### MOVING AVERAGE CROSS-STRATEGY AND DECISION TREE

# Introduction

This report analyses in detail a MATLAB-implemented moving average trading strategy for Task 1. The crossings of 7-day and 14-day moving averages by this mean-reverting trading technique at JustEat6M stock point out buy and sell signals. The strategy is implemented in a setting to pick lucrative trading opportunities.

Alongside this, we also have calculated profits/losses on each trade along with the date of trades and the remaining portfolio value for our next trade to calculate the total quantities to buy/sell shares. At last, we have plotted the stock along with 7-day and 14-day moving averages in MATLAB and as an output, we will be getting the remaining portfolio value and total buy-sell trades with the total profit or loss.

For Task 2, with 8 data points given and three attribute features along with the decision column. The model's modularity suggests discernment in the decision tree model-based choices into criterion, and parameters like MinParentSize will critically be looked at to attain the validity of the model. Detailed procedures will be able to build and validate a classification assignment model based on the decision tree. This critical calibration technique fine-tunes the underlying subtleties of a model's decision-making to ensure that it will be applied in a reliable and informed fashion for a couple of decisions related to that model.

# Data Overview and Calculations

## Task 1

We have data that underpins the daily closing pricing of JustEat6M, which is utilised in this job. The "JustEat6M.xlsx" Excel file that is attached. In this dataset, a row corresponds to a single trading day for 6 months starting from August 2020 to February 2021 and the daily closing price of the stock “Just Eat”.

## Task 2

With the dataset, we have 3 features (Stock Price, Market and News) also, we have a Decision column with Buy and Sell. For Stock Price and Market, we have up or Down or Stable. For News, we have Positive or Negative or Impartial.

Calculations for Entropy and Information Gain:

Let’s calculate the Entropy as E and information gain as I.G. for the given data set. Firstly, with the help of Decision Column,

The Entropy of (data) set S, with class C Initial Entropy for all,

Buyers: (B) 6/8 = 0.75 and sellers : (S) 2/8 = 0.25 Now,

Entropy(S)= -(𝑃𝐵𝑢𝑦) 𝑙𝑜𝑔2 (𝑃𝐵𝑢𝑦) – (𝑃𝑆𝑒𝑙𝑙) 𝑙𝑜𝑔2(𝑃𝑆𝑒𝑙𝑙) (Formula 1)

Now applying the formulas,

#### S = -((0.75) 𝑙𝑜𝑔2 (0.75) ) - ((0.25) 𝑙𝑜𝑔2 (0.25) )

= -(-0.31130)-(-0.50004)

Initial Entropy = 0.81134

Let’s find Information gain for the Stock price,

Firstly, we will find Entropy for each value of Stock price,

For “UP”:

B = 3/3 and S = 0/3, B = 1 and, S = 0,

Now using formula (1),

#### S = - ((1) 𝑙𝑜𝑔2 (1)) - ((0) 𝑙𝑜𝑔2 (0) )

#### S = -(0)-(0)

“UP” Entropy = 0

for “STABLE”:

B = 2/2 and S = 0/2, B = 1 and, S = 0,

Now using formula (1),

#### S = - ((1) 𝑙𝑜𝑔2 (1)) - ((0) 𝑙𝑜𝑔2 (0))

#### S = - (0) - (0)

“STABLE” Entropy = 0

for “DOWN”:

B = 1/3 and S = 2/3,

B = 0.3333 and S = 0.6667,

Now using formula (1),

#### S = - ((0.3333) 𝑙𝑜𝑔2 (0.3333)) - ((0.6667) 𝑙𝑜𝑔2 (0.6667))

S = -(-0.5283) -(-0.3899) “DOWN” Entropy = 0.9182

Now, that we have all individual entropy, we will do a summation of it concerning total data points,

Average Entropy in the Stock Price column, S= (3/8) (0) + (2/8) (0) + (3/8) (0.9182)

#### S= 0 + 0 + 0.34432

H (Stock Price) = 0.34432

For Information gain on the Stock Price attribute, we will subtract the Average Entropy from the Initial Entropy:

