2014 ASE BIGDATA/SOCIALCOM/CYBERSECURITY Conference, Stanford University, May 27-31, 2014

Prediction of Financial Time Series Using Hidden Markov Models

Lev Brailovskiy and Dr. Maya Herman

Dept. Mathematics and Computer Science, The Open University of Israel

ABSTRACT

Hidden Markov Models (HMM) is a powerful machine learning model. HMM's main usage has been in solving classification and pattern recognition problems in biology, speech and voice recognition. In recent days, attempts have been made to use HMM for prediction in general and prediction of time series in particular. However, this is not straightforward. To overcome the challenges in predicting time series with HMM some hybrid approaches have been applied. This paper has two main objectives. The first, is to compare HMM with other models when used for prediction of financial time series. We will show comparison between HMM and other models and also between different types of HMM's as unique contribution of this work. In recent years, prediction of stock market behavior became a field of great interest to many scientists. Hence, as a case study, we will use our implementation and examine HMM with Multivariate Normal Distribution and Gaussian Mixture Model in hidden states. We use nearest likelihood prediction algorithm and compare results with MAPE. As training dataset, we will use some popular stock from NASDAQ and S&P500 indexes historical data. The second, is to showcase a working system with generic HMM, advanced training and prediction algorithms implementation in C#. All the code is open sourced as contribution to the community. The system allows researchers from different disciplines to use HMM and other models to solve classification, pattern recognition and prediction on various series represented by k-dimensional vectors arrays and a way to compare and re-use the models under test.

METHODOLOGY

Time series – sequence of data points, measured typically at successive points in time spaced at uniform time intervals

Tuesday, Feb 15, 2011
AAPL
Open: 359.19
High: 359.97
Low: 357.55
Close: 359.90
EMA 30: 343.92
400
375
350
325
300
14. Feb
28. Feb
14. Mar
28. Mar
11. Apr
25. Apr
9. May
23. May
6. Jun
20. Jun
4. Jul
18. Jul
1. Aug
15. Aug
29. Aug

Objectives

- Forecasting of future values and trend for given time series
- Implementation of advanced models and machine learning algorithms

Analysis

Hidden Markov Model (HMM) – statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved states.

- Given observation sequence O=$\{o_0, \dots, o_T\}$ and model λ = (A, B,π), how we efficiently compute the probability of getting the observation sequence given the model P(O| λ).
- Given observation sequence O=$\{o_0, \dots, o_T\}$ and model λ = (A, B,π), how we efficiently compute Q=$\{q_0, \dots, q_T\}$, state sequence that best explains the observation sequence.
- How do we efficiently update model parameters λ = (A, B,π) to maximize P(O| λ).

Data

- Training data set
- Test data set

Training

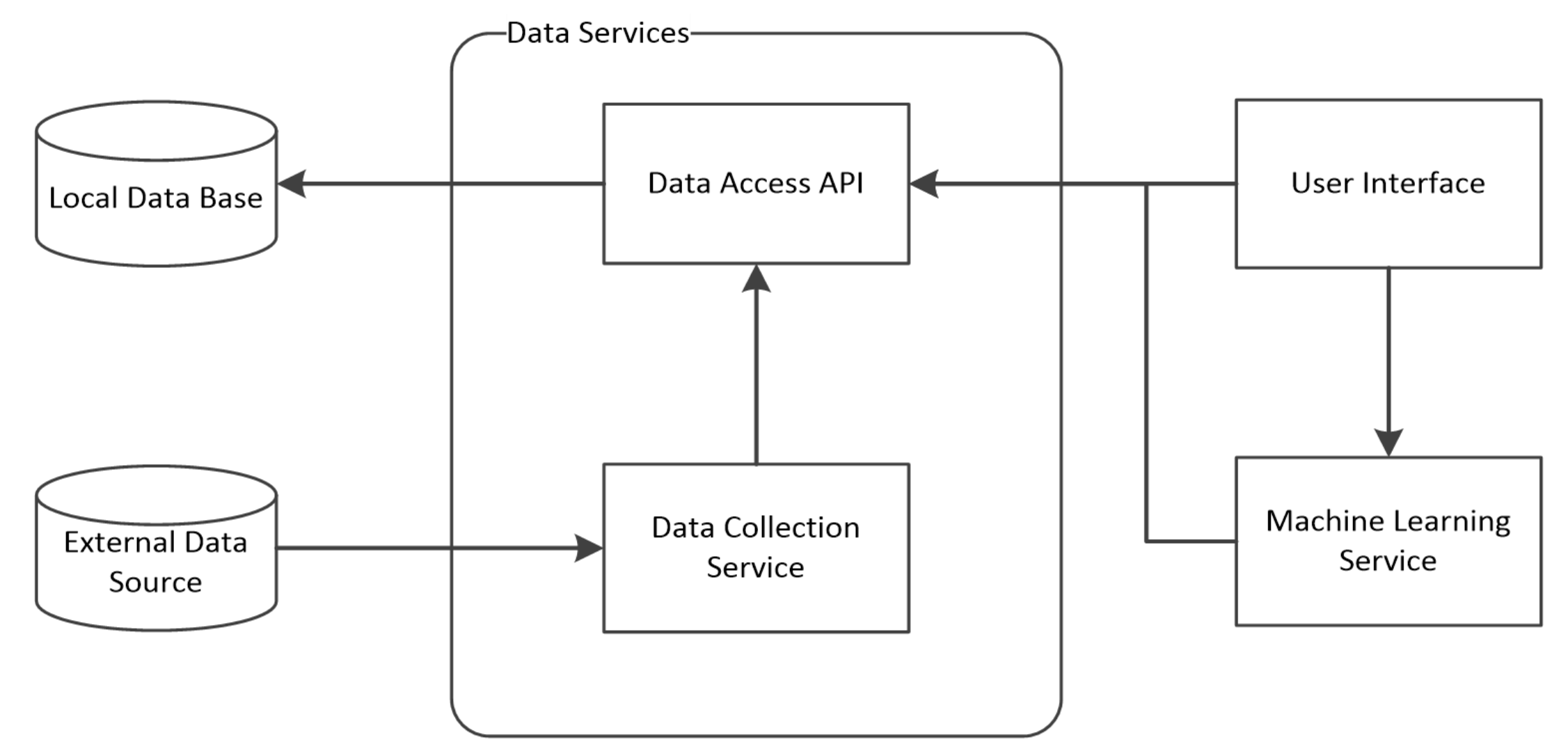
- Initialization with K-Means algorithm
- Baum-Welch algorithm (special case of EM algorithm). Training of HMM and it's Mixture Components

Prediction

- Likelihood based prediction using Forward-Backward procedure for likelihood calculations
- Viterbi based prediction
- Prediction error measures

SYSTEM DESIGN AND IMPLEMENTATION

System Architecture



System Modules

Mathematical module – extensions to basic functionality of System.Math in the .NET Framework. The main additions are implementations for vector and matrix actions. For more efficient calculation of determinant and inverted matrix we implemented LU and Cholesky decompositions.
Statistical module – base classes and implementation for various distribution functions. We implemented the following distributions :

