Data Analysis and Optimization of Asset Liability Management Problem

Chapter 1

Introduction

In this chapter, firstly we discuss about the risks associated with banking. Then we present Asset Liability Management (ALM) a solution to minimize risk associated with banking. Then we will discuss project objectives, problem statement, banking standards and the literature survey.

1.1 Risks Associated with Banking

The term risk can be defined in association with banking as the exposure to loss. The several risks associated with banking are viz., Operational risk, Credit risk, Liquidity risk, Market risk, Foreign exchange risk, Interest rate risk and Information risk.

1.1.1 Operational Risk

It is defined as the loss occurred due to failure of peoples or system. It is faced in the various departments such as of Information Technology department, Credit Department, Investment department and across delivery channels.

1.1.2 Credit Risk

It is defined as the loss occurred due to failure of the borrower to repay to bank on the agreed terms. There is uncertainty about the repayment of dues and in the repayment in the agreed time frame. It happens due to borrowers lack of income, failure of business, reluctance to repay, lack of underwriting frameworks and political culture of waving to loans.

1.1.3 Market risk

It is defined as the loss occurred due to fluctuation in the market prices. Its components are as follows:

- **Equity risk:** Probable failure to generate profit due to fluctuation in stock prices.
- **Foreign exchange risk:** Probable failure to generate profit due to the fluctuation of currency exchange rates as bank does transaction in multiple currencies.
- **Interest rate risk:** Probable failure to generate profit due to the fluctuation of interest rates.
- **Commodity risk:** Probable failure to generate profit due to change in commodity prices as metals (as gold, silver, platinum), Energy (oil and gas) and Agriculture (as wheat, cotton, coffee, tea,etc.). The change in these prices occurs due to variations in demand and supply.

1.1.4 Liquidity risk

It is characterized as the lack of liquid cash available in the hands of the bank to finance its day to day transactions or the situation where bank runs out of cash. Failure in managing the liquidity risk can lead to the destruction of the banks' reputation and losing its trust among customers.

The exposure of banks to these various risks and as the risks are also increasing with the liberalisation and increasing merger of local markets and global markets. Due to the deregulated operational space and the various products that requires the interest rates to be determined with the need to maintain the proper ratios between the profitability, spread and long term growth. Due to all these reasons, we need to perform Asset Liability management.

1.2 Asset Liability Management (ALM)

An asset is a valuable entity/resource that a person/organization owns for generating income. And the liability is a valuable entity/resource that a person/organization owes to pay for. At any given point of time, total assets must be greater than the total liability. To balance the equation the concept of asset liability management [?] had emerged.

In ALM we perform periodic monitoring of risk exposures involving collecting and analyzing the information in order to have the ability to anticipate, forecast and act so as to structure banks business to profit. ALM also involves transforming the asset and liability portfolio in a dynamic way to manage the risks which involve judgment and decision making.

1.3 ALM for Banks

As the business of banking involves lot of risks, the main problem of banking becomes risk management and the procedure to do so is ALM. The three pillars on which ALM resides are as follows:

1.3.1 ALM Information System

The risk policies and tolerance limits need to be specified by ALM information system. Information is the key to ALM process which is now available due to the computerization of all banks, its respective branches and transactions. It also includes performing experimentation in a specific branch and studying its effects. If the results are positive then replicating the changes to other branches.

1.3.2 ALM Organization

For risk management to be successful the need is strong commitment of the senior management to take strategic decisions and integrate the basic operations. The Asset Liability Committee (ALCO) is formed for performing the above-mentioned tasks. It includes the CEO and the senior management of the bank, and their task is to decide business strategies with respect to banks budget and decide risk management objectives according to present value of assets and liabilities.

1.3.3 ALM Processes

The ALM Processes are as shown in the following figure:

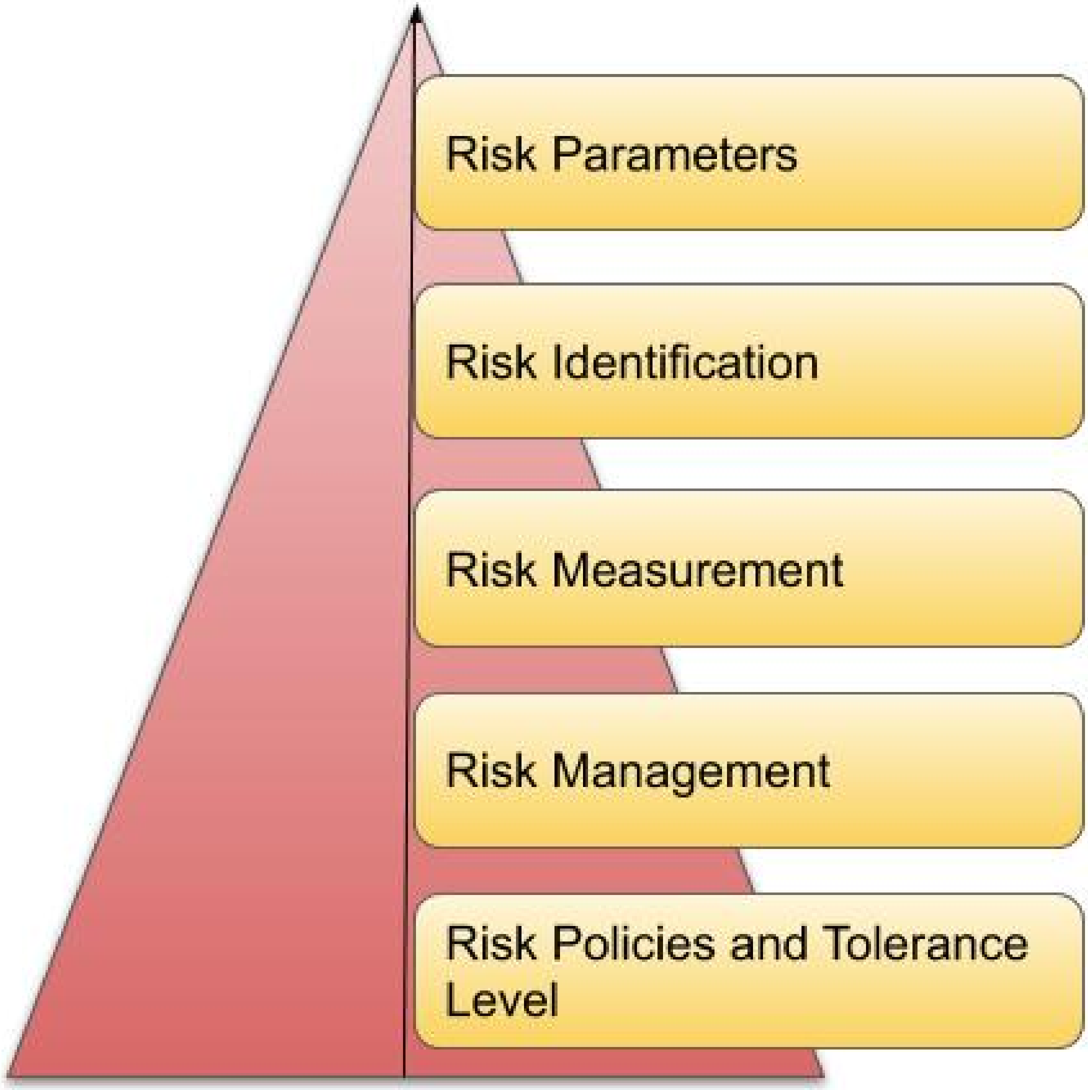

Figure 1.1: ALM Processes

All the risks discussed in section 1.1 are the problems to be handled by the ALM Processes. The assets and liabilities of a bank involve the following:

Liabilities	Assets
• Capital • Reserve Funds & Surplus • Deposits • Borrowings • Foreign Currency Liability	• Cash & Balances with RBI • Bal. With Banks & Money at Call and Short Notices • Investments • Fixed Assets, Loans and Advances • Foreign Currency Assets

Figure 1.2: Components of Banks Balance Sheet

The ALM cycle of a bank is as follows:

