

Using Trend Lines for Feature Extraction and Machine Learning for Prediction in Financial Time Series

Robert Bodley
The University of Cape Town
bdlrob001@myuct.ac.za

Prof. Deshen Moodley
The University of Cape Town
deshen@cs.uct.ac.za

ABSTRACT

This paper focuses on the use of trend lines as a feature extraction technique on financial time series data. Trend lines are extracted using segmentation algorithms, several algorithms will be evaluated and compared. Machine learning models will be trained using trend lines and compared to a baseline, to assess the effectiveness of using trend lines for prediction. It was found that using trend lines as a feature extraction technique and machine learning using a Long Shorter Term Memory or Multilayer Perceptron Neural Network for prediction does not compare to using point data as a feature and the same machine learning models for prediction.

CCS CONCEPTS

• **Information systems** → Data mining; • **Computing methodologies** → Feature selection;

KEYWORDS

Machine Learning, Time Series Analysis, Trend lines, Feature Extraction

1 INTRODUCTION

Time series data is generated and used in a very wide variety of applications, thus using past data to predict the future outcomes is extremely useful and can be utilised in a wide range of applications such as medical and biological experimental observations, daily fluctuation of stock markets, power consumption, monitoring of data center performance [12].

Predicting the stock market has been a topic researched by many and many different techniques have been tried, from technical analysis to machine learning, all with the common goal of making a monetary profit.

In recent years, computers have been leveraged to predict the stock market as they can process a huge amounts of data in a very short time period. Financial markets are represented by time series data which is a sequence of individual point data over a period of time [5, 13].

In this paper, the primary focus is on the data preparation, using trend lines for feature extraction while using machine learning models to make predictions. The paper will compare different segmentation algorithms to extract trend lines. The trend lines will then be used as features to train different machine learning models. A machine learning pipeline will be implemented as it is common method of implementing the process of using machine learning; from data collection to training and finally prediction as can be seen in the figure 1. The results obtained from using trend lines as a feature extraction technique and machine learning for prediction will then be compared to the baseline which is using point data as features and the same machine learning models for prediction.

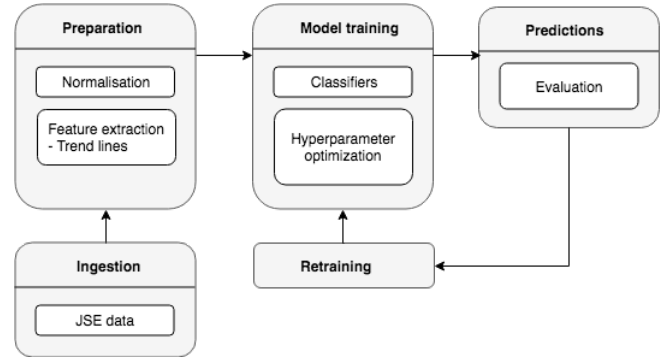


Figure 1: Machine learning pipeline

2 BACKGROUND AND LITERATURE REVIEW

2.1 Financial Market and the Johannesburg Stock Exchange:

A financial market is a market where people trade securities, commodities and pay a small transaction fee. The price of any single stock on an exchange is based off of the supply and demand currently on that market. The Johannesburg Stock Exchange, more commonly referred to as the JSE is a world class stock exchange based in Sandton, Johannesburg. The exchange is the oldest and largest in Africa with over 450 companies listed and a monthly trade volume of 6 Billion Dollars. This paper will focus on predicting certain stock prices on the JSE. The exact stocks can be seen in section 3.2.

2.2 History of Financial Market Prediction

Traditionally individuals would learn the principles of visual analysis [10], also known as technical analysis which is defined as the tracking and prediction of asset price movements using charts and graphs in combination [3].

Another approach to stock market prediction entails the use of statistical models such as linear and multiple regression. These models are used but are not very good at modeling the complexities within the stock market.

Machine learning, which this paper will focus on, has been one of the most popular techniques for time series analysis in recent years, especially on financial markets as obvious financial gains can be made if predictions can be made correctly. Machine learning is able to model very complex data with many features very quickly and able to make future predictions.

2.3 Time Series Data

A time series is a set of n ordered observations, $T = \{t_0, \dots, t_{n-1}\}$. The data points of the series are usually equally space apart and arranged in temporal order. Financial data is represented through a time series, this paper will use the daily closing price of a stocks time series.

Recently there has been an interest in mining time series data. This entails reducing the number of data points while still keeping in tact the data's meaning. Many different techniques have been used including, Shapelets, Symbolic Mappings and Piecewise Linear Regression (PLR). This paper will focus on using PLR to extract trend lines from financial time series data using segmentation algorithms.