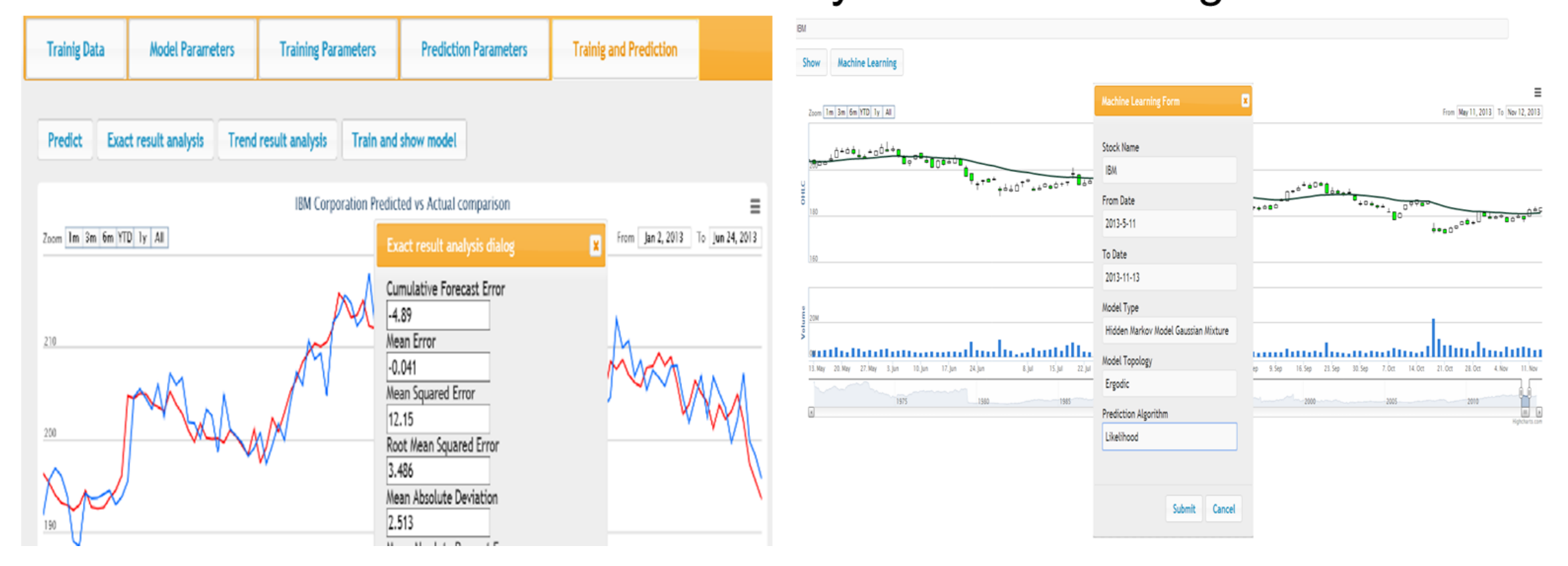
- Univariate distribution – discrete and continuous
- Gaussian distribution – Univariate and Multivariate
- Gaussian Mixture Model

Machine learning module - base classes of advanced machine learning models. Implementation of advanced algorithms.

Algorithm	Complexity
Forward-Backward	*$O(TN^2)$*
Viterbi	*$O(TN^2)$*
Baum-Welch	*$O(TN^2)$*
K-Means	*$O(TCID)$*
Likelihood based prediction	*$O(dTN^2)$*
Viterbi based prediction	*$O(TN^2)$*

Name
Hidden Markov Model – Univariate distribution
Hidden Markov Model – Multivariate distribution
Hidden Markov Model – Mixture distribution
Gaussian Mixture Model

User interface – web application that allows the user to select stock of interest, date range and model. Training and prediction are based on selected data and model and analysis of forecasting results

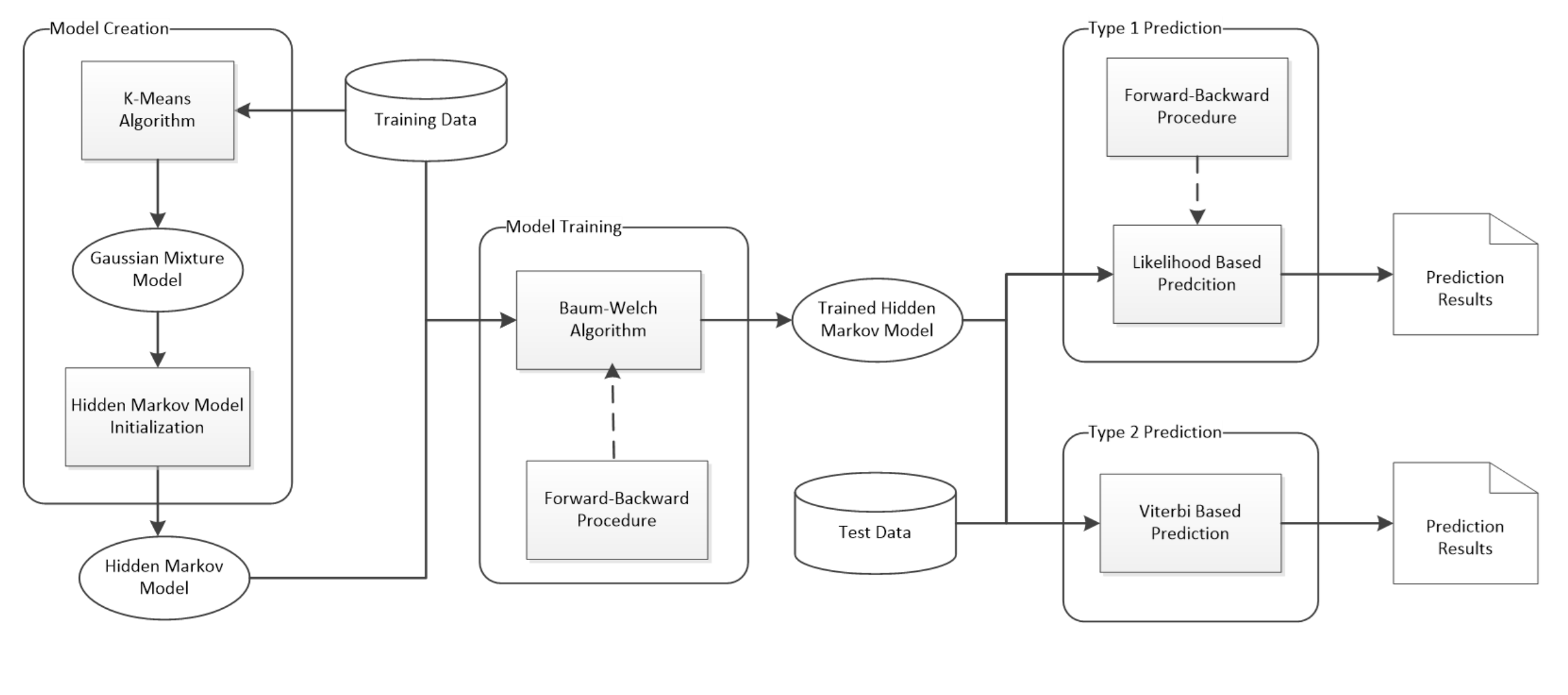


Cumulative Forecast Error: -4.89
Mean Error: -0.041
Mean Squared Error: 12.15
Root Mean Squared Error: 3.486
Mean Absolute Deviation: 2.513