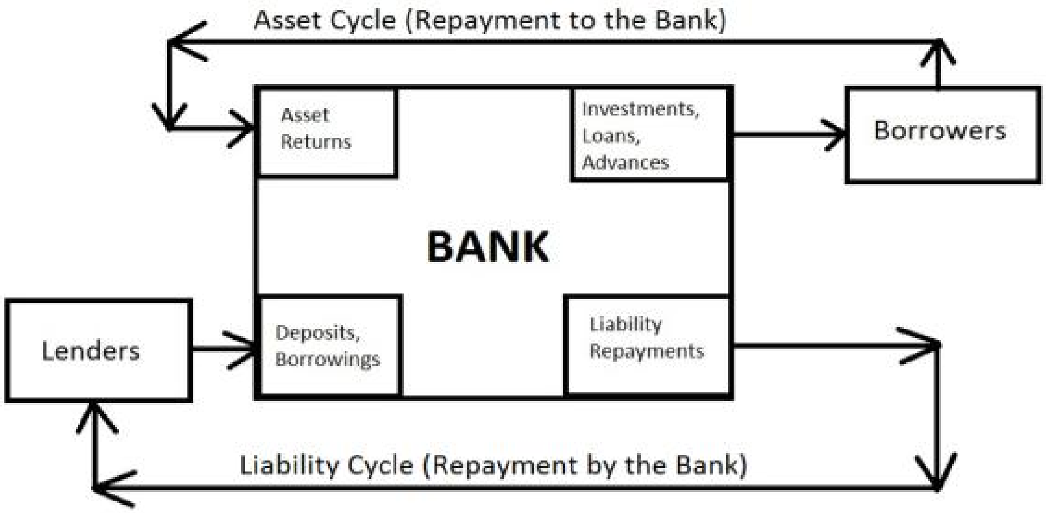

Figure 1.3: ALM Cycle of a Bank

Throughout the cycle of the ALM the mismatch in asset and liabilities (negative gap) should not exceed 20% of the cash outflow during 2-14 days and 15-28 days time bucket.

1.4 Project Objectives

To study the strategies to stabilize financial networks i.e. banks and to improve the profitability of them from various risks. The aim was to study the LPP optimization method and Deep Learning for finding the solution to risks associated with banking.

1.5 Problem Statement

The two problems that have been studied are as follows:

1.5.1 Liquidity Risk Management

Simulation of the ALM concept for bank using the 1999 Czech Financial data set and solve the single objective cash flow optimization for ALM to ensures Liquidity and Profitability of the bank.

1.5.2 Prediction of Stock Price

The Prediction of Stock Price is a typical problem of time series and we use a deep learning based solution methodology for future stock price prediction.

1.6 Literature Survey

For first problem the literature survey done is: Hongxi Li et al. (2017) assess for both positive and negative duration gap to increase the net value of bank when interest rates fluctuate favorably[?]. Nalan Gülpinar et al. (2013) uses the Vector Autoregressive process to model time-varying investment opportunities[?]. Teng Fan et al. (2011) studied the interest rate risk of Chinese life insurers' liability[?]. Mounika, P et al. (2011) have addressed the problem of single objective optimization for the maximization of wealth[?]. Chaudhury, Rahul et al. (2014) created a fuzzy rule-based asset liability optimization model[?].

For the second problem, the literature survey done is: Ashish Vaswani et al. (2017) proposed an attention mechanism for machine translation[?]. Yao Qin et al. (2017) proposed a DA-RNN. It serves the purpose of attention to time series and encoding information of long sequences[?]. Jian Liu et al. (2017) assesses the correlation between stock price movement with relevance to events happening in the world[?]. Hao Li et al. (2018) proposed the MI-LSTM using attention which filters noise and extracts information[?]. Huicheng Liu (2018) used attention based RNN for leveraging the news to predict stock price[?].

Below are the existing algorithms which helps as basis for problems we are studying. The implementation of some of these methods is available in a software package called *statsmodels, pandas, sklearn, tensorflow, keras* [?] [?] [?] [?], which are also necessary for statistical analysis.

1.6.1 Time Series Analysis for Stock Price Prediction

1.6.1.1 Introduction

The Time Series (TS) is defined as the collection of information or data at a regular interval of time. This TS contains the time-dependent data which may contain the seasonality (i.e the deviation in data at specific time frame) and trend (i.e. the changing mean with respect to time) in it; which is the change in data with respect to specific time frame.

Ex. The Stock Data is collected per day i.e. the value of stock at the start of the day, at the end of the day, the highest value of the day and the lowest value of the day.

	High	Low	Open	Close
Date				
2013-01-02	5.44500	5.32000	5.43625	5.35750
2013-01-03	5.47875	5.36375	5.38250	5.43375
2013-01-04	5.36750	5.29250	5.36750	5.33500
2013-01-07	5.41500	5.33500	5.33750	5.41125
2013-01-08	5.35000	5.30375	5.31125	5.31500

Figure 1.4: Example of Time Series data of a stock price

1.6.1.2 Issues with non-Stationarity of Data

The data is said to be stationary if the mean and variance of the data is stable over time and it has time-independent autocovariance. The statistical models which deal with the TS data have a premise that the data should be stationary. Because of this premise, we need to convert non-stationary data to stationary data. But how to check the data is stationary or not? There are two ways to do it:

- Plotting the Rolling Statistics
- Dickey-Fuller Test

1.6.1.2.1 Plotting the Rolling Statistics

In this procedure, we visualize the moving average and/or moving variance by plotting it against time. The data in Figure 1.5 is used to plot the Figure 1.6. The functions used for computing the data and plotting the graph are in Appendices.

Date	Close	moving_average	moving_variance
2013-01-02	0.079921	NaN	NaN
2013-01-03	0.093278	NaN	NaN
2013-01-04	0.075980	NaN	NaN
2013-01-07	0.089337	NaN	NaN
2013-01-08	0.072476	0.082198	0.008833
2013-01-09	0.075980	0.081410	0.009253
2013-01-10	0.103788	0.083512	0.013041

Figure 1.5: Moving Average and Moving Variance of the Closing Price with window size = 5

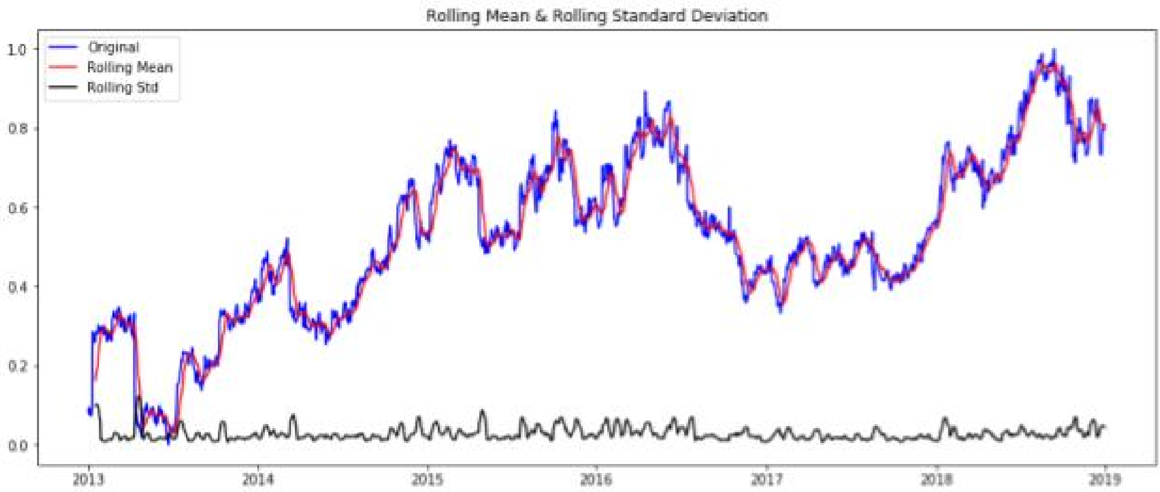

Figure 1.6: Plot of the Rolling Mean and the Rolling Standard Deviation of Moving Average of Closing Price

As we can infer from the plot, the rolling mean is changing with respect to time and there is a clear fluctuation in the standard deviation resulting in the data to be non-Stationary.

1.6.1.2.2 Dickey-Fuller Test (DF Test)

In statistics, there is a unit root test that is used to check the given TS is non-Stationary or not. DF Test [?] is the most widely used unit root test. In statistical testing we have to make the hypothesis, the same in this test are:
H0: TS is non-Stationary.
H1: TS is Stationary.

Where, H0 is the Null hypothesis and the H1 is the Alternate Hypothesis.
The hypothesis H0 is accepted if the p-value of the test is greater than 5% else the H0 is rejected and H1 is accepted.

For the data we are dealing with the results of the DF Test are as follows:

```
Results of Dickey-Fuller Test:
Test Statistic                   -2.467311
p-value                           0.123632
#Lags Used                        0.000000
Number of Observations Used    1509.000000
Critical Value (1%)              -3.434691
Critical Value (5%)              -2.863457
Critical Value (10%)             -2.567791
dtype: float64
```

Figure 1.7: DF Test Results for moving average of Closing price

As the p-value is 0.1236 i.e. 12.36% which is greater than the threshold 5% so the null hypothesis H0 is accepted; meaning the TS is non-Stationary.

