GROUP WORK PROJECT # 2
GROUP NUMBER: 6442

MScFE 660: RISK MANAGEMENT

FULL LEGAL NAME	LOCATION (COUNTRY)	EMAIL ADDRESS	MARK X FOR ANY NON-CONTRIBUTING MEMBER
Marin Stoyanov	Bulgaria	azonealerts@gmx.com	
Prabhdeep Kaur	India	prabhdeep089kaur@gmail.com	

Statement of integrity: By typing the names of all group members in the text boxes below, you confirm that the assignment submitted is original work produced by the group (excluding any non-contributing members identified with an "X" above).

Team member 1	Marin Stoyanov
Team member 2	Prabhdeep Kaur
Team member 3	

Use the box below to explain any attempts to reach out to a non-contributing member. Type (N/A) if all members contributed.
Note: You may be required to provide proof of your outreach to non-contributing members upon request.

N/A

Model Development

Forward algorithm's pseudocode:

1.) $For\ each\ state\ at\ the\ very\ beginning\ (\ let's\ call\ it:\ time\ 1\ or\ just\ \ t_1)$
$initialize\ the\ forward\ variable\ \alpha$
2.) $Start\ a\ cycle\ from\ t_2\ to\ T:$
$considering\ the\ transition\ probabilities,\ the\ previous\ \ \alpha_{t-1} and\ the$
$emission\ probabilities,\ then\ update\ \alpha_t$
3.) $Sum\ all\ the\ last\ \alpha\ values\ and\ derive\ the\ probability\ of\ the\ observed$
$sequence$

Backward algorithm's pseudocode:

1.) $At\ the\ last\ time\ step\ T:\ initialize\ the\ backward\ variable:\ \beta$
2.) $Considering\ the\ emission\ probabilities,\ the\ transition\ probabilities$
$and\ next\ \beta_{t+1},\ update\ \beta_t$
3.) $Sum\ the\ first\ \beta\ values\ to\ derive\ the\ final\ probability\ of\ the$
$observed\ sequence$

Baum-Welch Algorithm:

The Baum-Welch algorithm is also termed as Expectation-Maximization (EM) algorithm. It is generally used to find the unknown parameters of the Hidden Markov Model (HMM).

Pseudocode-

Step-1: Initialization

- Initialize the initial state distribution $\boldsymbol{\pi}$ and the two matrices, matrix A- transition probabilities matrix, matrix B- emission probabilities matrix.
- A small threshold ϵ is chosen for convergence.

Step-2: Expectation step

- For each observation sequence O, computing the forward probabilities $\boldsymbol{\alpha}$.

$$\alpha_t(i) = P(O_1, O_1, \ldots\ldots\ldots, O_t, S_t = i \mid \lambda)$$

- For each observation sequence O, computing the forward probabilities **β**.

$$\beta_t(i) = P(O_{t|1}, O_{t|2}, \ldots\ldots\ldots\ldots, O_T, S_t = i \mid \lambda)$$

Where: $\lambda = (A, B, \pi)$

Step-3: Maximization step

The model parameters will be updated using forward and backward probabilities-

- Expected no. of transitions from state i to state j is given by:

$$\xi_t(i,j) = \frac{\alpha_t(i) A_{ij} B_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i) A_{ij} B_j(O_{t+1}) \beta_{t+1}(j)}$$

- Expected no. of times model is in state i at time t is given by:

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

- Updated transition probability is given by:

$$A_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

- Updated emission probability is given by:

$$B_j(k) = \frac{\sum_{t=1}^{T} \gamma_t \cdot 1_{[O_t = k]}}{\sum_{t=1}^{T} \gamma_t(j)}$$

- Updated initial state distribution is given by:

$$\pi_i = \gamma_1(i)$$

For next step, check for the convergence, that is, stop the process if the log likelihood of the observation sequence is less than ϵ else repeat step-3.