2.4 What are Trend Lines?

Eamonn et al. [8] introduced trend lines as a smooth additive that can represent the local change of a time series. Trends are represented by a slope and a duration [12] and can be seen as a good data mining technique, more details discussed in sections 2.5.

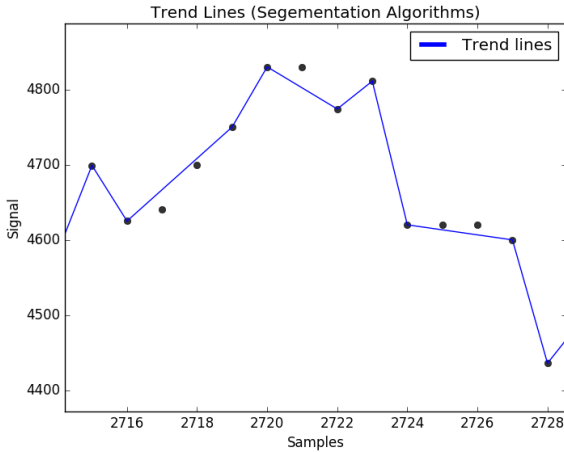


Figure 2: Trend line extraction

It can clearly be seen in figure 2 that the trend lines extracted smooth the time series while maintaining the overall shape, advantages of this will be discussed in section 2.5.

This paper defines a trend line by $t = \{d, s\}$ where d is the duration and s the slope of the trend line. All the trends present in a time series can be represented as $T = \{t_0, \dots, t_{n-1}\}$. We define K as the number of segments that are used to define the time series. L will be defined as the average length of the segments defined, $L = N/K$.

2.5 Why Use Trend lines?

Piecewise regression is similar to linear regression and is used when a single lines does not fit the data[1]. Piecewise regression groups data points in a time series into separate segments that are each represented by a single straight line, each line represents a single trend lines. These lines together still represent the data but

with added benefits discussed in below. Piecewise regression was originally done manually when it could be seen that a single line does not fit the time series. This paper would like to perform piecewise regression on extremely large amounts of data and do it very quickly, thus we need to define an algorithm that can perform the task. We will refer to these algorithms as segmentation algorithms which solve the segmentation problem.

The segmentation problem can be defined as:

- Given a time series T , as defined above, approximate the best representation such that each segments error is less than some specified threshold [8]. The greater this threshold the less segments will be present and the smaller the threshold the more segments will be present.

Trend lines have been useful in many different cases [8]:

- Extracting trend lines reduces the data size as many points can be represented by a single slope and duration pair. This allows the data to be processed a lot faster and stored using a lot less space while still maintaining the meaning in the data.
- A time series represented by mined trend lines are seen as a sequence of historical trends carrying the long-term contextual information which in essence effects the overall direction of the time series. This paper will leverage this using a special type of neural networks discussed in section 3.4.1.
- Trend lines also are used over raw data as raw data contains a lot of noise. Extracting the trend lines from the raw data reduces the amount of noise present.
- Similarity searches between data sets can be done very quickly.
- Used for clustering and classifying algorithms
- Used to detect change in data.

2.6 Machine Learning and Time Series Data

Machine learning has been used for prediction in many different areas. To name a few it has been used to predict corporate failure[9] through analysing a company's data, predicting the stock count required at a store for a certain item at a certain time by analysing past data and most importantly, for the purpose of this paper, analysing and predicting the stock market [5].

Neural Networks(NN) have a huge advantage over other prediction techniques as they can have non-linear learning properties when modeling and predicting. Due to the fact that stock markets possess a lot of data and the properties stated above, neural networks can be applied to stock market prediction.

Several different machine learning models have been used to predict the stock market and time series data in general.

2.6.1 Support Vector Machine: One approach is to use a Support Vector Machine (SVM) which is based on statistical theory and which has had very successful results in the prediction of financial markets. They use the theory of structural risk minimization to predict a function by minimizing the general error [13]. Support Vector Machines are known to be very resistant to over training unlike neural networks. The Paper by Paul D. et al. [13] describes how SVM outperform back propagation NN for predicting the stock market.

2.6.2 Sentiment Analysis: Another approach to predicting the stock market using NNs is using event information, traditionally gathered manually from news paper articles but in the recent years by trawling the web. Information can be gathered on a single topic or stock and sentiment analysis can be used to determine what is being said at a certain time. This is then combined with the stock price and a model is trained to recognise what will happen to the stock price when certain things are being said about that stock or market.