1.6.1.3 Converting non-Stationary TS to Stationary TS

To make the TS stationary we need to eliminate the trend and seasonality from the non-stationary TS. The first and the widely used approach to do it is Transforming the data using a log transform. Then use either Differencing or Decomposition. Let's see them one by one:

1.6.1.3.1 Log Transform

Taking the log of the data changes the recurrent pattern to a linear pattern and also stabilizes the variance of the data [?].

```
Results of Dickey-Fuller Test:
Test Statistic                   -2.607083
p-value                           0.091542
#Lags Used                        0.000000
Number of Observations Used    1509.000000
Critical Value (1%)              -3.434691
Critical Value (5%)              -2.863457
Critical Value (10%)             -2.567791
dtype: float64
```

Figure 1.8: DF Test Results for log-transformed Closing Price

Here we can see that the p-value is dropped from 12.36% to 9.15% which means that we can say we have 90% confidence that the TS is stationary.

1.6.1.3.2 Decomposing

A TS consists of three components: trend, seasonality and residual. If the decomposition be additive then:

$$Data_{\mathrm{t}} = Trend_{\mathrm{t}} + Seasonality_{\mathrm{t}} + Residual_{\mathrm{t}} \tag{1.1}$$

Where the subscript t denotes the time t. The multiplicative decomposition is:

$$Data_{\mathrm{t}} = Trend_{\mathrm{t}} * Seasonality_{\mathrm{t}} * Residual_{\mathrm{t}} \tag{1.2}$$

The addition or multiplication of these three components gives back the original data or TS. The multiplicative decomposition is used when the trend or seasonality of data is proportional to the time. The procedure of classical multiplicative decomposition [?] is as follows:

Algorithm 1 Classical Multiplicative Decomposition

Assumption m - seasional period or frequency, MA - moving average, $\mathrm{DT_t}$ - Detrended series.

1: If m is even number then

$$Trend_t = 2 * m - MA \tag{1.3}$$

Else

$$Trend_t = m - MA \tag{1.4}$$

2: Compute Detrended series:

$$DT_{\mathrm{t}} = Data_{\mathrm{t}} - Trend_{\mathrm{t}} \tag{1.5}$$

3: The seasonal component Seasonalitytof the respective season is the average of the $\mathrm{DT_t}$ of the given season.

4:

$$Residual_{\mathrm{t}} = Data_{\mathrm{t}}/(Trend_{\mathrm{t}} * Seasonality_{\mathrm{t}}) \tag{1.6}$$

The plot looks as follows:

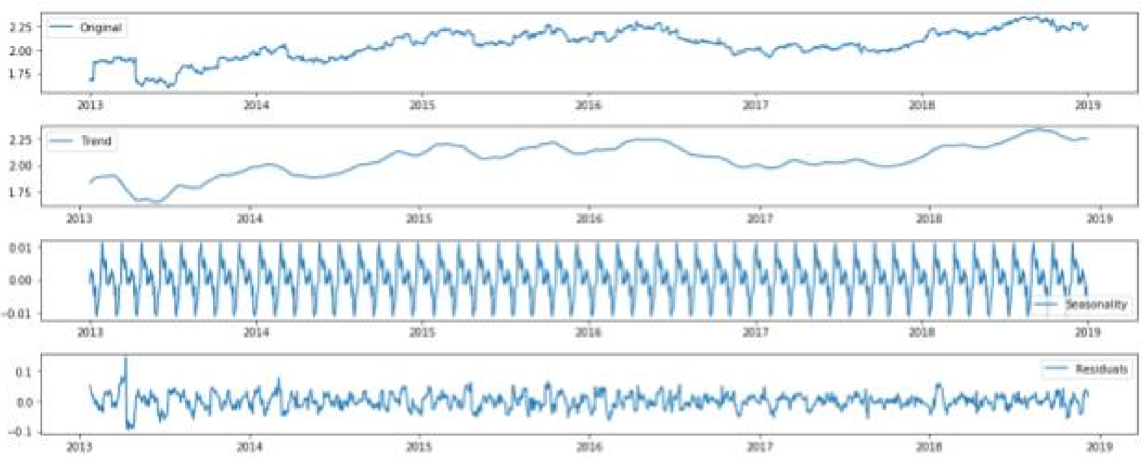

Figure 1.9: The trend, seasonality and residual of log-transformed data

The result of DF test results on the residual data is as follows:

```
Results of Dickey-Fuller Test:
Test Statistic                -1.056420e+01
p-value                        7.590940e-19
#Lags Used                     2.300000e+01
Number of Observations Used    1.456000e+03
Critical Value (1%)           -3.434849e+00
Critical Value (5%)           -2.863527e+00
Critical Value (10%)          -2.567828e+00
dtype: float64
```

Figure 1.10: DF Test Results on residual data

Here the p-value is 7.5e-17% which means that we can say the TS is stationary with 99.99% confidence.

1.6.1.3.3 Differencing

In this approach, we compute the difference between the data point at the current instance to the previous instance. Applying the DF test on the differencing [?] of the log-transformed data gives the following results:

```
Results of Dickey-Fuller Test:
Test Statistic                  -28.918288
p-value                           0.000000
#Lags Used                        1.000000
Number of Observations Used    1507.000000
Critical Value (1%)              -3.434697
Critical Value (5%)              -2.863460
Critical Value (10%)             -2.567792
dtype: float64
```

Figure 1.11: DF Test Results for log-transformed Closing price after differencing

Here the p-value is 0.0% which means that we can say the TS is stationary with 100% confidence.

1.6.1.4 Disadvantages of Making TS stationary

There is huge information loss when we are converting the non-stationary TS to Stationary TS. The elimination of trend and seasonality leads to information loss which makes the model linear and it fails to predict the future trend and seasonality appropriately.

1.6.1.5 Inference

The statistical methods are good to work with the stationary data, but when the data is non-stationary the statistical methods fail due to the premise discussed in section 3.2. Although due to the advances in the research area of neural networks we are able to deal with the non-stationary data without converting it to stationary. The methods used to predict the TS data are discussed in following section.

1.6.2 Deep Learning Architectures for Prediction of Stock Price

In this chapter, we will study the different deep learning techniques used to predict the sequence of output after feeding them the time-based input sequence.
For all the tasks consider
Data: X_i : $Source_i$; Y_i : $Target_i$
And the loss function: Mean Squared Error Loss as the problem is a regression problem.

$$MeanSquaredError(MSE) = \frac{1}{n}\Sigma_{i=1}^{n}(Predicted\_Y_i - Y_i)^2 \tag{1.7}$$

1.6.2.1 Recurrent Neural Network (RNN)

Artificial neural networks have a special class of neural networks that works best with the input sequences, called the RNN. As stated in the name the word Recurrent stands for the repeating structure of the neurons with respect to time/sequence based input to them until the whole input sequence is over. Unlike other neural network architectures where we need to feed whole input at once to the input neurons; in RNN [?], we can feed the sequence one by one to the input neurons and the input gets processed in the same recurrent fashion throughout at all the recurrent layers. The output of the neuron after processing the (t-1)th input sequence is fed again to the same neuron with the (t)th sequence of the input so that it can remember the whole input sequence. The idea basically is to remember the past, add it to the present and predict the future. The RNN looks as shown in the following Fig.

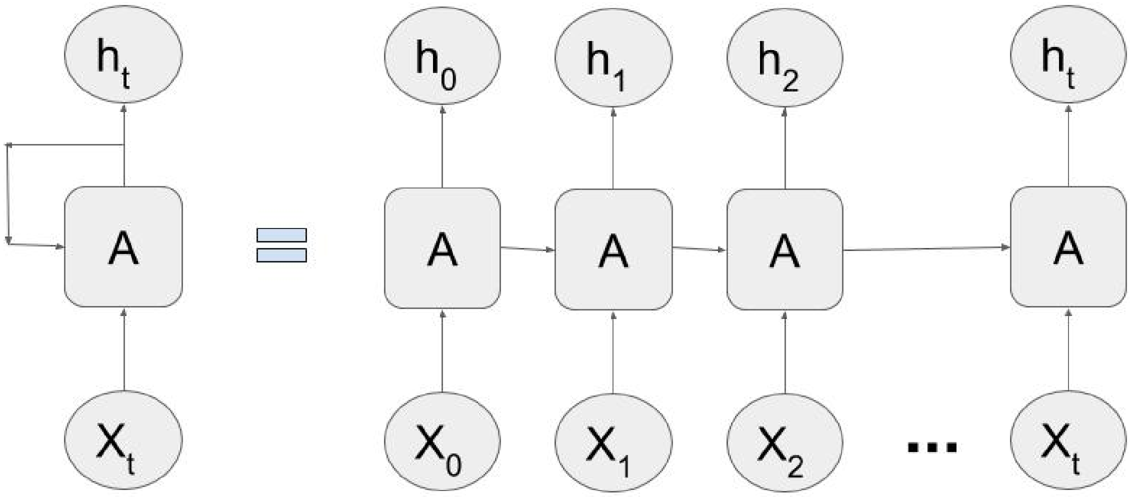

Figure 1.12: Unfolded Recurrent layer

Now let's consider the Fig.4.3 which shows the activation function tanh in the neurons of recurrent layers. The equation of the recurrent layer is as follows:

$$h_t = tanh(w_i * X_t + w_o * h_{t-1} + b) \tag{1.8}$$

Using weights w_i and w_o we will learn about the sequenced input. Here we are using tanh as activation function because to keep gradient in linear region of the activation function the second derivative of tanh function sustain for long range before going to zero, helping to learn about sequences.