Stock Name: IBM
From Date: 2013-5-11
To Date: 2013-11-13
Model Type: Hidden Markov Model Gaussian Mixture
Model Topology: Ergodic
Prediction Algorithm: Likelihood

Data Services – retrieving data from external data sources. Cleaning and transforming received data. Storing the data in local data base instance and exposing it to all other application modules

Data flow and Component Integration



ISBN: 978-1-62561-000-3

EXPERIMENTAL RESULTS

Case Study

- Three stocks from the NASDAQ Index with given date range
- Stocks data are divided to training and test data sets
- MAPE Error measure

Training Data

Observations	From Date	To Date
Microsoft Corp.	10 February 2004	10 September 2004
IBM Corp.	10 June 2004	10 September 2004
Apple Inc.	10 February 2004	10 September 2004

Test Data

Observations	From Date	To Date
Microsoft Corp.	13 September 2004	21 January 2005
IBM Corp.	13 September 2004	21 January 2005
Apple Inc.	13 September 2004	21 January 2005

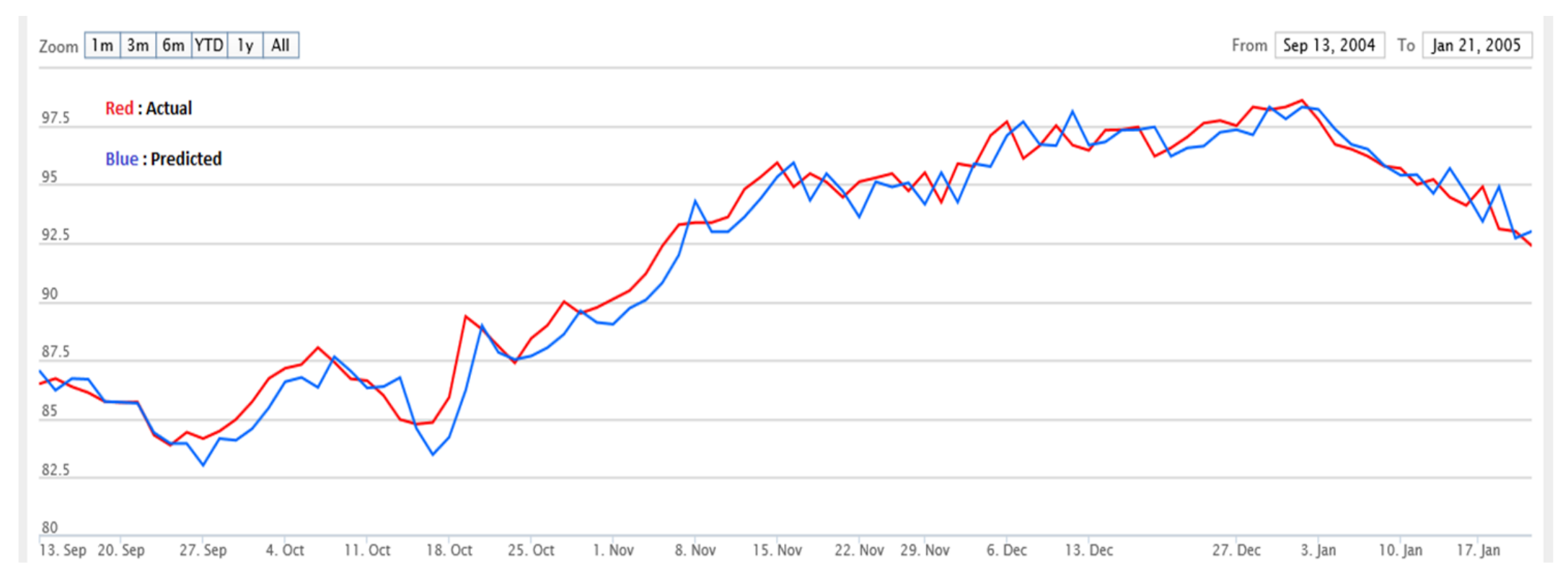
Comparison of HMM and other advanced machine learning models

- HMM Parameter

Parameter	Value
Number of States (N)	4
Number of Mixture Components (M)	5
Observation Dimension (D)	4
Mode Type	Ergodic

- MAPE of different approaches

Observation	*HMM*	*ARIMA*	*ANN*
Microsoft Corp.	1.022	1.324	1.326
IBM Corp.	0.777	0.972	0.972
Apple Inc.	1.962	1.801	1.801



Actual v/s Forecasted Value for IBM Corp. 2004-09-13 to 2004-01-21

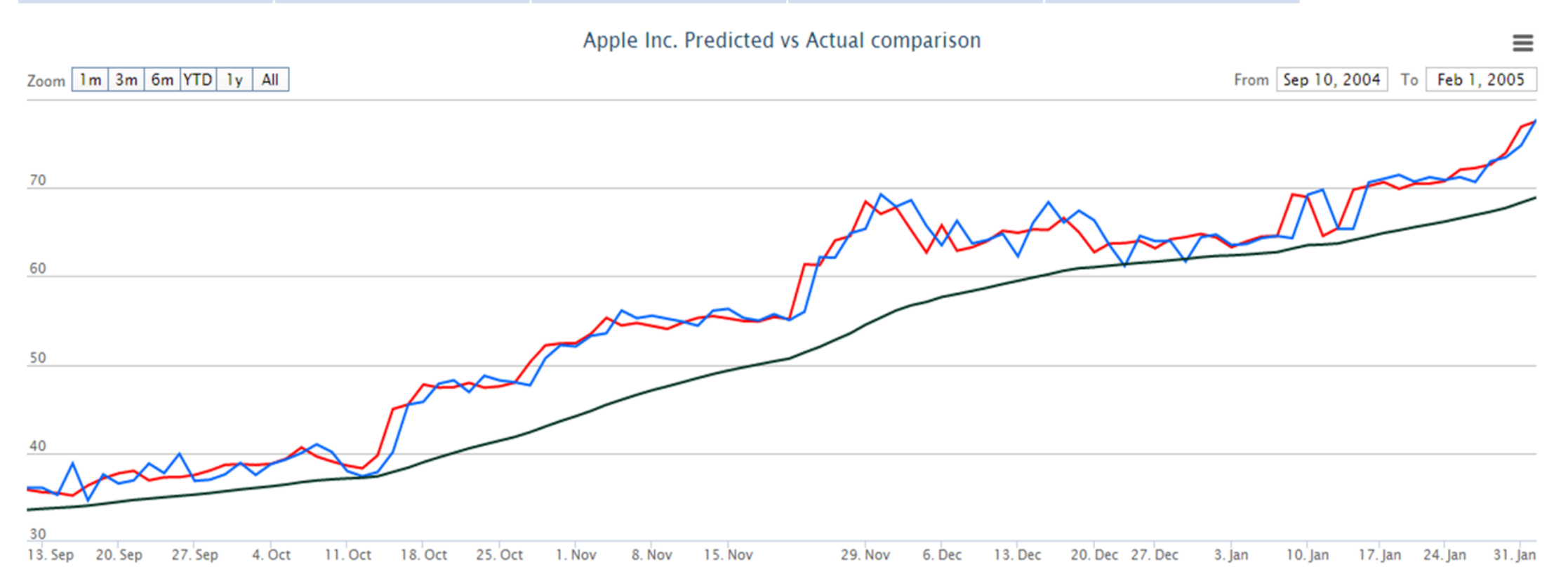
Comparison of HMM and other advanced machine learning models