2.6.3 Recurrent Neural Network: Recurrent neural networks (RNN) are very powerful for time series analysis and particularly Long Short Term Memory (LSTM) networks. RNN works well on sequence data with long-term dependencies, due to the internal memory mechanism. This paper will make use of an LSTM, this is discussed further in section 3.4.1

2.6.4 Convolutional Neural Networks: Convolutional neural networks excel in extracting effective representation of local salience from the raw data of time series, by enforcing a local connectivity between neurons. In the paper [12], a hybrid model is discussed which combines the advantages of both an RNN's and a CNN.

2.6.5 Pattern-based Hidden Markov Model: A method mentioned in the paper by Tao L. et al. [12] which takes the trend lines produced using a segmentation algorithm and models the dependency in segments with a Hidden Markov Model. The model is used to predict which state the current time series is in and then uses this to estimate the next trend based on the current state.

3 DESIGN AND IMPLEMENTATION

3.1 Overview of Design and Implementation

The analysis of using trend lines for feature extraction and machine learning for prediction will be done in 2 main parts:

- (1) Trend line extraction using 3 segmentation algorithms.
- (2) Prediction with machine learning using trend lines as features.

These two parts can be seen in the machine learning pipeline shown above in figure 1 and in more detail in figure 3.

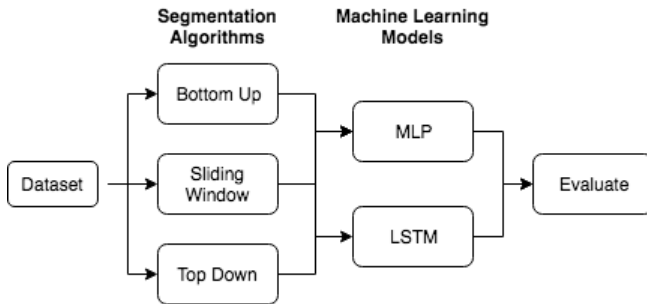


Figure 3: Feature extraction and machine learning for prediction

3.2 Dataset Choice

Three datasets were chosen, tested and evaluated.

- **Dataset 1:** Naspers was selected as a dataset as its daily closing price on the JSE is very volatile with a variance of 83783 and a mean of 52326 over a period of 9.5 years.
- **Dataset 2:** Redefine was selected as a dataset as its daily closing price on the JSE is very stable with a variance of 324 and a mean of 669 over 9.5 years.
- **Dataset 3:** TFG was selected as a dataset as its daily closing price on the JSE is relatively stable with a variance of 5228 and a mean of 5991 over 9.5 years.

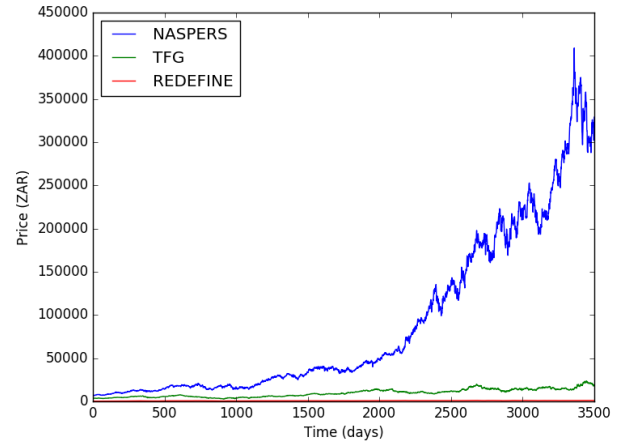


Figure 4: Datasets

3.3 Trend Line Extraction

Trend lines are extracted using segmentation algorithms. This paper will focus on using 3 algorithms. These algorithms being:

3.3.1 Algorithm 1 - Bottom Up Segmentation: A greedy approach can be used to minimise the search space. The greedy approach has two main goals, minimising the error between the segments and the actual data points, and using the fewest number of segments possible while still modeling the data well enough. The algorithm is a batch process and thus has to have all of the data up front, this means it is an offline algorithm and cannot take new data into account. The solution proposed is as follows.

- (1) Perform a piecewise regression with $n/2$ segments where n is the number of individual values that represent the time series. Each regression line is fit using least squares regression. Initially very little error will be detected.
- (2) Evaluate the increase in total error that would result if two neighbouring trend lines were merged into a single line. If the error increase is below $x\%$ more than the increase of any previous merge then the segments should be merged, or else they should stay separated. Usually, x is about 3% but can be altered, an increase would reduce the number of trend lines and a decrease would do the opposite.

3.3.2 Algorithm 2 - Sliding Window Segmentation: The sliding window technique is most probably the easiest to implement and understand. This approach is online and this is adopted by many who need to produce segments from live data. It is defined as follows:

- (1) The algorithm starts by anchoring the left most data point. It then extends the line trying to keep the approximation as close to the data as possible until some error threshold is broken, this defined as the end point.
- (2) A new segment is started from the endpoint. This continues until the whole time series has been segmented.

