Article

A Model for Feature Selection with Binary Particle Swarm Optimisation and Synthetic Features

Samuel Olusegun Ojo *, Juliana Adeola Adisa, Pius Adewale Owolawi and Chunling Tu

Department of Computer Systems Engineering, Faculty of Information and Communication Technology, Tshwane University of Technology, Pretoria 0001, South Africa; owolawipa@tut.ac.za (P.A.O.); duc@tut.ac.za (C.T.)
* Correspondence: samuelojo88@gmail.com

Abstract: Recognising patterns and inferring nonlinearities between data that are seemingly random and stochastic in nature is one of the strong suites of machine learning models. Given a set of features, the ability to distinguish between useful features and seemingly useless features, and thereafter extract a subset of features that will result in the best prediction on data that are highly stochastic, remains an open issue. This study presents a model for feature selection by generating synthetic features and applying Binary Particle Swarm Optimisation with a Long Short-Term Memory-based model. The study analyses the correlation between data and makes use of Apple stock market data as a use case. Synthetic features are created from features that have weak/low correlation to the label and analysed how synthetic features that are descriptive of features can enhance the model's predictive capability. The results obtained show that by expanding the dataset to contain synthetic features before applying feature selection, the objective function was better optimised as compared to when no synthetic features were added.

Keywords: feature selection; synthetic features; LSTM; correlation; Particle Swarm Optimisation; stock market

check for updates

Citation: Ojo, S.O.; Adisa, J.A.; Owolawi, P.A.; Tu, C. A Model for Feature Selection with Binary Particle Swarm Optimisation and Synthetic Features. *AI* **2024**, *5*, 1235–1254. https:// doi.org/10.3390/ai5030060

Academic Editor: Gianni D'Angelo

Received: 1 July 2024
Revised: 21 July 2024
Accepted: 22 July 2024
Published: 25 July 2024

1. Introduction

In machine learning, a feature can be described as some input variable on which a label (or output) is dependent. The terms feature and variable are used interchangeably in this study. Measuring or calculating the importance of a feature prior to or during training and prediction is not a new area of study and is relevant in various models such as Random Forest and other decision tree-based models. Given a set of features, selecting a subset of these features for training and discarding the rest is one approach which has been adopted, particularly when addressing the curse of dimensionality. Studies have shown that discarding some features based on feature importance ranking or correlation does not always result in better prediction or classification. Various methodologies for selecting important features, thus eliminating noise, have been proffered in the literature [1–3].

Fundamentally, the process of feature selection (FS) is aimed at improving prediction and achieving greater speed of data (pre)processing. FS can be categorised into three approaches, namely, the filter, wrapper, and embedded methods [4]. Filter methods form part of a data pre-processing stage whereby features are selected prior to, and independent of, the model training. Filter methods can be either univariate or multivariate, scoring a single feature or a set of features. The wrapper approach to FS is an iterative approach in which the optimal subset of features is selected as a function of the model's performance or accuracy and can be computationally intensive. The embedded method involves selecting the features as part of the model's training process.

The filter, wrapper, and embedded methods often rank features by a score using various algorithms. One popular algorithm is the Pearson Correlation Coefficient (PCC) [5]. The PCC, r, is a measure of the associative strength between two variables that have some form of linear relationship. This measure can be computed by the following formula, for some variables x and y:

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2}\sqrt{\sum(y_i - \bar{y})^2}} \tag{1}$$

where x_i is the ith value of the variable x in a sample, $\bar{x}$ is the mean of the x variable's values, y_i is the ith value of the variable y in a sample, and $\bar{y}$ is the mean of the y variable's values. The value of a PCC can indicate one of three facts: A negative value indicates a negative correlation, meaning as one value decreases the other will increase, and vice versa. A PCC value of 0 indicates no correlation at all. A positive value indicates a positive correlation, meaning as one value increases, the other too will increase.

In this study, a model that adopts the filter approach to FS and makes use of the definitions of relevance with respect to a feature being weakly relevant or strongly relevant to the label is proposed. With the aid of Binary Particle Swarm Optimisation (BPSO) [6,7], the proposed model selects the best combination of features that will minimise the objective function. The model proposed by this study initially expands the dataset by creating synthetic features generated from features that have a weak correlation to some output label. In so doing, new (synthetic) features which are partially composed of weak features and strong features are generated. Synthetic features are features that are not part of the original dataset but are created through some mathematical operation of two or more features. The synthetic features will be created by performing some mathematical operation on features with weak correlation and other features with strong correlation to the label; subsequently, a basic version of a Long Short-Term Memory network model is used with BPSO to select the combination of features that minimise the objective function. These selected features are then used to perform predictions for future values of the stock market data. The results obtained show that by creating synthetic features with features having a weak and strong correlation, features with a stronger correlation to the label are generated, and by applying BPSO, the optimum combination of features that would minimise the objective function are selected.

Two versions of an LSTM network model are used. The first LSTM network model is a very basic version of the model which will provide values of the objective function for subsets of features selected through BPSO, and the second LSTM network model will be used to validate the selected features by training and making predictions.

The model proposed in this study is based on the premise that generating synthetic features that have a high correlation to the label would result in better predictions. A novel approach to identifying weak features in a dataset and generating strong(er) features with a higher correlation is presented. This study makes use of stock market price behaviour prediction as a use case and the model presented is applicable for time series analysis and prediction problems.

2. Literature Review

In this section, work relating to FS, determining feature relevance, Binary Particle Swarm Optimisation, and LSTM networks is presented. These techniques form the basis of the model presented in this study.

Several approaches for determining and measuring the strength of feature relevance exist. The Chi-squared test measures the independence of two or more features [8]. Given a categorical feature and label, the Chi-squared test measured the independence of the label with respect to the feature. It can be denoted by the following:

$$\chi^2 = \sum \frac{(O - E)^2}{E} \tag{2}$$

where χ^2 is the Chi-squared, O is the observed value and E is the expected value. The Chi-squared test has been used extensively in use cases including measuring word importance when classifying textual data [9], cancer cell classification [10] and other areas where determining feature relevance of categorical data is relevant. One of the advantages of the Chi-squared test is that it does not require homoscedasticity in the data.

Tree-based models such as decision trees as well as ensemble models, such as Random Forest and Gradient Boosting, are able to derive and measure the relevance of each feature in a dataset by ranking each feature in terms of importance [11,12]. Decision trees generally have several nodes and at each node perform a split based on which feature(s) minimises Entropy or a Gini Index. Ensemble models are similar but instead of having a single tree with many nodes, they create several trees through techniques known as Bagging and Boosting. Each subsequent tree in an Ensemble model performs splits based on the measures of feature impurity from the previous trees, and thus, each subsequent tree aims to make improvements based on the decisions (splits) made by the previous trees. Each individual tree performs splits on a random subset of the available features and performs the decision as to which features are more important by calculating a Gini Index or Entropy [13]. The Gini Index can be represented as follows:

$$G = 1 - \sum_{i=1}^{C} (p_i)^2 \tag{3}$$

Entropy is calculated as follows:

$$E = \sum_{i=1}^{C} -p_i \times \log_2(p_i) \tag{4}$$

where p is the probability of a class i.

2.1. Particle Swarm Optimisation

Particle Swarm Optimisation (PSO) comes from the family of evolutionary algorithms that mimic the behaviours and social activities exhibited in nature. PSO looks particularly at the behaviour of flocks of birds, schools of fish, and human social interaction, where each agent of the collective (swarm) is represented as a particle [6,14]. The objective of the swarm is to collectively work towards finding the best variables such as position and velocity, at any given time, that will result in the swarm achieving a goal. For a particle i in a swarm, a local best $(pbest_i)$ is shared with its neighbours based on its experiences. A Nearest Neighbour algorithm is used to evaluate the neighbours with which information is shared. A neighbouring particle with whom information has been shared can compare this information with its local best and also the global best $(gbest)$ and update its best value with the shared value if the shared value is better. This optimisation process is iterative, with each iteration producing some metric, and with the objective of finding the optimum collection of features for which this metric is optimised. For a particle in an N dimensional space, the updates to the position and velocity of a particle can be represented as follows:

$$v_i^{k+1} = wv_i^k + c_1\mathbf{Rand}()_\mathbf{1}(pbest_i^k - x_i^k) + c_2\mathbf{Rand}()_\mathbf{2}(gbest^k - x_i^k) \tag{5}$$

$$x_i^{k+1} = x_i^k + v_i^{k+1} \tag{6}$$

where v_i^k is the velocity of the ith particle for iteration k, c_1 and c_2 represent constants, $\boldsymbol{Rand}()_\mathbf{1}$ and $\boldsymbol{Rand}()_\mathbf{2}$ are random numbers between 0 and 1, $\boldsymbol{w}$ is the weight of inertia as a result of changing velocity, $\boldsymbol{x_i^k}$ is the current position of the $\boldsymbol{i}$th particle for iteration $\boldsymbol{k}$, and $\boldsymbol{pbest_i^k}$ and $\boldsymbol{gbest^k}$ represent the $\boldsymbol{i}$th particles personal best and global best for iteration $\boldsymbol{k}$, respectively.

2.2. Long Short-Term Memory Networks

A Long Short-Term Memory (LSTM) network is a type of Recurrent Neural Network (RNN) which, unlike the standard RNN, does not suffer from a problem known as the vanishing gradient problem in which data are lost overtime during training. Figure 1 shows an LSTM cell, with the three sigmoid gates it uses to memorise long-term dependencies between data and update its internal state:

- An Input gate (i): Responsible for determining the current state of the LSTM cell by applying a sigmoid function and $tanh$ function.
- A Forget gate (f): Responsible for determining which information is fed into an LSTM cell and which information should be forgotten or discarded.
- An Output gate (o): Determines what information is passed out of the LSTM cell.

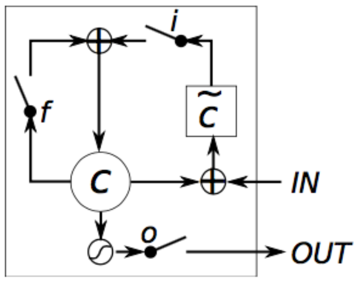


Figure 1. A Long Short-Term Memory network cell [15].

LSTM models have been developed and used successfully in several fields including text classification and Natural Language Processing [16,17], sentiment analysis [18,19], the prediction of future behaviour [20,21], and various other classification and regression problem areas.

2.3. Related Work

In a study, Ref. [22] proposed an FS model that combines both the filter and embedded approaches, using a decision tree model and the ReliefF algorithm [23]. A feature weight is calculated as part of the decision tree and is used to determine which features to select. The proposed algorithm was measured using the metrics accuracy, recall, and F1-score and performed well in terms of all three values.