- HMM Parameter

Parameter	Value	
	Ergodic	Right-Left
Number of States (N)	4	4
Number of Mixture Components (M)	5	5
Observation Dimension (D)	4	4
Delta	-	3

- MAPE of different approaches

Observation	Ergodic HMM Gaussian Mixture	Right-Left HMM Gaussian Mixture	Ergodic HMM Multivariate	Right Left HMM Multivariate
Microsoft Corp.	1.022	1.082	1.030	1.056
IBM Corp.	0.777	0.671	0.794	0.878
Apple Inc.	1.962	2.673	2.683	2.731



Actual v/s Forecasted Value of Apple. 2004-09-13 to 2004-01-21

CONCLUSIONS

- Review of theoretical aspects of HMM for discrete and continuous observations
- Usage of HMM for forecasting time series
- Experimental analysis shown in this study are very promising and proves that Ergodic HMM with Gaussian Mixture in output state is an excellent model for predicting time series future values compared to ANN, ARIMA or other HMM's with different distributions in output states or topologies
- Comparison analysis showed that HMM prediction MAPE is between 0.6 to 3, considered excellent
- Implementation of open source library in C# with advanced machine learning models and algorithms

FUTURE WORK

- Improvement of training algorithms with techniques from technical analysis for better time series behavior understanding
- Finding optimal number of states in HMM and length of training set using genetic algorithms
- Comparison of prediction abilities against more models like GV, SVR
- Improvement of prediction accuracy by replacing distribution in output states with Neural Networks.
- Improvement of the open source library developed by adding parallel execution for training and prediction algorithms
- Implementation of additional distributions for output states.

REFERENCES

[1] J. Han, M. Kamber and J. Pei, Data mining concept and techniques, Elsevier, 2011.
[2] R.J. Elliott and R.S Mamon, Hidden Markov Models in Finance, Springer, 2010.
[3] W. Zucchini and I.L. MacDonald, Hidden Markov Models for time series : an introduction using R, CRC Press, 2009.
[4] M. Mitzenmacher and E. Upfal, Probability and computing randomized algorithms and probabilistic analysis, Cambridge university press, 2009.
[5] C.M. Bishop, Pattern recognition and machine learning, Springer, 2006,
[6] L.R.Rabiner, A tutorial on Hidden Markov Model and selected applications in speech recognition, Proceeding of the IEEE, vol. 77, No 2, 1989.
[7] A.Gupta and B.Dhingra, Stock market prediction using Hidden Markov Models, Engineering and Systems (SCES), 2012.
[8] P.Arumugan and S.R.Siva Ambika, Stock market forecasting using Hidden Markov Model with clustering algorithm, International Journal of Bussiness Trends and Technology, vol. 33), 2013.
[9] G.Preethi and B.Santhi, Stock market forecasting techniques : a survey, Journal of Theoretical and Applied Information Technology, vol. 46, No. 1, 2012.
[10] R. Hassan and B. Nath, Stock market forecasting using Hidden Markov Model: A new approach, Proceedings of 5th International Conference of Intelegent System Design and Applications, 2005.
[11] Open source project , https://github.com/lev4ik/HmmDotNet

2014 ASE BIGDATA/SOCIALCOM/CYBERSECURITY Conference, Stanford University, May 27-31, 2014

Prediction of Financial Time Series Using Hidden Markov Models

Lev Brailovskiy and Maya Herman
Dept. Mathematics and Computer Science
The Open University of Israel, Raanana, Israel

Abstract

Hidden Markov Models (HMM) is a powerful machine learning model. HMM's main usage has been in solving classification and pattern recognition problems in biology, speech and voice recognition. In recent days, attempts have been made to use HMM for prediction in general and prediction of time series in particular. However, this is not straightforward. To overcome the challenges in predicting time series with HMM some hybrid approaches have been applied. This paper has two main objectives. The first, is to compare HMM with other models when used for prediction of financial time series. We will show comparison between HMM and other models and also between different types of HMM's as unique contribution of this work. In recent years, prediction of stock market behavior became a field of great interest to many scientists. Hence, as a case study, we will use our implementation and examine HMM with Multivariate Normal Distribution and Gaussian Mixture Model in hidden states. We use nearest likelihood prediction algorithm and compare results with MAPE. As training dataset, we will use some popular stock from NASDAQ and S&P500 indexes historical data. The second, is to showcase a working system with generic HMM, advanced training and prediction algorithms implementation in C#. All the code is open sourced as contribution to the community. The system allows researchers from different disciplines to use HMM and other models to solve classification, pattern recognition and prediction on various series represented by k-dimensional vectors arrays and a way to compare and re-use the models under test.

Keywords: Data Mining, Hidden Markov Models, Financial Time Series, Prediction, Gaussian Mixture Model, Multivariate Normal Distribution

1. Introduction

There are many different ways one can use to predict value or trends for given time series [1], for example known probability functions such as Gaussian distribution [2, 3, 5, 10], various moving averages [1], multivariate distributions [3 ,5, 10], mixture models [3, 5] and more. All approaches mentioned above have their limitations [3, 10]. For example, use of discrete or Gaussian single variable distributions will drastically reduce prediction accuracy. Prediction with Multivariate distributions disregards different behavior of data points in the series. With Mixture model, we can overcome most of the limitations above, but still prediction accuracy will be low because of lack of flexibility evident in Hidden Markov Models [2, 3, 5, 14, 15, 16].

Another approach is prediction using Markov chains. We should note first that stochastic process with discrete state space called chain. One of the major properties of a stochastic process is relations between different variables. In most cases, there is a relationship between variables in different periods of the process. We will differentiate between discrete and continuous Markov chains. In a discrete Markov chain state at time (t+1) depends only on state at time t and not on any other state. That is to say, all history of the process is described in its current state. In continuous Markov chain, state dimension is discrete and time dimension is continuous, state at time (t+1) depends only on the closest state to time (t+1) [4]. The most common way to describe a Markov chain is transition state matrix [4, 6]. However, this model is too limited to represent real life processes because of its lack of ability to handle data created from different distribution in the same time series.