The toy example for Baum-Welch algorithm is-

Let,

S be the two states = {1,2}

O be the two observations = {A,B}
O be the observation sequence = {A,B,A}
Initial transition probability matrix,

$$A = \begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix}$$

Initial emission probability matrix,

$$B = \begin{bmatrix} 0.5 & 0.5 \\ 0.6 & 0.4 \end{bmatrix}$$

Initial state distribution,

$$\pi = [\,0.6 \;\; 0.4\,]$$

For the above toy example, the Baum-Welch algorithm is implemented in python and the updated model parameters results are observed as follows-

```
Updated A: [[5.28642688e-10 9.99999999e-01]
 [1.00000000e+00 3.82364337e-62]]
Updated B: [[5.28642688e-10 9.99999999e-01]
 [1.00000000e+00 3.34087122e-44]]
Updated pi: [3.85651997e-20 1.00000000e+00]
```

Table.1. Results from Baum-Welch algorithm applied to the toy example

The Baum-Welch algorithm maximizes the likelihood of the observed sequence and updates the model parameters.

Bull Regimes

Bull Regimes in financial markets refers to the period where the market experiences upward trends due to rising prices, investor optimism or overall positive market movements.

Bear Regimes

Bear Regimes in financial markets refers to the period where the market experiences downward trends due to falling prices, investor pessimism or overall negative market movements.

Stagnant Regimes

Stagnant regimes refers to the periods where the states remain relatively unchanged. Identifying these regimes are crucial for investors to make informed decisions. Stagnant regimes holds different meaning for different data,

1. In Financial markets, the stagnant regimes are the period of low market volatility and minimum price movement.
2. In economic indicators, stagnant regimes are the periods where GDP growth, unemployment rates, inflation or other economic indicators remain constant.
3. In time series data, stagnant regimes are the periods where data shows minimum fluctuations.

We used the spot crude oil prices data series (WTISPLC) in order to showcase and to identify the periods of bull, bear and stagnant regimes. The following line chart is observed-

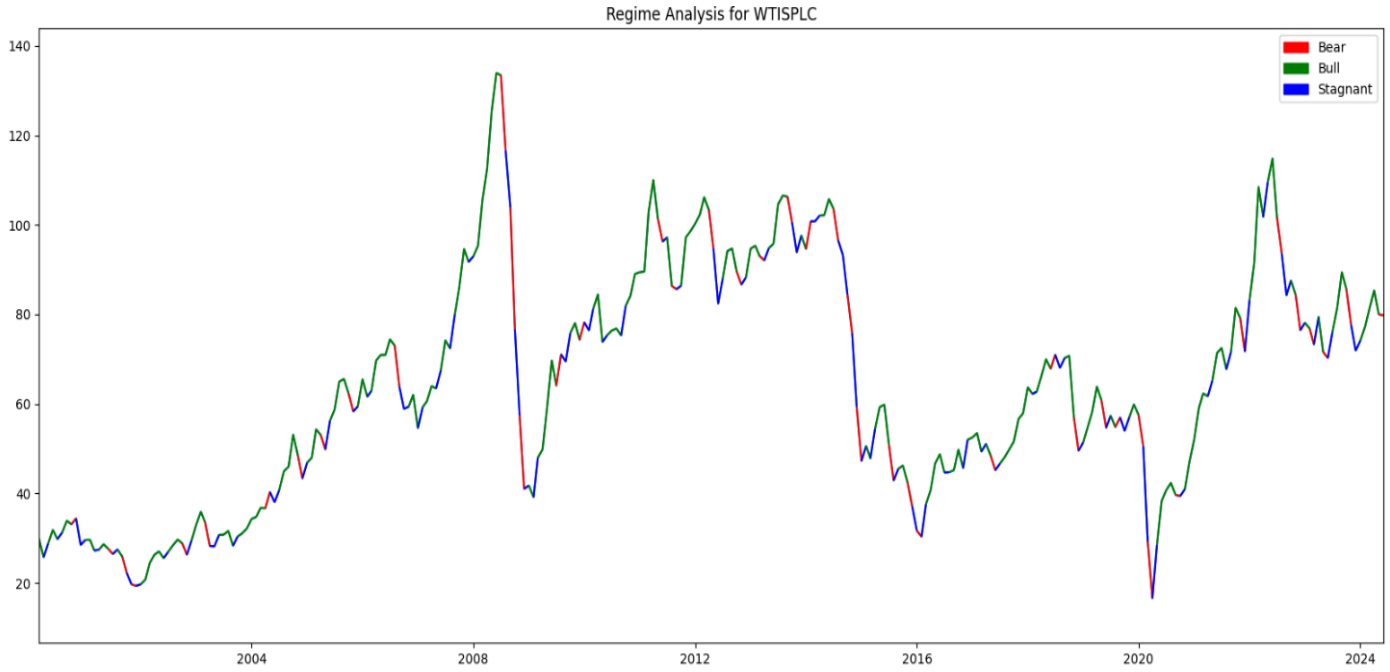

Fig.1. Regime analysis

The above plot shows the crude oil prices after the year 2000. Regime analysis is done on the prices of crude oil and it is observed on the line chart that the red color shows the period of bear regimes while the green color on the plot indicates the time period when the crude oil prices faced bull regime and for the time period when the prices of the crude oil were following stagnant regime is indicated by the blue color on the plot.