#### IG = 0.81134-0.34432

Hence, the final Information Gain for the Stock Price, IG (Stock Price) = 0.46702

Now, moving on to next column,

Let’s find Information gain for the Market,

Firstly, we will find Entropy for each value of Market,

For “DOWN”:

B = 1/2 and S = 1/2, B = 0.5 and, S = 0.5,

Now using formula (1),

#### S = - ((0.5) 𝑙𝑜𝑔2 (0.5)) - ((0.5) 𝑙𝑜𝑔2 (0.5))

#### S = -(-0.5000) - (-0.5000)

“DOWN” Entropy = 1

for “UP”:

B = 2/2 and S = 0/2, B = 1 and, S = 0,

Now using formula (1),

#### S = - ((1) 𝑙𝑜𝑔2 (1)) - ((0) 𝑙𝑜𝑔2 (0))

#### S = - (0) - (0)

“UP” Entropy = 0

for “STABLE”:

B = 3/4 and S = 1/4,

B = 0.75 and, S = 0.25,

Now using formula (1),

#### S = - ((0.75) 𝑙𝑜𝑔2 (0.75)) - ((0.25) 𝑙𝑜𝑔2 (0.25))

#### S = - (-0.31130) - (-0.50004)

“STABLE” Entropy = 0.81134

Now, that we have all individual entropy, we will do a summation of it concerning total data points,

Average Entropy in the Market column, S= (2/8) (1) + (2/8) (0) + (4/8) (0.81134)

S= 0.25 (1) + 0 + 0.40567 H (Market) = 0.65567

For Information gain on the Market attribute, we will subtract the Average Entropy from the Initial Entropy:

#### IG = 0.81134-0.65567

Hence, the final Information Gain for the Market, IG(Market) = 0.15567

Now, moving on to next column,

Let’s find Information gain for the News,

Firstly, we will find Entropy for each value of News,

For “IMPARTIAL”:

B = 1/1 and S = 0/1, B = 1 and S = 0

Now using formula (1),

#### S = - ((1) 𝑙𝑜𝑔2 (1) ) - ((0) 𝑙𝑜𝑔2 (0) )

S = (0) – (0) “IMPARTIAL” Entropy = 0

for “NEGATIVE”:

B = 3/4 and S = 1/4,

B = 0.75 and, S = 0.25,

Now using formula (1),

#### S = - ((0.75) 𝑙𝑜𝑔2 (0.75)) - ((0.25) 𝑙𝑜𝑔2 (0.25) )

S = - (-0.31130) - (-0.50004) “NEGATIVE” Entropy = 0.81134

for “POSITIVE”:

B = 2/3 and S = 1/3,

B = 0.6667 and, S = 0.3333,

Now using formula (1),

#### S = - ((0.6667) 𝑙𝑜𝑔2 (0.0.6667) ) - ((0.3333) 𝑙𝑜𝑔2 (0.3333) ) S = - (- 0.3899) - (-0.5283)

“POSITIVE” Entropy = 0.9182

Now, that we have all individual entropy, we will do a summation of it concerning total data points,

Average Entropy in the News column,

#### S= (1/8) (0) + (4/8) (0.81134) + (3/8) (0.9182)

S= 0 + 0.40567 + 0.3443 H (News) = 0.74997

For Information gain on the News attribute, we will subtract the Average Entropy from the Initial Entropy:

#### IG = 0.81134 - 0.74997

Hence, the final Information Gain for the News, IG(News) = 0.06137

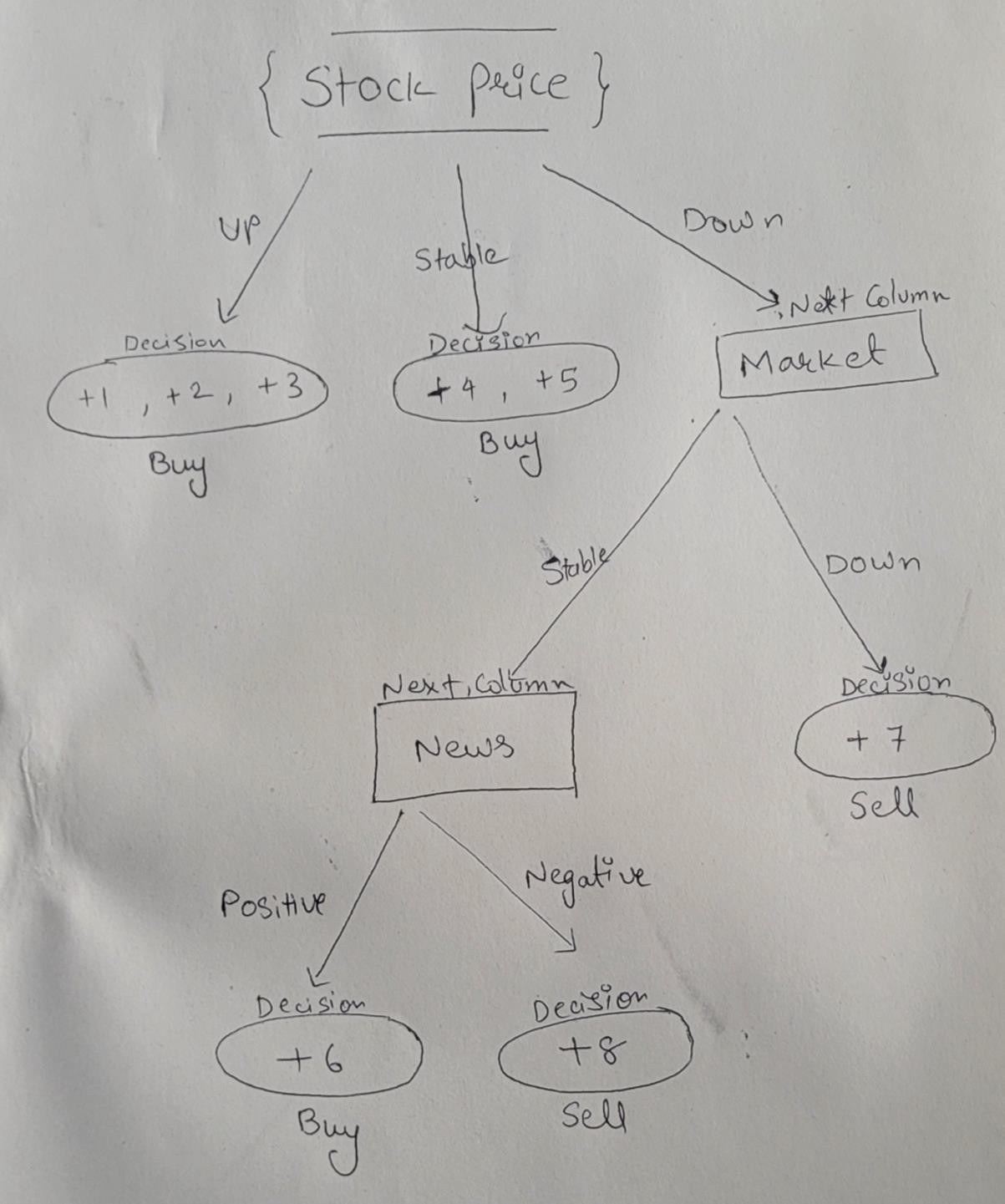
Finally, we have all the Entropy values for our data and individual Information Gain for each. Initial Entropy = 0.81134

H (Stock Price) = 0.34432 H (Market) = 0.65567

H (News) = 0.74997

IG (Stock Price) = 0.46702 IG(Market) = 0.15567

IG(News) = 0.06137



(Decision Tree)

The above is the hand-written decision tree based on the Entropy and Information Gain we got. By looking at the data we have the highest entropy in News: H (News) = 0.74997 and the highest Information Gain in Stocks: IG (Stock Price) = 0.46702, thus, we started our tree from Stocks till news with the decision nodes Buy/Sell along with the Market Column.

