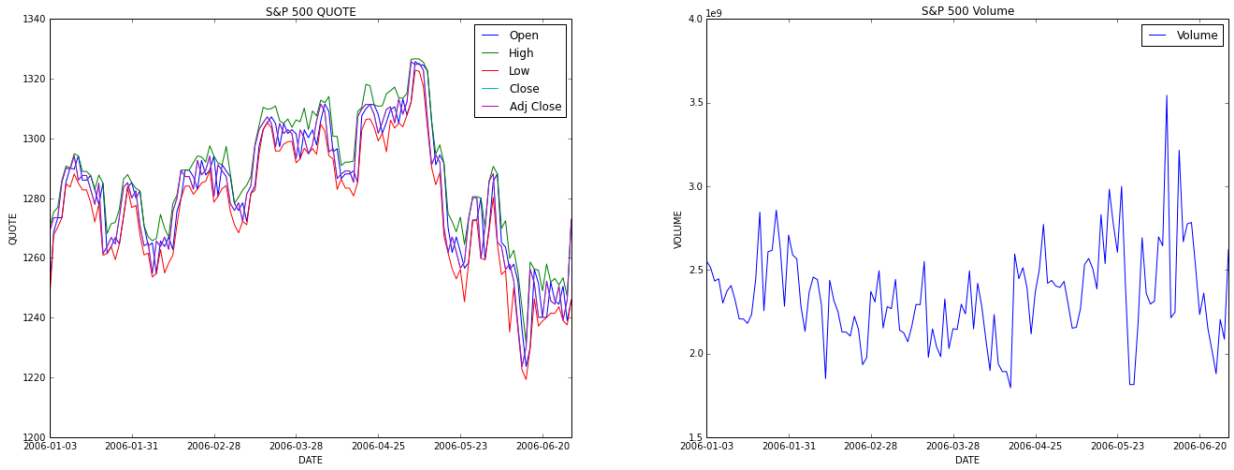
2.1.1 Market Indexes

When the Efficient Market Hypothesis (EMH) states that only "normal" returns can be earned in the stock market, an assumption is made that there is an average that summarizes the stock market performance [Fabozzi, 2008]. These averages, built with the intent to summarize part or an entire market , are called Market Indexes. These indexes are not assets as they cannot be traded, but they act as a benchmarks for the investor to understand how a market is behaving.

An example of a widely used stock market index is the Dow Jones Industrial Average (DJIA), which is computed by taking the price of each of 30 selected blue-chip American stocks, adding them, and dividing by a divisor. This divisor was originally the number of stocks in the index, but because of the changes in price caused by stock splits, the value of the divisor was reduced to account for each of these splits.

Another important index is the S&P 500, an index created by Standard & Poor's that takes in consideration 500 stocks and includes both the price per share of each stock and the number of shares outstanding, reflecting the total market value of all the stocks in the index. Despite being less popular than the DJIA, the S&P 500 is considered a better overall measure of the American stock market performance when compared to DJIA because of the amount of stocks involved in the computations, and also because of its statistical properties.

The Time Series from Market Indexes normally contain the Open, Close, Low, High and Volume values for the time period considered. These values are computed considering the assets under each index, as well as the particular average formula used by it. The figure 2.1 shows how the time series for the S&P 500 varied between 03-Jan-2006 and 29-Jun-2006:



(a) (b)

Figure 2.1: S&P 500 Time Series: (a) Quotes and (b) Volume

2.1.2 Commodities

According to Fabozzi et. al.[Fabozzi, 2008], there is a consensus among academics and practitioners that commodities compared to other assets can be considered, in a portfolio context, as an asset class of their own. An asset class in that context consists of assets that show a high correlation in their risk/return ratio, and a different ratio when compared to other assets. Commodities, as an exception, possess intra-class heterogeneity - in other words, every commodity has their own specific properties.

Despite being considered as uncorrelated assets, Commodities are generally grouped under two different categories: soft commodities and hard commodities. Hard commodities are products from the energy, precious and industrial metals, and soft commodities are usually

perishable commodities from the agricultural sector. The figure 2.2 shows the components of each of these categories:

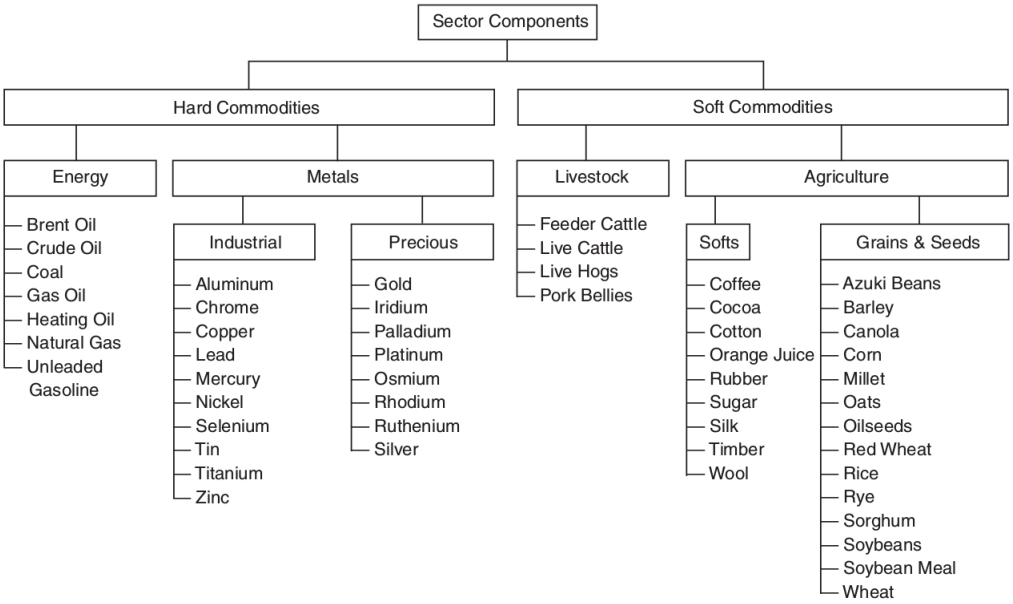


Figure 2.2: Commodity categories [Fabozzi, 2008]

Commodities are traded in exchanges in the form of commodity futures contracts, which is an agreement to buy or sell a certain amount of a specific commodity at a specific price and date. Fluctuations in the price and traded volume of these contracts generate historical information in the form of time series, which are then used for econometric studies and building portfolios. The raw value for a certain amount of a commodity can also be used as a time series.

The figure 2.3 show how the time series for the price of the troy once (373.24g) of Gold varied between 03-Jan-2006 and 29-Jun-2006:

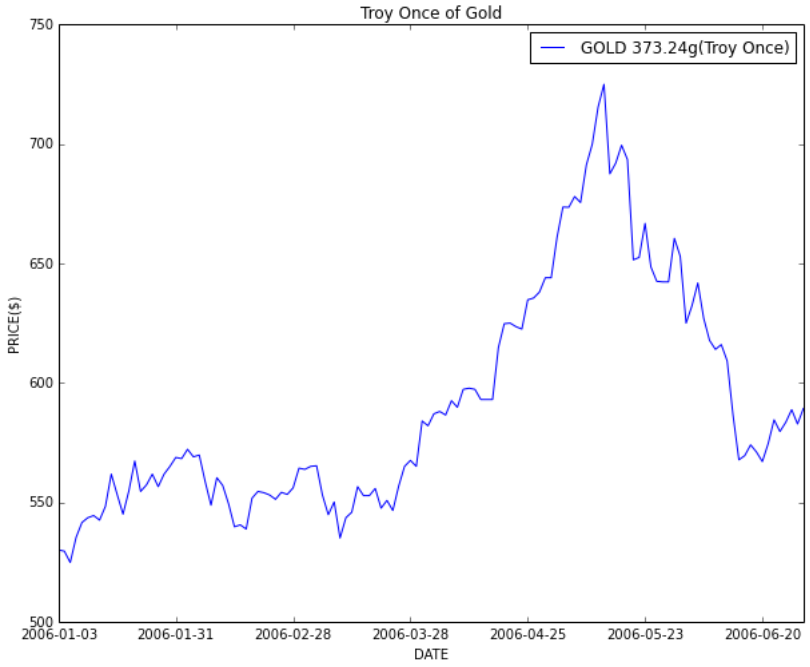


Figure 2.3: Price of 373.24g of Gold in USD

2.1.3 Currency Rates (FOREX)

A foreign exchange or currency rate is the price of one country's money in terms of another's, and the foreign exchange market is a global network of buyers and sellers of currency [Fabozzi, 2008]. Currency markets, much like the stock market, are affected by an ever-changing mix of events which makes supply-and-demand of each currency change, and along this change, the price of one currency in relation to another shifts accordingly. It is said that no other market encompasses as much of what is going on in the world at any given time as foreign exchange.

80% of the FOREX transactions have the American (US) dollar as one of the currencies when the currency rate is computed [Fabozzi, 2008]. The US dollar plays such an important because:

- It is used as an investment currency throughout the world
- It is a reserve currency held by many central banks

- It is a transaction currency in many international commodity markets

Additionally, the most traded currency pairs are:

- 30% of the global operations: European Euro/American dollar (EUR/USD)
- 20% of the global operations: American dollar/Japanese Yen (USD/JPY)
- 11% of the global operations: British Pound/American dollar (GBP/USD)
- American dollar/Swiss Franc (GBP/CHF)

The Time Series from Currency Rates come in the form of ratios, indicating how much the first currency is worth in respect to the second currency in the pair name. As an example, the value of the currency pair EUR/USD indicates how much a Euro is worth in respect to the US dollar - in other words, a value higher than one Euro is worth more than one dollar.

2.1.4 Look-Ahead Bias

Look-ahead bias can be defined as the inadvertent use of information that is not available until a later date; in other words, forecasting the future using future data. Look-ahead bias might be added to research protocols or back-tests in subtle ways; as explained by [Mahfoud and Mani, 1996], commercially and publicly available financial data might contain look-ahead bias from the start, with data associated to Governmental economic indicators for example going through review processes that might modify past figures.

Aside from look-ahead bias added to the data itself, the use of certain techniques as a pre-processing step might also be problematic. As an example, normalization techniques are very popular pre-processing steps in studies with financial time series. The min-max and the z-score normalizations, arguably the two most popular normalization techniques, make use of statistical variables that might change as new information is added to time series, such as the

minimum and maximum values, as well as the standard deviation. Normalizing the entire time series will add look-ahead bias to the evaluating protocol as variables that might change over time (min, max and standard deviation) are known and fixed from the start. For the min-max normalization in specific, note that using the percentage change, or Rate of Change (RoC), of the prices instead of the prices themselves as input minimizes the effect of the look-ahead bias as the minimum and maximum values change much less overtime when compared to the prices, since trend component is eliminated from the time series.