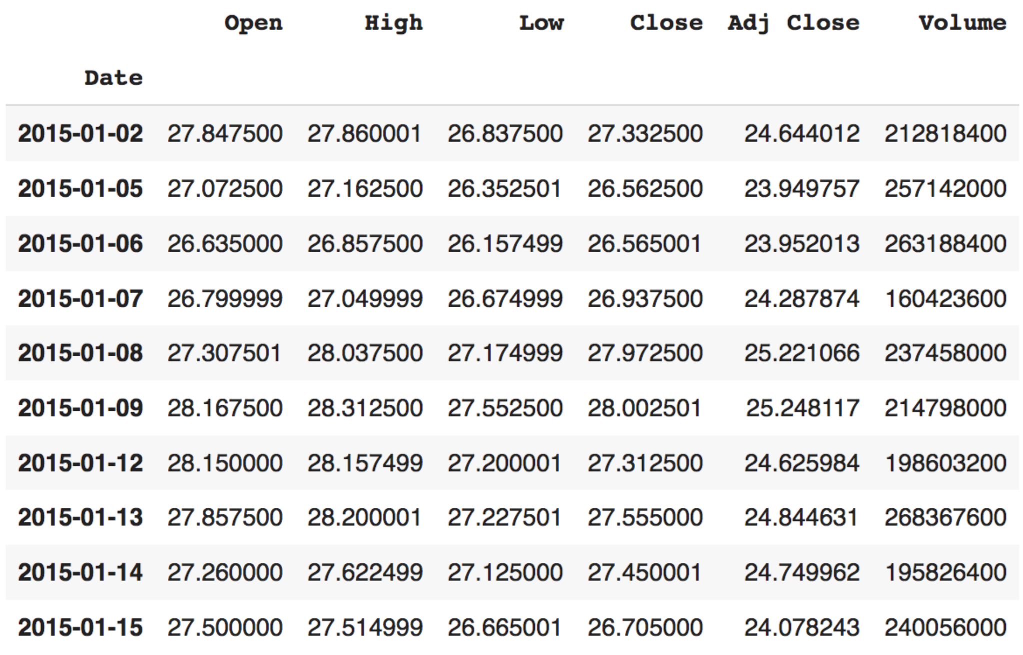
A study by Dhiman et al. [24] proposed an algorithm for FS using the binary meta-heuristic algorithm, Binary Emperor Penguin Optimiser (EPO), which like PSO is modelled after behaviours found in nature. EPO is modelled after the huddling behaviour of emperor penguins found in the arctic as they move in a swarm. The proposed algorithm is compared against other binary meta-heuristic algorithms and outperforms the other binary algorithms using various benchmark functions.

3. Materials and Methods

This study presents a model for FS for the purpose of stock market behaviour prediction by taking the correlation of features to the label into account, and creating synthetic features that have strong(er) correlation to the label from features which have weak correlation. The Binary Particle Swarm Optimisation (BPSO) algorithm is used to select the best set of features that minimise the objective function with a very basic version of an LSTM model having a single layer and dropout. The combination of this basic LSTM model and BPSO is used as a filter mechanism for FS. The selected features are then used as input data to train the main model.

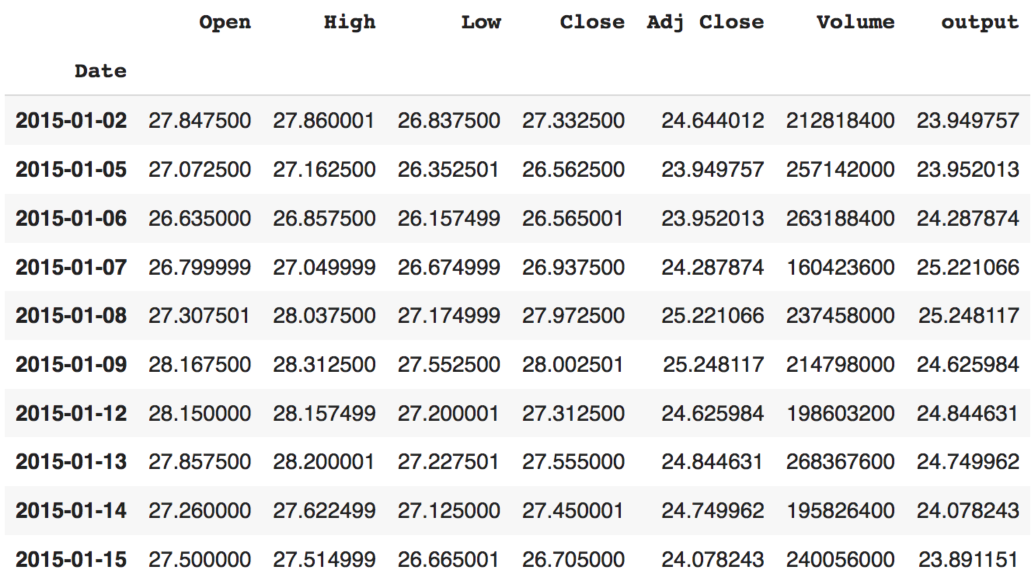
3.1. Data Analysis and Pre-Processing

Apple stock market data for a 5-year period between 1 January 2015 and 31 December 2019 obtained from Yahoo Finance were used for the dataset. The dataset consists of six features including Open, High, Low, Close, Adj Close prices, the Date recorded, and Volume. The purpose of this study is to make use of a multivariate analysis of all these features to predict the future price and behaviour of the "Adj Close" price. A label called "output" is created which represents the next day's Adj Close price for each day. Figure 2 shows a sample of the original dataset with the various features as well as a Date column indicating a daily time-step, while Figure 3 shows a sample of the dataset with a new column "output" introduced. This "output" column represents the next day's Adj Close price and is the label which is to be predicted.



Date	Open	High	Low	Close	Adj Close	Volume
2015-01-02	27.847500	27.860001	26.837500	27.332500	24.644012	212818400
2015-01-05	27.072500	27.162500	26.352501	26.562500	23.949757	257142000
2015-01-06	26.635000	26.857500	26.157499	26.565001	23.952013	263188400
2015-01-07	26.799999	27.049999	26.674999	26.937500	24.287874	160423600
2015-01-08	27.307501	28.037500	27.174999	27.972500	25.221066	237458000
2015-01-09	28.167500	28.312500	27.552500	28.002501	25.248117	214798000
2015-01-12	28.150000	28.157499	27.200001	27.312500	24.625984	198603200
2015-01-13	27.857500	28.200001	27.227501	27.555000	24.844631	268367600
2015-01-14	27.260000	27.622499	27.125000	27.450001	24.749962	195826400
2015-01-15	27.500000	27.514999	26.665001	26.705000	24.078243	240056000

Figure 2. Sample of Apple stock market dataset.

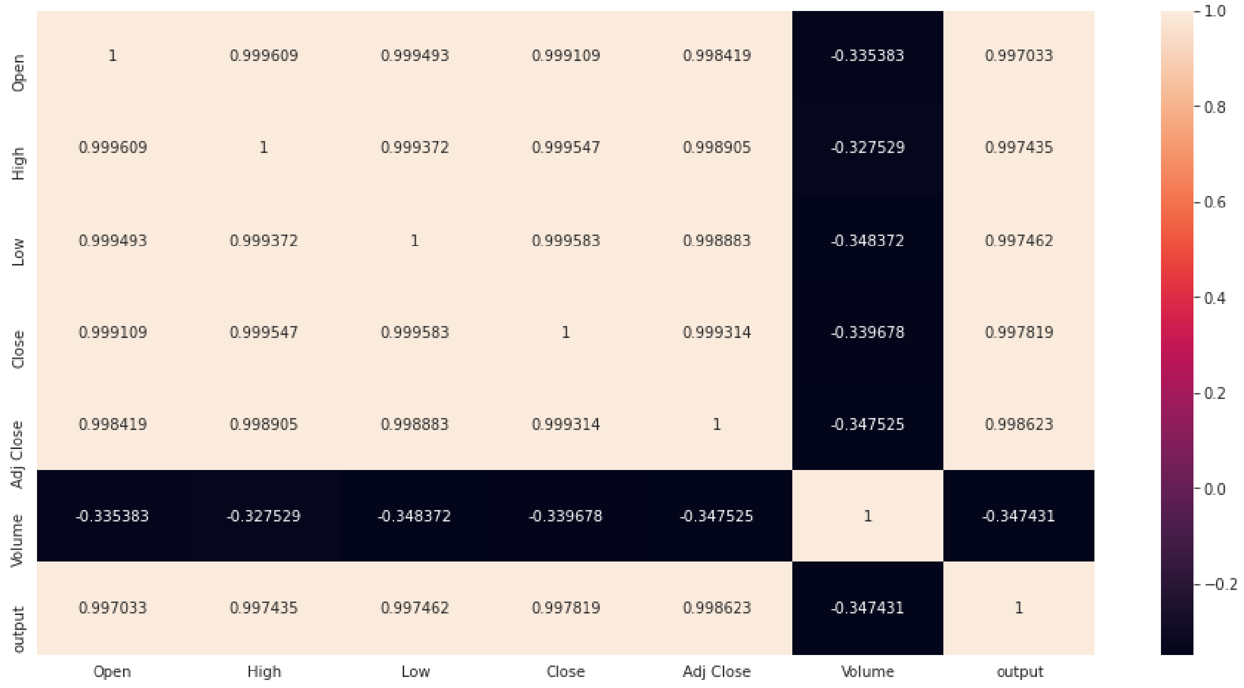


Date	Open	High	Low	Close	Adj Close	Volume	output
2015-01-02	27.847500	27.860001	26.837500	27.332500	24.644012	212818400	23.949757
2015-01-05	27.072500	27.162500	26.352501	26.562500	23.949757	257142000	23.952013
2015-01-06	26.635000	26.857500	26.157499	26.565001	23.952013	263188400	24.287874
2015-01-07	26.799999	27.049999	26.674999	26.937500	24.287874	160423600	25.221066
2015-01-08	27.307501	28.037500	27.174999	27.972500	25.221066	237458000	25.248117
2015-01-09	28.167500	28.312500	27.552500	28.002501	25.248117	214798000	24.625984
2015-01-12	28.150000	28.157499	27.200001	27.312500	24.625984	198603200	24.844631
2015-01-13	27.857500	28.200001	27.227501	27.555000	24.844631	268367600	24.749962
2015-01-14	27.260000	27.622499	27.125000	27.450001	24.749962	195826400	24.078243
2015-01-15	27.500000	27.514999	26.665001	26.705000	24.078243	240056000	23.891151

Figure 3. Sample of Apple stock market dataset with output label column.