# Code Implementation

## Task 1

Import Data: the first code reads the data that was imported from a file, done using the command ‘readtable’ and containing all Excel data. From it, dates are fetched with their closing price per date and saved in 2 different variables.

Code:

data = readtable('JustEat6M.xlsx'); dates = data.Date;

closingPrice = data.Close;

Calculating the Moving Average: Given below is the MATLAB script which will be used to calculate the moving average for 7 days and 14 days of the closing price. This moving average will help estimate a trend in stock movements.

Code:

ma7Days = movmean(closingPrice, 7); ma14Days = movmean(closingPrice, 14);

Trading Strategy: The trading strategy is a critical part based on moving average crossovers. The script traverses through the data and creates buy signals in case of a crossover of the 7-day moving average above the 14-day moving average and opposite with the sell signals when it happens. It records each transaction, and profit/loss is computed for sell transactions.

Code:

budget = 1000000; % £1 million is up for grabs portfolio = 0; % At first, no shares are owned

transactions = zeros(size(dates)); % Records buy and sell transactions total\_profit\_loss = 0; % Initialize total profit/loss as 0

Buy\_count = 0; % Initialize total buy count as 0 in order to add in it Sell\_count = 0; % Initialize total sell count as 0 in order to add in it totalPortfolioValue = 1000000; %Counting the value of portfolio for P&L

for i = 15:length(dates) % Start from the 15th day to ensure enough data

if ma7Days(i) > ma14Days(i) && ma7Days(i-1) <= ma14Days(i-1) % Buy signal numOfSharesToBuy = floor(budget / closingPrice(i));

if numOfSharesToBuy > 0 buyPrice = closingPrice(i);

portfolio = portfolio + numOfSharesToBuy;

budget = budget - numOfSharesToBuy \* closingPrice(i); transactions(i) = numOfSharesToBuy;

Buy\_count= Buy\_count+1;

fprintf('Bought %d shares on %s at price of %.2f\n', numOfSharesToBuy, char(dates(i)), closingPrice(i));

end

elseif ma7Days(i) < ma14Days(i) && ma7Days(i-1) >= ma14Days(i-1) % Sell signal if portfolio > 0

% Calculating the profit/loss before selling the stock profit\_loss = (closingPrice(i)-buyPrice) \* portfolio;

total\_profit\_loss = total\_profit\_loss + profit\_loss; % Updating total profit/loss here budget = budget + portfolio \* closingPrice(i);

Sell\_count=Sell\_count+1;

totalPortfolioValue = totalPortfolioValue+profit\_loss;

fprintf('Sold %d shares on %s at price of %.2f.\n', portfolio, char(dates(i)), closingPrice(i)); fprintf('P&L: £%.2f\n', profit\_loss);

fprintf('Portfolio value: £%.2f\n',totalPortfolioValue); transactions(i) = -portfolio;

portfolio = 0; end

end end

Total Profit/Loss Calculation: This, in other words, is obtained by the addition of different profits/losses obtained from all the sell transactions. The count of Trades: These are the values of total transactions that happened on the buy side as well as sell side.

Code:

fprintf('\nTotal profit/loss: £%.2f\n', total\_profit\_loss);

fprintf('\nTotal Buy Orders: %.0f\n', Buy\_count); fprintf('\nTotal Sell Orders: %.0f\n', Sell\_count);

Last but not least, the script produces a graphical representation of JustEat6M stock closing prices vs. 7 and 14 moving averages, so that it is possible to get some understanding of whether the strategy would perform well in different timings.

Code:

plot(dates(15:end), closingPrice(15:end),'LineWidth',2); hold on;

plot(dates(15:end), ma7Days(15:end),'-g'); hold on;

plot(dates(15:end), ma14Days(15:end),'-r');

legend('Closing Prices for JustEat', '7Days Moving Average', '14Days Moving Average'); xlabel('Date');

ylabel('Price');

title('JustEat with Moving Average Crossover');

## 

## Task 2

The first definition in the code is that of four arrays that will carry the category data: stockPrice, market, news, and decision. These will stand for things that, for example, have changes in stock prices and the condition of the market; through the form of the news and acts happening, they are just about to act in response to stimuli.