The general case is that extra care must be taken when using techniques that makes use of information only available at time t_1 to modify data points at time t_0 (where $t_1 > t_0$). Another example of such method is Empirical Mode Decomposition (EMD), which adds look-ahead bias if used inappropriately. As part of its algorithm, EMD performs successive searches for local minima and maxima, with a subsequent spline interpolation between those points to generate an upper and lower envelopes of the signal. Because EMD stores and subtracts the highest-frequency signal from the original signal on each iteration to create the IMFs, the higher-order components (lower frequency) will be generated by interpolating points that are far away from each other in the original series. Through these successive interpolations, future information is embedded on the IMFs.

Note that the existence of look-ahead bias might or might not affect the results obtained using a particular protocol, but the fact that its existence might heavily skew results should be enough for striving to remove it. For this reason, it is crucial that experimental protocols using EMD or its ensemble variation EEMD, and other pre-processing techniques for that matter, be carefully crafted to account for algorithmic peculiarities, in such a way that bias is not accidentally added.

Figure 2.4 depicts from a high level perspective, the protocol used in the literature: In Figure 2.4, normalization methods might or might not be used as part of the protocol. The

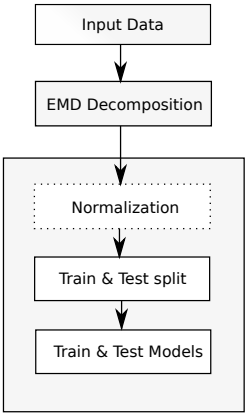


Figure 2.4: Protocol with look-ahead bias

dataset split and the way the models are trained and tested vary as well. However, using EMD as a pre-processing step seems to be common practice.

The addition of bias of such nature is unfortunately often overlooked in research works found in the literature, and protocols that contain look-ahead bias due to the usage of EMD seems to be prevalent. [Al-Hnaity and Abbod, 2015, Fenghua et al., 2014a, Xiong et al., 2011, Yu et al., 2008].

2.2 Time series Decomposition

Crafting custom features for prediction problems on time-series can be seen as one of the few areas where there is still human interaction in the machine learning pipeline; this is even more problematic in the realm of finance, where there is no consensus among experts as to which are the most useful features for trend prediction. For this reason, automating the process of decomposing these time-series in a number of different signals that can be used as features would be ideal.

Time-series Decomposition techniques, applied with the goal of extracting features, aim at solving the problem of creating hand-crafting features by extracting simpler, periodic signals from the original time-series, which can be used as inputs to machine learning or other statistical models. This study focus on the use of two particular techniques: Singular Spectrum Analysis and Ensemble Empirical Mode Decomposition.

2.2.1 Ensemble Empirical Mode Decomposition

Empirical Mode Decomposition (EMD) is an adaptive method created to separate the spectrum of non-linear and non-stationary signals ([Wu and Huang, 2009]). It decomposes a given time series, or signal, in components with different frequencies and amplitudes, called Intrinsic Mode Functions (IMFs). IMFs have two properties that distinguish them from other signals:

- The number of extrema and zero crossings must differ at most by one.
- The mean value of the upper and lower envelope is zero.

These conditions make the IMFs be quasi-periodic, similar to harmonic signals, with the biggest difference between them being that there is no guarantee that the IMFs will have the same amplitude and frequency along the time axis. These IMFs, or simply modes as they are also known for, are extracted from the original time series through a process called sifting, where the order of IMFs extraction is from high-frequency to low-frequency signals; as the component extraction process progresses, the modes look more and more periodic and have less noise embedded in them. Algorithm 1 describes this process in details:

There are variations of stopping criterion for the component extraction process, but they are normally derived from the condition proposed by [Huang et al., 2003], where the algorithm is executed for a specific number of iterations after the residue satisfies the restriction of its

Algorithm 1 Empirical Mode Decomposition

Require: $x(t)$
Ensure: IMFs
$IMFs = []$
while Stopping Criterion is not reached **do**
  Identify all the maxima and minima values of $x(t)$
  Generate upper and lower envelopes, $e_{min}(t)$ and $e_{max}(t)$, with cubic spline interpolation.
  Compute point-by-point average of upper and lower envelopes: $m(t) = (e_{min}(t) + e_{max}(t))/2$
  Compute the difference between $x(t)$ and $m(t)$: $d(t) = x(t) - m(t)$
  if d(t) fits the IMF criterion **then**
    IMFs.append($d(t)$)
    $x(t) = x(t) - d(t)$
  else
    $x(t) = d(t)$
  end if
end while
return IMFs

zero-crossings and number of extrema differing by at most one, and becoming a monotonic function from which no more IMFs can be extracted.

EEMD operates very similarly to EMD, but instead of decomposing the original signal once, it decomposes various copies of the original signal with different white Gaussian noises added to it, and averages all the IMFs generated by decomposing each of those copies. The addition of the noise helps the sifting process to avoid mode mixing, which is one of the main problems of the conventional EMD technique.

Algorithm 2 describes succinctly how EEMD operates:

Algorithm 2 Ensemble Empirical Mode Decomposition

Require: $x(t)$, N, $Noise\ strength$

IMFs = []

Copy $x(t)$ K times

Add white noise to the copies of $x(t)$

IMFs.append(EMD(composed signals))

return Mean(IMFs)

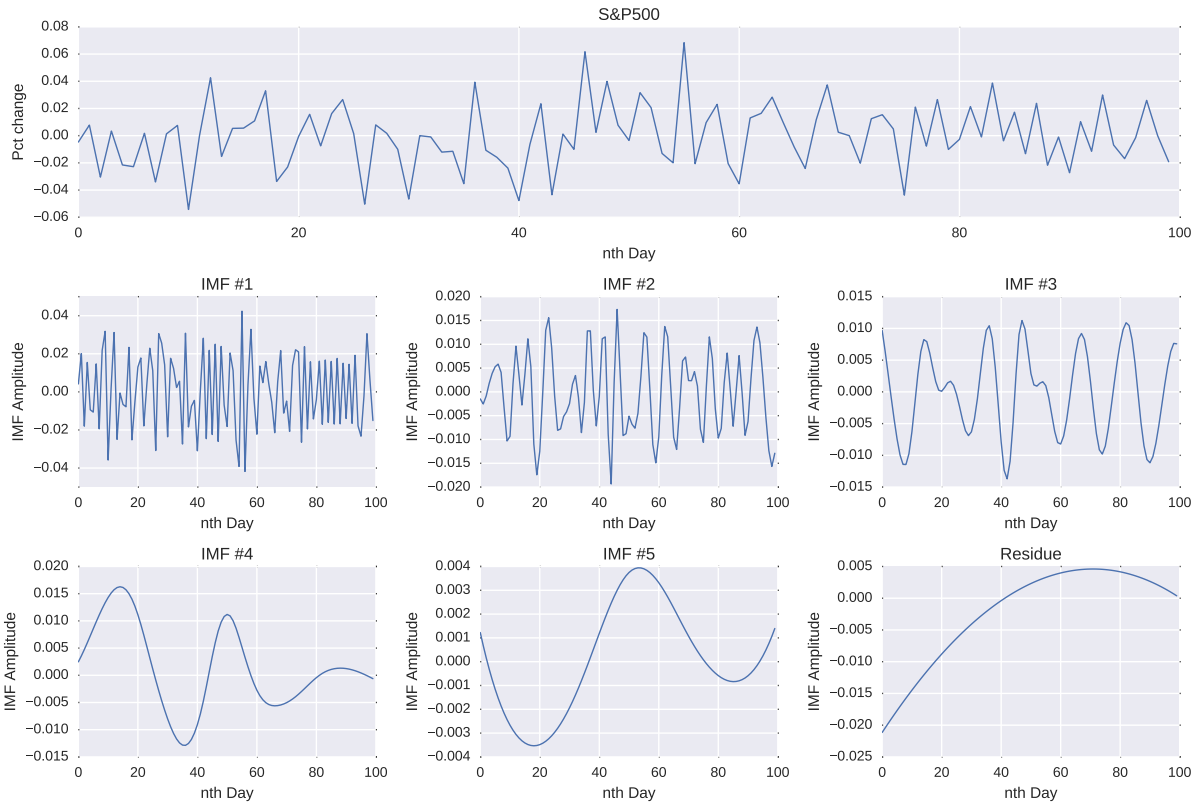


Figure 2.5: EEMD decomposition applied to 100 days of data from the S&P500 index.

Figure 2.5 shows the IMFs extracted from a portion of the S&P500 index time series that is used in this study. The IMFs were plotted from first to last component that is extracted from the series, where the last plot contains the residue.

In addition to the input signal, the result of the decomposition using EEMD is also affected by a few additional parameters (as show in algorithm 2):

- **Ensemble size** (K): The number of replicas of the input signal to be used in the ensemble
- **Noise strength**: Standard deviation of the Gaussian random noise added to the original signal before the sifting process starts.
- **S-number (Stopping Criterion for EMD)**: For S consecutive iterations, the number of zero crossings and extrema differ at most by one, and these numbers stay the same.

- **Maximum number of siftings (Stopping Criterion for EMD)**: A maximum number of total iterations can be set. This is done to increase the speed of the algorithm, prevent oversifting and to prevent the sifting procedure from being in a never-ending loop.

The "S-number" and "Maximum number of siftings" parameters play an important role, affecting the number of IMFs produced by the algorithm. No matter the combination of variables though, the upper bound for the total number of IMFs extracted from a signal will be close to $\log_2(nPoints)$ ([Wu and Huang, 2009]).

2.3 Evaluation Metrics

Trend prediction on financial time series is a binary classification problem, where the true positive rate (TPR), or the number of "up" predictions that were correct, is as important as the false positive rate (FPR), or the number of "up" predictions that were incorrect. In both of these situations a potential investor would take the fact in consideration to sell or buy a financial instrument. It is also important to mention that the distribution of classes in these time series is mostly uniform – in other words, the numbers of occurrences of ups and downs are very similar.

Taking those facts in consideration, two metrics will be used to evaluate the results:

- Classification Accuracy
- Cumulative Return

2.3.1 Classification Accuracy

The classification accuracy, along with the confusion matrix, are arguably the most popular metrics for reporting the performance of a binary classifiers. The accuracy is given by the formula $\frac{(TP+TN)}{P+N}$, where TP, TN, P and N refer to the number of True Positives, True

Negatives, Positives in the original data and Negatives in the original data, respectively. Tom Fawcett [Fawcett, 2006] created a confusion matrix that makes the relationship between these quantities very clear:

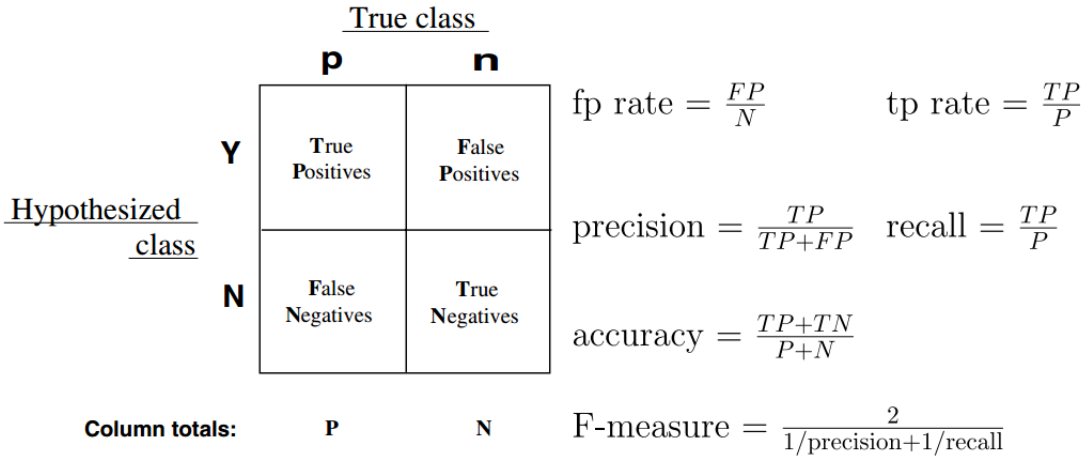


Figure 2.6: Confusion matrix and performance metrics.