Hidden Markov Model is an extension to Markov chain and it has encapsulated two stochastic processes. The first stochastic process is hidden and we can learn about it from observations produced by the second process [6]. Most common use of Hidden Markov Model is for classifications and pattern recognition problems. This model is extensively used in text, voice and image recognition, biology and more [3, 5, 6]. A Hidden Markov Model can represent the behavior of a time series, for example trending up or down, with a state machine encapsulated in it. With use of multivariate distribution in output states, we expand the model and can work on time series represented by a k-dimensional vector. Use of mixture model in output state will add even more flexibility to our model in describing time series behavior. Hence, integration between Hidden Markov Model, Multivariate Gaussian Distribution and Gaussian Mixture model will produce predictions that are more accurate [13, 14, 15 ,16]. HMM has the following advantages as stated in [15]

- Strong mathematical foundations
- Computational efficiency in learning and classifying/predicting
- Robust handling of over time changing data

GMM with Multivariate Gaussian Distribution as components has the ability to describe behavior of time series with predefined pattern [5]. We will utilize this by splitting different time series behaviors to be handled by different GMM components in the HMM.
This paper organized as follows: Section II describes theoretical and practical aspects behind integration and implementation of Gaussian Mixture Model and Multivariate Gaussian Distribution with Hidden Markov Model.

In section III we will provide a detailed description of the system including system architecture and its main modules. In those modules we implemented advanced machine learning models like GMM and HMM, different types of probability functions and advanced learning and prediction algorithms. In section IV a comparison analysis of prediction results produced by our system is done. Section V concludes this work with conclusion and future work opportunities.

2. Methodology

Hidden Markov Model is a special case of a finite state machine with a predefined number of states. Each state has another, hidden, state connected to it which can produce values based on some predefined probability function. This model gives a probabilistic framework for time series modeling, when k-dimensional vector represents each value.

2.1 Hidden Markov Model

Hidden Markov Model is defined as 5-tupple (S, M, A, B, π). In the rest of the paper we will use the following notation regarding HMM

N – Number of states in the model

S – $\{n_0, \dots, n_N\}$ is finite set of states

M – Finite set of output symbols.

A – Transition probability matrix, A, is N x N matrix where N is the number of states. Cell a[i, j] represents probability of transitioning from state i to state j. The sum of row i is equal to 1 and represents a probability function.

B – Observation probabilities matrix M x N. B_{mn} is probability of getting symbol m in state n.

π – Probability function for selecting starting state.

It is common to denote Hidden Markov Model as λ = (A, B, π), states and output symbols deduced from this definition. To work with Hidden Markov Model one should answer three main question [6]:

- Given observation sequence O=$\{o_0, \dots, o_T\}$ and model λ = (A, B, π), how we efficiently compute the probability of getting the observation sequence given the model P(O| λ).
- Given observation sequence O=$\{o_0, \dots, o_T\}$ and model λ = (A, B, π), how we efficiently compute Q=$\{q_0, \dots, q_T\}$, state sequence that best explains the observation sequence.
- How do we efficiently update model parameters λ = (A, B, π) to maximize P(O| λ).

To answer the first question we implemented Forward-Backward algorithm. The second question covered by Viterbi algorithm implementation. Third question covered by Baum-Welch algorithm implementation, special case of Expectation Maximization algorithm. In [6, 13] we have detailed description of this algorithms and Expectation Maximization theorem.

2.2 Continous Hidden Markov Model

In case of a time series with a k-dimensional observation vector a Hidden Markov Model with discrete distribution in output state is not enough. To overcome this we introduce Multivariate Gaussian Distribution as Gaussian Mixture Model component. The observation probability matrix will be changed to a vector of length N. B_n will hold the probability density function for state n. In the case of a Gaussian Mixture model:

$$b_j(o_t) = \sum c_{ij} N(o_t, \mu_{jl}, \sigma_{jl})$$

in which:

T – Length of time series

o_t – Observation at time t, 0<t<T

c_{ij} – Mixture coefficient for the l mixture in state j.

$N(o_t, \mu_{jl}, \sigma_{jl})$ – Multivariate Gaussian distribution for m mixture in state j.

μ_{jl} – mean vector of l mixture component in state j

σ_{jl} – co-variance matrix of l mixture component in state j

Integration of continuous distributions with Hidden Markov Model are described in [6]. The main changes are in the learning phase, Baum-Welch algorithms, which with continuous components in general and mixture model in particular need to learn mixture components and their internal components distribution functions. Despite everything written above the complexity of Baum-Welch stays unchanged.

2.3 Hidden Markov Model Types

In this work, we examined two types of Hidden Markov Models, Right-Left and Ergodic [6].

In the Ergodic model all states must be recurrent and aperiodic, we can reach each state from every other state, i.e. the transition probability matrix describes a complete graph.

In the Right-Left model we cannot move from higher order states to lower order states, the index always increases or stays the same.

$$a_{ij} = 0 \qquad i < j, j > i + \Delta$$

This type of model is useful for time series analysis and speech recognition. There is hybrid model type that will not be covered here [6].

2.4 Prediction Using Hidden Markov Model

Hidden Markov Models are heavily used to solve classification problems. Prediction is a new domain and not much work been done here.

In literature, we can find some prediction algorithms developed in recent years for HMM. A prediction algorithm based on the Viterbi algorithm is covered in [10] in more details. Clustering based prediction is used by the authors in [14]. Maximum a Posteriori (MAP) prediction approach used by authors in [15]. In this paper we will cover Likelihood based prediction algorithm [12, 17] in theory and use its implementation in our experimental results discussion.

To predict with HMM we will utilize our knowledge of the past data. The rational is if something happened in the past i.e. some pattern appeared, than it will with high probability appear in the future. Hence we will look for resembling pattern and within this patter most probable observation that will be used to calculate our prediction results.

Given trained Hidden Markov Model we calculate the likelihood value of last observation, o_t, in the sequence. From historical data we collect all observations with likelihood in predefined interval,Δ, from likelihood of ot. The base assumption is that the predicted observation will act similarly to past observation patterns. We calculate the difference between most similar likelihood observation to o_t, o_i and observation o_{i+1} that follows it. The difference o_t+(o_{i+1}-o_i) is returned as our prediction. In case that no observations are found in predefined likelihood range Δ we use adaptive approach and take observation that is closest to o_t without range limitation. Our algorithm can run on single day or on predefined days range. In case of date range we add predicted values to historical data after each iterations.

3. Implementation

The Microsoft based open source community does not include many advanced machine learning models and algorithms. One of the goals for this work was a creation of a library in C# with generic machine learning models implementation and contribution to the open source community. The source code and system documentation that describes in details the models and algorithms implemented is stored in GitHub with the following link https://github.com/lev4ik/HmmDotNet.

3.1 Use Case

This work will cover the specific use case of financial time series. Each observation in financial time series is a 4-dimentional vector:

o_t=(open, high, low, close)

To handle 4-dimentional vector we will combine HMM with GMM and Multivariate Gaussian Distribution in output states. In such a hybrid model we will get all the advantages listed above for each model. Hence, our model will give better prediction results because of the ability to better handle time series described by a combination of different distributions in different intervals.