Hidden Markov Model

The hidden markov model (HMM) refers to a system which shows the transition between the set of hidden states in a manner characterized by the first order markov process and each hidden state produces an output which is observable according to a certain probability distribution, thus generating the symbols which are observable.

The HMM model consists of (Q, Σ, Π, A, B) where:

$Q = \{q_1, \dots, q_N\}$ *is the set of states that are finite count* N

$\Sigma = s_1, \dots, s_N$ *is the set of* M *possible symbols*

$\Pi = \{\pi_i\}$ *is the vector of initial probabilities*

$A = \{a_{ij}\}$ *is the probability matrix of state transitions*

$B = \{b_i(v_k)\}$ *is the emission probability matrix*

Combining all this the HMM can be denoted like this: $\lambda = (\Pi, A, B)$, *while following these constraints:*

The total transition probability is 1 *for:*

- *from the initial state to all hidden states:* $\sum_{i=1}^{N} \pi_i = 1$
- *from a hidden state* q_i *to all other hidden states:* $\forall i \in Q, \sum_{j=1}^{N} a_{ij} = 1$
- *From the hidden states to all observable states* $\forall i \in Q, \sum_{j=1}^{M} b_i(v_k) = 1$

The macroeconomic research is also called top to bottom approach of Financial data and analyses [1] which means that first of all we are looking at macro level (world view) i.e. we are looking at the world economy as a combination of all the countries. In our case we are looking at the US economy and as such we are looking at the oil supply/demand.

In such cases we are looking at the macro indicators like: oil consumption, production, consumer price index, producer price index etc. These indicators show us the state of the oil supply and demand that we are currently in. How much is being produced, what the capacity of the production is engaged, what are the prices and costs etc? This is important data that can

show us the trend, momentum or change of momentum and mean reversion in production and usage of oil so thus we can derive the supply and demand for oil. [2]

In order to get this data we are using the Federal Reserve Economic Data (FRED) [3] where all kinds of statistical data for the USA can be retreived. Of course this data can be corrupted or full of missing values so we have to apply some data preparation techniques and methods in order to clean the data and make it useful for machine learning usage.

First we check for missing data and as usual either remove it or impute it, depending on the volume of the missing data. If there is a lot of missing data we can impute it with the sample mean or back fill it or forward fill it depending on the data itself (categorical or numerical data).

In our case the data is all numerical and furthermore there is not that much of missing values so we decide to just drop it. Next we filter out the outliers so that they do not skew the data. Something very important is to check whether specific data is of the right kind, I mean that, for example, sometimes datetime is classified as a string data type or integer data type so in such cases we should turn it into a datetime type.

When we are finished with the data cleaning procedures we concatenate and combine all the data into one dataset for further analyses and then we proceed to the next steps like splitting the data into a training set and a testing set and then proceed into the modeling phase.

The ‘pgmpy’ (‘pgmpy’ refers to the probabilistic graphical models library in python) library is used to build and train the Bayesian Network [4] which in turns represents the relationship between the crude oil prices and the indicators. In order to train the network, first the bayesian network structure is defined and historical data is used to estimate the parameters and at last the network is trained using the ‘pgmpy’ library.

The next step is to validate the parameters and the model. In order to ensure that the network captures the relationship between indicators and crude oil prices accurately, it is crucial to validate the model. For validation, the model is first tested on a separate validation dataset and then the predicted outcomes are compared with the actual values. Validation metrics like RMSE, MAE are calculated to achieve this.

In case the computed validation error comes out to be high then in order to improve the model’s performance, hyperparameter tuning is performed. Hyperparameter tuning involves retraining the model after adjusting the model parameters. To achieve this, first the number of

hidden states in HMM is adjusted. Then the structure of the Bayesian network is changed and training parameters like learning rate, number of iterations etc. are modified.