Code:

%% Define the data

stockPrice = ["Up"; "Up"; "Up"; "Stable"; "Stable"; "Down"; "Down"; "Down"];

market = ["Down"; "Up"; "Stable"; "Up"; "Stable"; "Stable"; "Down"; "Stable"];

news = ["Impartial"; "Negative"; "Positive"; "Negative"; "Negative"; "Positive"; "Positive"; "Negative"]; decision = ["Buy"; "Buy"; "Buy"; "Buy"; "Buy"; "Buy"; "Sell"; "Sell"];

Data combining into a table: A table called dataTable is created by combining the data from the arrays. A factor is represented by each column in the table.

Code:

dataTabe = table(stockPrice, market, news, decision);

Decision Tree object: Using the fitctree function, the informative dataTable creates an object of the decision tree model, and the prediction of the outcome is made in the tree1 object according to the criteria.

Code:

%% Creation of a decision tree

tree1 = fitctree(dataTabe, 'decision');

Display the decision tree as text: The decision tree (tree1) is displayed as both a graphical representation and text format.

Code:

%% Display Tree 2

disp('Decision Tree1:'); view(tree1, 'Mode','graph')

view(tree1,"Mode", 'text')

Define the target values: The decision column from the dataTable is extracted as the target variable for model training.

Code:

%% Define the target

target = dataTabe.decision;

Initialize MinParentSize: A variable named MinParentSize is initialized to 1. This parameter controls the minimum number of observations required to split a node during the construction of the decision tree.

Code:

%% Initialize MinParentSize

MinParentSize = 1;

Find the maximum value for MinParentSize with zero loss: A loop iterates to find the maximum value of MinParentSize where the decision tree (tree2) achieves zero classification loss. Inside the loop: A decision tree model (tree2) is created with the current MinParentSize. The loss of the model is calculated using the loss function. If the loss is zero, the loop breaks. Otherwise, MinParentSize is incremented.

Code:

%% Find the maximum value for MinParentSize with 0 loss

while true

tree2 = fitctree(T(:, 1:3), target, 'MinParentSize', MinParentSize);

if loss(tree2, T(:, 1:3), target) == 0

break;

else

MinParentSize = MinParentSize + 1;

end

end

Display results: After the loop, the maximum MinParentSize achieving zero loss is displayed along with the decision tree (tree2) represented both in text and graphical formats.

Code:

%% Display Tree 2 and maximum minparentsize with zero loss

disp(['Maximum MinParentSize for zero loss: ', num2str(MinParentSize

disp('Decision Tree2:'); view(tree2, "Mode", 'text');

view(tree2, 'Mode', 'graph');

# Output Analysis

## Task 1

The output of the script provides in-depth information on each buy and sell transaction. For each transaction, the following details are displayed:

* + - shares acquired or sold
    - transaction date
    - price
    - profit or loss
    - portfolio value
    - total profit or loss

The moving averages and the JustEat6M stock closing prices are also shown in figure 1 for easy visual scrutiny.

Output:

Bought 119 shares on 10-Sep-2020 at price of 8384.00

Sold 119 shares on 15-Sep-2020 at price of 8422.00. P&L: £4522.00

Portfolio value: £1004522.00

Bought 119 shares on 18-Sep-2020 at price of 8428.00 Sold 119 shares on 05-Oct-2020 at price of 8732.00.

#### P&L: £36176.00

Portfolio value: £1040698.00

Bought 117 shares on 13-Oct-2020 at price of 8842.00 Sold 117 shares on 23-Oct-2020 at price of 9174.00.

#### P&L: £38844.00

Portfolio value: £1079542.00

Bought 125 shares on 03-Nov-2020 at price of 8636.00 Sold 125 shares on 11-Nov-2020 at price of 8414.00.

