CF963 Assignment

Reg no: 1802559

Task 1:

The first task is to choose and use a trading strategy to buy and sell shares. For this purpose the moving averages strategy was selected for buying stocks at a given price before dumping them all onto the market when the trend of prices changed to a decreasing stock price.

The algorithm (see appendix) for the solution loops of over n rows until the end of the file is reached. A moving window is then created that over each new row moves the window one space through the data space in order to tally the closing prices on the stock over seven days or one week to be averaged at the end of that week. The main part of the algorithm responsible for working out profits works in two parts. The first part of the algorithm is responsible for buying stocks while the price is increasing, and the second is responsible for selling those stocks when the price dips below a certain threshold compared between the average last stock price and the present average stock price.

The first part of the algorithm works by appending to a list the trading prices for bought stocks. This is an essential measure to work out the profit by comparing the price of bought stocks with stocks to be sold on the market. This comparison is done by using an equation for net gain of stocks, to work out the percentage positive or negative change in the value of the bought stocks and can be expressed with the equation:

(new stock price - old stock price) / old stock price.

When multiplied against an investment_step i.e. batch of investment it is possible to see how much a stock has appreciated or depreciated since buying. In order to work out the profit the profit is taken as 0 and then profits calculated from the aforementioned process are added to it. In the end negative profits when added to the existing profit will move the profits variable away from 0 or higher profit and vice versa if they are positive.

The final part of the algorithm is the cleanup of the variables used to generate a profit evaluation. This includes resetting the list to hold bought stock prices and then updating the old average with the new average for future evaluation of stock prices, and moving the indexes for the window for the next iteration of the algorithm. In a perfect world the algorithm would buy when stock prices go up and sell immediately as stocks tend to decrease in price. However in practice this doesn't always work out, and is exacerbated by using such a long window of seven days. If a smaller window was used, then the profitability of the algorithm could be improved. The reason for this is that the actual price may peak or bottom outside the mean average i.e. that the actual value traded at may be much higher or lower than the mean price.

The performance of the algorithm is best judged by working out what the overall profit is for the trading session. In the given application the result was an overall loss, but as aforementioned changing the window size could lead to much better performance.

Task two:

a)

① PROFIT FUNCTIONS

GIVEN $P = 120 - Q$; $C_1 = 10 + 3q_1$

$C_2 = 12 + 6q_2$; $Q = q_1 + q_2$

$$\pi_1 = q_1(120 - (q_1 + q_2)) - (10 + 3q_1)$$

$$\pi_2 = q_2(120 - (q_1 + q_2)) - (12 + 6q_2)$$

② Quantities

i) $\frac{\partial \pi_1}{\partial q_1} = 1(120 - (q_1 + q_2)) + q_1(-1) - 3 = 0$ solve for q_1

$$120 - (q_1 + q_2) - q_1 - 3 = 0$$

$$120 - 3 - q_1 - q_2 - q_1 = 0$$

$$117 - q_1 - q_2 - q_1 = 0$$

$$q_1^* = \frac{117 - q_2}{2}$$

ii) Substitute into 2nd PROFIT FUNCTION, FIRM 1's quantity

$$q_2\left(120\left(\frac{117 - q_2}{2} + q_2\right) - (12 + 6q_2)\right)$$ USING PRODUCT RULE

$$q_2' \cdot \frac{1}{2}\left(120 - \left(\frac{117 + q_2}{2}\right)\right) + q_2\left(-\frac{1}{2}\right)$$

$$120 - \left(\frac{117 + q_2}{2}\right) - \frac{q_2}{2} - 6$$

$$120 - 6 - \frac{117}{2} - \frac{q_2}{2} - \frac{q_2}{2} = 0$$

$$114 - \frac{117}{2} - q_2 = 0$$

$$114 - \frac{117}{2} = q_2$$

$$\frac{228}{2} - \frac{117}{2} = q_2$$

$$\frac{111}{2} = q_2$$

$55.5 = q_2$, $Q_1 = \frac{117 - q_2}{2}$ ∴

$$Q_1 = \frac{117 - 55.5}{2}$$

$Q_1 = 30.75$; $Q_2 = 55.5$

(3) PROFITS BASED ON QUANTITIES

$\pi_1 = q_1(120 - (q_1 + q_2)) - (10 + 3q_1)$

$30.75 \times (120 - (55.5 + 30.75)) - (10 + 3 \times 30.75)$

$\pi_1 = \$935.5625$

$\pi_2 = 55.5(120 - (55.5 + 30.75)) - (12 + 6 \times 55.5)$

$\pi_2 = 1528.125$

∴ FIRM 1 PROFIT = 935.5625
FIRM 2 PROFIT = 1528.125

(4) SURPLOSSES

$CS = 3719.53125$

$TS = \pi_1 + \pi_2 + CS$

$TS = 6183.21875$

$Q = q_1 + q_2$

$Q = 86.25$

$CS = \frac{1}{2} \times 86.25 \times 86.25$

B)