Once everything is done, the model parameters are optimized and the performance of validation also comes out to be satisfactory then the model is tested on unseen data for the final step. To perform this, a separate test data will be used and the model's performance on the test set will be evaluated. In the end, the predicted and actual outcomes will be compared. This will in turn evaluate the ability of the model to make accurate predictions on new data.

The design process consists of multiple steps from macro research, data retrieval to data cleaning, regime identification, training and testing the network, validation etc. each and every step is important in order to make an accurate model. By training the network using 'pgmpy' and testing and validating the parameters ensures that the model will capture the underlying factors driving crude oil prices and helps in increasing reliability of the forecasts.

The very first step with dealing with the time series is to turn them into returns. This is important because based on these returns we will do the mapping procedure where we go through the returns data and all the values that are bigger than zero can be mapped as positive so we "name" them as 1 and the rest as 0. This is important and will be later used for the regime detection method.

For regime detection we use the Viterbi Algorithm to identify the most probable transition-state regime. [5] Furthermore, while considering that the bull regime has positive returns, while the bear regime has negative returns and the stagnant regime has returns that are gravitating around zero, we will use this specification for mapping and coloring the regimes with the corresponding colors: bull = green, bear = red and stagnant = blue.

To find most likely sequence of hidden states using the Viterbi algorithm-
HMM, the hidden markov models, are proved to be the most powerful and useful tools to identify the underlying regimes in the time series data like the financial time series data of crude oil prices. HMM can help in showing the hidden states that influence the observed sequence of emissions but are not directly observable. In order to achieve and uncover these hidden states, the Viterbi algorithm works in order to find the most likely sequence of hidden states or regimes that generated the observed data [5] [6]. Considering the Viterbi algorithm as the black box, we'll understand how hidden states or regimes are identified.

First, the time series data is transformed with the observed emissions sequence which is the monthly changes in the prices of crude oil and the difference in price between consecutive months is computed. The Baum-Welch algorithm is used to train the model and learn the parameters from the sequence of observed emissions.

After training the parameters of the model, Viterbi can be used now to find the most likely sequence of hidden states (regimes) that generated the observed emissions sequence. The Viterbi algorithm computes the path with the highest probability by using dynamic programming to find the sequence of observed emissions. At last, the identified hidden states are mapped to the corresponding regimes of bull, bear or stagnant based on average price change to interpret the results.

Hence, the Viterbi algorithm, treated as black box, helps in finding the most likely sequence of hidden states and for complex financial datasets, it enables robust regime detection.

The hidden states in regime detection represent the different market conditions or regimes for the financial time series data. For our model, there are three hidden states of bull, bear and stagnant market regimes and we can analyze their characteristics in order to infer their latent meanings.

The model's first hidden state is the bull regime which is characterized by increase in prices indicating the rising market. It reflects an upward trend as the average price change of this state is positive. The latent meaning of bull market regime refers to the period when crude oil prices are increasing. This hidden state has strong demand and favorable economic conditions.

The second hidden state is the bear regime which is characterized by decrease in prices indicating the falling market. It reflects a downward trend as the average price change of this state is negative. The latent meaning of bear market regime refers to the period when crude oil prices are consistently decreasing. This hidden state has weak demand and unfavorable economic conditions.

The third and the last hidden state of the model is the stagnant market regime which is characterized by a low frequency of both increase and decrease in prices indicating a stable market. It reflects a lack of clear market direction or overall stability as the average price change of this state is close to zero. The latent meaning of stagnant market regime refers to the

period when crude oil prices are experiencing minimal changes. This hidden state has balanced supply and demand and market stability.

In order to interpret the market conditions and to make informed trading or investment decisions, it is important to understand the latent meanings of the hidden states in regime detection.

In order to implement the minimal working example of a hill climb search for learning Bayesian Network, the provided example in the thesis is followed on the combined data of macroeconomic and financial datasets that we have and the pgmpy library is used.