#### P&L: £-27750.00

Portfolio value: £1051792.00

Bought 130 shares on 18-Nov-2020 at price of 8030.00 Sold 130 shares on 19-Nov-2020 at price of 8052.00.

#### P&L: £2860.00

Portfolio value: £1054652.00

Bought 133 shares on 27-Nov-2020 at price of 7920.00 Sold 133 shares on 03-Dec-2020 at price of 7790.00.

#### P&L: £-17290.00

Portfolio value: £1037362.00

Bought 130 shares on 14-Dec-2020 at price of 7948.00 Sold 130 shares on 15-Dec-2020 at price of 7860.00.

#### P&L: £-11440.00

Portfolio value: £1025922.00

Bought 132 shares on 16-Dec-2020 at price of 7758.00 Sold 132 shares on 18-Dec-2020 at price of 7970.00.

#### P&L: £27984.00

Portfolio value: £1053906.00

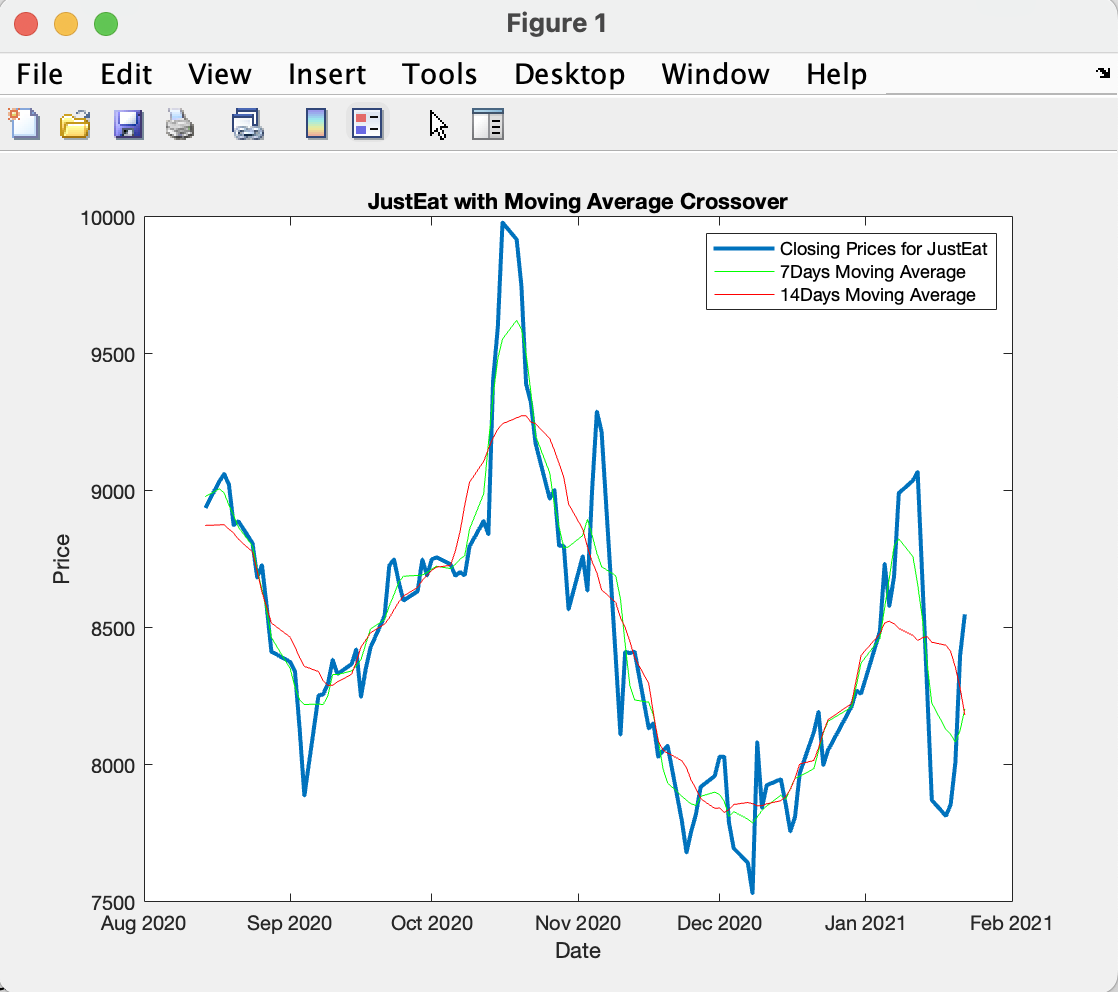
Bought 120 shares on 05-Jan-2021 at price of 8734.00 Sold 120 shares on 14-Jan-2021 at price of 8280.00.