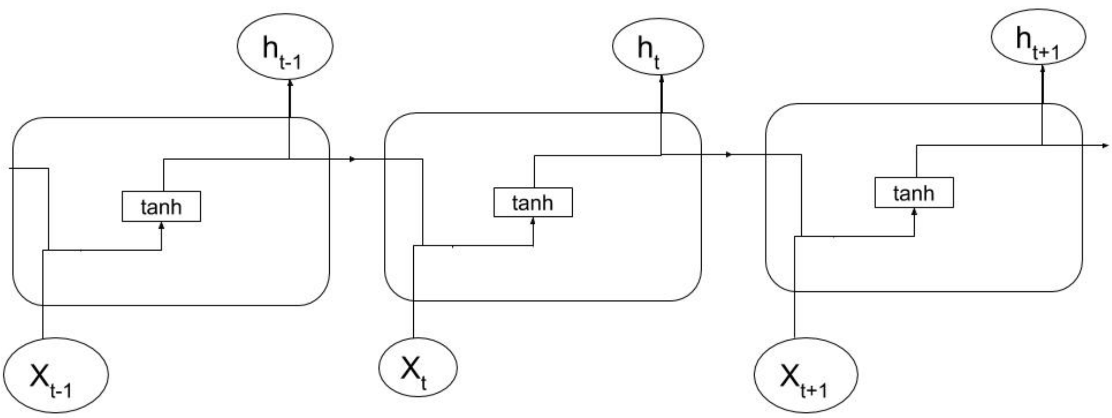

Figure 1.13: Unfolded Recurrent layer with activation function

1.6.2.1.1 Problems of Long Term Dependency

If the input sequences are small then the RNN is the best choice, but what if the input sequence is long, can RNN successfully learn the information from the past and using present information predict the future? The answer is No. In RNN we don't have an equation for how much to remember from the past or what to remember from the past. This is called the problem of long term dependency [?] as shown in Fig.4.4. If the output h_{t+1} depends on the input X_1 then RNN can't handle this dependency. The solution to this is the improved version of RNN is discussed later.

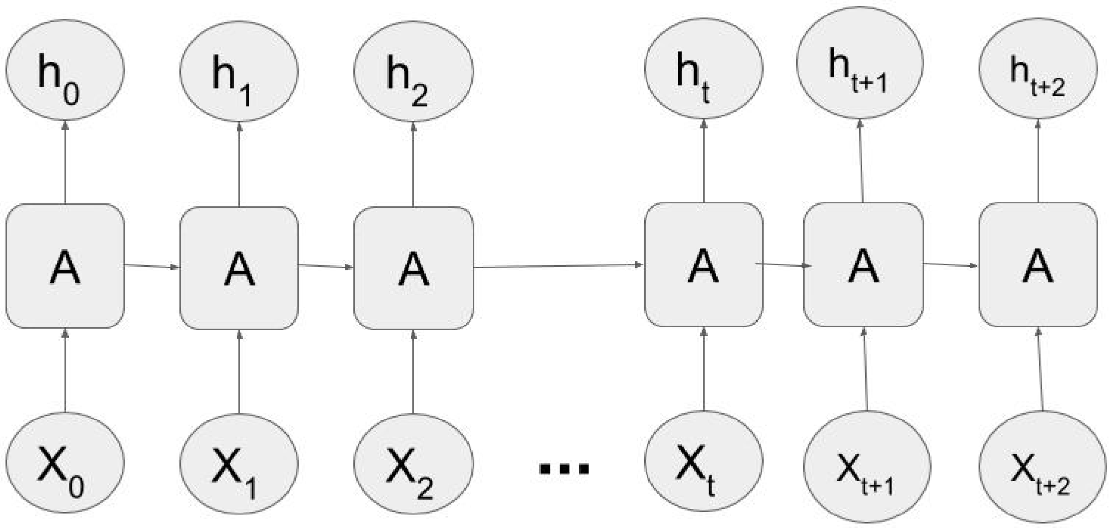

Figure 1.14: Long term dependency problem

1.6.2.1.2 Long Short Term Memory (LSTM)

The word LSTM [?] can be interpreted as storing the Short information in the Memory for the Long Term. This happens due to the functional change in the cell of RNN. Instead of the single neuron function in the RNN, the LSTM has four neurons connected in a circuit creating three gates: input gate, forget gate and the output gate. Other than these gates there is a cell state which computes the update for the next state. The following are the notations that will be used to describe the architecture of the LSTM:

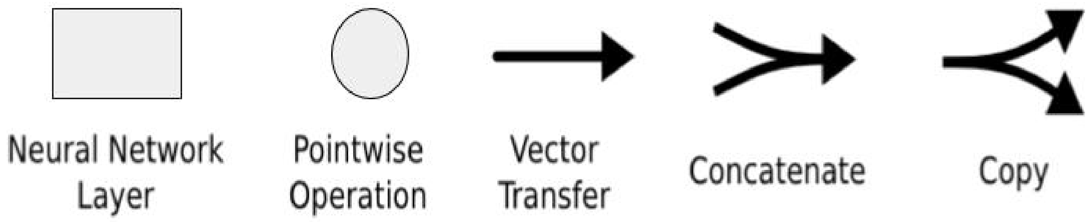

Figure 1.15: Notations

The comparison of RNN and LSTM is as follows:

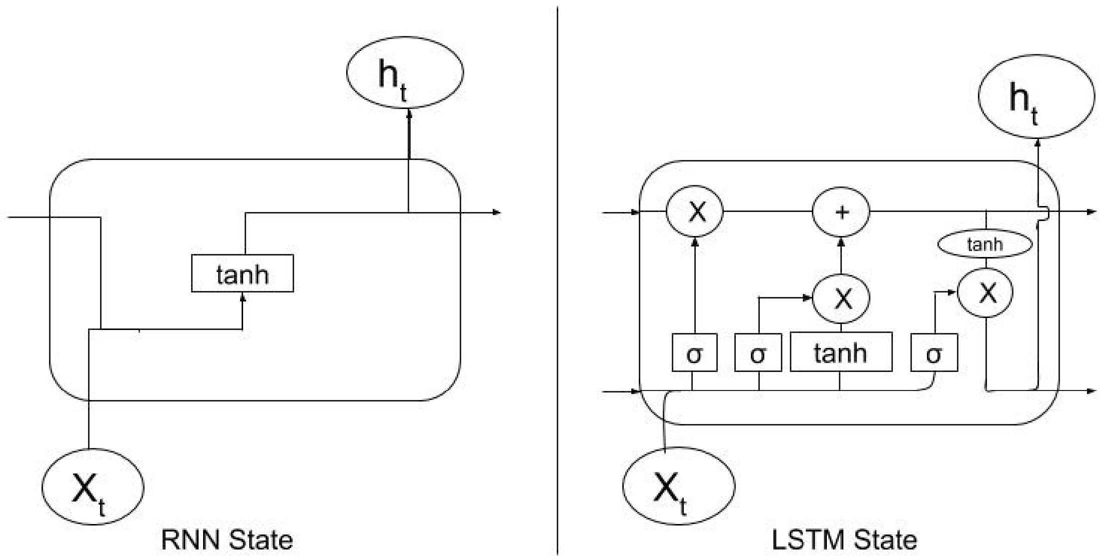

Figure 1.16: Comparison of RNN and LSTM State

Cell State The cell state decides how much of the original information from previous state is to be retained to the next state (Forget gate) and how much new information from the current state is to be added after forgetting the old information from the previous state (Input gate). It's like scaling and shifting operation. The information from the previous state is scaled between 0 to 1 and information from the current state is added (shifting).