Analysing the correlation between the individual features and the label is crucial as part of understanding the effects of each feature on the label values. Figure 4 shows the correlation matrix when PCC is applied for each of the features and output label. Figures 5–9 all show a positive and high correlation between each feature and the label, whereas Figure 10 shows a negative and low correlation between "Volume" and the label.

In analysing this correlation data, the absolute value of each feature's correlation against the "output" label is observed. In addition, the absolute value of each correlation is measured on a scale from 0 to 1, where the closer to 1, the stronger the correlation value, and the closer to 0, the weaker the correlation value. With the dataset used, the correlation matrix shows that the "Adj Close" feature has the highest correlation to the "output" label and "Volume" has the lowest. The model proposed by this study aims to initially create synthetic features from features with a weak correlation to the label, and thus generate [synthetic] features with a stronger correlation in the dataset. The "Volume" feature is a candidate for creating synthetic features for this dataset.



	Open	High	Low	Close	Adj Close	Volume	output
Open	1	0.999609	0.999493	0.999109	0.998419	-0.335383	0.997033
High	0.999609	1	0.999372	0.999547	0.998905	-0.327529	0.997435
Low	0.999493	0.999372	1	0.999583	0.998883	-0.348372	0.997462
Close	0.999109	0.999547	0.999583	1	0.999314	-0.339678	0.997819
Adj Close	0.998419	0.998905	0.998883	0.999314	1	-0.347525	0.998623
Volume	-0.335383	-0.327529	-0.348372	-0.339678	-0.347525	1	-0.347431
output	0.997033	0.997435	0.997462	0.997819	0.998623	-0.347431	1

Figure 4. Correlation matrix.

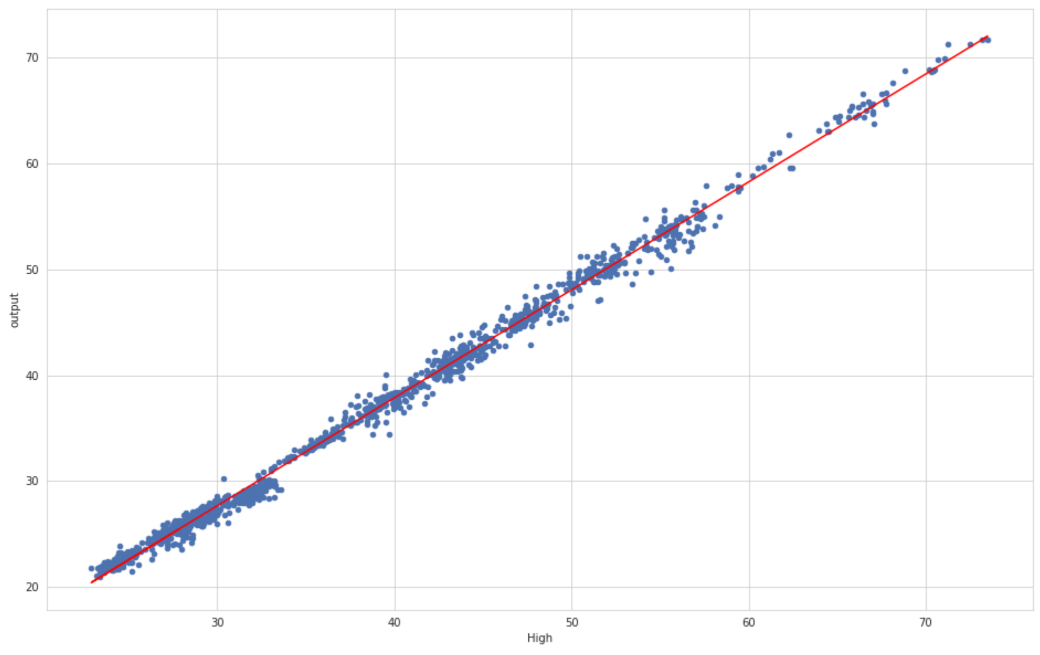


Figure 5. High vs. Output.

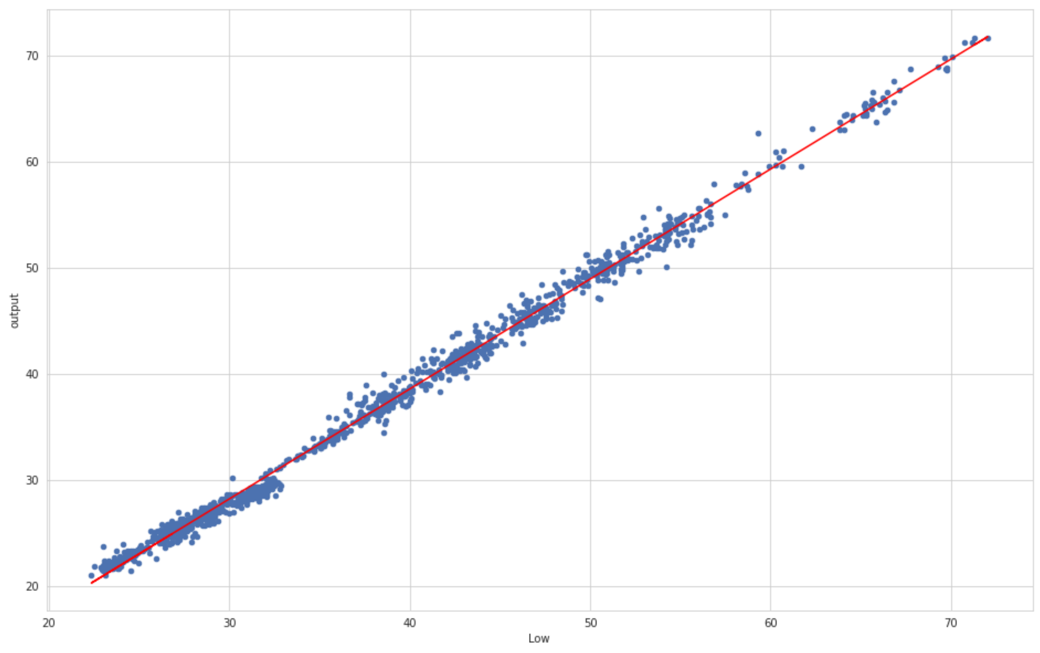


Figure 6. Low vs. Output.

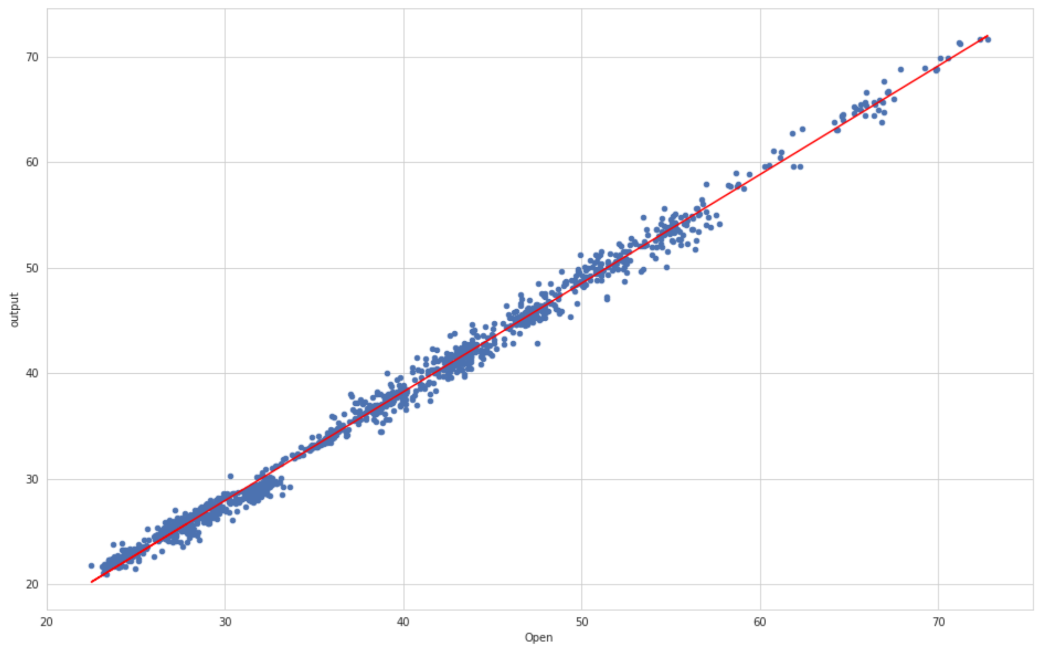


Figure 7. Open vs. Output.

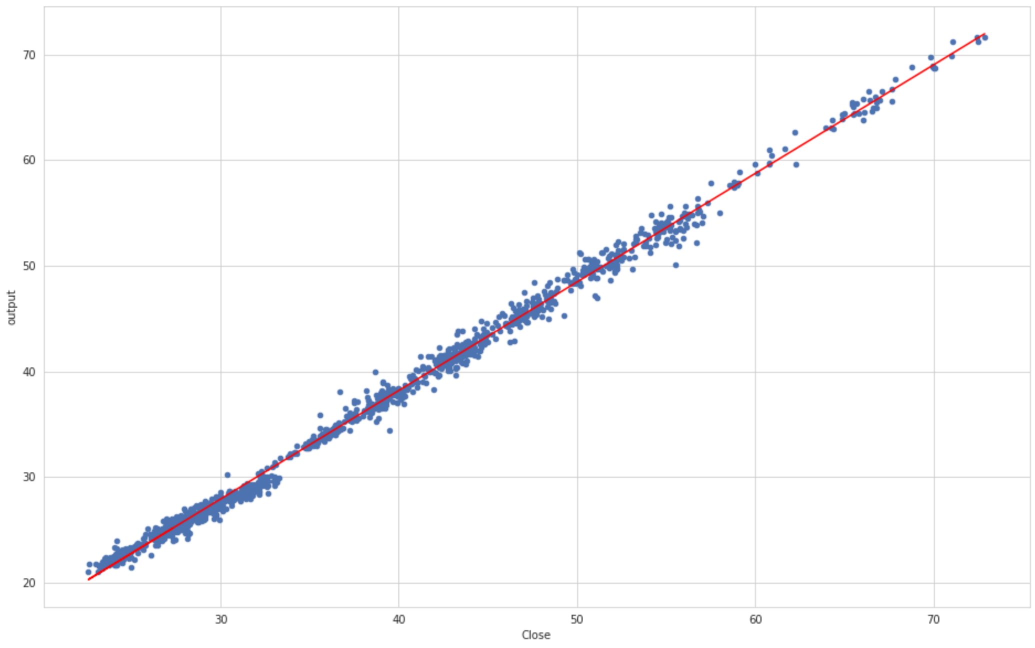


Figure 8. Close vs. Output.