3.2 Measurement and Units

The system is based on SOA and is shown in Figure. 1. The three main services are:

- Data extractor service – responsible for getting data from external data sources. For example Yahoo finance or Google Finance API
- Data service – responsible for storing and retrieving data in system local database, thus encapsulating the repository
- Machine learning service – responsible for running learning and prediction algorithms and presenting results to end user

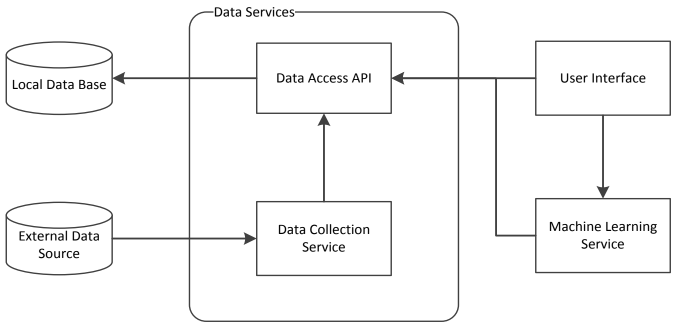


Figure. 1. System architecture.

3.3 Data Flow and Integration

Data flow and integration points between different learning and prediction algorithms is a crucial aspect in understanding how the system is built. We will describe a scenario when HMM with Gaussian Mixture in output states is used with Likelihood prediction algorithm. The scenario is shown in Figure.2.

To initialize the HMM we first run K-Means algorithm to initialize mixture components with calculated means. After this we apply the Baum-Welch algorithm on a given training set and initialized HMM. Baum-Welch is using a Forward-Backward procedure to calculate P(O| λ) for variable re-estimation. As an output we have trained HMM that passed to prediction sub system.

In the prediction sub-system we apply Likelihood based prediction algorithm on a given test set and trained HMM. Forward-Backward procedure is used to calculate likelihoods in the prediction procedure. In addition, we can run Viterbi based prediction algorithm instead of Likelihood based prediction.

Description of data flow on Figure. 2. Is given bellow:

- Solid arrows – indicate input and output of model or algorithm
- Dashed arrows – usage of algorithms by another algorithm
- Rectangle – algorithms
- Diamond – data set

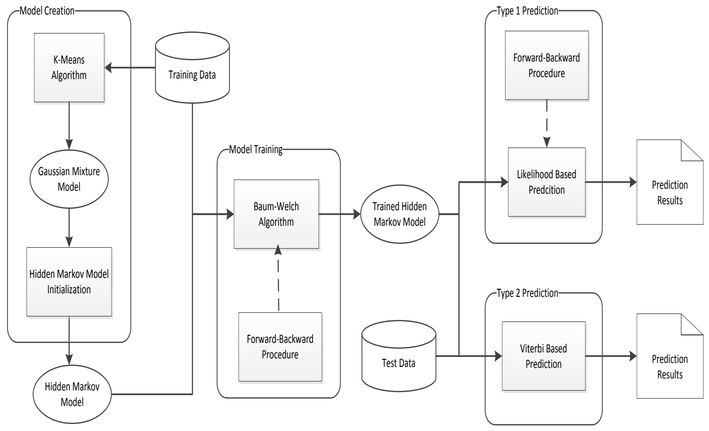
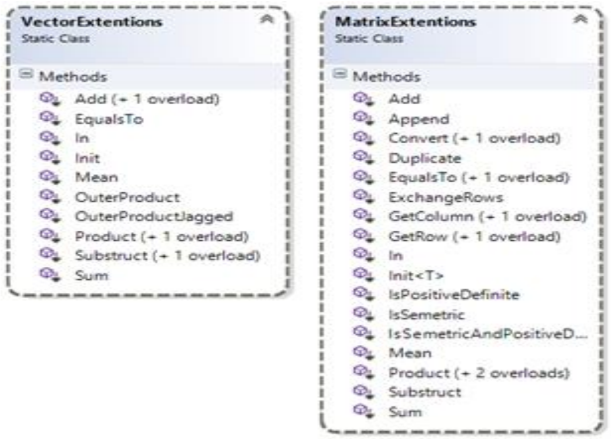


Figure. 2. Data flow and integration scenario

3.4 Mathematical Module

This module contains extensions to basic functionality of System.Math in the .NET Framework. The main additions are implemented for vector and matrix actions described in Figure. 3. For more efficient calculation of determinant, inverted matrix we implemented LU and Cholesky decompositions.



VectorExtentions
Static Class

Methods
- Add (+ 1 overload)
- EqualsTo
- In
- Init
- Mean
- OuterProduct
- OuterProductJagged
- Product (+ 1 overload)
- Substruct (+ 1 overload)
- Sum

MatrixExtentions
Static Class

Methods
- Add
- Append
- Convert (+ 1 overload)
- Duplicate
- EqualsTo (+ 1 overload)
- ExchangeRows
- GetColumn (+ 1 overload)
- GetRow (+ 1 overload)
- In
- Init<T>
- IsPositiveDefinite
- IsSemetric
- IsSemetricAndPositiveD...
- Mean
- Product (+ 2 overloads)
- Substruct
- Sum

Figure. 3. Vector and matrix actions implementation.

Also we have implemented extension for Log function to perform numerically stable statistical operation [6, 11, 13].

3.5 Statistical Module

This module contains base classes and implementation for various distribution functions. Base level interfaces shown in Figure. 4. In this work we implemented following concrete distribution functions and models:

- Univariate distribution
  - Continuous
  - Discrete
- Gaussian distribution
  - Multivariate
  - Univariate
- Mixture distribution
- Gaussian Mixture Model

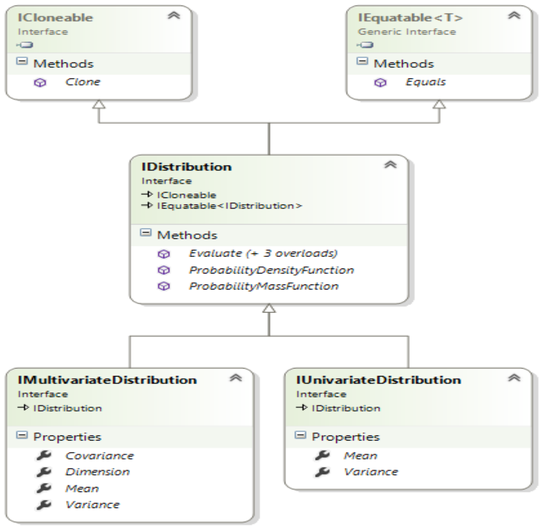


Figure. 4. Interfaces for all distribution functions implementation

More distributions can be added by implementing interfaces and base classes shown above.