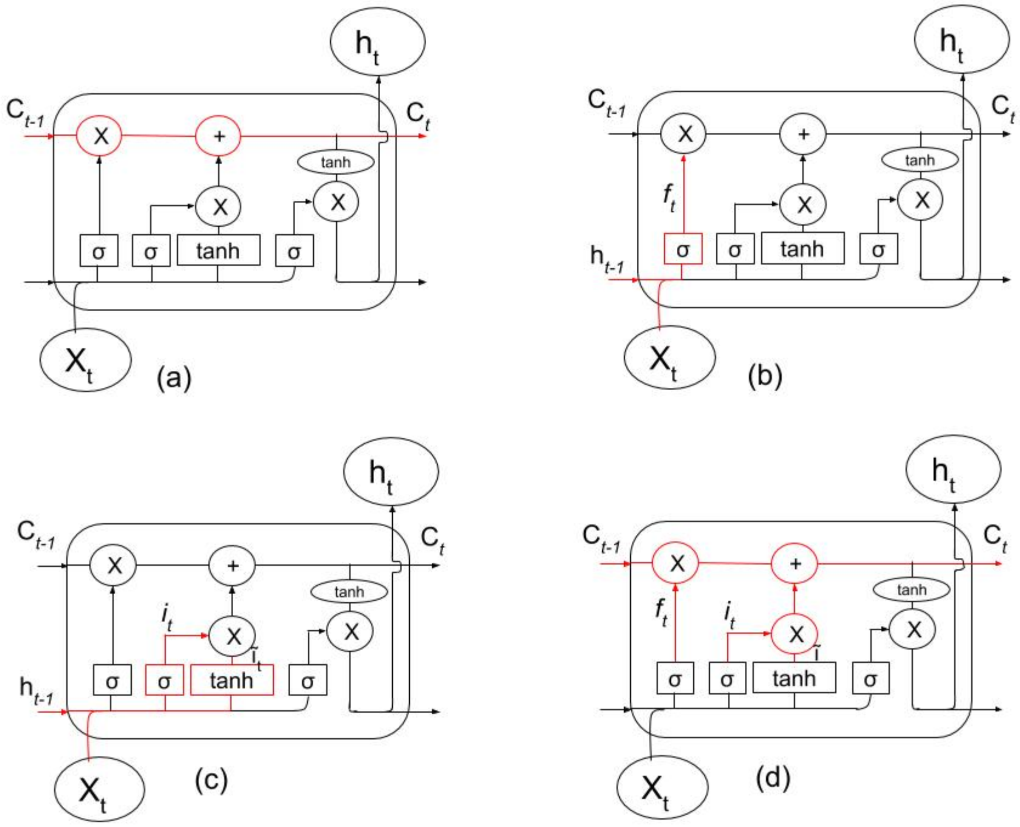

Figure 1.17: Flow of input through various gates. (a) Cell State, (b) Forget gate, (c) Input Gate and (d) Update to New Cell State

Forget Gate As discussed in the cell state this gate uses the sigmoid function which gives the values between 0 to 1. These values determine the scaling of the forget gate. Equation for forget gate:

$$f_t = \sigma(W_f.[h_{t-1}, X_t] + b_f) \tag{1.9}$$

Input Gate This gate decides how much information from the current state is to be added to the cell state. It uses a sigmoid and a tanh function for this purpose. The equation is as follows:

$$i_t = \sigma(W_i.[h_{t-1}, X_t] + b_i) \tag{1.10}$$

$$\tilde{I}_t = tanh(W_c.[h_{t-1}, X_t] + b_c) \tag{1.11}$$

0 to 1 is the range of sigmoid function and -1 to1 is the range of tanh function which shifts the input accurately and creates the output of the input gate.

Updating Cell State The eq(2.8 to 2.10) computed the values for updating the cell state, now we just need to do pointwise multiplication and addition. The equation is as follows:

$$C_t = f_t * C_{t-1} + i_t * \tilde{I}_t \tag{1.12}$$

Output Gate Using sigmoid function on current states input and tanh function on the updated cell state outputs the balanced combination of current input and previous cell output . The following are the equations:

$$o_t = \sigma(W_o.[h_{t-1}, X_t] + b_o) \tag{1.13}$$

$$h_t = o_t * tanh(C_t) \tag{1.14}$$

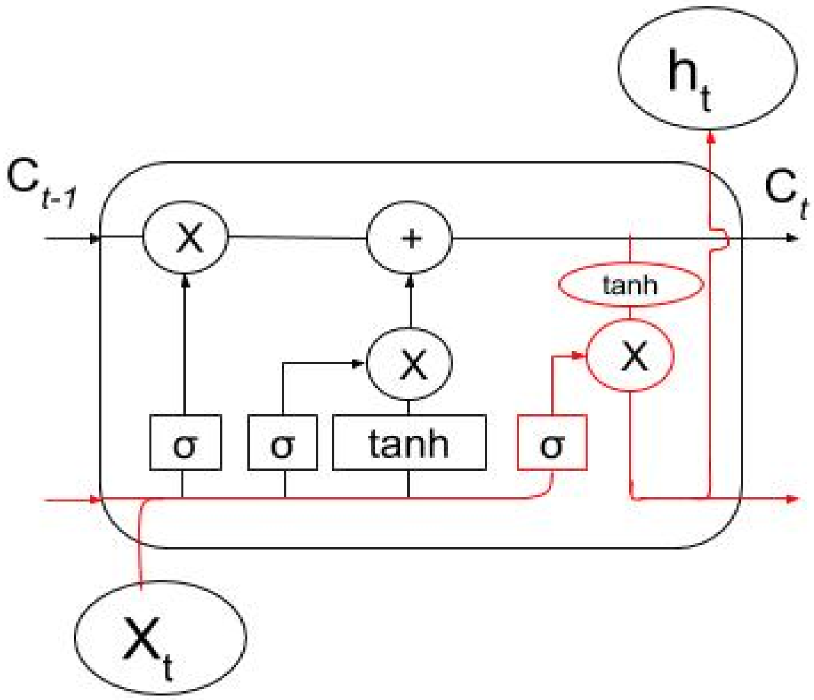

Figure 1.18: Output Gate

For the understanding of the later equations let us consider the following equations:
For RNN:

$$h_t = RNN(h_{t-1}, X_t) \tag{1.15}$$

For LSTM:

$$h_t, C_t = LSTM(h_{t-1}, C_{t-1}, X_t) \tag{1.16}$$

1.6.2.1.3 Encoder - Decoder Models (Sequence to sequence models)

The Encoder - Decoder model [?] is very much useful in lot of applications. The Encoder which is a neural network works for computing the representation of the input while the decoder which is also a neural network uses this input representation from Encoder and the target output to learn about the relation between the input and output and makes the appropriate predictions. For our problem, both encoder and decoder use the RNN/LSTM as the encoding and decoding function. The Encoder - Decoder mechanism can be represented as in Fig. below:

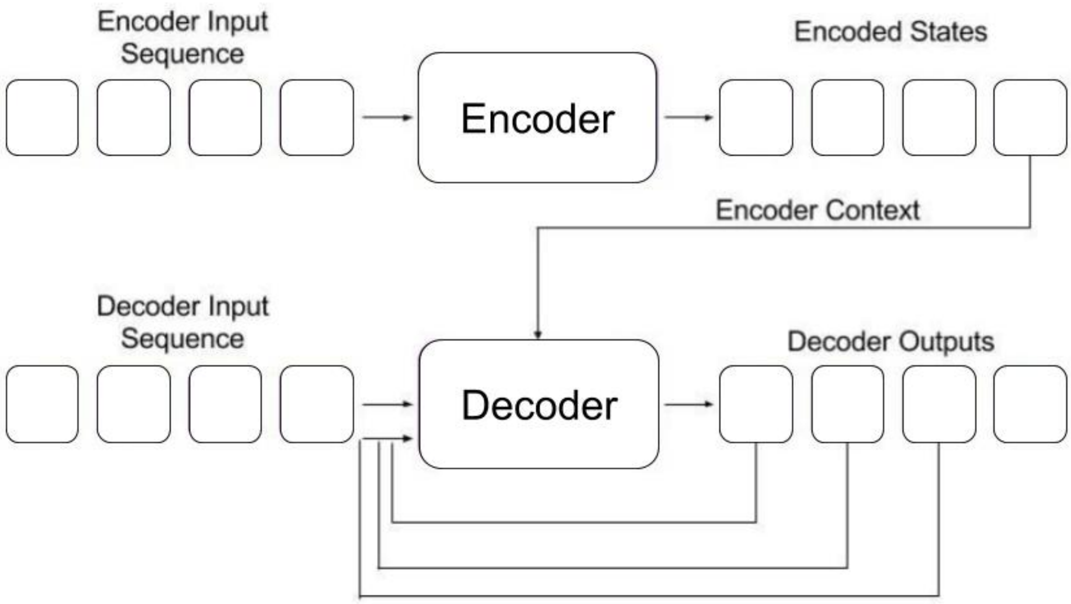

Figure 1.19: Encoder - Decoder Mechanism

The equations for the encoder-decoder model using RNN as the neural network architecture is as follows:

Encoder:

$$h_t = RNN(h_{t-1}, X_t) \tag{1.17}$$

Decoder:

$$C_0 = h_t \tag{1.18}$$

$$C_t = RNN(C_{t-1}, [h_{t-1}, e(\hat{y}_{t-1})]) \tag{1.19}$$

1.6.2.1.4 Encoder - Decoder Models with attention (Sequence to sequence models with attention)

The attention is a technique which tells the decoder how much to focus on the information at the given point of time. The attention of the overall input sequence adds to 1 due to the softmax function. The Encoder - Decoder mechanism with attention [?], [?] can be represented as in Fig.4.13.

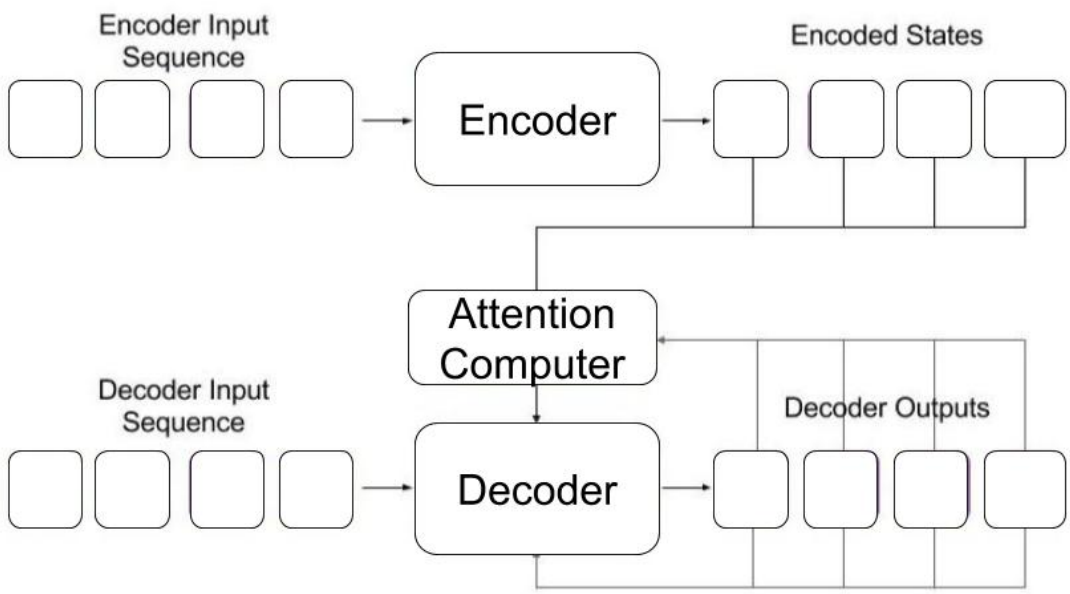

Figure 1.20: Encoder - Decoder Mechanism with Attention

The equations for the encoder-decoder model with attention using RNN as the neural network architecture is as follows:
Encoder:

$$h_t = RNN(h_{t-1}, X_t) \tag{1.20}$$

$$C_0 = h_t \tag{1.21}$$

Decoder:

$$e_{jt} = V_{attn} tanh(U * h_j + W * C_t) \tag{1.22}$$

$$\alpha_{jt} = softmax(e_{jt}) \tag{1.23}$$

$$S_t = \sum_{j=1}^{t} \alpha_{it} * h_j \tag{1.24}$$

$$C_t = RNN(C_{t-1}, [e(\hat{y}_{t-1}), S_t]) \tag{1.25}$$

Where e_{jt} is the importance of j^{th} input for decoding the t^{th} output

α_{jt} is the focus probability on j^{th} input with respect to t^{th} output

V_{attn}, U, W: Parameters to be learned.

1.7 Overview of Project Report

The overview of the project report is as follows:

The chapter 1 contains the overall introduction to the basics of thesis. In chapter 2, we have simulation of the ALM scenario for a bank using a data-set and some insights from the data-set. In chapter 3, we have used LPP method for the optimization of the assets. The chapter 4 contains implementation details and the results for the prediction of the stock price. The thesis ends with chapter 6 containing conclusion and future work.

Chapter 2

Simulation and Analysis of ALM

2.1 Analysis of Saving and Loan Accounts Data

Here we will discuss about the work on a 1999 Czech Financial Data-set with real anonymized transactions [?]. The aim was to learn the concept of ALM for banks and for that purpose we tried to simulate the ALM portfolio based on the given data set. As depicted in Figure 1.3 ALM Cycle of a Bank, the saving account deposits are the liability to the bank and the loans are assets for the bank. While neglecting all other parameters of the bank we are considering just deposits and loan to simulate the ALM concept.

2.2 Dataset Description

The Czech Financial dataset consists of 8 files in it, but we will be using only two namely transaction and loan. The description of both the files is as follows:

- **Relational Transaction:** This file consist of 1056320 rows, where each row is a transaction to an account. The attributes of the transaction file are :

1. Trans id: unique transaction number.

2. Account id: unique account number.

3. Date: date of the transaction.

4. Type: type of transaction: credit or debit.

5. Operation: mode of transaction: credit card withdrawal, credit in cash, collection from another bank, withdrawal in cash, remittance to another bank.

6. Amount: Transaction amount

7. Balance: remaining balance after transaction

- **Relational Loan:** This file consist of 682 rows, where each row is a loan given per account. The attributes of the loan file are :

1. Account id: unique account number.

2. Date: date of the transaction.

3. Amount :loan amount

4. Duration: Duration of loan

5. Payment: amount to be paid monthly.

6. Status : Loan account status: cleared loan, client in debt, defaulter, paying loan

The overview of the transaction file is as follows:

```
## # A tibble: 6 x 10
##   trans_id account_id   date type  operation amount balance k_symbol bank
##      <dbl>      <dbl>  <dbl> <chr> <chr>      <dbl>   <dbl> <chr>    <chr>
## 1   695247       2378 930101 PRIJ~ VKLAD        700     700 <NA>     <NA>
## 2   171812        576 930101 PRIJ~ VKLAD        900     900 <NA>     <NA>
## 3   207264        704 930101 PRIJ~ VKLAD       1000    1000 <NA>     <NA>
## 4  1117247       3818 930101 PRIJ~ VKLAD        600     600 <NA>     <NA>
## 5   579373       1972 930102 PRIJ~ VKLAD        400     400 <NA>     <NA>
## 6   771035       2632 930102 PRIJ~ VKLAD       1100    1100 <NA>     <NA>
## # ... with 1 more variable: account <dbl>
```

Figure 2.1: Description of Transaction File

The summary of the transaction file is as follows:

```
##     trans_id          account_id          date               type
##  Min.    :       1   Min.    :    1   Min.    :930101   Length:1048575
##  1st Qu.: 428084   1st Qu.: 1204   1st Qu.:960113   Class :character
##  Median : 855438   Median : 2434   Median :970406   Mode  :character
##  Mean    :1327034   Mean    : 2936   Mean    :965560
##  3rd Qu.:1982606   3rd Qu.: 3659   3rd Qu.:980212
##  Max.    :3682986   Max.    :11382   Max.    :981219
##   operation              amount             balance            k_symbol
##  Length:1048575     Min.    :     0.0   Min.    :-41126   Length:1048575
##  Class :character   1st Qu.:   135.9   1st Qu.: 22388   Class :character
##  Mode  :character   Median : 2100.0   Median : 33099   Mode  :character
##                     Mean    : 5942.1   Mean    : 38457
##                     3rd Qu.: 6817.0   3rd Qu.: 49502
##                     Max.    :87400.0   Max.    :209637
##      bank               account
##  Length:1048575     Length:1048575
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
```