3.3.3 Algorithm 3 - Top Down Segmentation: The top down segmentation algorithm works by recursively considering every possible partitioning of the time series and splitting it at the best position. This algorithm is also a batch process and thus cannot perform work with incoming streams of data. The algorithm:

- (1) Splits the time series in half then algorithmically defines a linear regression segmentation line for each half.
- (2) Then checks the error in each segment and if the error is above a user specified threshold it splits the segment again until all segments are below a certain specified threshold.

3.4 Machine Learning For Prediction On Time Series Data

3.4.1 Machine Learning Model Design And Implementation: Two different machine learning models will be used. Both models will take several trend lines as input and attempt to predict the next trend line.

- (1) A Multilayer Perceptron Neural Network: A Multilayer Perceptron (MLP) Neural Network is one of the most basic neural networks.
- (2) A type of Recurrent Neural Network called a Long Short Term Memory Neural Network: LSTM's are good at prediction using sequence data with long-term dependencies, due to their internal memory mechanism.

Both types of networks will be trained using the daily closing price of point data and extracted trend lines using different look backs and training time periods. The results from these models will then be evaluated according to evaluation measures discussed in section 4.4.

The LSTM and MLP will have a similar structure, discussed further in section 3.4.2. The basic model input and output will be similar to figure 5 below.

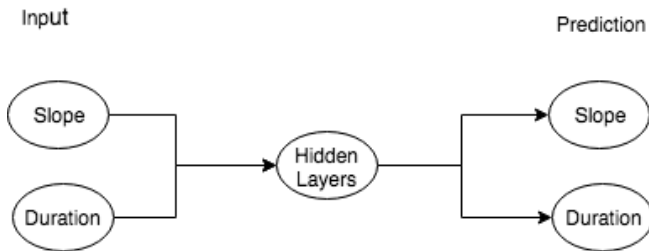


Figure 5: Basic model input and output

3.4.2 Hyper-parameter Choices: For both types of neural network used, for consistency, similar network structures, optimizers and model evaluation techniques will be used.

- **Structure:** Both neural networks will be trained using two internal layers. The two layers performed better than one empirically. In the case of the LSTM, two layers help store more long term dependency within the data and thus aid making better predictions.
- **Dropout:** Dropout is a technique for reducing over-fitting in a neural networks by adapting training data. A dropout of 15% was applied to both layers of each neural network.
- **Optimiser:** The optimiser is used, in our case, to minimize the models evaluation measure by adapting the weights in the neural network. The Adam optimiser was chosen as it converges faster than Gradient Descent and Stochastic Gradient Descent optimisers. This was seen through empirical testing.
- **Epochs:** The amount of epochs refer to the amount of times the whole time series is trained on. When training an LSTM model using trend line extracted on dataset 3 over a period of 1500 epochs and evaluating the loss and validation loss at each epoch, as seen in figure 6, we can see at about 400 epochs the validation loss and loss start to diverge. This is a clear indication that the model starts to over fit the dataset.

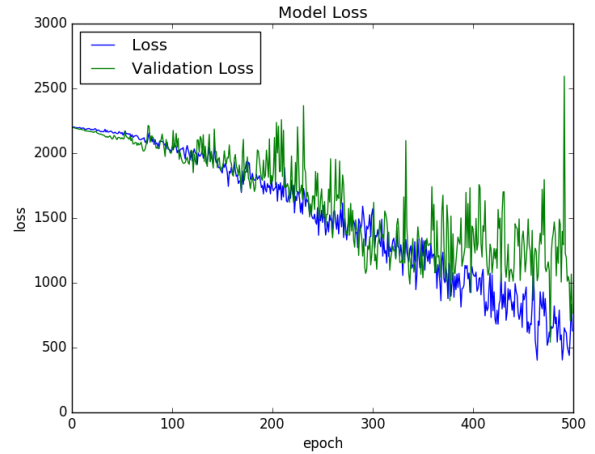


Figure 6: Model Loss (first 500 epochs)

- **Test Train Split:** The time series data is split between training and test, 70% of the data will be used to train the ML models and the remaining 30% used for testing.
- **MSE used for trend extraction:** When using different MSE's to extract trend lines from the time series data, different trend lines are extracted. Increasing the MSE, decreases the amount of trend lines extracted and smooths the data more. To find the appropriate MSE to use we evaluate the LSTM and MLP models at different MSE's and training for 400 epochs. In table 1 it can be seen that both the MLP and LSTM performed best, with the lowest RMSE with trends extracted using an MSE of 1000.