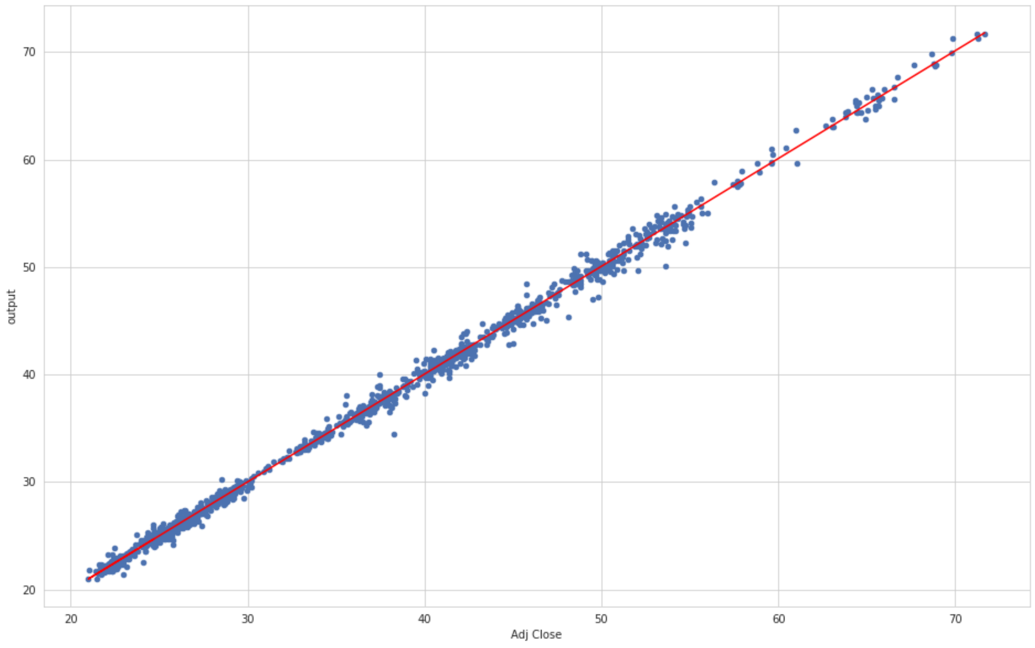


Figure 9. Adj Close vs. Output.

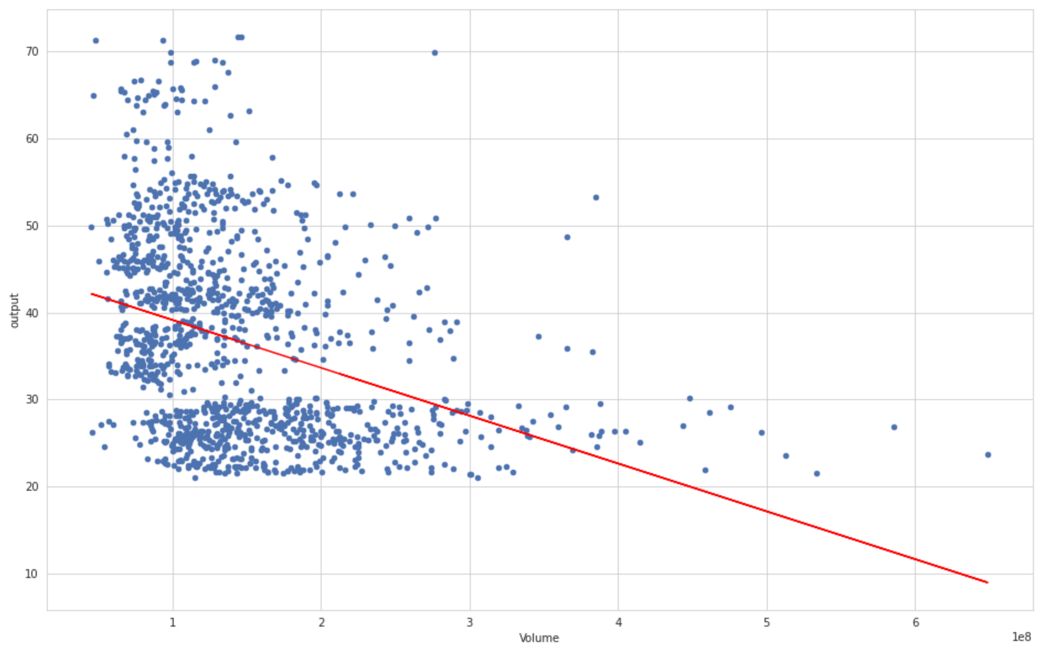


Figure 10. Volume vs. Output.

3.2. Building Synthetic Features

An algorithm was developed to determine a weak feature for a given dataset and identify potential candidate features from which synthetic features can be created. A benchmark needed to be defined that would computationally define what a weak correlation meant for a specific dataset. To this end, a weak correlation is defined as any correlating value which is less than the mean correlation of all features. This mean correlation r_{Mean} can be represented by Equation (7).

$$r_{mean} = \frac{\sum_{n=1}^{N} |r(f_n)|}{N} \tag{7}$$

where f_n is the nth feature in a dataset of N features, and $r(f_n)$ is the correlation of the nth feature.

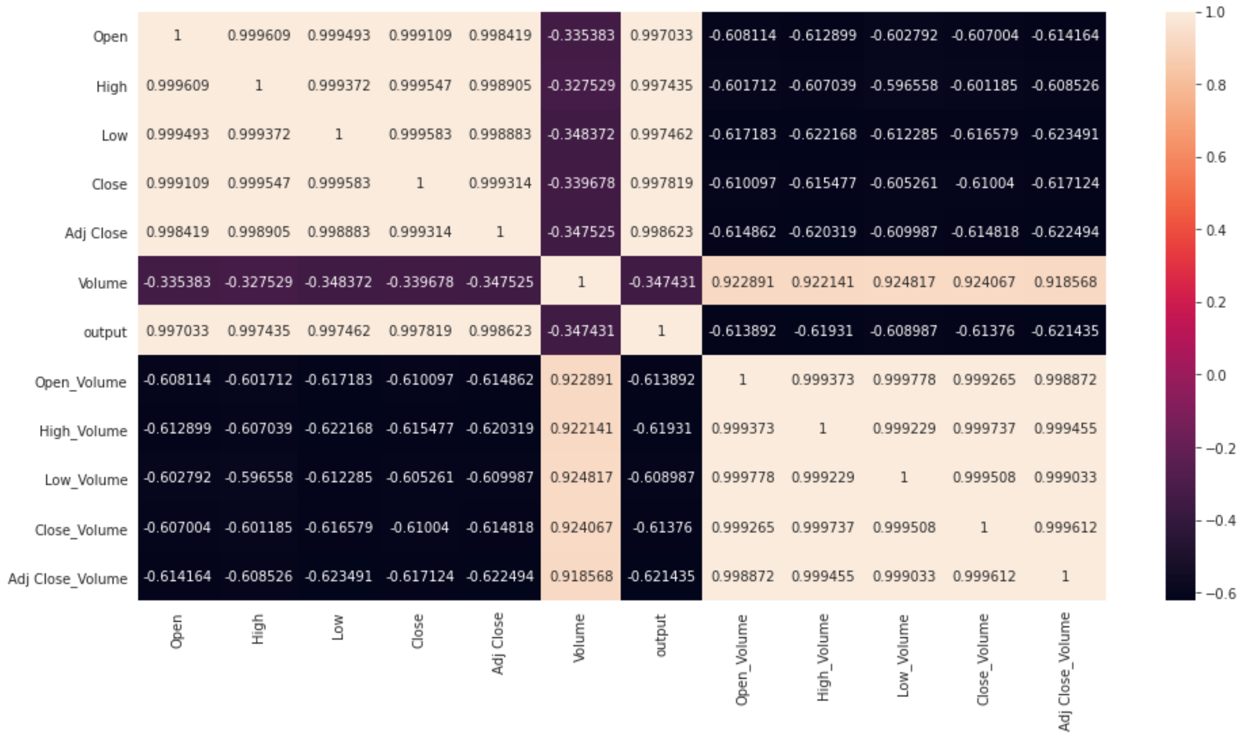
In this study, a feature with a weak correlation is defined as one for which the absolute value of its correlation against the label is less than r_{mean}. Synthetic features are then created from such features by performing a mathematical operation using this feature and all other features in the dataset. The pseudocode for this operation is represented by Algorithm 1.

Algorithm 1: Create synthetic features from features with weak correlation to the label

```
Data: $D$ = Input dataframe of $N$ number of raw features, $L$ = Dataframe of output label
Result: $D'$ = Dataframe of Raw and synthetic features: $D \subseteq D'$
$D' = D$ for $f_n \in D$ do
    if $| r(f_n, L) | < r_{mean}$ then
        for do
            $D'_i = f_n/D_i$
        end
    end
end
```

In Algorithm 1, a synthetic feature D'_i is created by dividing the value of a weak feature f_n by the value of a strong feature D_i. In so doing, a new value which represents a fraction made up of two of the values from the raw features is created. For data that are numerical, Algorithm 1 highlights a means by which synthetic data can be created by performing a mathematical operation on weak and strong features. In Algorithm 1, $\boldsymbol{D}$ represents the raw dataset as input, which contains $\boldsymbol{N}$ number of features. $\boldsymbol{L}$ is the label in the dataset, which is represented as "output". $\boldsymbol{f_n}$ is a feature of the raw dataset $\boldsymbol{D}$ at position $\boldsymbol{n}$. $\boldsymbol{L}$ is a dataset containing only the label values, and $\boldsymbol{D'}$ is the resulting dataset which contains the raw and synthetic features. The overall process shown in Algorithm 1 involves looping through the dataset of raw features and for each feature, computing the absolute value of the PCC of the raw feature to the label, denoted as $\mid \boldsymbol{r(f_n, L)} \mid$. If the absolute value of the computed PCC is less than $\boldsymbol{r_{mean}}$ from Equation (7), then the value of the feature $\boldsymbol{f_n}$ is divided by the values of each of the other features at that specific index $\boldsymbol{n}$ that are deemed strong features, resulting in a new (synthetic) feature that represents a fractional value made up of the weak and strong feature. This process is repeated for each feature, producing a new expanded dataset $\boldsymbol{D'}$ that is made up of raw features and synthetic features that have a higher PCC value to the label than the weak features. The premise of creating synthetic features is that a model's accuracy of prediction can be enhanced by expanding the dataset to contain synthetic features that have a high correlation to the label.