The accuracy, as shown in its formula, is used when the number of true positives and true negatives have similar importance, which happens to be the case for trend prediction in finance time series. Considering a hypothetical situation where every action of buy/sell will be based on predictions made by a binary classifier, predicting when the market will go up is as important as predicting when it is going to go down.

2.3.2 Cumulative Return

The cumulative return is the quantity, in percentage value, an investment has gained or lost over time. This is a specially important metric in the field of finance, widely used to assess the performance of trading strategies. The standard formula for cumulative return (Rc) between days a and b is given by Equation 4.2

$$\mathrm{Rc} = \prod_{i=a}^{b}(1 + \frac{P_{i+1} - Pi}{P_i}) \tag{2.1}$$

where P_i is the closing price of the financial instrument at the i^{th} day.

It is important to note that the cumulative return is far more important than the accuracy of prediction when evaluating a predictive model on financial time series. While these metrics are directly correlated, it is possible for different predictive models to achieve different cumulative returns, while achieving similar prediction accuracy. Financial instruments are known to present a fat tail distribution in respect to their returns, so models that incorrectly classify strong movements present much smaller cumulative returns.

Chapter 3

State of the Art

Studies that focus on applying machine learning techniques to financial series generally do so by selecting a testing protocol, the machine learning techniques to be used for the problem and by defining a new group of financial assets (e.g. stocks) and a time period, which define the series that is going to be extracted. The problem with this approach is that comparing results among studies becomes very challenging since results are obtained and reported on different datasets almost every time a new study is done. These time series vary in the number of samples, sampling rate (days, weeks or months) and financial instrument.

Because of the difficulty in finding overlaps between studies and datasets used, this state of the art was structured to give the reader a better idea of the best results achieved by state of the art techniques for tasks on financial time series, in addition to providing references to studies that make use of time series decomposition techniques for machine learning tasks.

3.1 Artificial Neural Networks

A large body of knowledge exists on the application of Neural Networks on predicting financial time series. This machine learning technique has been used for decades in one of its different architectures, normally feed-forward, radian basis function (RBF) or recurrent, as a stand-alone technique or in ensemble methods in prediction efforts of financial nature.

The reasoning behind using Neural Networks for stock or market indexes prediction is that, as pointed on Vui et. al. On [Vui et al., 2013] and on various other studies, Neural Networks have been shown to be able to predict the volatility and non-linear stock market prices due to its learning, mapping, generalizing and self-organizing characteristics. Among all the Neural Network architectures, Feed-forward is the most commonly used in stock market forecasting probably due to its simplicity and the considerable number of studies backing up its efficiency.

The financial datasets where ANN have been tested are many and varied. Taking only stock markets in consideration, they include stock and market index time series from the New York Stock Exchange, NASDAQ, Bombay Stock Exchange (BSE), Brazil Stock Exchange (Bovespa), Hong Kong Stock Exchange, London Stock Exchange, Tokyo Stock Exchange and Malaysia Stock Exchange. Daily prices time series are the most used, but weekly and monthly time series are also featured in studies; this is likely due to the high variance of stock markets. As far as the size of the time series, they vary by a great margin – The time series range from one year to over a decade of information.

Artificial Neural Networks have been shown to outperform classical prediction techniques such as ARIMA, ARMA and MA in most studies where these methods were compared. When compared to SVMs, the results seems to depend highly on the architecture used and how much time was dedicated to tune the network. Preethi et. al. [Preethi and Santhi, 2012] surveys a number of different techniques used for market prediction up to the year of 2012, and points

out Neural networks, along with fuzzy systems, as the leading machine learning techniques in stock market index prediction on that point in time.

In [Patel et al., 2015], Patel et. al. proposes a fusion of machine learning techniques to predict the stock market indexes CNX Nifty and S&P Bombay Stock Exchange (BS) Sensex from Indian Stock Markets. The study proposes an ensemble approach involving ANN, Random Forest and SVR, resulting in hybrid models that are compared to each other and also to the individual techniques. The results suggest that some of the hybrid models perform better than their individual counterparts, and that the model composed of SVMs and a ANN performs better than the others and also better than its individual parts (SVMs and ANN).

3.2 Support Vector Machines (SVM)

The Support Vector Machine (SVM) method, developed by Vapnik et. al. in 1995, is used for many machine learning tasks such as pattern recognition, object classification, and in the case of time series prediction, regression analysis. Support Vector Regression, or SVR, is the methodology by which a regression function is estimated using observed data, which in turn "trains" the SVM. In most studies involving SVM, this technique is classified as one of the techniques with the most promising results when compared to other prediction methods such as different architectures of Artificial Neural Networks as well as more traditional techniques, such as MA, ARIMA, ARMA, GARCH and different kinds of filters.

In the survey [Sapankevych and Sankar, 2009], Sapankevych et. al. states that of all the practical applications using SVR for time series prediction, financial data time series prediction appears to be the most studied along with electrical load forecasting. According to the study, over 30% of the papers reviewed were on the subject. The noisy, non-stationary and chaotic

nature of this type of time series are pointed as reasons for the vast use of non-linear algorithms such as SVR on such problems.

Few different types of SVMs have been used with success in prediction tasks, more specifically Least Square SVMs (LS-SVM)[Ismail et al., 2011], Fuzzy SVMs (fSVM)[Bao et al., 2005], Quasi-linear SVM[Lin et al., 2013] and the original SVM formulation, proposed by Vapnik. Other studies involving comparing ensemble methods with the application of individual techniques have also been found [Sun and Li, 2012].

Lin et. al. [Lin et al., 2013] proposes a prediction system that uses a correlation-based SVM filter to rank and select a subset of financial indexes as the best features and uses a quasi-linear SVM to predict the market trend using those indexes as input. Their results indicate that the quasi-linear kernel outperformed non-linear kernels such as RBF, polynomial and the sigmoid.

Hossain et. al.[Hossain et al., 2009] compares GARCH, Neural Network and SVM in financial time series prediction. The time series were composed of monthly index data for the stock markets from Japan (Nikkei 225), Hong Kong (HS), U.K. (FTSE 100) and Germany (Dax). The time periods vary, but they range from 1984-1990 to 2006. The researchers found that SVM surpasses Neural Networks and also the classical statistical method GARCH in prediction accuracy (although the SVM used wasn't much better than GARCH in this study).

Sun et. al.[Sun and Li, 2012] compared the performance of SVMs and SVMs ensembles in predicting Financial distress. The experimental results indicate that the SVM ensemble is significantly superior to individual SVM classifier when the number of base classifiers in the SVM ensemble is properly set. Additionally, it also shows that the RBF SVM based on features selected by stepwise multi discriminant analysis is a good choice for this kind of prediction when the individual SVM classifier is applied.

In [Wang and Choi, 2013], Wang et. al. compare the performance of PCA-SVM, SVM, PCA-ANN, ANN and Random Walk in predicting the Market Index and Stock Price Direction in the Korean SKOSPI and HSI markets, with data ranging from January 1st, 2002 to January 1st, 2012. The results he presented show that the best performing technique was the SVM using principal components as features for both markets.

Ou el. al. [Ou and Wang, 2009] have also compared the accuracy of 10 different techniques to predict the movement of the Hang Seng index of Hong Kong stock market and have reported much higher accuracy values for all the tested techniques, but SVMs still performed better than all the other methods.

There are many studies that draws attention to the accuracy of SVMs when compared to other techniques. The technique is also flexible enough to be used in classification problems (market trend) and regression problems (price prediction) alike. However, the studies show that the correct choice of input features is crucial, and the choice of the correct kernel also plays an important part in how well the SVM performs in a given problem. These choices, ultimately, add bias to the model and hinders the ability of this technique to uncover hidden patterns.

3.3 Evolutionary Computing

As described in the work by Antonin et. al. [Ponsich et al., 2013], many problems in all sorts of domains can be formulated as optimization problems, which need the application of specialized methods for their solution. As great importance was traditionally placed on mathematical programming methods, the complexity of the models would have led researchers to concentrate their efforts on the development of solution heuristics analogous with biological, social, or physical phenomena observed in nature. In this framework, great attention has been dedicated to evolutionary algorithms (EAs).

This class of bioinspired algorithms relies on swarm behaviour, mutation, selection and a number of other heuristics to evolve a population of solutions toward a good adaptation to their environment (i.e., to produce solutions that are a good approximation of the global optimum).

In the finance industry, EAs have been used mainly to find optimal parameters for models such as SVMs, but they have also been used extensively as a way to solve a specific class of problems with this domain – the portfolio management problem. For these problems, Evolutionary Algorithms achieve good results when the problem involves multiple objectives, which is normally the case for portfolio optimization.

As described by Man Mohan Rai in his work [Rai, 2006], EAs have an advantage over conventional gradient-based search procedures because they are capable of finding global optima of multi-modal functions and searching design spaces with disjoint feasible regions. They are also robust in the presence of noisy data. Another desirable feature of these methods is that they can efficiently use distributed and parallel computing resources since multiple function evaluations can be performed simultaneously and independently on multiple processors. However, a crucial problem with genetic and evolutionary algorithms is that they often require many more function evaluations than other optimization schemes to obtain the optimum.

Other methods such as Genetic Algorithms and Particle Swarm Optimization have been used with success. Hanhong Zhu et. al.[Zhu et al., 2011] results show that PSO performs well when approximating the Pareto frontier produced by different portfolio allocations and perform better than traditional methods as we scale the number of assets we use to build the portfolio. Antonin et.al. [Ponsich et al., 2013] also draws the attention of researchers to the success achieved by Multi objective Evolutionary Algorithms (MOEAs) on financial problems. In his survey, he elaborates on the use of MOEAs to approximate complex Pareto frontiers, which might be a problem for other types of optimization algorithms.