3.6 Machine Learning Module

This module contains base classes of advanced machine learning models. We implemented Gaussian Mixture Model and Hidden Markov Model. To achieve learning and predicting capabilities we implemented algorithms from Table 1:

Forward-Backward procedure, Viterbi and Baum-Welch are generic implementation of the algorithms and can be applied to any kind of Hidden Markov Model. For Baum-Welch applied to HMM with GMM in output states we needed in addition to implement learning phase of GMM. Complexity data taken from [6,10,13,15,17].

Algorithm	***Complexity***
Forward-Backward Procudere	$O(TN^2)$
Viterbi	$O(TN^2)$
Baum-Welch	$O(TN^2)$
K-Means	$O(TCID)$ *I – number of iterations* *D – number of attributes*
Likelihood based prediction	$O(dTN^2)$ *d – number of attributes*
Viterbi based prediction	$O(TN^2)$

Table 1. Complexity of different algorithms

K-Means is implemented to initialize GMM or other distribution components in case we need to predict based only on time series data. In this case HMM will be initialized automatically and improved by applying Baum-Welch algorithm.

Basic prediction error is defined as difference between the actual and predicted value. For this work we have implemented 6 error measures for model accuracy:

Cumulative Forecast Error (CFE) – Sum of all prediction errors

Mean Error (ME) – Arithmetic average of all prediction errors

Mean Squared Error (MSE) – Arithmetic mean of sum of the squares of the prediction error

Root Mean Squared Error (RMSE) – Square root of mean squared error

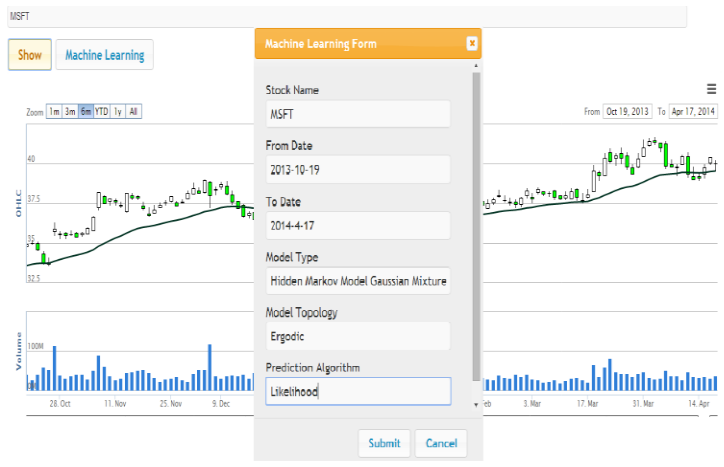
Mean Absolute Deviation (MAD) – Average of absolute value of prediction error

Mean Absolute Percent Error (MAPE) – This measure corrects the 'canceling out' effects and also keeps into account the different scales at which this measure can be computed and thus can be used to compare different predictions

All prediction algorithms calculate this error measure as their output with prediction results. The main error measure that we based our analysis on is MAPE. In general, a MAPE of 10% is considered very good, a MAPE in the range 20% - 30% or even higher is quite common.

3.7 User Interface

We implemented a web user interface using ASP.NET MVC and various open source JavaScript libraries. In the main screen the user can select stock data for analysis. After selecting the desired stock the user can choose a desired date range to use as the test set. Model type, output state distribution and prediction algorithm configuration is also available as shown in Figure. 5.



Stock Name: MSFT
From Date: 2013-10-19
To Date: 2014-4-17
Model Type: Hidden Markov Model Gaussian Mixture
Model Topology: Ergodic
Prediction Algorithm: Likelihood

Figure. 5. Stock selection for analysis with starting training and prediction parameter.

On the second screen the user is presented with the following menus:

Training Data – advanced configuration of training and test data sets

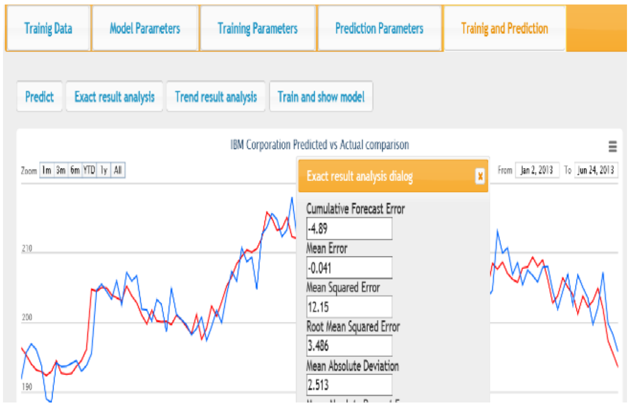
Model Parameters – advanced configuration of the model like model type, model topology, number of state and components

Training Parameters – advanced settings of training algorithm, training tolerance and number of iteration till convergence

Prediction Parameters – advanced settings of prediction algorithm, algorithm type and tolerance. Prediction range can also be changed

Training and Prediction – option to run training prediction algorithms of choice with all selected parameters. Ability to view prediction results by exact value and trend. Saving of created model in JSON format

For example the user can set what is the tolerance of the prediction algorithms or how many iterations the learning algorithms should perform. After setting the various parameters the user can run training and prediction procedures, see actual v/s predicted values and model error analysis as shown in Figure. 6.



Cumulative Forecast Error: -4.89
Mean Error: -0.041
Mean Squared Error: 12.15
Root Mean Squared Error: 3.486
Mean Absolute Deviation: 2.513

Figure. 6. Training and prediction screen with model error results

3.8 Data Retrieval and Transformation

For our use case the Yahoo finance API has been used to get historical stock quotes and store them in a local instance of Microsoft SQL Server. No transformation was applied to the data because the data is already processed and stored in an appropriate way. In general case we would apply some preprocessing logic to complete missing values and transform the data into an appropriate format. The data exposed to the other parts of the application via dedicated Data Service.

3.9 Testing

The algorithms and models in our system were tested by two types of tests, unit and integration. In our unit tests we tested each model and algorithm in isolation. Testing in isolation is good to prove that a specific unit is working correctly without its dependencies. In our integration test we used real time series to see the full pipe line at work, from learning phase to prediction analysis. We used Matlab and Wolfram Mathematica to re-create the same models and apply the same learning and prediction algorithms to them. We compared results of our learning and prediction phases with their results to insure that our models and algorithms are properly implemented. To accomplish all this we implemented more than three hundred unit and integration test.

4. **Experimental Results**

In this chapter we will present two comparison:

- Comparison between HMM and other advanced machine learning models
- Comparison between different types of HMM's

For our comparison analysis, we selected three big companies trading in NASDAQ index, Microsoft, IBM and Apple. As the training set for Microsoft and Apple a period from 10 February 2004 was selected. For IBM a period from 10 June 2004 till 10 September of the same year. As the test set a period of 130 days between 13 September, the end of our training set data, until 21 January 2005 was selected.

In the first comparison analysis we examined ANN, ARIMA against Ergodic HMM with Gaussian Mixture in output state. For the second comparison analysis we examined Ergodic HMM with Gaussian Mixture (EHMM-GM) and with Multivariate Gaussian Distribution (EHMM-M) and compared them with Right-Left HMM with Gaussian Mixture (RLHMM-GM) and Multivariate Gaussian Distribution (RLHMM-M).