For the dataset, it was observed that when Algorithm 1 was applied, the feature "Volume" had the weakest correlation to the label. A dataset of raw and synthetic features is then created based on this. The resulting dataset contains synthetic features with a stronger correlation than "Volume". A naming convention for these synthetic features is adopted by concatenating the original feature name with an underscore and the name of the feature with low correlation. Figure 11 shows the correlation matrix of the new dataset. Taking the absolute value of each feature's correlation to the label, it was observed that the synthetic features (Open_Volume, High_Volume, Low_Volume, Close_Volume, and Adj Close_Volume) have stronger correlation than the original Volume feature.



	Open	High	Low	Close	Adj Close	Volume	output	Open_Volume	High_Volume	Low_Volume	Close_Volume	Adj Close_Volume
Open	1	0.999609	0.999493	0.999109	0.998419	-0.335383	0.997033	-0.608114	-0.612899	-0.602792	-0.607004	-0.614164
High	0.999609	1	0.999372	0.999547	0.998905	-0.327529	0.997435	-0.601712	-0.607039	-0.596558	-0.601185	-0.608526
Low	0.999493	0.999372	1	0.999583	0.998883	-0.348372	0.997462	-0.617183	-0.622168	-0.612285	-0.616579	-0.623491
Close	0.999109	0.999547	0.999583	1	0.999314	-0.339678	0.997819	-0.610097	-0.615477	-0.605261	-0.61004	-0.617124
Adj Close	0.998419	0.998905	0.998883	0.999314	1	-0.347525	0.998623	-0.614862	-0.620319	-0.609987	-0.614818	-0.622494
Volume	-0.335383	-0.327529	-0.348372	-0.339678	-0.347525	1	-0.347431	0.922891	0.922141	0.924817	0.924067	0.918568
output	0.997033	0.997435	0.997462	0.997819	0.998623	-0.347431	1	-0.613892	-0.61931	-0.608987	-0.61376	-0.621435
Open_Volume	-0.608114	-0.601712	-0.617183	-0.610097	-0.614862	0.922891	-0.613892	1	0.999373	0.999778	0.999265	0.998872
High_Volume	-0.612899	-0.607039	-0.622168	-0.615477	-0.620319	0.922141	-0.61931	0.999373	1	0.999229	0.999737	0.999455
Low_Volume	-0.602792	-0.596558	-0.612285	-0.605261	-0.609987	0.924817	-0.608987	0.999778	0.999229	1	0.999508	0.999033
Close_Volume	-0.607004	-0.601185	-0.616579	-0.61004	-0.614818	0.924067	-0.61376	0.999265	0.999737	0.999508	1	0.999612
Adj Close_Volume	-0.614164	-0.608526	-0.623491	-0.617124	-0.622494	0.918568	-0.621435	0.998872	0.999455	0.999033	0.999612	1

Figure 11. Correlation of new dataset containing synthetic features and the label.

3.3. Model Efficiency Considerations

Machine learning models are prone to lengthy training times and resource utilisation, particularly when trained on complex or large datasets. It is therefore important to optimise the design of a machine learning model by considering the amount of resources such as time used when training the model. In order to find a balance between accuracy and resource efficiency, an experiment was carried out to determine the impact of model complexity on training time. A stacked LSTM with a complex architecture (having multiple LSTM layers) was developed and trained on the dataset and the time taken for training was observed. Similarly, a simple LSTM model with a single LSTM layer was developed and trained on the same dataset and the training time was observed. Figure 12 shows the complex LSTM model, with several LSTM layers and dropout layers in between each LSTM layer. Figure 13 shows the total time taken for the complex LSTM model to complete training on the Apple stock market dataset. In comparison, Figure 14 shows the simple LSTM model experimented with and evaluated for training time, with a single LSTM layer and a single dropout layer. Figure 15 illustrates the associated training time for the simple LSTM model on the Apple dataset.

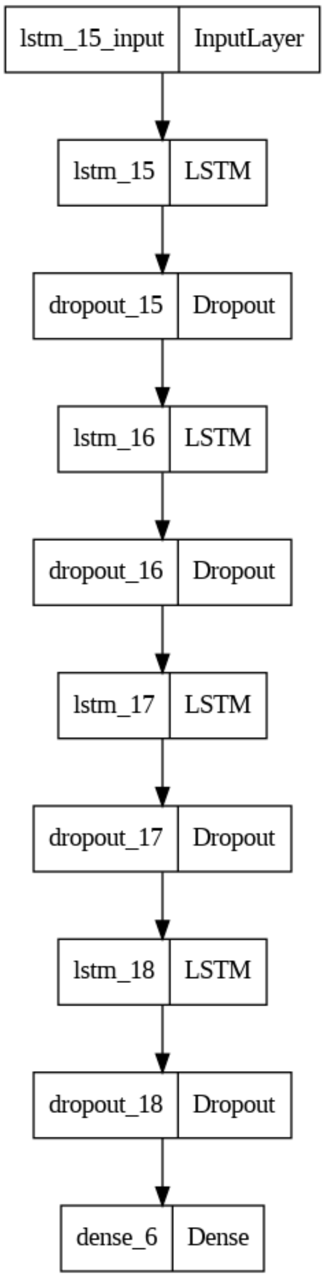


Figure 12. Complex LSTM model.

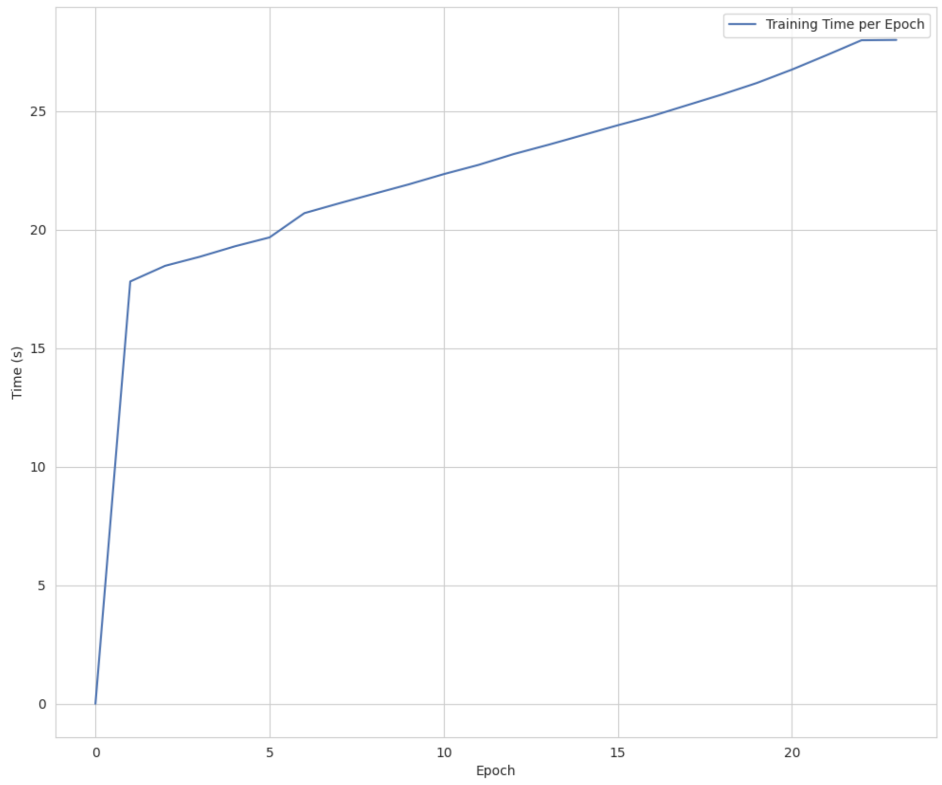


Figure 13. Complex LSTM model: training time per epoch.

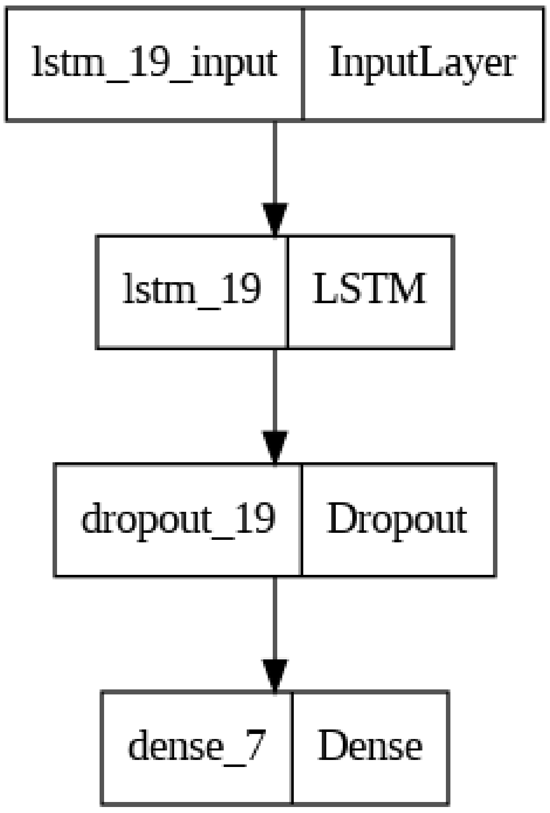


Figure 14. Simple LSTM model.

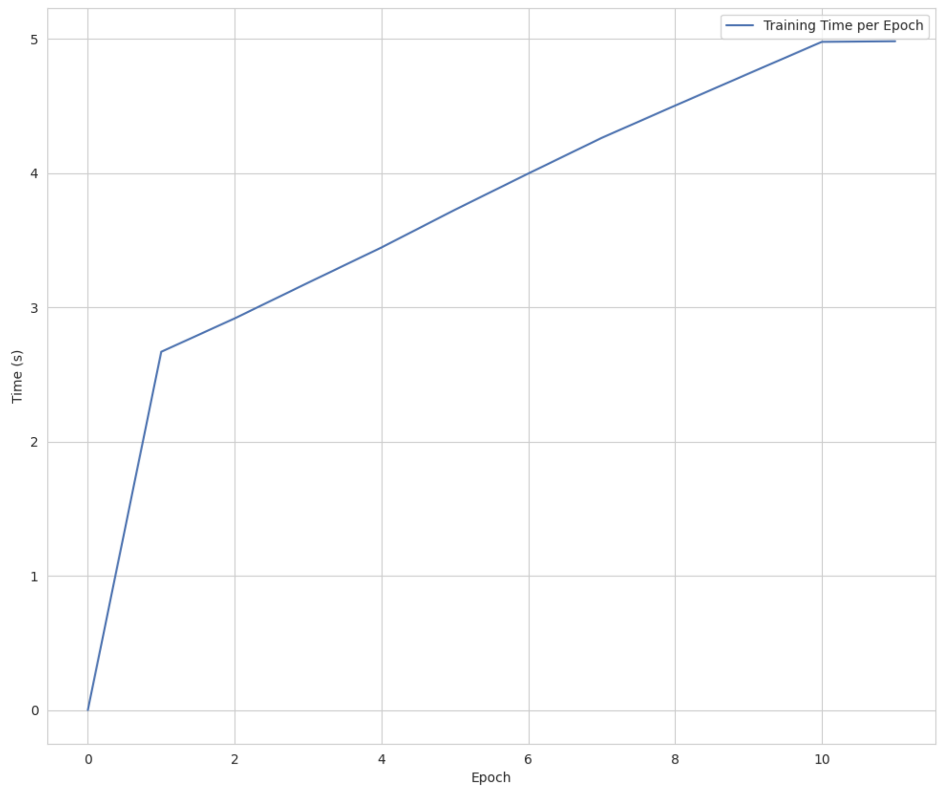


Figure 15. Simple LSTM model: training time per epoch.