3.4 Ensembles

Classifier ensembles, a classification approach based on combining multiple single classifiers, aims at obtaining more accurate classifiers composed of a group of weak, less accurate classifiers. The intuition behind combining classifiers is that by grouping them in a single classification effort, each individual model covers for the errors of the others on the input space. Following this reasoning, the performance of classifier ensembles is expected to be better than even the best single classifier. In [Cheng et al., 2012], Cheng et. al. compare 3 ensemble algorithms for the problem of of short-term prediction of the Shanghai Composite Index. He uses 18 features to describe each data point in the time series, and the following base models to compose the ensembles: SVR, BPNN, RBGNN and LWL (linear regression). The ensemble techniques evaluated in this study are Bagging, Stacking and Random Subspace, and the evaluation criteria where the Root Mean Squared Error (RMSE) and the Relative Absolute Error. Under these constrains, the authors concluded that the Bagging algorithm provides a more stable and had better improvements compared to the accuracy the individual models presented. Tsai et. al. [Tsai et al., 2011] mentions in their study (2010) that "in the area of machine learning and pattern recognition, the combination of multiple classifiers (e.g. classifier ensembles), have been shown to perform better than single classifiers. However, this technique has not been widely examined to predict stock returns in financial markets.". This affirmation, to the best of this author's knowledge, still holds true given the lack of work with ensembles in this area. This work uses a dataset composed of financial ratios and stock returns of 70% of the Taiwan stock market, and present the results with the ensemble techniques voting and bagging to the positive/negative trend prediction problem. The authors make use of MLP, CART and linear regression as the base models, and test homogeneous and heterogeneous ensembles. The conclusion was that, for this

specific problem, voting homogeneous ensembles performed best, most specifically ensembles of MLPs.

Ballings et. al.[Ballings et al., 2015] studies indicates that Random Forest is the best technique for Stock Price direction prediction, even when compared with techniques such as ANN and SVM; Kernel Factory and AdaBoost were also surpassed by the far simpler Random Forest, and ANN performed even worse.

In [Patel et al., 2015], Patel et. al. proposes a fusion of machine learning techniques to predict the stock market indexes CNX Nifty and S&P Bombay Stock Exchange (BS) Sensex from Indian Stock Markets. The study proposes an ensemble approach involving ANN, Random Forest and SVR, resulting in hybrid models that are compared to each other and also to the individual techniques. The results suggest that some of the hybrid models perform better than their individual counterparts, and that the model composed of SVMs and a ANN performs better than the others and also better than its individual parts (SVMs and ANN).

3.5 Deep Learning

Deep Learning techniques have been gaining a lot of momentum recently, mostly on the fields of speech recognition and image analysis. Deep Recurrent Neural Networks, Convolutional Neural Networks (CNN) and Deep Belief Networks (DBN) have been used successfully in identifying context in images, improving speech recognition and other applications. However, despite the success of these techniques in other areas, they didn't achieve the same level of popularity for financial applications. There is a very limited number of studies with Deep Belief Networks, no application of Convolutional Neural Networks and very few studies with Deep Recurrent Neural Networks have been found. Reasons that could explain the lack of work in this area are the unusually larger number of data points needed to train these networks, the

time taken to adjust the hyper parameters of the network and the time required to train these networks properly. It is important to note that because we are dealing with time series that are non-stationary, the networks need to be retrained constantly, and at most hundreds of data points (if we are dealing with daily quotes) from the recent past would be relevant for short-term trend prediction, leaving not much data to be used for training these networks.

Kuremoto et. al. [Kuremoto et al., 2014] proposes a method for time series prediction using Deep Belief Networks built by stacking RBMs. They used a 3-layer DBN composed by two RBMs to capture the input space of the time series raw data and pre-trained them using energy functions. They also made use of Kennedy and Eberhart's PSO to find the optimal numbers of units in the input layer, hidden layer and the learning rate of the RBMs. The model was compared to other techniques that participated in the CATS benchmark of the IJCNN'04, and although it performed better than models such as MLPs and ARIMA, it still couldn't beat the best methods, which were Kalman Smoother and Ensemble methods.

Cao et. al. [Cao et al., 2015] propose a deep learning approach to capture the complex couplings across multiple financial markets by learning hidden features. The empirical results of trading the market trends predicted by the model in real financial market such as DJI (USA), Bovespa (Brazil) and BSESN (India), show that the approach achieves better accuracy than state-of-the-art methods such as Neural Networks trained with back propagation, Chain HMM, RBM and classical techniques such as ARIMA and logistic regression.

In his Master's thesis, Xi Chen [Chen, 2015] studied the application of Deep Belief Networks to stock price prediction. The conclusion of his study was that DBN can better forecast, when compared to MLP Neural Networks, the financial price within a predefined price percent change range if the raw data is properly preprocessed and the model properly trained. The data

used for this test were daily Open, High, Low and Close values for the S&P 500 index from Jan 1st 2000 to Dec 31st 2014.

Takeuchi et. al. [Takeuchi and Lee,] used an auto encoder composed of stacked restricted Boltzmann machines to extract features from historical time series of stock prices. The model was able to discover an enhanced version of the momentum effect in stocks without extensive hand-engineering of input features and deliver an annualized return of 45.93% over the 1990-2009 test period versus 10.53% for basic momentum. The data used were ordinary shares traded on NYSE, AMEX and NASDAQ; they mitigated the noise in the dataset by excluding stocks with monthly prices below $5 per share – this is done just so shares with small values present large percentage variations with small value variations.

In his Master's thesis, Estrada [Batres-Estrada, 2015] proposed a DBN architecture which was composed of 3 RBMs stacked with a MLP, and used log-returns as an input variable. This study is very similar to Takeuchi et al. [Takeuchi and Lee,], but uses RBMs instead of auto encoders. He mentions that no techniques to select the best hyper parameters were used, and points this as one area for improvement. As benchmark, he used a standard MLP with 20 Neurons, Linear Regression and Naive benchmark, which states that Y(t+1) = Y(t). Although his proposed architecture beat the alternatives, there is no mention of SVM or other Deep Learning architectures.

3.6 Recurrent Neural Networks

As described by Bengio et. al. On [Pascanu et al., 2012], a recurrent neural network (RNN) is a neural network model proposed in the 80's (Rumelhart et al., 1986; Elman, 1990; Werbos, 1988) for modelling time series. The structure of the network is similar to that of a standard multilayer perceptron, with the distinction that connections exist among hidden units

associated with a time delay. Through these connections the model can retain information about the past inputs, enabling it to discover temporal correlations between events that are possibly far away from each other in the data (a crucial property for proper learning of time series), which is of particular interest for time series that have high auto-correlation score. This class of networks is subdivided in a number of different architectures, out of which Fully Recurrent Networks, Elman Networks, Echo State Networks and Long-short term memory networks are the most used.

Although powerful, RNNs have been considered mostly for almost a decade after its conception because of the exploding and vanishing gradients problem, which refers to the large increase or decrease in the norm of the gradient during training. Few techniques have been proposed to mitigate the problem, but it was only on 1997 with the creation of Long-short term memory units, which allowed RNN to learn effectively.

RNNs have achieved state of the art results in speech recognition [Graves et al., 2013], and have been applied to a number of other forecasting problems such as forecasting wind speed, solar radiation, and energy consumption and on financial time series. Not many papers have been released in the past four years on the later; the relevant papers found in this category where [Dan et al., 2014], [Lin et al., 2009] and [Wei and Cheng, 2012] which deals with the application of Echo State Networks and Elman Networks to the problem of predicting stock prices and in building a new system for stock trading. In these studies, the RNN are compared against more conventional machine learning techniques such as BPNN and RBF and a buy and hold strategy, which are all surpassed by the RNN technique used in the study.

3.7 Time series Decomposition Techniques

Time series decomposition techniques are used to deconstruct and represent signals as various components, each with different characteristics and associated with the underlying cyclical nature of the original signal. These methods can be used for a variety of tasks, from denoising the signal to making inferences about its periodic behaviour and predictive tasks from the extraction of fetal heart signals from maternal ECG ([Ghodsi et al., 2010]), analysis of seismic signals ([Wang et al., 2012]) to prediction on financial time series ([Fenghua et al., 2014a]) being just a few applications.

Decomposition techniques can be divided into two main categories: Parametric and Non-parametric. Parametric methods make initial assumptions about the characteristics of the decomposed components, such as modelling these signals as sinusoidal waves of different amplitudes and frequencies (e.g. Fourier Transform). Parametric techniques are over-represented in the works found in the literature, where Fourier Transform and Wavelets are the most popular representatives.

Non-parametric techniques, on the other hand, make no a priori assumption for the generated components but are computationally more intensive than parametric models. These models are particularly good at decomposing non-linear, non-stationary time series due to their higher flexibility when compared to parametric models. The most well-known non-parametric techniques are the EEMD and Singular Spectrum Analysis (SSA) ([Vautard et al., 1992]), with the EEMD being the most under-represented out of these two techniques. The main reason for this under-representation can be attributed to EEMD's very recent history: it was developed in 2009 [Wu and Huang, 2009] as an improvement to Empirical Mode Decomposition (EMD) was introduced in 1998 ([Huang et al., 1998]), while SSA was proposed in 1992.

Works in different domains have reported promising results with the use of non-parametric decomposition techniques. Hassani et. al. [Hassani et al., 2009] have used SSA (Singular Spectrum Analysis) in conjunction with a linear model to predict the values of industrial production for sectors of the German, French and UK economies. The authors obtained the best results with this method when compared to ARIMA and Holt-Winters (a double exponential smoothing method).

In [Hassani et al., 2010], the authors use Singular Spectrum Analysis to predict the trend of the daily dollar exchange rate with respect to Euro and the Yen. The results obtained in this work were not conclusive, with a Multivariate SSA outperforming the Random Walk model for the Euro/Dollar, but with underwhelming results when predicting the Yen/Dollar trend.

EEMD and SSA were also used for Stock Price Prediction [Fenghua et al., 2014b], where the authors reported a trend prediction accuracy of approximately 68% with a combination of SSA and SVM and 63% using a combination of EEMD and SVM, both superior to accuracy obtained by SVM with raw data. This work also reported an accuracy of approximately 63% when using EEMD+SVM on the same dataset.

Chau et. al. [Chau and Wu, 2010] used a hybrid model of SSA for feature extraction and ANN in an attempt to improve the accuracy of daily rainfall forecasting. The authors reported that this fusion of techniques considerably improved the ANN performance, reducing the RMSE by over 50%.

In their review work [Lei et al., 2013], the authors explain that EMD has been widely applied and studied in fault diagnosis of rotating machinery. [Tang et al., 2011] have used EEMD and Least Squares Support Vector Regression (LSSVR) to predict Nuclear Energy Consumption, with results reporting a reduction of 40% in the RMSE in comparison to LSSVR using raw data.

These results show the range of application and effectiveness of these techniques when applied in conjunction with Machine Learning models, specially for noisy and non-stationary signals.

3.8 Final Considerations

Prediction in financial time series is still considered an open problem despite being an active field of study for decades.

There are very few studies focusing on Deep Learning models. Due to the problems mentioned in the section dedicated to Deep Learning, effectively applying Deep Neural Networks to problems in the field of finance have proved very difficult.

Wildly popular models such as SVM, Neural Networks and Random Forest perform better than other models most of the time, but the results vary by a large extent from study to study, depending on the features used as input and the testing protocol. Furthermore, considering the studies evaluated as part of this work, no base model or ensemble of models has shown to perform better than others across studies.

Finally, a comparison with mature fields such as pattern recognition on images reveal a notable lack of focus on the effectiveness of gauging the effectiveness of financial and time series features to predictive tasks.