Table 1: Model training with trends extracted using different MSE

MSE Evaluation		
MSE	Model	Model Test RMSE
100	LSTM	117.32
	MLP	360.92
250	LSTM	118.30
	MLP	94.89
500	LSTM	116.08
	MLP	199.37
1000	LSTM	85.81
	MLP	100.48
2500	LSTM	110.18
	MLP	183.46
1000	LSTM	170.15
	MLP	173.46

- **Trend Look-back:** Look-back refers to the number of previous samples used to make a single prediction. A look-back of 10 when using trend lines for training means 10 previous trends will be used to predict a single trend line. When analysing table 2, it is clear that when the look-back increases above 5, the MLP's RMSE starts to increase drastically. The LSTM's decreases until a look-back of 25 and then slowly starts to increase. To train the MLP a look-back of 1 will be used and to train the LSTM a look-back of 25 will be used.

Table 2: Model training with trends extracted using different MSE

Look-back Evaluation		
Look-back	Model	Model Test RMSE
1	LSTM	85.09
	MLP	72.40
5	LSTM	89.38
	MLP	84.02
10	LSTM	105.98
	MLP	110.10
25	LSTM	74.43
	MLP	113.96
35	LSTM	82.27
	MLP	211.76
50	LSTM	90.31
	MLP	1066.66
75	LSTM	82.02
	MLP	1481.79
100	LSTM	85.03
	MLP	1554.63
150	LSTM	87.66
	MLP	2099.12

4 RESULTS AND EVALUATION

4.1 Segmentation Algorithms Evaluation Measures

The evaluation measures used to compare the different models are:

- **Time Complexity:** The time complexity of an algorithm is used to quantify the time taken by the algorithm to come to a result.
- **Model Accuracy:** Using the same model and the model evaluation techniques mentioned below, in section 4.4, to compare the results generated from the 3 segmentation algorithms.
- **Online vs Offline Algorithms:** Online segmentation algorithms can process incoming streams of data while offline algorithms need all the data points upfront. For real time segmentation, online algorithms are required.
- **Amount of Trend lines Extracted:** Fewer trend line generated with the same error used

Table 3: Segmentation algorithm comparison

Algorithm	Online
Bottom Up	No
Sliding Window	Yes
Top Down	No

Both the Bottom Up and Top Down algorithms are offline segmentation algorithms. This is a negative as they require batch processing (all the data required upfront) and any new data that comes in will require the entire algorithm to be run on the whole dataset which could return completely different trend line. In relation to machine learning, if the trend lines generated are different than the ones trained on before, the ML model will have to be retrained. This is not viable for use on the live stock market as training a model takes time and often manual refinement. On the other hand the Sliding Window algorithm is online, thus is able to take in new data and process it adequately, making it more desirable for predicting on an ever changing stock market.

4.2 Segmentation Algorithm Results

The segmentation algorithms will be compared using both dataset 1, 2 and 3 at different MSE values. A larger MSE allows for more error in each trend line, therefore less trend lines should be extracted and each trend line should have a shorter length.

Table 4: Segmentation algorithms using dataset 1.

Segmentation Algorithms Analysis Analysis (Dataset 1)			
Algorithm	MSE	Runtime (seconds)	Trend Count
Bottom Up	250	0.14	2610
	500	0.20	2574
	1000	0.16	2549
	2500	0.23	2466
	10000	0.23	2292
Sliding Window	250	0.31	2610
	500	0.28	2575
	1000	0.29	2552
	2500	0.27	2473
	10000	0.26	2318
Top Down	250	4.51	2631
	500	4.66	2597
	1000	4.63	2581
	2500	4.83	2526
	10000	4.54	2394

On dataset 1, the volatile dataset, when comparing run-time the Top Down algorithm is slow compared to the Sliding Window and Bottom Up algorithms. The run-time of the Sliding window is marginally slower than the bottom up segmentation method but as the MSE increases this difference becomes less and less. The Bottom Up algorithm gets slower as the MSE increases while the Sliding Window run-time decreases. When comparing the amount of trends extracted, it can clearly be seen that when increasing the MSE from 250 to 10000, the amount of trend lines extracted does not decrease a lot. This is because increasing the length of a segment increases the MSE of that segment by a lot as the time series data is very volatile.