#### P&L: £-54480.00

Portfolio value: £999426.00

Bought 116 shares on 22-Jan-2021 at price of 8550.00

Total profit/loss: £-574.00 Total Buy Orders: 10 Total Sell Orders: 9



(Figure 1)

We got a total loss of -574.00 pounds during our 6-month back testing journey in Just Eat Stock and at last, we still have one buy position open due to the limitation of data we have which is till February 2021.

## Task 2

The output provides descriptions of two decision trees: Tree 1 and Tree 2

For Tree 1, there is only one node in this decision tree. Regardless of the supplied attributes, it predicts the class to be "Buy".

The maximum MinParentSize value that achieves zero loss during model training is 1.

This decision Tree 2 is more complex, with more nodes. The stock price is confirmed by the root node, or node 1. If the stock price is "Down," Node 2 is reached; if the stock price is "Stable" or "Up," Node 3 is achieved. Node 2 looks at the overall health of the market. If it is "Down," it goes to node 4, and if it is "Stable," it goes to node 5. "Sell" is forecast if not. Node 3 predicts "Buy" immediately. Node 4 forecasts "Sell" right away. Node 5 evaluates the news's tone. If it's "Negative," it goes to node 6, and if it's "Positive," it goes to node 7. "Buy" is forecasted if not. “Sell" is the clear prediction made by Node 6. Node 7 predicts "Buy" immediately.

Output:

Decision Tree1:

Decision tree for classification

7 class = Buy

Maximum MinParentSize for zero loss: 1 Decision Tree2:

Decision tree for classification

7 if stockPrice=Down then node 2 elseif stockPrice in {Stable Up} then node 3 else Buy