In both cases we trained the models and compared their prediction abilities using MAPE error measure. Acceptable range of MAPE values is between 10 to 20. Values between 0 and 10 as we got in our case defined as excellent for a given predictor.

4.1 HMM Comparison with other Models

We used HMM with Gaussian Mixture Model in output states with following parameters [14, 15, 16, 17] :

- Number of states N=4
- Number of mixture components for each state M=5
- Observation dimension D=4
- Ergodic HMM

In related works Right-Left HMM has been used due to. the claim that it's structure better describes the stock trading process. To initialize HMM mixture components, when provided only with time series data, we used the k-means algorithm. Each cluster created by k-means is translated to a separate mixture component with weights that the cluster received as weights of this component. Starting distribution π and transition probability matrix are uniform across all states.

Observation	***From date***	***To Date***
Microsoft Corp.	10 February 2004	10 September 2004
IBM Corp.	10 June 2004	10 September 2004
Apple Inc.	10 February 2004	10 September 2004

Table 2. Training Dataset

Observation	***From date***	***To Date***
Microsoft Corp.	13 September 2004	21 January 2005
IBM Corp.	13 September 2004	21 January 2005
Apple Inc.	13 September 2004	21 January 2005

Table 3. Test Dataset

Observation	***HMM***	***ARIMA***	***ANN***
Microsoft Corp.	1.022	1.324	1.326
IBM Corp.	0.777	0.972	0.972
Apple Inc.	1.962	1.801	1.801

Table 4. MAPE for different approaches

In Table 4 we present MAPE values for three stocks predicted using our trained model and prediction algorithm. As can be seen HMM performed better for IBM Corp. stock and with little difference in favor of ANN and ARIMA with Apple Inc. stock. Figure. 7. shows actual and forecasted IBM Corp. stock values.

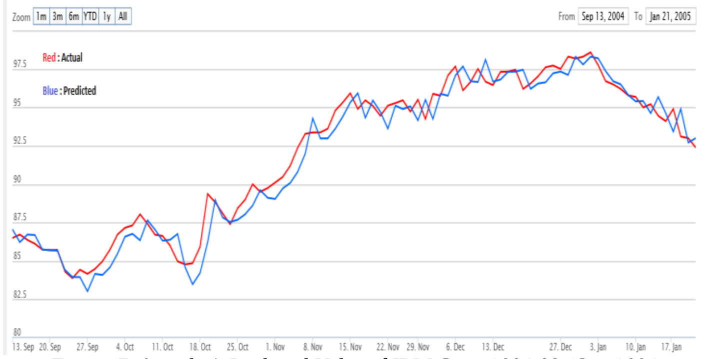


Figure. 7. Actual v/s Predicted Value of IBM Corp. 2004-09-13 to 2004-01-21.

4.2 Identify the Headings

Same setup and initialization used for Ergodic HMM from previous section. For Right-Left HMM we set Δ =3. Table 5 present comparison between HMMs with different distribution in output states. As can be seen HMMs with Gaussian Mixture produce better results than their counterparts with Multivariate Gaussian Distribution.

Observation	***Ergodic HMM-GM***	***Right-Left HMM-GM***	***Ergodic HMM-M***	***Right-Left HMM-M***
Microsoft Corp.	1.022	1.082	1.030	1.056
IBM Corp.	0.777	0.671	0.794	0.878
Apple Inc.	1.962	2.673	2.683	2.731

Table 5. MAPE for different types of HMM

5. Conclusions

This paper covered theoretical aspects of HMM for discrete and continuous observations. We explained how HMM can be used not only for classification but also for prediction. In our comparison analysis showed that HMM prediction MAPE is between 0.6 to 3. Experimental analysis shown here is very promising and proves that Ergodic HMM with Gaussian Mixture in output state is an excellent model for predicting time series future values compared to ANN, ARIMA or other HMM's with different types.

This work is a first pillar for future research. We can improve our training algorithms with techniques from technical analysis for better time series behavior understanding. Finding of optimal number of states in HMM and length of training set using genetic algorithms. Improvement of prediction accuracy by replacing distribution in output states with Neural Networks. More comprehensive analysis that includes more times series and advanced models such as GV or SVR.

The open source library developed will be improved by adding parallel execution for training and prediction algorithms. Implementation of additional distributions for output states. Improved user interface to give users ability to run more than one model and analyze multiple run results or run selected model on more than one time series.

References

[1] J. Han, M. Kamber and J. Pei, Data mining concept and techniques, Elsevier, 2011.
[2] R.J. Elliott and R.S Mamon, Hidden Markov Models in Finance, Springer, 2010.
[3] W. Zucchini and I.L. MacDonald, Hidden Markov Models for time series : an introdution using R, CRC Press, 2009.
[4] M. Mitzenmacher and E. Upfal, Probability and computing randomized algorithms and probabilistic analysis, Cambridge university press, 2009.
[5] C.M. Bishop, Pattern recognition and machine learning, Springer, 2006,
[6] L.R.Rabiner, A tutorial on Hidden Markov Model and selected applications in speech recognition, Proceeding of the IEEE, vol. 77, No 2, 1989.
[7] C. Lavergne and M.Saidane, Optimal prediction with conditionally heteroskedastic factor analysed Hidden Markov Models, Springer Science and Business Media, pp.323-364, 2009.
[8] S.Park,J.Lee,J.Song and T.Park, Forecasting change directions for financial time series using Hidden Markov Models, Springer-Verlag Berlin Heidelberg, pp.184-191, 2009.
[9] C. Lavergne and M.Saidane, On factorial HMMs for time series in finance, The Kyoto Economic Review, Vol 75, No. 1, pp.63-90, 2006.
[10] Y,Zhang, Prediction of financial time series with Hidden Markov Models, Simon Fraser University, 2004, unpublished.
[11] T.P.Mann, Numerically Stable Hidden Markov Model implementation, February 2006, unpublished.
[12] B.Nobakht,C.E.Joseph and B.Loni, Stock market analysis and prediction using Hidden Markov Models, unpublished.
[13] L.R.Rabiner and B.H.Juang, Fundamentals of speech recognition, Prentice Hall Signal Processing Series, 1993.
[14] A.Gupta and B.Dhingra, Stock market prediction using Hidden Markov Models, Engineering and Systems (SCES), 2012.
[15] P.Arumugan and S.R.Siva Ambika, Stock market forecasting using Hidden Markov Model with clustering algorithm, International Journal of Bussiness Trends and Technology, vol. 33), 2013.
[16] G.Preethi and B.Santhi, Stock market forecasting techniques : a survey, Journal of Theoretical and Applied Information Technology, vol. 46, No. 1, 2012.
[17] R. Hassan and B. Nath, Stock market forecasting using Hidden Markov Model: A new approach, Proceedings of 5th International Conference of Intelegent System Design and Applications, 2005.
[18] Open source project , https://github.com/lev4ik/HmmDotNet