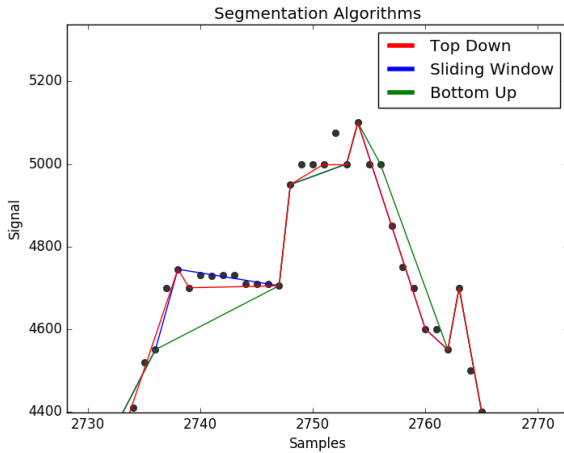


Figure 7: Trends Extracted

When inspecting the trends extracted, as seen in figure 7 we can clearly see that the Top Down algorithm and Sliding Window both extract very similar trends. The Bottom Up algorithm does

not follow the point data and smooths the data too much and thus loses valuable information about the signal. The Sliding Window algorithm follows the data the best while creating the longest trend lines thus reducing the data set to few trend lines that represent the data the best.

Table 5: Segmentation algorithms using dataset 2.

Segmentation Algorithms Analysis Analysis			
Algorithm	MSE	Runtime (seconds)	Trend Count
Bottom Up	250	0.20	295
	500	0.20	214
	1000	0.21	150
	2500	0.22	87
	10000	0.23	39
Sliding Window	250	0.09	346
	500	0.08	247
	1000	0.07	171
	2500	0.08	94
	10000	0.08	43
Top Down	250	1.55	451
	500	1.44	319
	1000	1.68	222
	2500	1.34	132
	10000	1.31	55

When comparing the trend lines extracted on the stable dataset, it is important to notice the vast difference in the amount of trend lines extracted. With an MSE of 250, 295 trends were extracted compared to the 2610 trends extracted on the volatile dataset. It can also be seen that as the MSE increases the amount of trends decreases drastically compared to the volatile dataset. The run-time of the Sliding Window algorithm is twice as fast as the Bottom Up algorithm on a stable dataset and extracts similar trends to the Bottom Up Algorithm. The Top Down algorithms run-time is slow and as the volatility of the dataset decreases the run-time decreases.

Table 6: Segmentation algorithms using dataset 3.

Segmentation Algorithms Analysis Analysis			
Algorithm	MSE	Runtime (seconds)	Trend Count
Bottom Up	250	0.24	2423
	500	0.28	2275
	1000	0.33	2079
	2500	0.39	1761
	10000	0.52	1220
Sliding Window	250	0.34	2436
	500	0.27	2301
	1000	0.27	2126
	2500	0.24	1830
	10000	0.21	1327
Top Down	250	5.53	2545
	500	5.35	2443
	1000	5.42	2316
	2500	5.26	2065
	10000	5.16	1602

On the relatively stable dataset the run-time and trends extracted of the Bottom Up and Sliding Window are similar but the Top Down algorithm is very slow.

When looking at all of the results gathered on segmentation algorithms a lot of insightful information can be seen. As the dataset gets more stable the run-time of the Sliding Window algorithms run-time decreases while the Bottom Up algorithms run-time increases. It can also be seen that as the MSE set increases the Sliding Window Algorithm run time decreases while the Bottom Up algorithms run-time increases. The Top Down algorithm does not seem to compare to either of the other 2 algorithms as its run-time is extremely slow when compared to them. The trends extracted also do not fit the data very well, as seen in figure 7.

Further comparisons will be made in section 4.4 when comparing the results of different models trains with the different trend lines extracted.

4.3 Machine Learning Model Evaluation

Figure 3 depicts how each of the machine learning models will be compared when being trained using trend lines. Trend lines will be extracted from the three datasets using all 3 segmentation algorithms with the best hyper-parameters discussed in section 4.4 to train the models. These results will then be compared using the measures mentioned in section 4.4.

The models will be trained using the hyper-parameters evaluated and chosen in section 3.4.2 and an overview can be seen in table 7.

Table 7: Hyper-parameter choice overview

Hyper-Parameters	
Structure	2 Internal Layers
Dropout	0.15 on each layer
Optimiser	adam
Epochs	400
MSE (trends)	1000
Lookback	1(MLP) 25(LSTM)

4.4 Machine Learning Model Evaluation Measures

The evaluation measures used to compare the different models are:

- **Root Mean Squared Error:** The models are trained to optimise the RMSE and thus this will be a good comparison between models. To compare predicted trend lines to the predicted point data the trend lines are extrapolated and turned into point data, this helps make the comparison more fare.
- **Training Speed:** Models will be assessed on the rate at which they are trained to reach their optimal performance.

4.5 Machine Learning Model Evaluation Using Trend Lines

Different machine learning models are trained using the best hyper parameters seen in table 7. Comparisons will are then made between the different segmentation algorithms and the MLP vs LSTM.