7 if market=Down then node 4 elseif market=Stable then node 5 else Sell

7 class = Buy

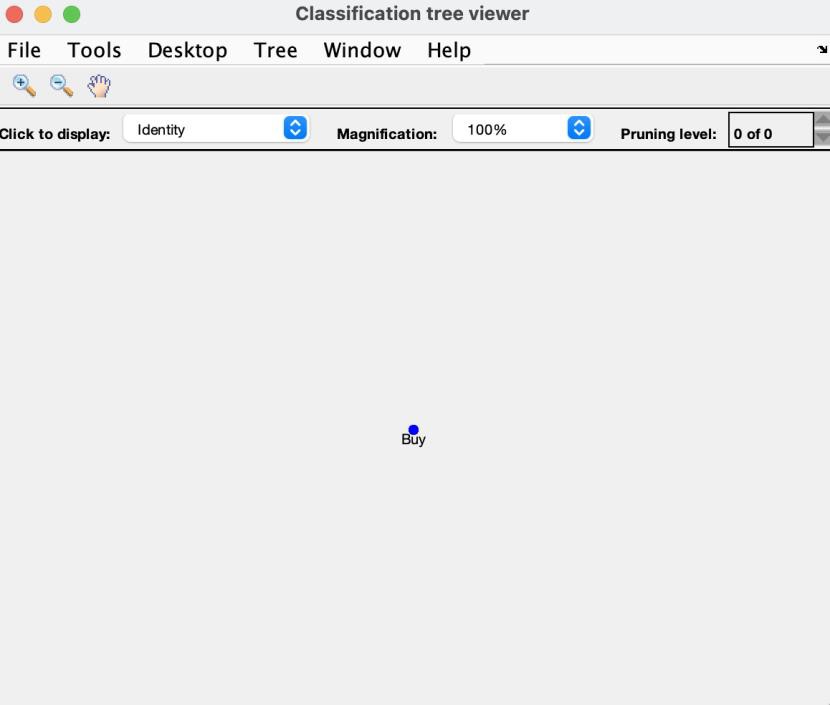
7 class = Sell

7 if news=Negative then node 6 elseif news=Positive then node 7 else Buy

7 class = Sell

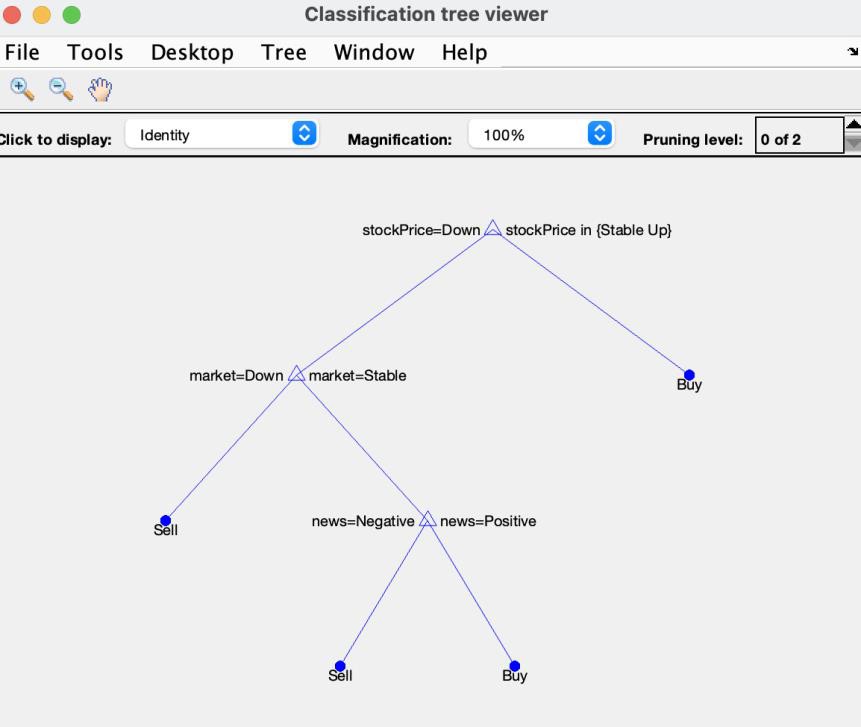
7 class = Buy

### Tree 1



(Figure 2)

### Tree 2



(Figure 3)

# Conclusion

Task 1 concludes that the use of a trading strategy, such as moving averages, points to the great importance of technical indicators in making the right trading decisions. The study gives examples of how crossovers with moving averages in this method may bring benefit in profitable deals which is not for this stock in this timeframe, but a lack of information and case studies shows a strong effectiveness of the method. Sharpening of the performance of the method could be made through more studies and improvements. It could be custom-tailored in such a way that it is fit for changing market conditions. In one word, the code does proper justice to make understanding moving averages used concerning algorithmic trading strategies available today.

Task 2: Classify stock trading decisions using a decision-tree-based classification model that has the factors associated with the movement of stock prices, the market factors, and the sentiment from the news. First, the input data are defined, and after that, it is combined into a table which creates a decision tree model (tree1). The loop then iterates with different MinParentSize from 1 to 6 to get the maximum that results in zero classification loss. The output has given two decision trees: tree1 predicts 'Buy' for all cases, while tree2 gives a detail of the demarcation for the decision-making based on the input features. From the above, one can notice that there are various combinations based on which the model tries to arrive at a decision - whether to buy the stock or not. Inference from the output is that such a model of the decision tree could learn the decision rules from the data given. This gives insight into factors that affect stock trading decisions. Given the preceding output, it may be inferred that starting from an appropriate MinParentSize value, the model could get a zero-loss value, and in practice, it means the model can predict accurately.