Chapter 4

Methodology

4.1 Datasets

One of the datasets used for the experiments is the "Istanbul Stock Exchange", created and used by [Akbilgic et al., 2014] in their work and made available on the UCI repository [Lichman, 2013]. The dataset contains 536 data points, each representing a day and composed of nine floating point numbers indicating daily returns between January 5, 2009, to February 22, 2011, for the market indexes in Table 4.1.

Table 4.1: Istanbul Stock Exchange dataset

Market Index	**Description**
ISE100_TL	Istanbul Stock Market Index(TL)
ISE100_USD	Istanbul Stock Market Index(USD)
S&P 500	Standard & Poor's 500
DAX	German Stock Market Index
FTSE	London's Stock Market Index
NIKKEI	Tokyo's Stock Market Index
BVSP	Sao Paulo's Stock Market Index
EU	MSCI European Index
EM	MSCI Emerging Markets Index

The data was processed by the authors just so the days on which the Turkish stock exchange was closed were removed. Missing values on the time series indexes were replaced by their immediate valid past value. The original work reported the accuracy and the cumulative return obtained by trading ISE100 based on the predictions of their model, a Hybrid RBF Neural Network. ISE100_TL and ISE100_USD represent the same market index (ISE100), but one of them computed with respect to US Dollars and the other one with respect to the Turkish Lira. ISE100_TL was used in this work exclusively to generate results that could be compared to the accuracy and cumulative return reported in the original work. As an abstraction, in this work, we assume the existence of a portfolio that tracks the market indexes and can be rebalanced at will, as to effectively function as a single financial instrument, since Market Indexes cannot be traded.

The second dataset was built specifically for this study and will be reference as the "Global Market" dataset. This dataset contains 10 years of daily data, or 2608 days, from 03-Jan-2006 to 31-Dec-2015, of some of the most important market indexes in the world [Cetorelli and Peristiani, 2009], as well as the exchange rates between the dollar and other currencies that account for big parts of the composition of foreign exchange reserves [Ma and Villar, 2014]. Daily prices for the commodities Crude Oil, Natural Gas and Gold were also added to this dataset.

The data for all market indexes were taken from Yahoo! Finance, and the Exchange Rate data and Commodities were taken from the Federal Reserve Economic Data (FRED) - St. Louis Fred using the API they make available. The only pre-processing step between gathering the data and saving it was to fill the missing values with a valid predecessor value; no other normalization or transformation was applied such that other researchers can make use of the raw data and do whatever transformation they see fit.

Table 4.2: Global Market dataset

Instrument	Description
DJIA	Dow Jones Industrial Average
S&P500 (GSPC)	S&P Dow Jones index
Nasdaq (IXIC)	Nasdaq Composite index
NYSE (NYA)	New York Stock Exchange
CAC40 (FCHI)	Paris Stock Exchange
FTSE100 (FTSE)	London Stock Exchange
DAX (GDAXI)	Frankfurt Stock Exchange
Nikkei225 (N225)	Tokyo Stock Market
SSE (SSEC)	Shanghai Composite Index
BVSP (BVSP)	São Paulo Stock Exchange
JPUS	U.S Dollar x Japan Yen Exchange Rate
BZUS	U.S Dollar x Brazil Reais Exchange Rate
USEU	U.S Dollar x EU Euro
USUK	U.S Dollar x UK Pound Sterling
SZUS	U.S Dollar x Swiss Francs
CHUS	U.S Dollar x China Yuan
DCOILWTICO	WTI Crude Oil
DHHNGSP	Natural Gas
GOLDPMGBD228NLBM	Gold

4.2 Protocol

The protocol was designed to compare the trend accuracy and the theoretical cumulative return obtained by trading financial instruments based on the classification from 4 different classifiers, using 1-day lagged values of the raw time series in one case and the 1-day lagged values of the components obtained with EEMD.

Training and testing the models consisted of an interactive process that simulates daily trading. For the "Istanbul Stock Exchange", the initial training set consisted of the first 250 days, and the test set a single data point, the 251^{th} day. After each iteration, the day used as the test set gets added to the training set, and the next day with respect to the last test day is used as the new test set, up until the 450^{th} day is used as the test set. The initial training size, 250 days, and final testing day, 450, were chosen just so the results of this work could be compared to the results published by the authors of the dataset.

For the "Global Market" dataset, a subset was used to validate the results obtained on the "Istanbul Stock Exchange". This subset is composed of all the exchange rates daily data from 01-Jan-2012 to 31-Dec-2015. The initial training set consisted of 250 days just so the tests done on the Istanbul dataset could be replicated. The train/test iterations were carried out in a similar fashion, with one difference - since there are multiple years worth of data, the size of the window used as the training set increases by one day until the limit of 504 was reached, which is the equivalent of 2 years worth of "trading" days (252 days). When the limit size is reached, the oldest day in the training set is dropped, and a new day is added on each iteration. Since the financial markets change rapidly, the size limit of the window is set in such a way that the models are trained considering the most recent market dynamics.

For each dataset, each pre-processing step is done within the training and testing iterations as to not add look-ahead bias; as explained previously, decomposing the time series with EEMD as a pre-processing step would add a flaw to the protocol. After the decomposition, each component is normalized.

Figure 4.1 is a high-level description of the protocol used. Algorithms 3 and 4 is a detailed description of the protocol:

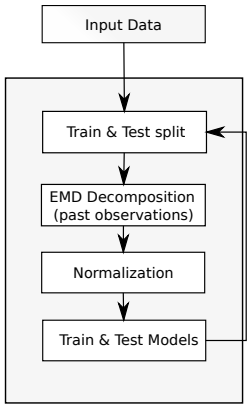


Figure 4.1: Protocol without look-ahead bias.

Algorithm 3 High level protocol

```
Require: data_set, nFeats, nIMFs
Ensure: results
results = []
for each combination of parameters do
    Get all feature combinations with nFeats features
    for Each feature combination do
        models = LinearSVM, RBF-SVM, RF and LogRegress
        lag_dataset = CreateLaggedDataset(dataset)
        results.append(RunTrainTest(0, 250, 450, lag_dataset, nIMFs, models))
    end for
end for
return results
```

Algorithm 4 RunTrainTest

```
Require: start_train, end_train, FINAL_DAY, dataset, nIMFs, models
Ensure: results
results = []
while end_train < FINAL_DAY do
    train_set = dataset[start_train:end_train - 1]
    test_set = dataset[end_train]
    imfs = EEMD(train_set + test_set, nIMFs)
    norm_Imfs = Normalize(imfs)
    imfs_train_set = norm_Imfs[start_train:end_train - 1]
    imfs_test_set = norm_Imfs[end_train]
    for each model in models do
        Train model 4-fold cross validation on imfs_train_set
        Test model on imfs_test_set
    end for
    end_train += 1
end while
for each model in models do
    results.append(model.results)
end for
return results
```

On algorithm 4, despite the training set and the testing set being concatenated before the decomposition, this does not add look-ahead bias to the protocol. The only information we do not possess at the end of each day is the class associated to the most current observation, which is defined by whether the price will go up or down by the end of the day tomorrow. We also need to consider that the test set is always composed of a single data point, which is computed with yesterday's and today's closing price; this is all data we possess. This protocol maps to a real

situation where the training set is composed of all the historical observations we have but the last data point of the time series, which is the percent change value computed between yesterday's closing price and today's closing price. The predictive models are trained and tested after the markets close, so we can predict, at the end of each day, what will happen tomorrow.

In order to evaluate the effectiveness of EEMD alone, we do an exhaustive search over all the possible combinations of market indexes by training and testing the models with each combination and storing the results for later comparison. The 1-day lagged values of the target index was also part of the feature pool.

The parameters S_number and number of siftings of EEMD were set to 4 and 50 respectively, the default value in the library used for the tests ([Luukko et al., 2016]). These parameters were not changed during the tests because preliminary simulations have shown that the impact different values had on the IMFs generated were negligible. The values of noise strength and the size of the ensemble were set in accordance to the guidelines presented on EEMD's seminal paper [Wu and Huang, 2009]: The noise strength amplitude was set to be 0.2 of the standard deviation of the input signal, and the size of the ensemble, K, was set at 250. Despite being parameters that do affect the decomposition accuracy, results in the literature indicate that increasing noise amplitudes and ensemble size do not alter the decomposition considerably as long as the added noise has moderate amplitude and the ensemble is large enough [Wu and Huang, 2009].

A total of 324 possible combinations of classifiers and features were used for the Istanbul dataset to build the pool of models using raw values and EEMD components as input. For each pool, four different classifiers were trained (Linear SVM, RBF-SVM, Logistic Regression and Random Forests) with all the possible combinations of 4 or more indexes in the feature vector. For the cases where EEMD was used to extract the components, the number of features is

multiplied by eight since this is the number of components extracted from the indexes, which matches the theoretical number of components extracted from time series of size n, $log_2 n$ ([Wu and Huang, 2009]).

For the forex data from the Global Market dataset, a total of 84 possible combinations of classifiers and features were used to build the pool of models using raw values and EEMD components as input. For each pool, the classifiers which performed the best on the Istanbul dataset were used (Linear SVM and Random Forests) with all the possible combinations of 3 or more indexes in the feature vector.

The metrics used to compare the results among the different models were the Trend accuracy and the Cumulative Return, the later being a popular metric to compare the performance of different financial instruments. The Trend accuracy is given by Equation 4.1:

$$\text{TrendAcc} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

where the True Positive (TP) and True Negative (TN) are the number of correct predictions for up trends and down trends, respectively. The denominator sums up to the total number of predictions performed by the model.

The standard formula for cumulative return (Rc) between days a and b is given by Equation 4.2

$$\text{Rc} = \prod_{i=a}^{b} (1 + \frac{P_{i+1} - Pi}{P_i}) \tag{4.2}$$

where P_i is the closing price of the financial instrument at the i^{th} day. However, to compute the cumulative return taking in consideration the accuracy of the predictions, Equation 4.2 needs to be slightly different:

$$\text{Rc} = \prod_{i=a}^{b} \begin{cases} 1 + abs(\frac{P_{i+t} - Pi}{P_i}) \mapsto TP \quad \text{and} \quad TN \\ 1 - abs(\frac{P_{i+t} - Pi}{P_i}) \mapsto FP \quad \text{and} \quad FN \end{cases} \tag{4.3}$$

In this scenario, we consider a hypothetical situation where we are able to short or long any one of these market indexes for a day, with the profit from this transaction being the full percentage change from today's to tomorrow's closing price.

The Cumulative Return is a specially important metric. Despite having a high accuracy, a specific model can present a lower cumulative return if it is doesn't perform well in detecting strong up or down movements. These two metrics were also used in the study that introduced the dataset used in this study ([Akbilgic et al., 2014]), which allows us to use the reported results as another data point. Both metrics are very popular in studies using financial data.

Chapter 5

Results

The experiments were conducted with the public dataset "Istanbul Stock Exchange" and the forex data from the dataset built for this study, both broken down in details in the “Dataset & Pre-processing” section. The procedure used for the tests is the same described in the methodology chapter.