Figure 2.2: Description of Transaction File

The overview of the loan file is as follows:

```
## # A tibble: 6 x 7
##   loan_id account_id   date amount duration payments status
##     <dbl>      <dbl>  <dbl>  <dbl>    <dbl>    <dbl> <chr>
## 1    5314       1787 930705  96396       12     8033 B
## 2    5316       1801 930711 165960       36     4610 A
## 3    6863       9188 930728 127080       60     2118 A
## 4    5325       1843 930803 105804       36     2939 A
## 5    7240      11013 930906 274740       60     4579 A
## 6    6687       8261 930913  87840       24     3660 A
```

Figure 2.3: Description of Loan File

The summary of the loan file is as follows:

```
##       udate           loan_id       account_id        amount
## Min.    :930101   Min.    :   0   Min.    :    0   Min.    :     0
## 1st Qu.:940723   1st Qu.:   0   1st Qu.:    0   1st Qu.:     0
## Median :960129   Median :   0   Median :    0   Median :     0
## Mean    :956088   Mean    :1829   Mean    : 1725   Mean    : 44857
## 3rd Qu.:970715   3rd Qu.:5352   3rd Qu.: 1989   3rd Qu.: 45258
## Max.    :981219   Max.    :7308   Max.    :11362   Max.    :590820
##     duration         payments       status
## Min.    : 0.00   Min.    :    0   A    : 203
## 1st Qu.: 0.00   1st Qu.:    0   B    :  31
## Median : 0.00   Median :    0   C    : 403
## Mean    :10.81   Mean    :1242   D    :  45
## 3rd Qu.:12.00   3rd Qu.:1845   None:1620
## Max.    :60.00   Max.    :9910
```

Figure 2.4: Description of Loan File

2.3 Simulation of the Experiment

Algorithm 2 Pseudo Code for Computation of Maturity Table for ALM

1: Initialization:
Start ← [The starting day of each time bucket]
End ← [The ending day of each time bucket]
i ← 0
max ← length(Start)

2: *while(i < max)*:

3:
$$Liability = \Sigma_{j=Start[i]}^{End[i]} deposit\_amount[j] \tag{2.1}$$

4:
$$Asset[i] = \Sigma_{j=Start[i]}^{End[i]} loan\_amount[j] \tag{2.2}$$

5:
$$Withdrawal[i] = \Sigma_{j=Start[i]}^{End[i]} withdrawal\_amount[j] \tag{2.3}$$

6:
$$Mismatch[i] = Asset[i] - Withdrawal[i] \tag{2.4}$$

7:
$$\%\_Mismatch[i] = \frac{Mismatch[i]}{Liability[i]} \tag{2.5}$$

8: Cumulative % Mismatch = Cumulative_Sum(%_Mismatch)

2.4 Results of the Experiment

Table 2.1: Results of the ALM Simulation 1

B.N.	#Start	#End	Liability	Withdrawal	Assets
Bucket#1	1	1	3200	0	0
Bucket#2	2	7	28964	0	0
Bucket#3	8	14	268887	0	0
Bucket#4	15	28	296622	0	0
Bucket#5	29	90	6348345	1166308	0
Bucket#6	91	180	23510757	13085553	0
Bucket#7	181	265	89174600	64642130	2619276
Bucket#8	366	1095	669010939	577121716	26724276
Bucket#9	1096	1825	1458973798	1261537518	48700920
Bucket#10	1826	2179	942042305	832618418	25217268

B.N. : Bucket Number
#Start: Starting day of time bucket
#End: Ending day of time bucket
Liability: Total amount deposited in each time bucket
Withdrawal: Total amount withdrawn from deposited amount in each time bucket
Assets: Total loan amount granted in each time bucket

Table 2.2: Results of the ALM Simulation 2

B.N.	#Start	#End	Mismatch	% Mismatch	Cumulative % Mismatch
Bucket#1	1	1	3200	-100.00000	-100.0000
Bucket#2	2	7	28964	-100.00000	-200.0000
Bucket#3	8	14	268887	-100.00000	-300.0000
Bucket#4	15	28	296622	-100.00000	-400.0000
Bucket#5	29	90	6348345	-99.99665	-499.9966
Bucket#6	91	180	23510757	-99.98661	-599.9833
Bucket#7	181	265	89174600	-85.65359	-685.6368
Bucket#8	366	1095	669010939	-51.31571	-736.9526
Bucket#9	1096	1825	1458973798	-55.54921	-792.5018
Bucket#10	1826	2179	942042305	-50.90293	-843.4047

Mismatch: Difference in assets and liabilities
% Mismatch: Mismatch as % to outflow i.e. liability
Cumulative % Mismatch: Cumulative sum of % Mismatch

2.5 Analysis of the Results

As per the RBI guidelines the negative cumulative mismatch in the first four buckets should not exceed 5%, 10%, 15% and 20% of the cash outflow of the respective time bucket. Here a lot of parameters are from the ALM sheet are missing that's why the cumulative mismatch is exceeding the limits as shown in Table 2.2.

Although the difference between the liability and the withdrawal column is significantly positive the customers are not withdrawing all the amount deposited in the account. This lead for us to investigate more on the data .

2.5.1 Post Result Analysis of Data

Here we have assumed the different window for the transaction amount and the frequency of the transactions in that amount window is as follows:

Table 2.3: The amount window and frequency of transactions

Sr. No.	Window	Frequency
1	[0,100)	218988
2	[100,500)	160913
3	[500,1e+03)	48191
4	[1e+03,1e+04)	423892
5	[1e+04,2e+04)	104482
6	[2e+04,5e+04)	88270

The frequency density plot of the above table looks like this:

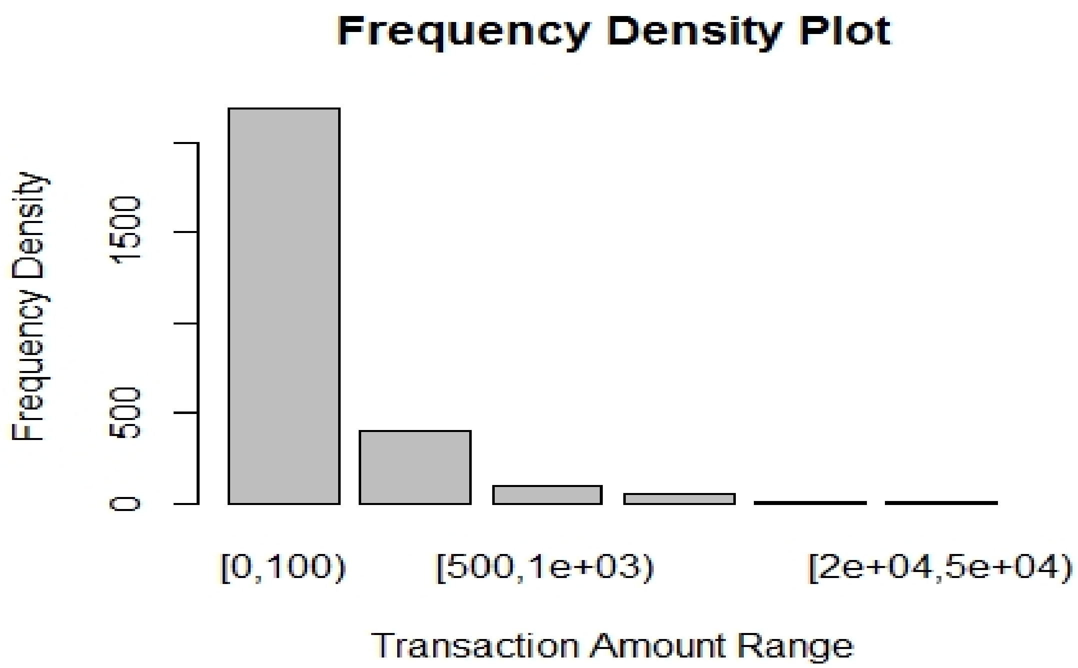

Figure 2.5: Frequency density plot of the Transaction Amount

The plot seems to be following Pareto distribution, which means that 80% of the transactions belong to the first window. Lets again select the random account numbers and plot their deposits and withdrawal transactions.