From the training times observed in Figures 13 and 15, the complex LSTM model took approximately 28 s for training, while the simple LSTM model took about 5 s for training. For the purpose of this study, balancing accuracy with resource utilisation was vital. For algorithms in which a balance is required between accuracy and resource utilisation, the use of a less resource-intensive model can be adopted. The applicability of this design consideration is discussed further in Section 3.4 where the BPSO technique is applied.

3.4. Model Training and Binary Particle Swarm Optimization

With the expanded dataset containing synthetic features along with the original features and label, Binary Particle Swarm Optimisation (BPSO) is applied to a very basic LSTM network to determine the best combination of features that best optimise the objective function. BPSO works in the same way as the PSO previously described; however, the position of a particle in BPSO is discrete. With respect to the problem of feature relevance, BPSO will determine whether a feature is relevant or not. The position of a particle in BPSO can be denoted by Equation (8) [7]:

$$\boldsymbol{X_{ij}(t+1)} = \begin{cases} \boldsymbol{0, if\ Rand() \geq S(v_{ij}(t+1))} \\ \boldsymbol{1, if\ Rand() < S(v_{ij}(t+1))} \end{cases} \tag{8}$$

where $\boldsymbol{Rand()}$ is a random number, $\boldsymbol{v_{ij}}$ is the velocity of the $\boldsymbol{i}$th particle for iteration $\boldsymbol{j}$, and $\boldsymbol{S(v_{ij}(t+1))}$ is the sigmoid function:

$$\boldsymbol{S(v_{ij}(t+1)) = \frac{1}{1+e^{-(v_{ij})}}} \tag{9}$$

The BPSO is applied to a very basic LSTM network, which provides MSE values for each iteration of the BPSO. The purpose of using a very basic LSTM network at this stage is the BPSO algorithm is not interested in achieving an MSE value that is as close to 0 as possible; for now, rather, it is interested in whether the MSE value with a set of features

$\boldsymbol{F_x}$ is less than or greater than the MSE for another set of features $\boldsymbol{F_y}$. In addition, a small LSTM network will save us long computation times when trying to select the optimum set of features. Ultimately, the BPSO and LSTM network will provide a set of features for which the smallest MSE value overall is obtained. The BPSO algorithm is initialised with the following values in relation to Equation (5): $\boldsymbol{c_1} = 0.5$, $\boldsymbol{c_2} = 0.5$, $\boldsymbol{w} = 0.9$, and $\boldsymbol{k} = 5$ with a total of 10 iterations.

Table 1 shows the features that were selected as a result of the BPSO and LSTM network FS process, with all features that have a value of **1** being part of the set of relevant features that provide the smallest MSE value. From Table 1 it is observed that the BPSO algorithm has selected High, Volume, Open_Volume, Low_Volume, Close_Volume, and Adj Close_Volume as the combination of features that will result in the smallest MSE value. Of the six features selected, four are synthetic features.

Table 1. Result of BPSO applied to LSTM network to select set of features that provide a minimal MSE.

Feature	Relevance Indicator
Open	0
High	1
Low	0
Close	0
Adj Close	0
Volume	1
Open_Volume	1
High_Volume	0
Low_Volume	1
Close_Volume	1
Adj Close_Volume	1

To validate the relevance of the feature set selected by the BPSO algorithm, features that are deemed relevant by the BPSO and LSTM network are selected as an input dataset of relevant features to our main LSTM network model, which is a larger LSTM network model that has more input neurons and layers. This LSTM network is trained on the dataset of relevant features and is used to make prediction for future stock market behaviour.

3.5. Data Pre-Processing and Model Design

The LSTM model is developed in the Python programming language along with the Tensorflow (V2.2.0) library for machine learning. Google Colaboratory is used to develop and run the model on a GPU providing 13 GB of memory and 68.4 GB disk space. The dataset of relevant features is scaled to values between 0 and 1, and split into 80% training data and 20% test data. During training, 20% of the training data are reserved as a validation set. Figure 16 illustrates the split of training data and test data for the Adj Close daily price which the model will make predictions of. A stacked LSTM model is used and the Adaptive Moment Estimator (Adam) algorithm [25] is adopted for updating the internal network weights. Early stopping is used to prevent overfitting with 200 epochs and a patience value of 20 epochs.

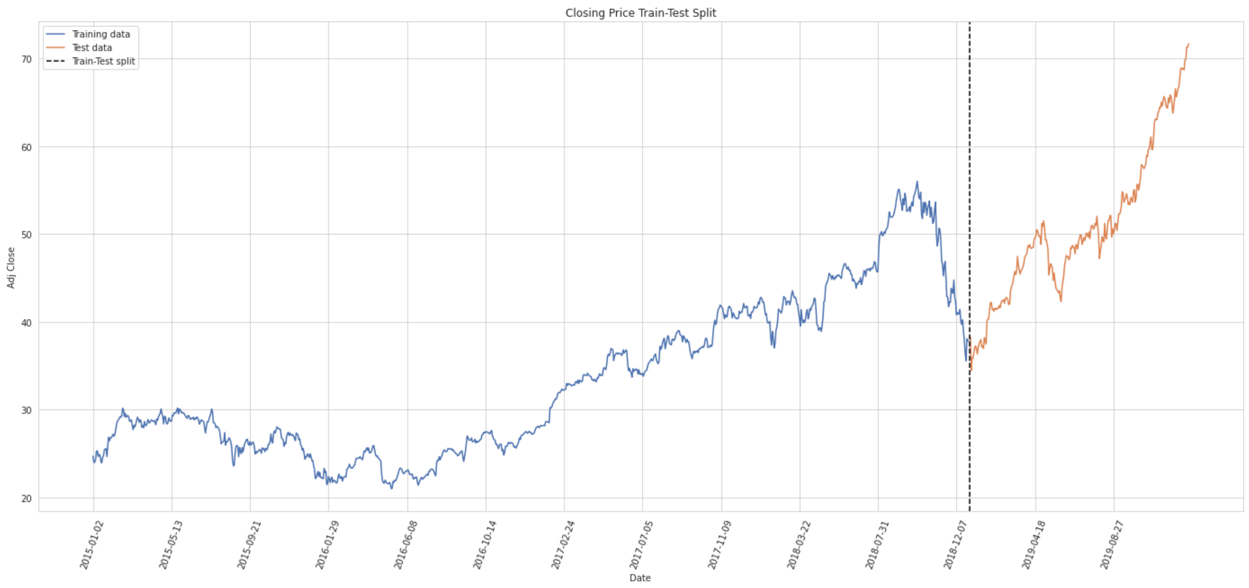


Figure 16. Training and test data split of Adj Close daily price.

3.6. Model Evaluation

To evaluate the model's performance, values of the Mean Absolute Error (MAE) and Mean Squared Error (MSE) were computed. These evaluation metrics are expressed by Equation (10) and Equation (11), respectively [26]:

$$\boldsymbol{MAE = \frac{1}{N}\sum_{t=1}^{N}|y_t - \hat{y}_t|} \tag{10}$$

$$\boldsymbol{MSE = \frac{1}{N}\sum_{t=1}^{N}(y_t - \hat{y}_t)^2} \tag{11}$$

where $\boldsymbol{N}$ is the number of observed values, $\boldsymbol{y_t}$ is the set of actual values, and $\boldsymbol{\hat{y}_t}$ is the set of predicted values.

4. Results

In this section, the results of applying the most relevant features selected by the BPSO algorithm for predicting the future behaviour of the stock market are presented as compared with making use of a dataset with no synthetic features and no FS model applied.

Figure 17 shows the observed loss for the duration of training. Due to the adoption of the early stopping strategy, the model completed training after 17 epochs. If the early stopping strategy was not in place, the model would have gone beyond 17 epochs and this would have possibly caused overfitting. In addition, Figure 18 and Figure 19 show the observed MSE and MAE, respectively, for the duration of training. The figures also show the model completed training after 17 epochs and that the MSE and MAE values for training and validation gradually decrease and eventually come to a place of near convergence towards the end of the training cycle.

Predictions are made based on the test dataset and evaluate our model's prediction by analysing the MSE and MAE values. In addition, the LSTM network model is trained on the original dataset for which no synthetic features were created and was not subjected to the FS process proposed by this study, and evaluated using the same metrics. Table 2 shows the metrics from both experiments. The metrics show that by subjecting the dataset to the FS model proposed, smaller values for the MSE, MAE, and MAPE were obtained, indicating that by applying the proposed model as a filter approach prior to model training and prediction, the LSTM model's predictability was enhanced.

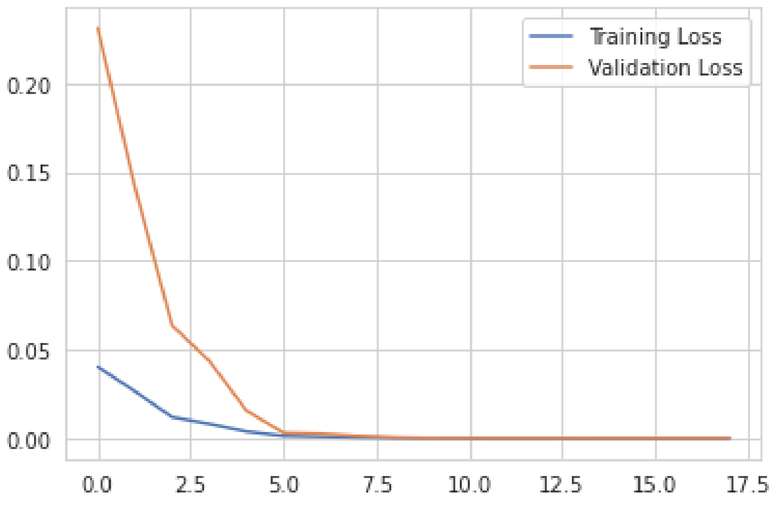


Figure 17. Training and Validation Loss.