Table 8: Segmentation Algorithms for Feature extraction and Machine Learning for prediction on Naspers Dataset

Trend Line Analysis				
Dataset	Algorithm	Model	Test RMSE	Training
Dataset 1 Naspers	Bottom Up	MLP	373.33	32s
		LSTM	376.45	17m 14s
	Top Down	MLP	320.61	34s
		LSTM	378.28	16m 20s
	Sliding Window	MLP	316.44	31s
		LSTM	355.71	17m 35s

The Naspers dataset is extremely volatile, this means that a lot of trend lines are extracted at an MSE of 1000. The more trend lines that are extracted the more data there is to train the machine learning model, this does not always mean a better model will be trained but means the model will take longer to train.

When comparing the RMSE's produced by the ML models trained using the trend lines extracted by the different segmentation algorithms the Sliding Window algorithm does marginally better than the Top Down a lot better than the Bottom Up extraction algorithm.

It is clear that the MLP produces a lower test RMSE in all cases on the Naspers dataset compared to the LSTM mode. It also trains a lot faster than the LSTM.

Table 9: Segmentation Algorithms for Feature extraction and Machine Learning for prediction on Redefine Dataset

Trend Line Analysis				
Dataset	Algorithm	Model	Test RMSE	Training
Dataset 2 Redefine	Bottom Up	MLP	39.57	3s
		LSTM	30.42	7m 25s
	Top Down	MLP	19.21	3.72s
		LSTM	21.30	7m 41s
	Sliding Window	MLP	18.72	3.27s
		LSTM	21.13	7m 47s

The Redefine dataset is very stable, refer to figure 4, this means very few trends will be extracted, refer to section 4.2. Less trend lines means the models can be trained faster and the trend lines will also be more stable. The RMSE's are vastly better than the models trained on the Naspers dataset.

The Sliding window algorithm also gets the lowest RMSE and very similar training time to the other models.

The MLP produces better result and trains a lot faster in all cases compared to the LSTM.

Table 10: Segmentation Algorithms for Feature extraction and Machine Learning for prediction on TFG Dataset

Trend Line Analysis				
Dataset	Algorithm	Model	Test RMSE	Training
Dataset 3 TFG	Bottom Up	MLP	78.67	26s
		LSTM	81.03	11m 31s
	Top Down	MLP	71.80	20s
		LSTM	80.05	11m 59s
	Sliding Window	MLP	70.46	25s
		LSTM	74.72	11m 55s

The TFG dataset it in between volatile and stable. The amount of trend lines extracted can be seen in the analysis in section 4.2 and more information on the dataset in figure 4.

The Sliding Window extraction algorithm produces the best results when compared to the feature extraction techniques.

The MLP outperforms the LSTM in both train time and test RMSE.

In conclusion, as the dataset gets more volatile the MLP and LSTM models take longer to train and produce larger RMSE's. In terms of which segmentation algorithm performs the best when used to train an LSTM or MLP, it is clear that the Sliding Window approach is superior but when comparing the trends predicted vs the actual trend lines, as seen in figure 8, it is clear that machine learning using either an MLP or LSTM is not able to make useful predictions.

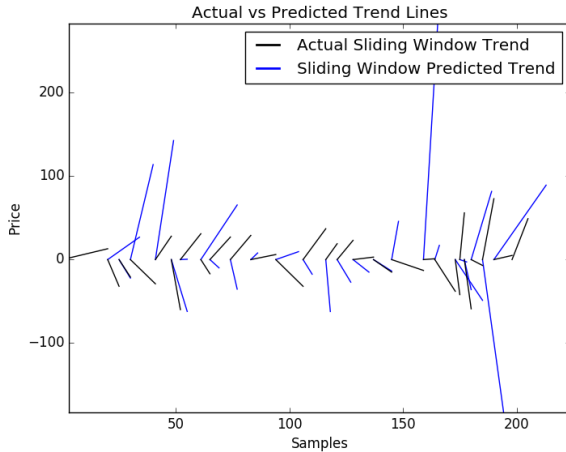


Figure 8: Predicted Trends vs Actual Trends

4.6 Machine Learning Model Evaluation Using Point Data

As a baselines, models trained on point data will be used. Through empirical testing the best hyper-parameters are chosen and an MLP and LSTM are trained on all 3 datasets.

Table 11: Point Data Machine Learning Analysis

Point Data Analysis				
Dataset	Best Lookback	Model	Test RMSE	Training
Dataset 1 Naspers	10	MLP	166.24	47s
Dataset 2 Redefine	10	LSTM	36.33	3m 14s
Dataset 3 TFG	25	MLP	9.65	45s
	25	LSTM	6.87	3m 45s
	25	MLP	90.34	51s
	25	LSTM	30.57	3m 35s

It can clearly be seen that when using point data to train machine learning models on all 3 datasets the LSTM is far superior to the MLP in terms of test RMSE while the run time is about 350% slower.