Python was used to develop the code for the simulations, mainly due to its various auxiliary packages that were incredibly useful for reading and processing the data, training the models and generating the visualizations. Worth noting among these packages are:

- scikit-learn: A powerful and very popular framework for Machine Learning and Data Mining
- numpy: The defacto python package for array and matrix manipulations
- matplotlib: Provides support for a large variety of different visualizations
- pyeemd: A python wrapper around a C library for performing the ensemble empirical mode decomposition (EEMD)

- pandas: This package was developed within a hedge fund, and is particularly useful for working with financial time series data.

As the benchmarks for Accuracy and Cumulative Return, we use the coin flip probability of predicting accurately the trend and a buy-and-hold strategy, respectively. The buy-and-hold (BH) return for a financial instrument after t days is simply defined by Equation 5.1:

$$\mathrm{BH}_i = 1 + \frac{P_{i+t} - Pi}{P_i}, \tag{5.1}$$

where P_i is the closing price of the financial instrument at the i^{th} day. A return larger than 1 implies earning with respect to the initial capital, and a loss when the value is smaller.

The best results were reported in respect to the model, accuracy, cumulative return and input representation in Tables 5.2, 5.3, 5.4, 5.5, Figures 5.1, 5.2, **??** and 5.4. The bar charts show the best results across all models for the Istanbul dataset, and on the tables we report the best results per model for each dataset. The results obtained with EEMD used as a pre-processing step (with look-ahead bias) are referenced by the label EEMD*. We also present a comparison between the results obtained by the Akbilgic et.al. using a model of their authorship (HRBF-NN) and our algorithm using EEMD on table 5.1. The values in bold represent the best accuracy and cumulative return obtained by each model using the best combination of features found via exhaustive search, as explained previously.

For the sake of brevity, the list of features per model were not reported, but worth noting is the fact that the best number of market indexes for the models using the raw values were, on the large majority of the cases, larger than for the models using EEMD components. A single index added to the list of feature actually adds 8 components to the feature vector due to the decomposition, so this difference can be explained by the curse of dimensionality. For the results reported under EEMD*, there is a strong tendency of the best results being obtained with all

Table 5.1: Comparing study results to the one in the literature.

	ISE_TL					
	Raw		**EEMD**		**EEMD***	
	Accuracy	$R_c(\%)$	**Accuracy**	$R_c(\%)$	**Accuracy**	$R_c(\%)$
Lin SVM	0.660	246.65	0.600	**256.70**	0.660	331.25
RBF SVM	0.655	**271.53**	**0.620**	246.61	0.675	343.58
Log. Reg.	0.650	233.18	0.600	228.85	**0.685**	**360.44**
RF	0.615	207.17	0.560	162.47	0.615	217.25
HRBF-NN ([Akbilgic et al., 2014])	**0.68**	202	-		-	-

the features, which indicates the look-ahead bias embedded useful information about future behaviour in every market index decomposed.

The models, whether using the raw values or the EEMD components, were able to consistently beat the buy and hold strategy for the marked indexes in the Istanbul dataset, and also for the Forex data, or Exchange Rates, present in the dataset built for this study, as shown on Tables 5.3 and 5.5.

The results however show the difference that exists between the performance of the models when using a proper protocol for the tests and one with look-ahead bias added to it. In the large majority of the tests the models using the components extracted from the entire time series with EEMD as a pre-processing (tests with look-ahead bias) beat all the other models.

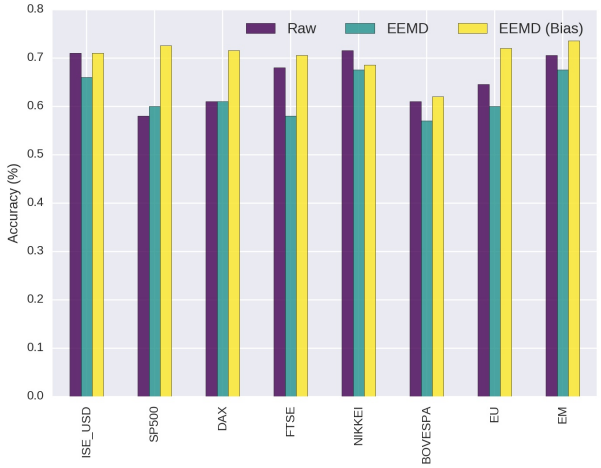


Figure 5.1: Best Accuracy per Market Index in the Istanbul dataset

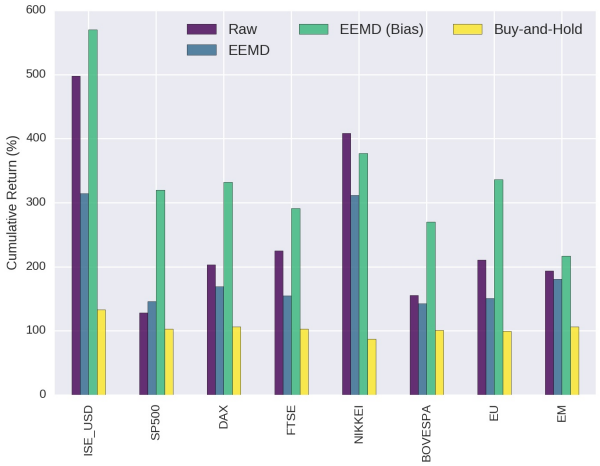


Figure 5.2: Best Cumulative Returns (R_c) per Market Index in the Istanbul dataset

Table 5.2: Best Accuracy results per predictive model (with best parameters & best feature combination) for the Istanbul dataset

	ISE_USD			**SP500**			**DAX**			**FTSE**		
	raw	eemd	eemd*	raw	eemd	eemd*	raw	eemd	eemd*	raw	eemd	eemd*
L.svm	**0.710**	0.660	0.700	0.545	**0.600**	0.725	0.600	**0.610**	0.710	0.665	0.575	0.705
R.svm	0.705	0.640	0.710	0.550	0.595	0.715	0.590	0.595	0.715	0.635	0.560	0.635
L.reg	0.700	0.625	0.700	0.535	0.590	0.715	0.600	**0.610**	0.715	0.665	0.570	0.685
RF	0.635	0.620	0.640	0.580	0.580	0.640	**0.610**	0.600	0.605	**0.68**	0.580	0.565
	NIKKEI			**BOVESPA**			**EU**			**EM**		
	raw	eemd	eemd*	raw	eemd	eemd*	raw	eemd	eemd*	raw	eemd	eemd*
L.svm	**0.715**	0.675	0.675	0.500	0.510	0.600	0.630	0.590	0.705	**0.705**	0.675	0.715
R.svm	0.705	0.660	0.675	0.505	0.520	0.620	0.595	0.590	0.700	0.695	0.650	0.735
L.reg	0.710	0.660	0.685	0.505	0.515	0.600	0.630	0.590	0.720	**0.705**	0.655	0.715
RF	0.680	0.635	0.600	**0.610**	0.570	0.580	**0.645**	0.600	0.615	0.695	0.650	0.645

In contrast to these results, a protocol created to eliminate the look-ahead bias from the application of EEMD tells a different story. For the Istanbul dataset, despite the fact that models using the EEMD components achieved a considerable improvement in the cumulative return with respect to the original study, the best performing models when predicting the trend direction, and also with respect to the cumulative return were, in their majority, models using the raw percent change values instead of the components generated with EEMD. For the ISE_USD, DAX, FTSE, NIKKEI, BOVESPA, EU and EM indexes, the best models were the models using the raw percent change values instead of the values extracted with EEMD. For the S&P500

Table 5.3: Best Cumulative Return results per predictive model (with best parameters & best feature combination) for the Istanbul dataset

	ISE_USD			**SP500**			**DAX**			**FTSE**		
	Raw	eemd	eemd*	raw	eemd	eemd*	raw	eemd	eemd*	raw	eemd	eemd*
L.svm	481.9	317.6	541.0	108.9	120.1	319.6	**203.3**	154.2	332.2	**224.7**	147.0	290.8
R.svm	492.1	314.0	570.0	116.2	124.7	306.6	190.3	136.7	295.8	196.4	145.8	262.9
L.reg	**497.6**	311.4	553.7	104.6	133.5	297.4	202.7	168.7	316.4	214.8	146.4	277.1
RF	276.5	279.9	309.8	128.2	**146.0**	230.5	177.5	157.2	204.3	179.4	154.9	196.2
BH	132.7	132.7	132.7	103.0	103.0	103.0	106.1	106.1	106.1	103.0	103.0	103.0
	NIKKEI			**BOVESPA**			**EU**			**EM**		
	raw	eemd	eemd*	raw	eemd	eemd*	raw	eemd	eemd*	raw	eemd	eemd*
L.svm	**408.3**	294.9	376.9	106.7	100.1	240.6	**210.3**	150.6	322.8	**193.5**	180.3	207.2
R.svm	391.7	311.3	369.2	116.0	104.1	270.0	201.5	141.0	290.1	185.0	176.5	216.7
L.reg	390.0	299.7	373.1	107.1	100.7	246.7	202.4	144.8	335.8	189.9	169.6	208.7
RF	297.7	217.0	223.3	**155.6**	142.1	180.7	174.2	144.1	206.9	189.8	157.4	180.6
BH	87.5	87.5	87.5	100.7	100.7	100.7	99.2	99.2	99.2	106.5	106.5	106.5

market index, the models trained with the EEMD components performed consistently better than the models trained with the raw values.

In terms of the results obtained by each individual model with the raw values and the EEMD components without look-ahead bias, it is interesting to note that the Linear SVM was the best performing model when predicting the trend of 5 out of 8 market indexes contained in the dataset. Random Forests, despite achieving good accuracy results, are the worst performing model in general regarding cumulative returns, with less accurate predictions when detecting large movements.

For the models trained with the exchange rates from the "Global Market" dataset, we identify the same pattern. The best models trained with the EEMD components generated from the entire time series as a pre-processing step perform considerably better than all the others.

Aside from these results, the protocol without look-ahead bias generated models using the EEMD components and the raw features that were fairly matched. The models using the EEMD components performed, in general, slightly better in terms of maximum cumulative return, and slightly worse in terms of maximum accuracy. The best models for this dataset were

Random Forests, and the models trained with the raw values outperformed the models trained with the EEMD components on 4 out of the 6 exchange rates in terms of accuracy (BZUS, CHUS, SZUS, USEU), but performed better in terms of cumulative return only on CHUS. The best models on each case performed better than buy and hold in terms of cumulative return for all but 2 of the exchange rates, BZUS and JPUS.