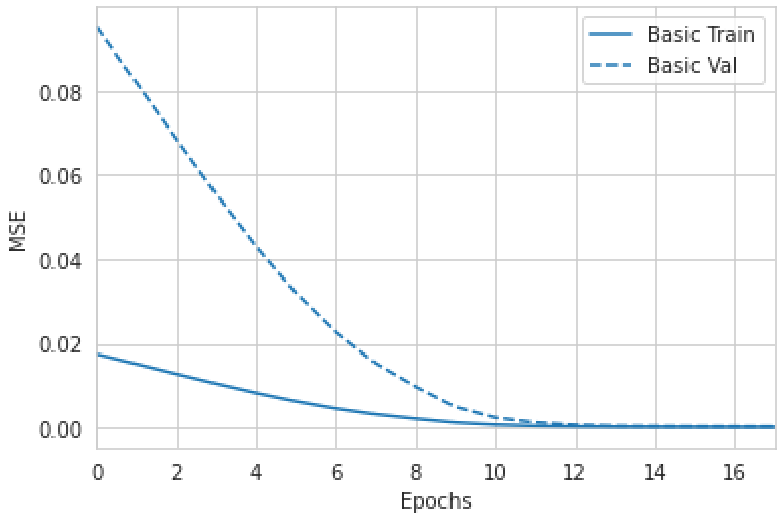


Figure 18. Training and Validation MSE observed for the duration of training.

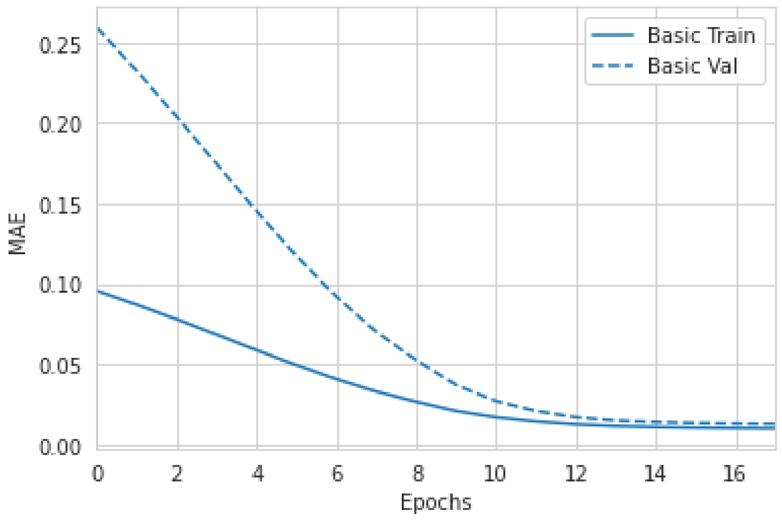


Figure 19. Training and Validation MAE observed for the duration of training.

Table 2. LSTM model MAE, MSE, and MAPE with and without proposed feature selection model.

Proposed Feature Selection	MAE	MSE
YES	0.000305	0.012886
NO	0.000433	0.016486

The future behaviour of the Adj Close price is predicted using the test data. Figure 20 shows the Adj Close training, test, and predicted values plotted for each day. Figure 21 shows an enhanced plot of the test and predicted values of the daily Adj Close price. Additionally, Figure 22 shows a plot of the Adj Close training, test, and predicted values when the FS model was not applied, and Figure 23 shows an enhanced plot of the test and predicted values when the FS model was not applied.

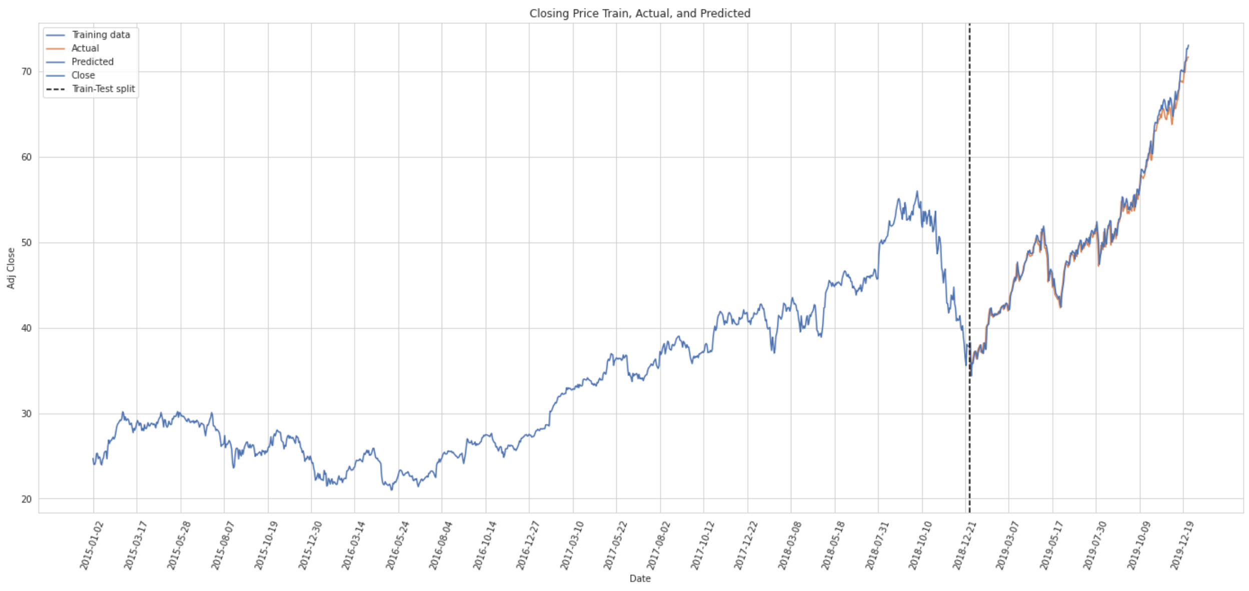


Figure 20. Training, test, and predicted daily Adj Close price with proposed feature selection applied.

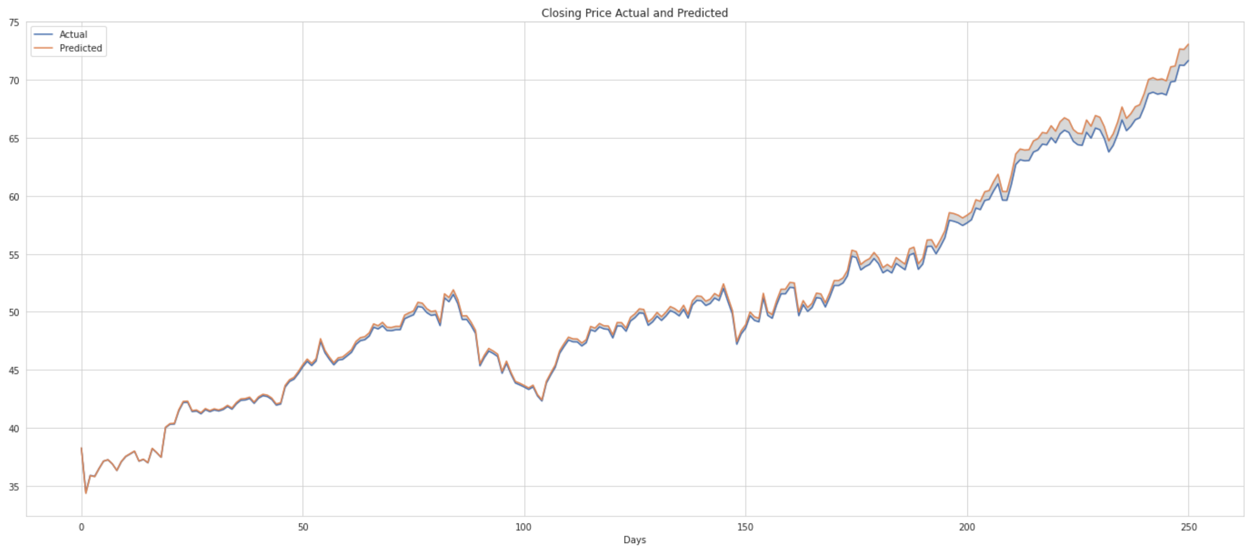


Figure 21. Test and predicted daily Adj Close price with proposed feature selection applied.

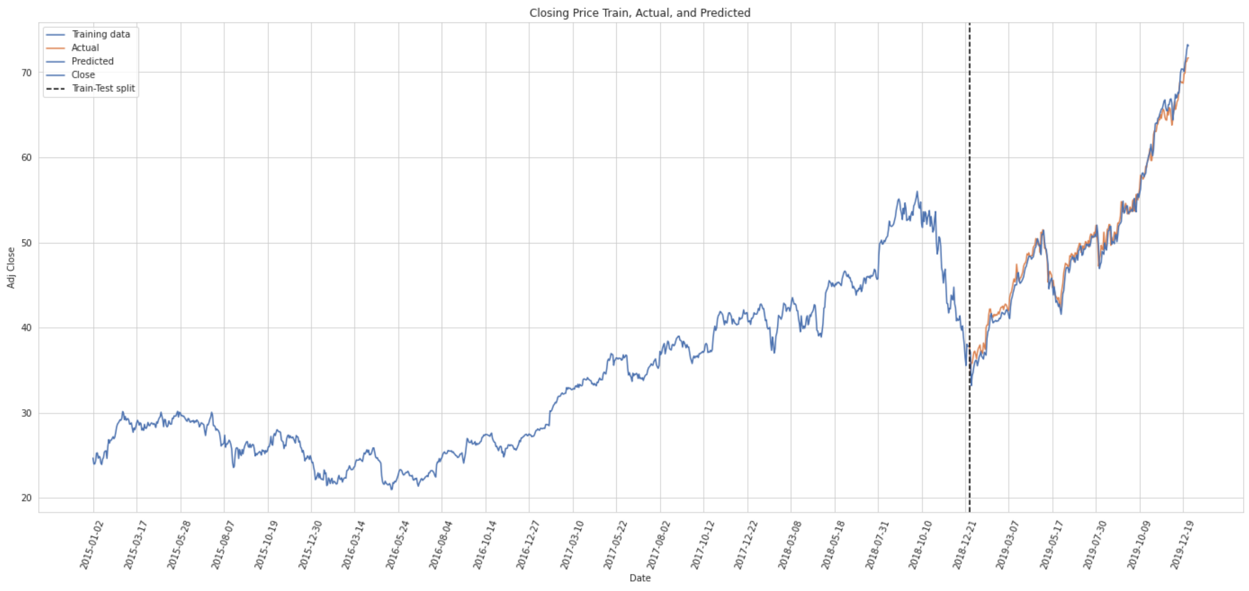


Figure 22. Training, test, and predicted daily Adj close price without feature selection.