Firstly, the data is filtered and then discretized into binary values of 1 for increase and 0 for decrease. After that, with the help of discretized data and an initial expert knowledge model, the hill climb search is performed. The data is trained and the learned Bayesian network model is fitted into the data. Finally, the resulting network is visualized and observed as follows-

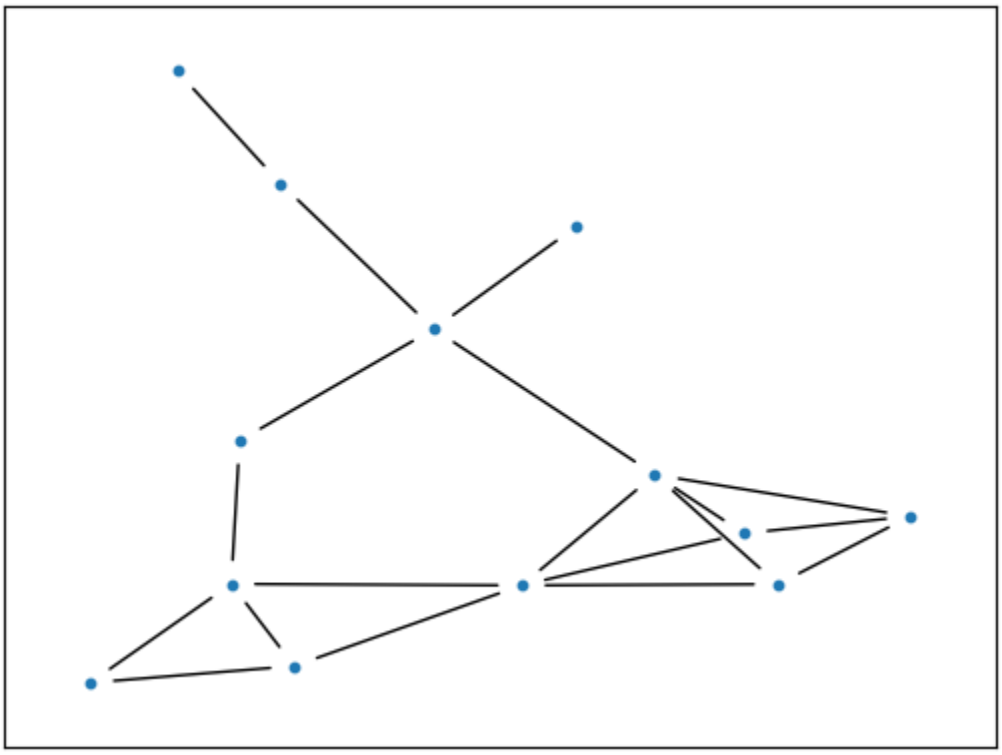

Fig.2. Learned Bayesian Network (Hill Climb Search)

In the above plot, the nodes represent the variables of the dataset and edges represent the dependencies between them. Here, the nodes with many connections can be the key variables. Now, the model has been fitted using the hill climb search and inferences can be made using forecasts as evidence.

Here, the different states of the variables within the Bayesian network are referred to as the hidden states. These hidden states might be different economic or market conditions that are derived from the relationships between the observed variables. In the discretized dataset, the state values are binary and each hidden state in the model is a combination of these binary values.

The first hidden state of the bear market is the state where the majority of variables show a decreasing trend. For example, a decrease in spot crude oil prices might be due to a decrease in industrial production index and SP500.

The second hidden state of the bull market is the state where the majority of variables show an increasing trend. For example, an increase in spot crude oil prices might be due to an increase in Consumer Price Index for All Urban Consumers: Energy and Canadian dollar to US dollar exchange rate.

The third hidden state of stagnant markets is the state which represents a mixed condition where some variables increase while others decrease.

REFERENCES:

1. De Grauwe, P., (2009), “Top-down versus Bottom-up Macroeconomics”, DG ECFIN Annual Research Conference, Brussels, 15 October 2009
2. Lutz Kilian. Not All Oil Price Shocks Are Alike:Disentangling Demand and Supply Shocks in the Crude Oil Market. University of Michigan and CEPR, 2008.
3. Federal Reserve Economic Data, FRED, https://fred.stlouisfed.org/
4. Pure python implementation for Bayesian Networks with a focus on modularity and extensibility, https://pgmpy.org/
5. Danish A. Alvi, Application of Probabilistic Graphical Models in Forecasting Crude Oil Price, Department of Computer Science, University College London, 2018
6. Cheng Xiang Zhai. “A Brief Note on the Hidden Markov Models (HMMs)”, Department of Computer Science, University of Illinois at Urbana-Champaign, 2003
7. Hidden Markov Model - an overview | ScienceDirect Topics.

8. https://pypi.org/project/pgmpy/