$$\pi_1 = q_1[120 - q_1 - q_2] - 10 - 3q_1$$

$$\pi_1 = q_1\left[120 - q_1 - 57 - \frac{q_1}{2}\right] - 10 - 3q_1$$

$$\frac{\partial \pi'}{\partial q_1} = 120q_1 - q_1^2 - 57q_1 - \frac{q^2}{2} - 10 - 3Q$$

$$\frac{\partial \pi'}{\partial q_1} = 120 - 2q_1 - 57 - 3 = 0$$

$$120 - 60 = q_1$$

$$60 = q_1$$

$$57 - 30 = 27$$

REACTION PR:

Equilibrium quantity is 60.

Task 3:

The Nash equilibrium couldn't be applied for this task but what was achieved was a program that allows bidding to take place between two different bidders using the generalised second place auction. The algorithm used awards the winning bidder the price of the competitor and with the second place slot a price of 0. It also takes input for the actual values that the advertising firms are willing to pay versus the price they paid to work out the payoffs for either firm respectively.

Task four:

This paper examines the work of Singh in introducing game theory through the application to electric power markets as an example. It applies lessons learnt from Maria Kyropoulou in her lectures on game theory, and particular the Nash equilibrium, cooperative and non-cooperative games and Cournot duopoly.

In his paper, Singh[1] introduces to the reader concepts on game theory and applies them to the domain of energy markets, specifically for energy production. He begins by partitioning the concept of a game into two typologies: co-operative or non-cooperative. For these games they can be further classified into zero sum or non zero sum games. The two are differentiated further as games where only one person can win outright and games where the winnings can be distributed in some mutually beneficial way, for which the Nash equilibrium can be applied. For the example of non-cooperative games there are two kinds of formats: normal and extensive. The normal and extensive varieties can now be examined further.

A normal game has a set of players, a set of choices and a set of payoffs for given strategies [1]. The end payoff depends on not only the strategies of the first player but also that of the second, third, fourth or so on. In this case the classic case of applying the Nash equilibrium in this case is the prisoners dilemma.

In the example given by Singh [1] contracts for differences (CFDs) exist to get a price that is averaged or buffered against market forces (prices). The difference comes from the difference in actual price versus that of a contract price for the end consumer and the supplier. This leads to an agreed price between the buyer of energy and the seller of energy in an energy market. In such a market a firm exists to exert its market power i.e. the ability of a market participant to raise prices above the competitive level.

Output (MW)		Generator B High	Generator B Low	
Generator A	High	75 75	75 20	A's output B's output
	Low	20 75	20 20	A's output B's output

Figure 1. Output decisions of A and B

Price ($/MWh)		Generator B High	Generator B Low	
Generator A	High	40	45	
	Low	45	150	

Figure 2. Prices corresponding to production decisions.

In the given example (see above) two generators exist, generator A and generator B. In this case the production of energy by A and B must be set to maximise the profits of the energy company responsible for generating the energy. This is the strategic decision for generators A and B and suggests using the Cournot model to create the environment for a duopoly, thus maximising the profits of both generators. In the example each generator must choose between output decisions high or low. The payoff for this case is best if

both generators select a low output in megawatts.

Profit ($)		Generator B: High	Generator B: Low	
Generator A	High	2250	2625	A's profit
		2250	700	B's profit
	Low	700	2800	A's profit
		2625	2800	B's profit

Figure 3. Profits without cfd [1]

Profit ($)		Generator B: High	Generator B: Low	
Generator A	High	2250	2575	A's profit
		2250	650	B's profit
	Low	650	1700	A's profit
		2575	1700	B's profit

Figure 4. Profits with contract (cfd) for 30 mw [1]

Profit ($)		Generator B: High	Generator B: Low	
Generator A	High	2250	2475	A's profit
		2250	550	B's profit
	Low	550	-500	A's profit
		2475	-500	B's profit

Figure 5. profits with cfd for 10 mw. [1]

If significant price changes occur then this would lead to a reevaluation of the amount of production as in the two figures above. In this example there is an agreed price for the CFD that is beneficial for both generators. In all three figures (3-5) a Nash equilibrium is achieved. This is because neither generator has an incentive to change the amount of production because both parties gain the best profit for themselves, and all other alternatives are less optimal for both parties. When payoffs change like in the above figure 4 a revaluation must take place so that each firm chooses a strategy that is most beneficial for itself.