4.7 Comparing Trend Lines and Point Data for prediction using Machine Learning

To compare trend lines as a feature extraction technique we will use the best trend line model results vs the best point data model results on each dataset.

Table 12: Point data vs trend lines extracted

Point Data vs Trend Lines		
Dataset	Extraction	Test RMSE
Dataset 1 Naspers	Point	36.33
	Trend	316.44
Dataset 2 Redefine	Point	6.87
	Trend	18.72
Dataset 3 TFG	Point	30.57
	Trend	70.46

It is clear that using trend lines for feature extraction while using either an LSTM or MLP machine learning model for prediction does not compete with using point data. The machine learning models trained using point data are a lot better at making predictions compared to trend lines. It is important to take into account that trend lines make predictions a lot further into the future than point data does. Point data only predicts a single point into the future, on the datasets used in this paper, that is a single day. The trend lines predict on average 4.37 days into the future with an MSE of 1000 as used when training the models. It should also be noted that training both an MLP or LSTM on trend lines takes considerably longer than when training using point data. This is because NN's trained using point data produced optimal results with less internal layers and they only had to take a single input and produce a single output, when looking at the internal mechanisms of the NN's, this allows a lot less calculations to be made.

5 CONCLUSION AND FUTURE WORK

In conclusion, the Sliding Window segmentation algorithm is superior to the Bottom Up and Top Down algorithms as it is an online algorithm, it also makes the most accurate predictions when using an MLP while also has the fastest extraction run-time. It also smooths the data well by reducing the data size while keeping as much information about the signal as possible.

When comparing the MLP to the LSTM the MLP performs slightly better in terms of error in most cases and train time in all cases when trained using trend lines as features.

When comparing the predictions made by point data to the predictions made by trend lines it is easy to see that using an LSTM or MLP on point data produces a lot better results than trend lines do while training a lot faster.

Future work for on this topic could include comparing prediction using point data at a further lookahead instead of a single day in advance. Other types of neural networks could be implemented to try get better predictions when using trends lines as features because when using either an LSTM or MLP the predictions are not useful.

REFERENCES

- [1] [n. d.]. Piecewise regression: when one line simply isn't enough. <https://www.datadoghq.com/blog/engineering/piecewise-regression/>. Accessed: 2018-05-1.
- [2] Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. <https://www.tensorflow.org/> Software available from tensorflow.org.
- [3] R Chan. 2014. Primary Tools for Technical Analysis: Technical Analysis and Financial Asset Forecasting. (2014), 17–25. https://doi.org/10.1142/9789814436250_0002
- [4] François Chollet et al. 2015. Keras. <https://keras.io>. (2015).
- [5] Osman Hegazy, Omar S. Soliman, and Mustafa Abdul Salam. 2013. A Machine Learning Model for Stock Market Prediction. 4 (12 2013), 17–23.
- [6] John D. Hunter. 2007. Matplotlib: A 2D graphics environment. *Computing In Science & Engineering* 9, 3 (May-Jun 2007), 90–95.
- [7] Eric Jones, Travis Oliphant, Pearu Peterson, et al. 2001–. SciPy: Open source scientific tools for Python. (2001–). <http://www.scipy.org/> [Online; accessed 11-07-2018].
- [8] Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. 1993. Segmenting Time Series: A Survey and Novel Approach. In *In an Edited Volume, Data mining in Time Series Databases. Published by World Scientific*. Publishing Company, 1–22.
- [9] S Korkin. 2004. Machine learning for corporate failure prediction : an empirical study of South African companies. University of Cape Town. (2004).
- [10] Peter M. Nelson, Ethan R. Van Norman, and Theodore J. Christ. 2017. Visual Analysis Among Novices: Training and Trend Lines as Graphic Aids. *Contemporary School Psychology* 21, 2 (01 Jun 2017), 93–102. <https://doi.org/10.1007/s40688-016-0107-9>
- [11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
- [12] Karl Aberer Tao Lin, Tian Guo. 2017. Hybrid Neural Networks for Learning the Trend in Time Series. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*. 2273–2279. <https://doi.org/10.24963/ijcai.2017/316>
- [13] P. D. Yoo, M. H. Kim, and T. Jan. 2005. Machine Learning Techniques and Use of Event Information for Stock Market Prediction: A Survey and Evaluation. In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, Vol. 2. 835–841. <https://doi.org/10.1109/CIMCA.2005.1631572>