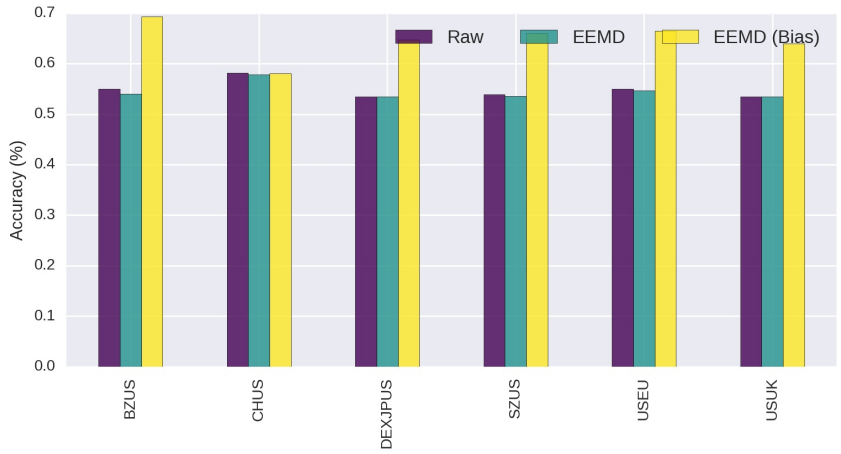


Figure 5.3: Best Accuracy per Exchange Rate in the Global Market dataset

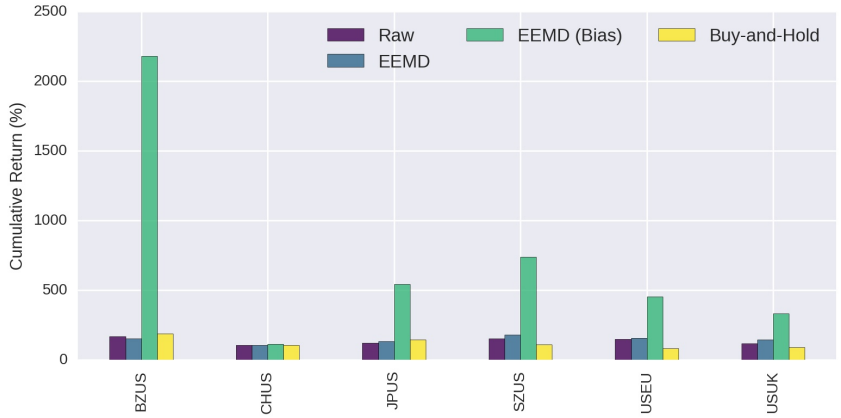


Figure 5.4: Best Cumulative Returns (R_c) per Exchange Rate in the Global Market dataset

Table 5.4: Best Accuracy results per predictive model (with best parameters & best feature combination) for Exchange Rates (Global Market)

		BZUS			**CHUS**			**JPUS**	
	raw	eemd	eemd*	raw	eemd	eemd*	raw	eemd	eemd*
Lin SVM	0.503	0.540	0.693	0.571	0.578	0.58	0.521	0.518	0.532
RF	**0.55**	0.530	0.580	**0.582**	0.557	0.580	**0.534**	**0.534**	0.647

		SZUS			**USEU**			**USUK**	
	raw	eemd	eemd*	raw	eemd	eemd*	raw	eemd	eemd*
Lin SVM	0.518	0.536	0.660	0.534	0.547	0.665	0.512	0.526	0.640
RF	**0.539**	0.510	0.573	**0.550**	0.522	0.568	**0.535**	**0.535**	0.550

Table 5.5: Best Cumulative Return results per predictive model (with best parameters & best feature combination) for Exchange Rates (Global Market)

		BZUS			**CHUS**			**JPUS**	
	raw	eemd	eemd*	raw	eemd	eemd*	raw	eemd	eemd*
Lin SVM	94.0	150.7	2181.4	102.3	106.6	111.3	122.1	134.7	543.3
RF	169.6	137.6	357.4	**107.4**	106.4	113.8	116.8	112.5	170.6
BH	**189.2**	-	-	103.88	-	-	**143.7**	-	-

		SZUS			**USEU**			**USUK**	
	raw	eemd	eemd*	raw	eemd	eemd*	raw	eemd	eemd*
Lin SVM	122.1	**180.3**	739.1	128.1	**155.4**	452.4	115.8	**145.4**	333.1
RF	150.8	129.2	223.0	146.7	136.9	230.6	105.9	142.9	162.4
BH	109.19	-	-	82.46	-	-	91	-	-

Chapter 6

Conclusion

The objective of this study was to evaluate the effectiveness of Ensemble Empirical Mode Decomposition at generating features to be used as trend predictors for market indexes. For this purpose, two different datasets were used: the Istanbul Market Index dataset ([Akbilgic et al., 2014]) and a dataset that was built exclusively for this thesis, the Global Market dataset. The testing protocol designed for this study, in contrast to protocols defined in other studies found in the literature, does not add look-ahead bias through the use of EEMD, and thus better represents how this technique would be used in a real case scenario - on an actual trading system.

The results obtained with the use of our protocol indicate, in contrast to the results presented in the literature, that the models trained with the EEMD components do not outperform, in terms of Accuracy and Cumulative Return, the models trained with the raw percent values for the majority of the market indexes contained in the "Istanbul Stock Exchange" dataset. The exception to this was S&P500, which was an isolated case where the difference between both models was small compared to the other use cases.

For the forex data from the Global Market dataset, the differences between the best models trained with the raw values and the EEMD components were not substantial. Despite achieving a slightly better performance in terms of cumulative returns, the models trained with the EEMD components did worse than the models trained with the raw values in terms of accuracy.

On the other hand, the models trained with the EEMD components extracted as a pre-processing step from the entire time series outperformed all the other models in the majority of the cases on both datasets, indicating that the look-ahead bias contained in protocols found in the literature will, most of the time, affect the accuracy and the cumulative return of the models by generating components that encode information about the future on past data points. These results reinforce the need to be extremely careful when using techniques that might make use of values that are not contained in the training set.

These results, however, do not necessarily mean that the EEMD components are not useful as predictors. They might not be better predictors on their own when compared to the raw percentage values, but these components might be useful if used in conjunction with the raw values. Additionally, the models did perform better when predicting the trend of S&P500, and performed slightly better than their raw values counterpart for Exchange Rates, which indicate this technique might be useful for prediction tasks on specific time series - just not to the level of what some studies would lead us to believe. Care must be taken though, as a time series of size n will be decomposed in $log_2 n$ components, which makes the curse of dimensionality a problem to be considered even when there are just a few predictors.

As future work, the author could look further into the impact caused by the rough decomposition at the end of the time series when using EMD/EEMD. The impossibility of interpolating the very last value of series to subsequent points generate a crude decomposition at

the end, which might actually be one of the main culprits for the drastic reduction in accuracy obtained by the classification models. In addition, the use of other decomposition techniques such as SSA might be evaluated and compared to the results obtained in this study.

Bibliography

[Akbilgic et al., 2014] Akbilgic, O., Bozdogan, H., and Balaban, M. E. (2014). A novel Hybrid RBF Neural Networks model as a forecaster. *Statistics and Computing*, 24(3):365–375.

[Al-Hnaity and Abbod, 2015] Al-Hnaity, B. and Abbod, M. (2015). A novel hybrid ensemble model to predict FTSE100 index by combining neural network and EEMD. In *Control Conference (ECC), 2015 European*, pages 3021–3028. IEEE.

[An et al., 2013] An, N., Zhao, W., Wang, J., Shang, D., and Zhao, E. (2013). Using multi-output feedforward neural network with empirical mode decomposition based signal filtering for electricity demand forecasting. *Energy*, 49:279–288.

[Atsalakis and Valavanis, 2009] Atsalakis, G. S. and Valavanis, K. P. (2009). Surveying stock market forecasting techniques–part ii: Soft computing methods. *Expert Systems with Applications*, 36(3):5932–5941.

[Ballings et al., 2015] Ballings, M., Van den Poel, D., Hespeels, N., and Gryp, R. (2015). Evaluating multiple classifiers for stock price direction prediction. *Expert Systems with Applications*, 42(20):7046–7056.

[Bao et al., 2005] Bao, Y.-K., Liu, Z.-T., Guo, L., and Wang, W. (2005). Forecasting stock composite index by fuzzy support vector machines regression. In *Machine Learning and*

Cybernetics, 2005. Proceedings of 2005 International Conference on, volume 6, pages 3535–3540. IEEE.

[Batres-Estrada, 2015] Batres-Estrada, B. (2015). Deep learning for multivariate financial time series.

[Cao et al., 2015] Cao, W., Hu, L., and Cao, L. (2015). Deep modeling complex couplings within financial markets. In *AAAI*, pages 2518–2524.

[Cetorelli and Peristiani, 2009] Cetorelli, N. and Peristiani, S. (2009). Prestigious stock exchanges: a network analysis of international financial centers. *Federal Reserve Bank of New York, Staff Report*, (384).

[Chau and Wu, 2010] Chau, K. and Wu, C. (2010). A hybrid model coupled with singular spectrum analysis for daily rainfall prediction. *Journal of Hydroinformatics*, 12(4):458–473.

[Chen, 2015] Chen, X. (2015). *Stock Price Prediction via Deep Belief Networks*. PhD thesis, UNIVERSITY OF NEW BRUNSWICK.

[Cheng et al., 2012] Cheng, C., Xu, W., and Wang, J. (2012). A comparison of ensemble methods in financial market prediction. In *Computational Sciences and Optimization (CSO), 2012 Fifth International Joint Conference on*, pages 755–759. IEEE.

[Dan et al., 2014] Dan, J., Guo, W., Shi, W., Fang, B., and Zhang, T. (2014). Deterministic echo state networks based stock price forecasting. In *Abstract and Applied Analysis*, volume 2014. Hindawi Publishing Corporation.

[Deng et al., 2009] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

[Fabozzi, 2008] Fabozzi, F. J. (2008). *Handbook of Finance, Financial Markets and Instruments*, volume 1. John Wiley & Sons.

[Fawcett, 2006] Fawcett, T. (2006). An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874.

[Fenghua et al., 2014a] Fenghua, W., Jihong, X., Zhifang, H., and Xu, G. (2014a). Stock Price Prediction based on SSA and SVM. *Procedia Computer Science*, 31:625–631.

[Fenghua et al., 2014b] Fenghua, W., Jihong, X., Zhifang, H., and Xu, G. (2014b). Stock price prediction based on ssa and svm. *Procedia Computer Science*, 31:625–631.

[Furlaneto et al., 2017] Furlaneto, D. C., Oliveira, L. S., Menotti, D., and Cavalcanti, G. D. (2017). Bias effect on predicting market trends with emd. *Expert Systems with Applications*, 82:19–26.

[Gershenfeld and Weigend, 1994] Gershenfeld, N. and Weigend, A. (1994). The future of time series: Learning and understanding. time series prediction: Forecasting the future and understanding the past.

[Ghodsi et al., 2010] Ghodsi, M., Hassani, H., and Sanei, S. (2010). Extracting fetal heart signal from noisy maternal ECG by singular spectrum analysis. *Journal of Statistics and its Interface, Special Issue on the Application of SSA*, 3(3):399–411.

[Graham, 2014] Graham, B. (2014). Fractional max-pooling. *arXiv preprint arXiv:1412.6071*.

[Graves et al., 2013] Graves, A., Mohamed, A.-r., and Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 6645–6649. IEEE.

[Guo et al., 2012] Guo, Z., Zhao, W., Lu, H., and Wang, J. (2012). Multi-step forecasting for wind speed using a modified emd-based artificial neural network model. *Renewable Energy*, 37(1):241–249.