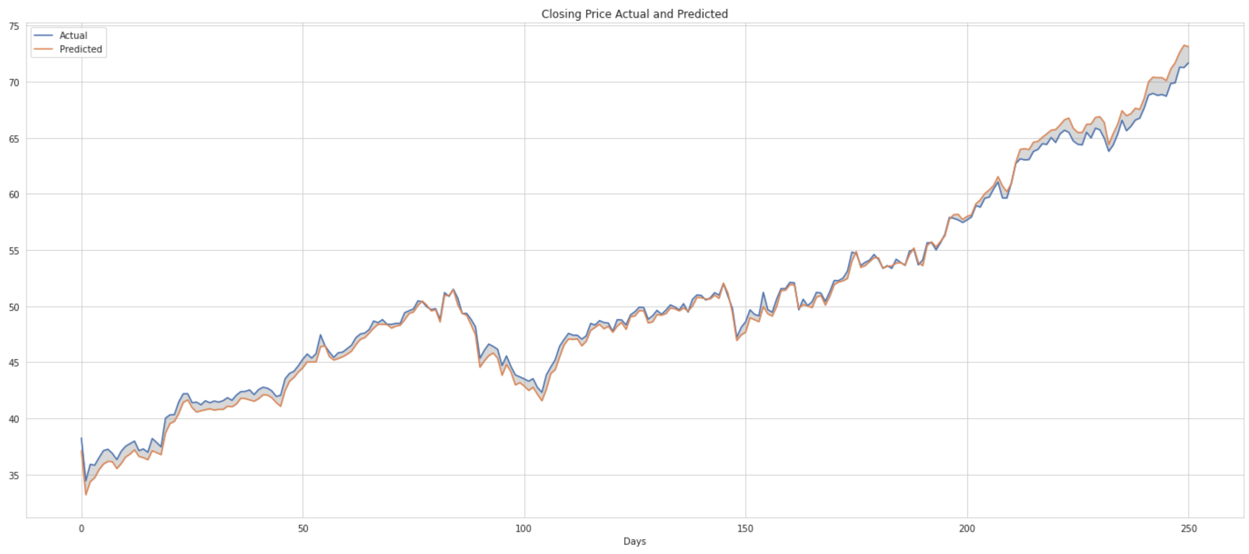


Figure 23. Test and predicted daily Adj Close price without feature selection.

From the metrics in Table 2 and the plots in Figures 20–23, it was observed that the model performed well in making predictions of the future Adj Close price when the proposed FS model was applied as compared with when no feature selection was applied. From the results obtained, when the proposed FS model was used, the LSTM model had better prediction and achieved values of 0.000305 and 0.012886 for the MAE and MSE, respectively. When compared with the MAE and MSE of 0.000433 and 0.016486 when the FS model was not applied, the overall predictions were improved through the application of the FS model.

5. Conclusions

In this study a model for FS was presented, making use of synthetic features, Pearson Correlation Coefficient, and Binary Particle Swarm Optimization. The results show that by initially expanding the dataset to create synthetic features that have a greater PCC to the label, and when the FS model was applied before training an LSTM network model, the predictive capability of the LSTM network was enhanced and the evaluation error metrics (MSE and MAE) were smaller than when the FS model was not used. As part of future work, the use of synthetic features with Pearson Correlation Coefficient without Binary Particle Swarm Optimisation can be experimented with for other models which have feature importance/ranking such as Random Forest and other tree-based models, the results of which can be compared with the FS model presented by this study. In addition, the use of wrapper and embedded approaches to FS may provide better results. The study does not provide any techniques to evaluate the continued relevance of a feature overtime as more (new) data become available for training. As more data become available, it may be necessary to periodically evaluate the degree of a feature's relevance in an effort to ensure the model is continuously able to generalise and provide predictions with a high accuracy.

Author Contributions: Conceptualization, S.O.O., P.A.O. and C.T.; investigation, J.A.A.; methodology, S.O.O. and J.A.A.; software, S.O.O.; supervision, P.A.O. and C.T.; validation, S.O.O.; writing—original draft, S.O.O. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used in this study can be sourced from: https://finance.yahoo.com/quote/AAPL/ (accessed on 10 July 2020).

Acknowledgments: The authors would like to thank the maintainers of the Pyswarms library for Particle Swarm Optimization [7]. Lester James V. Miranda, Aaron Moser, and Siobhán K. Cronin, along with all those who contributed towards the project, thank you for making this library available for use.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

LSTM	Long Short-Term Memory
BPSO	Binary Particle Swarm Optimisation
FS	Feature selection

References

1. Rostami, M.; Berahmand, K.; Nasiri, E.; Forouzandeh, S. Review of swarm intelligence-based feature selection methods. *Eng. Appl. Artif. Intell.* **2021**, *100*, 104210. [CrossRef]
2. Dhal, P.; Azad, C. A comprehensive survey on feature selection in the various fields of machine learning. *Appl. Intell.* **2022**, *52*, 4543–4581. [CrossRef]
3. Pudjihartono, N.; Fadason, T.; Kempa-Liehr, A.W.; O'Sullivan, J.M. A review of feature selection methods for machine learning-based disease risk prediction. *Front. Bioinform.* **2022**, *2*, 927312. [CrossRef] [PubMed]
4. Effrosynidis, D.; Arampatzis, A. An evaluation of feature selection methods for environmental data. *Ecol. Inform.* **2021**, *61*, 101224. [CrossRef]
5. Pearson, K.; Lee, A. Mathematical contributions to the theory of evolution. VIII. on the inheritance of characters not capable of exact quantitative measurement. Part I. introductory. Part II. on the inheritance of coat-colour in horses. Part III. on the inheritance of eye-colour in man. *Philos. Trans. R. Soc. Lond. Ser. A Contain. Pap. Math. Phys. Character* **1900**, *195*, 79–150.
6. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95-International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.
7. Miranda, L.J.V. PySwarms, a research-toolkit for Particle Swarm Optimization in Python. *J. Open Source Softw.* **2018**, *3*, 433. [CrossRef]
8. McHugh, M.L. The chi-square test of independence. *Biochem. Medica* **2013**, *23*, 143–149. [CrossRef] [PubMed]
9. Alshaer, H.N.; Otair, M.A.; Abualigah, L.; Alshinwan, M.; Khasawneh, A.M. Feature selection method using improved CHI Square on Arabic text classifiers: Analysis and application. *Multimed. Tools Appl.* **2021**, *80*, 10373–10390. [CrossRef]
10. Jahan, S.; Islam, M.S.; Islam, L.; Rashme, T.Y.; Prova, A.A.; Paul, B.K.; Islam, M.M.; Mosharof, M.K. Automated invasive cervical cancer disease detection at early stage through suitable machine learning model. *SN Appl. Sci.* **2021**, *3*, 806. [CrossRef]
11. Cañete-Sifuentes, L.; Monroy, R.; Medina-Pérez, M.A. A review and experimental comparison of multivariate decision trees. *IEEE Access* **2021**, *9*, 110451–110479. [CrossRef]

12. Mienye, I.D.; Sun, Y. A survey of ensemble learning: Concepts, algorithms, applications, and prospects. *IEEE Access* **2022**, *10*, 99129–99149. [CrossRef]
13. Raileanu, L.E.; Stoffel, K. Theoretical comparison between the gini index and information gain criteria. *Ann. Math. Artif. Intell.* **2004**, *41*, 77–93. [CrossRef]
14. Kennedy, J. The particle swarm: Social adaptation of knowledge. In Proceedings of the 1997 IEEE International Conference on Evolutionary Computation (ICEC'97), Indianapolis, IN, USA, 13–16 April 1997; pp. 303–308. [CrossRef]
15. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv* **2014**, arXiv:1412.3555.
16. Chen, C.; Dai, J. Mitigating backdoor attacks in lstm-based text classification systems by backdoor keyword identification. *Neurocomputing* **2021**, *452*, 253–262. [CrossRef]
17. Huang, W.; Liu, M.; Shang, W.; Zhu, H.; Lin, W.; Zhang, C. LSTM with compensation method for text classification. *Int. J. Wirel. Mob. Comput.* **2021**, *20*, 159–167. [CrossRef]
18. Behera, R.K.; Jena, M.; Rath, S.K.; Misra, S. Co-LSTM: Convolutional LSTM model for sentiment analysis in social big data. *Inf. Process. Manag.* **2021**, *58*, 102435. [CrossRef]
19. Gandhi, U.D.; Kumar, P.M.; Babu, G.C.; Karthick, G. Sentiment Analysis on Twitter Data by Using Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM). *Wirel. Pers. Commun.* **2021**, 1–10. [CrossRef]
20. Zhang, B.; Zou, G.; Qin, D.; Lu, Y.; Jin, Y.; Wang, H. A novel Encoder-Decoder model based on read-first LSTM for air pollutant prediction. *Sci. Total Environ.* **2021**, *765*, 144507. [CrossRef] [PubMed]
21. Nguyen, H.; Tran, K.; Thomassey, S.; Hamad, M. Forecasting and Anomaly Detection approaches using LSTM and LSTM Autoencoder techniques with the applications in supply chain management. *Int. J. Inf. Manag.* **2021**, *57*, 102282. [CrossRef]
22. Zhou, H.; Zhang, J.; Zhou, Y.; Guo, X.; Ma, Y. A feature selection algorithm of decision tree based on feature weight. *Expert Syst. Appl.* **2021**, *164*, 113842. [CrossRef]
23. Cui, X.; Li, Y.; Fan, J.; Wang, T. A novel filter feature selection algorithm based on relief. *Appl. Intell.* **2022**, *52*, 5063–5081. [CrossRef]
24. Dhiman, G.; Oliva, D.; Kaur, A.; Singh, K.K.; Vimal, S.; Sharma, A.; Cengiz, K. BEPO: A novel binary emperor penguin optimizer for automatic feature selection. *Knowl.-Based Syst.* **2021**, *211*, 106560. [CrossRef]
25. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
26. Bukhari, A.H.; Raja, M.A.Z.; Sulaiman, M.; Islam, S.; Shoaib, M.; Kumam, P. Fractional neuro-sequential ARFIMA-LSTM for financial market forecasting. *IEEE Access* **2020**, *8*, 71326–71338. [CrossRef]