However the paper can be criticised for its over simplification of oligopolies, and limited or no detail on a number of key concepts. This includes some of the intricacies in dominant strategies, and does not seek to quantify demand functions for a Cournot duopoly and does not explain the application of the Bertrand model in a great amount of detail. A Cournot duopoly could be specified if in figure 1 a greater range of energy production was allowed so that the amount supplied could be specified with further fidelity, rather than a choice between two functions high or low. In such a market the two generators of energy would have to judge the best quantity of energy to be defined so that both firms can monopolise the market, rather than the Bertrand model which is based just on pricing alone. The Bertrand model on the other hand can be applied directly to the example given based on price, and not quantity, leading to a prisoners dillema situation where both firms will compete to give the best price unless they cooperate to charge a price that benefits both

companies mutually. This means that in order to reach the best mutual price both companies must announce the prices that they are going to charge. This means setting some kind of equilibrium so that the prime strategies of both companies lead to a monopolisation of the market. For this case there are a number of possible outcomes. The first outcome is that the price of firm 1 is greater than the monopoly price. In this case the second firm will monopolise the revenue from the market. In the second case the firm could charge a price below the monopoly price but this would lead the first company to undercut the market at until a point where the price becomes uncompetitive given the costs to produce a good. This leaves two possibilities: the first is that the second firm charges att cost or they sell at a loss to monopolise the market by producing at a lower cost than the cost of production. In this case it may lead to a situation where the competing firms can be priced out of the market by a larger company bullying them out of the market. In the given example the best strategy of both firms is to try to monopolise the energy market by reaching a Nash equilibrium that gives more of an incentive to charge a monopolistic price than to cut the price, assuming that a larger firm does not exist that can simply bully out other firms from the equation. In a Cournot model the outcome would be different, based upon the production of the firms.

References:

[1] H. Singh, "Introduction to game theory and its application in electric power markets", *IEEE Computer Applications in Power*, vol. 12, no. 4, pp. 18-20, 1999. Available: https://0-ieeexplore-ieee-org.serlib0.essex.ac.uk/stamp/stamp.jsp?tp=&arnumber=795133&tag=1. [Accessed 23 March 2019].

Appendix:

For ease of reference the code written to achieve the goals of the first task is appended below. For further information consult the files attached to this report.

```
csv_file_table = readtable('AAL.csv');
column = 2;
total = 0;
average = 0;
row_top = 1;
row_bottom = 7;
window_size = 7;
rows = height(csv_file_table)-6;
investment = 1000000;
initial_investment = 1000000;
investment_step = 100000;
list_of_buying_prices = [];
profit = 0;
format long
for frame = 1:rows
    % sum over the window to get the total to be used for average
    for row = row_top:row_bottom
        table = csv_file_table(row,column);
        total = total + table.('Var2');
    end
    %calculate present average to be compared with previous average
    tmp_average = total/window_size
    current_price = getCurrentPrice(row_bottom,column,csv_file_table);
    bos = buyorsell(tmp_average,average)

    switch bos
        case 'buy'
```

```
        %buy - add to cell price bought at with default step of £100k
        % investment
        if(investment>investment_step)
            list_of_buying_prices = [list_of_buying_prices,current_price];
            investment = investment - investment_step
        else
            disp("cannot afford to invest any more")
        end
    case 'sell'
        %sell - compare revenue from buying to selling and update
        %investment
        if(~isempty(list_of_buying_prices))
            % work out net change in price and then multiply investment
            % step in order to work out the real change in pounds.
            % iterate through a list of prices
            for k=1:length(list_of_buying_prices)
                change_in_value = (((current_price - list_of_buying_prices(k))/
list_of_buying_prices(k))*investment_step);
                investment = investment + (investment_step + change_in_value);
                fprintf('change_in_value %.4f \n', change_in_value);
                fprintf('current bank account: %.4f \n',investment);
                profit = profit + change_in_value;
                fprintf('profit: %.4f ', profit)
            end
            % at end of sale setlist of buying prices to 0
            list_of_buying_prices = [];
        else
            disp("cannot sell as nothing is invested ")
        end
  end

  % at end of buy or sell evaluation set the new average to tmp_average
  % for comparison and reset total. Increment window with row_top+1 and
  % row_bottom+1
  average = tmp_average;
  total = 0;
  row_top = row_top+1;
  row_bottom = row_bottom+1;
end

% get and return the current price.
function [return_price] = getCurrentPrice(row_bottom,column,csv_file_table)
  table = csv_file_table(row_bottom,column);
  return_price =  table.('Var2');
end

% decide whether to buy or  sell given new average larger than average.
function [buy_or_sell] = buyorsell(tmp_average,average)
  if(tmp_average>average)
    buy_or_sell = "buy";
  else
    buy_or_sell = "sell";
  end
end
```