[Guresen et al., 2011] Guresen, E., Kayakutlu, G., and Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, 38(8):10389–10397.

[Hassan and Nath, 2005] Hassan, M. R. and Nath, B. (2005). Stock market forecasting using hidden markov model: a new approach. In *Intelligent Systems Design and Applications, 2005. ISDA'05. Proceedings. 5th International Conference on*, pages 192–196. IEEE.

[Hassani et al., 2009] Hassani, H., Heravi, S., and Zhigljavsky, A. (2009). Forecasting european industrial production with singular spectrum analysis. *International journal of forecasting*, 25(1):103–118.

[Hassani et al., 2010] Hassani, H., Soofi, A. S., and Zhigljavsky, A. A. (2010). Predicting daily exchange rate with singular spectrum analysis. *Nonlinear Analysis: Real World Applications*, 11(3):2023–2034.

[Hossain et al., 2009] Hossain, A., Zaman, F., Nasser, M., and Islam, M. M. (2009). Comparison of garch, neural network and support vector machine in financial time series prediction. In *Pattern Recognition and Machine Intelligence*, pages 597–602. Springer.

[Hsu et al., 2016] Hsu, M.-W., Lessmann, S., Sung, M.-C., Ma, T., and Johnson, J. E. (2016). Bridging the divide in financial market forecasting: machine learners vs. financial economists. *Expert Systems with Applications*, 61:215–234.

[Huang et al., 1998] Huang, N. E., Shen, Z., Long, S. R., Wu, M. C., Shih, H. H., Zheng, Q., Yen, N.-C., Tung, C. C., and Liu, H. H. (1998). The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 454, pages 903–995. The Royal Society.

[Huang et al., 2003] Huang, N. E., Wu, M.-L. C., Long, S. R., Shen, S. S., Qu, W., Gloersen, P., and Fan, K. L. (2003). A confidence limit for the empirical mode decomposition and Hilbert spectral analysis. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 459, pages 2317–2345. The Royal Society.

[Ismail et al., 2011] Ismail, S., Shabri, A., and Samsudin, R. (2011). A hybrid model of self-organizing maps (som) and least square support vector machine (lssvm) for time-series forecasting. *Expert Systems with Applications*, 38(8):10574–10578.

[Kaastra and Boyd, 1995] Kaastra, I. and Boyd, M. S. (1995). Forecasting futures trading volume using neural networks. *Journal of Futures Markets*, 15(8):953–970.

[Kim, 2003] Kim, K.-j. (2003). Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1):307–319.

[Kumar and Thenmozhi, 2006] Kumar, M. and Thenmozhi, M. (2006). Forecasting stock index movement: A comparison of support vector machines and random forest. In *Indian Institute of Capital Markets 9th Capital Markets Conference Paper*.

[Kuremoto et al., 2014] Kuremoto, T., Kimura, S., Kobayashi, K., and Obayashi, M. (2014). Time series forecasting using a deep belief network with restricted boltzmann machines. *Neurocomputing*, 137:47–56.

[Lakonishok and Smidt, 1988] Lakonishok, J. and Smidt, S. (1988). Are seasonal anomalies real? a ninety-year perspective. *Review of Financial Studies*, 1(4):403–425.

[Lee et al., 2015] Lee, C.-Y., Gallagher, P. W., and Tu, Z. (2015). Generalizing pooling functions in convolutional neural networks: Mixed. *Gated, and Tree, arXiv e-print sarXiv*, 1509.

[Lei et al., 2009] Lei, Y., He, Z., and Zi, Y. (2009). Application of the eemd method to rotor fault diagnosis of rotating machinery. *Mechanical Systems and Signal Processing*, 23(4):1327–1338.

[Lei et al., 2013] Lei, Y., Lin, J., He, Z., and Zuo, M. J. (2013). A review on empirical mode decomposition in fault diagnosis of rotating machinery. *Mechanical Systems and Signal Processing*, 35(1):108–126.

[Lichman, 2013] Lichman, M. (2013). UCI machine learning repository.

[Lin et al., 2009] Lin, X., Yang, Z., and Song, Y. (2009). Short-term stock price prediction based on echo state networks. *Expert systems with applications*, 36(3):7313–7317.

[Lin et al., 2013] Lin, Y., Guo, H., and Hu, J. (2013). An svm-based approach for stock market trend prediction. In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–7. IEEE.

[Luukko et al., 2016] Luukko, P., Helske, J., and Räsänen, E. (2016). Introducing libeemd: A program package for performing the ensemble empirical mode decomposition. *Computational Statistics*, 31(2):545–557.

[Ma and Villar, 2014] Ma, G. and Villar, A. (2014). Internationalisation of emerging market currencies. *BIS Paper*, (78d).

[Mahfoud and Mani, 1996] Mahfoud, S. and Mani, G. (1996). Financial forecasting using genetic algorithms. *Applied Artificial Intelligence*, 10(6):543–566.

[Malkiel, 2003] Malkiel, B. G. (2003). The efficient market hypothesis and its critics. *The Journal of Economic Perspectives*, 17(1):59–82.

[Mikosch and Stărică, 2004] Mikosch, T. and Stărică, C. (2004). Nonstationarities in financial time series, the long-range dependence, and the igarch effects. *Review of Economics and Statistics*, 86(1):378–390.

[Ou and Wang, 2009] Ou, P. and Wang, H. (2009). Prediction of stock market index movement by ten data mining techniques. *Modern Applied Science*, 3(12):p28.

[Pascanu et al., 2012] Pascanu, R., Mikolov, T., and Bengio, Y. (2012). On the difficulty of training recurrent neural networks. *arXiv preprint arXiv:1211.5063*.

[Patel et al., 2015] Patel, J., Shah, S., Thakkar, P., and Kotecha, K. (2015). Predicting stock market index using fusion of machine learning techniques. *Expert Systems with Applications*, 42(4):2162–2172.

[Polat and Güneş, 2007] Polat, K. and Güneş, S. (2007). Detection of ecg arrhythmia using a differential expert system approach based on principal component analysis and least square support vector machine. *Applied Mathematics and Computation*, 186(1):898–906.

[Ponsich et al., 2013] Ponsich, A., Jaimes, A. L., and Coello, C. A. C. (2013). A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications. *Evolutionary Computation, IEEE Transactions on*, 17(3):321–344.

[Preethi and Santhi, 2012] Preethi, G. and Santhi, B. (2012). Stock market forecasting techniques: A survey. *Journal of Theoretical & Applied Information Technology*, 46(1).

[Rai, 2006] Rai, M. M. (2006). Single-and multiple-objective optimization with differential evolution and neural networks. *VKI lecture series: introduction to optimization and multidisciplinary design*.

[Sapankevych and Sankar, 2009] Sapankevych, N. I. and Sankar, R. (2009). Time series prediction using support vector machines: a survey. *Computational Intelligence Magazine, IEEE*, 4(2):24–38.

[Sharma et al., 2011] Sharma, N., Sharma, P., Irwin, D., and Shenoy, P. (2011). Predicting solar generation from weather forecasts using machine learning. In *Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on*, pages 528–533. IEEE.

[Shumway and Stoffer, 2013] Shumway, R. H. and Stoffer, D. S. (2013). *Time series analysis and its applications*. Springer Science & Business Media.

[Sun and Li, 2012] Sun, J. and Li, H. (2012). Financial distress prediction using support vector machines: Ensemble vs. individual. *Applied Soft Computing*, 12(8):2254–2265.

[Takeuchi and Lee,] Takeuchi, L. and Lee, Y.-Y. A. Applying deep learning to enhance momentum trading strategies in stocks.

[Tang et al., 2011] Tang, L., Wang, S., and Yu, L. (2011). EEMD-LSSVR-Based Decomposition-and-Ensemble Methodology with Application to Nuclear Energy Consumption Forecasting. In *Computational Sciences and Optimization (CSO), 2011 Fourth International Joint Conference on*, pages 589–593. IEEE.

[Tsai et al., 2011] Tsai, C.-F., Lin, Y.-C., Yen, D. C., and Chen, Y.-M. (2011). Predicting stock returns by classifier ensembles. *Applied Soft Computing*, 11(2):2452–2459.

[Tsay, 2005] Tsay, R. S. (2005). *Analysis of financial time series*, volume 543. John Wiley & Sons.

[Vautard et al., 1992] Vautard, R., Yiou, P., and Ghil, M. (1992). Singular-spectrum analysis: A toolkit for short, noisy chaotic signals. *Physica D: Nonlinear Phenomena*, 58(1):95–126.

[Vui et al., 2013] Vui, C. S., Soon, G. K., On, C. K., Alfred, R., and Anthony, P. (2013). A review of stock market prediction with artificial neural network (ann). In *Control System, Computing and Engineering (ICCSCE), 2013 IEEE International Conference on*, pages 477–482. IEEE.

[Wan et al., 2013] Wan, L., Zeiler, M., Zhang, S., Cun, Y. L., and Fergus, R. (2013). Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1058–1066.

[Wang et al., 2012] Wang, T., Zhang, M., Yu, Q., and Zhang, H. (2012). Comparing the applications of EMD and EEMD on time–frequency analysis of seismic signal. *Journal of Applied Geophysics*, 83:29–34.

[Wang et al., 2015] Wang, W.-c., Chau, K.-w., Xu, D.-m., and Chen, X.-Y. (2015). Improving forecasting accuracy of annual runoff time series using arima based on eemd decomposition. *Water Resources Management*, 29(8):2655–2675.

[Wang and Choi, 2013] Wang, Y. and Choi, I.-C. (2013). Market index and stock price direction prediction using machine learning techniques: An empirical study on the kospi and hsi. *arXiv preprint arXiv:1309.7119*.

[Wei and Cheng, 2012] Wei, L.-Y. and Cheng, C.-H. (2012). A hybrid recurrent neural networks model based on synthesis features to forecast the taiwan stock market. *International Journal of Innovative Computing Information and Control*, 8(8):5559–5571.

[Wiener, 1949] Wiener, N. (1949). *Extrapolation, interpolation, and smoothing of stationary time series*, volume 2. MIT press Cambridge, MA.

[Wu and Huang, 2009] Wu, Z. and Huang, N. E. (2009). Ensemble empirical mode decomposition: a noise-assisted data analysis method. *Advances in adaptive data analysis*, 1(01):1–41.

[Xiong et al., 2011] Xiong, T., Bao, Y., Hu, Z., Zhang, R., and Zhang, J. (2011). Hybrid decomposition and ensemble framework for stock price forecasting: a comparative study. *Advances in Adaptive Data Analysis*, 3(04):447–482.

[Yu et al., 2008] Yu, L., Wang, S., and Lai, K. K. (2008). Forecasting crude oil price with an EMD-based neural network ensemble learning paradigm. *Energy Economics*, 30(5):2623–2635.

[Zhu et al., 2011] Zhu, H., Wang, Y., Wang, K., and Chen, Y. (2011). Particle swarm optimization (pso) for the constrained portfolio optimization problem. *Expert Systems with Applications*, 38(8):10161–10169.