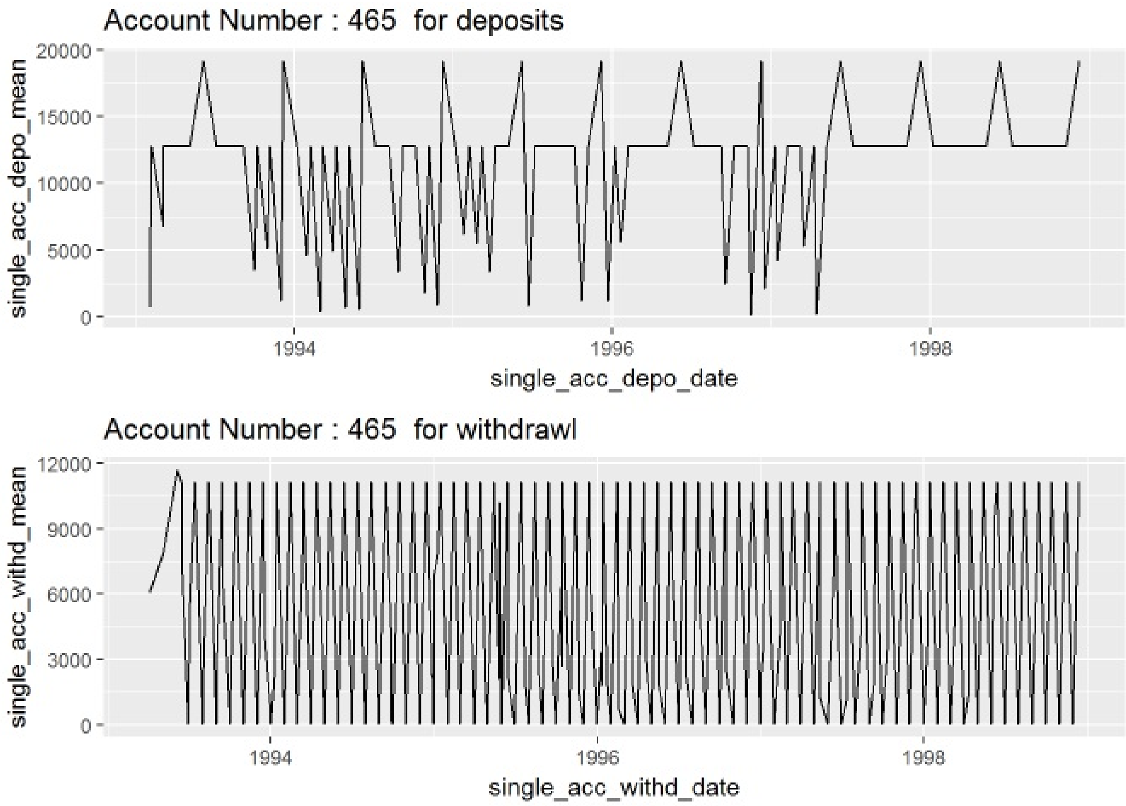

Figure 2.6: The deposit and withdrawal plot for randomly chosen account number

The plot shows that the mean amount deposited is maximum than mean amount withdrawal and the frequency of deposits is less than the frequency of withdrawal. But there were some cases where this condition failed. After analyzing such more plots we can say that there will always be some amount remaining in the account. The formula for finding the assets from each saving accounts can be given as:

$$ASA = \{AAD - AAW\}_{overgiventimebucket} \tag{2.6}$$

Where,

- ASA: Assets from Saving Account
- AAD: Average Amount Deposited
- AAW: Average Amount Withdrawal

Using this formula for first 5 time buckets, we got following results:

##	account_nos	acc_depo_mean	acc_withd_mean	differences
## 1	2378	14433.676	10355.634	4078.042
## 2	465	6371.113	5145.206	1225.906
## 3	5270	23756.868	10567.704	13189.164
## 4	1019	10565.604	3661.526	6904.078
## 5	1637	2906.244	1367.747	1538.497
## 6	485	14473.943	6214.987	8258.956
## 7	3367	12334.281	5660.907	6673.374
## 8	2486	17729.990	9340.120	8389.870
## 9	3678	5079.980	3149.784	1930.196
## 10	1127	12698.727	6580.909	6117.818

Figure 2.7: Average balance of randomly selected saving account

Where,

- account_nos: account number of customer
- acc_depo_mean: average amount deposited per account_nos
- acc_withd_mean: average amount withdrawal per account_nos
- differences: difference in acc_depo_mean and acc_withd_mean

The results above shows that there will be some amount always left in the account of customer. The bank can focus on such customers and encourage them to invest in their various investment schemes, so that this money can be utilized as asset for some fixed duration of time.

2.6 Conclusion

In this chapter, we have simulated the concept of ALM for banks and analyzed the data for getting some meaningful insights.

Chapter 3

Linear optimization of Liquidity Risk Management

3.1 LPP for Optimization of Assets

Linear Programming is the famous method used for the optimization of a given objective function and constraints. In this section we will use this method for the macro optimization of assets with respect to liabilities to ensure the liquidity risk.

3.2 Algorithm

The algorithm for the LPP optimization using simplex method is as follows:

Algorithm 3 The Algorithm for LPP Optimization

1: Objective Function:

$$MaximizeProfit = \Sigma_{j=1}^{m} Asset_j - \Sigma_{i=1}^{n} Liability_i \tag{3.1}$$

which can further be simplified as

$$MaximizeG_b = \sum_{a=1}^{k-1}\sum_{c=1}^{m}\{Y_c^{a,b}+Y_c^{a,b}*IA_c^{a,b}*TA_c^{a,b}\}-\sum_{a=1}^{k-1}\sum_{d=1}^{n}\{X_d^{a,b}+X_d^{a,b}*IL_d^{a,b}*TL_d^{a,b}\} \tag{3.2}$$

2: Assumption:
a. No amount is expected to be paid or received from previous time buckets as there are no assets and liabilities in previous time bucket.
b. We have 5 assets and 5 liabilites.

3: Constraints :
For First Time Bucket:

$$X_1^{1,3} + X_2^{1,4} + X_3^{1,5} + X_4^{1,5} + X_5^{1,6} + X_1 = Y_1^{1,2} + Y_2^{1,3} + Y_3^{1,5} + Y_4^{1,6} \tag{3.3}$$

For Second Time Bucket:

$$[Y_1^{1,2}+Y_1^{1,2}*IA_1^{1,2}*TA_1^{1,2}]+X_1^{2,4}+X_2^{2,4}+X_3^{2,5}+X_4^{2,6}+X_5^{2,7}+G_2 = Y_1^{2,3}+Y_2^{2,4}+Y_3^{2,5}+Y_4^{2,7}+Y_5^{2,7} \tag{3.4}$$

and so on until we have time buckets left.

4: Convert constraint inequality to equality by adding slack variables to the constraints.

5: Use objective function and constraints of LPP to create initial Simplex table.

6: Find out initial solution by assigning 0 to decision variable

7: Optimality Test:
a. Calculate c_j - z_j
b. If the calculated values are positive then optimal solution is the current basic solution. The greatest value column is the key column.
c. If any one value is negative then choose the greatest value corresponding variable.

8: Feasibility Test:
Computr the ratios by dividing the value under XB Column by corresponding valueo of key column. The minimum value of ratio identifies the key row.

9: Key Element:
The intersection of key column and key roe gives the key element.

10: Updating the Table: a. For key row use formula

$$New\_Value = \frac{Old\_Value}{Key\_Value} \tag{3.5}$$

b. For Other rows use formula:
New_Value = Old_Value - $\frac{Corresponding\_key\_column\_value*corresponding\_key\_row\_value}{Key\_Value}$

11: Repeat step 7 to 10 until all the values of c_j - z_j are 0 or negative.

3.3 Illustration

The problem is as described in Figure 3.1:

- A1, A2,....A9 are the asset variables to be optimized.
- L1, L2,...L6 are the given liabilities.
- Interest rates on assets are given.

A	B	C	D	E	F
A3	0			Six month (L3)	5000
A4	0			One year deposit (L4)	15000
A5	0			Three year deposit (L5)	25000
A6	0			Five year deposit (L6)	20000
A7	0			**Total Liability**	**100000**
A8	0				
A9	0				
				Assets	**Interest Rates for Assets**
Objective				Cash	0
				Reserve againse deposit	1.725
Maximize Z	0			Provision	1.725
				Deposit head office	4.5
Constraints				Three months loan	4.65
		InEquality		Six month loan	4.65
1	0	>=	600	One year loan	4.875
2	0	<=	1500	Three year loan	4.9
3	0	>=	600	Five year loan	5.025
4	0	>=	5000		
5	0	>=	5000		
6	0	<=	7500		
7	0	<=	54000		
8	0	>=	0		
9	0	<=	45000		
10	0	<=	60000		
11	0	<=	65000		
12	0	<=	80000		
13	0	<=	20000		
14	0	>=	100000		
15	0	>=	0		
16	0	>=	0		
17	0	>=	0		
18	0	>=	0		
19	0	>=	0		
20	0	>=	0		
21	0	>=	0		
22	0	>=	0		
23	0	>=	0		

Figure 3.1: LPP Problem Formulation in Excel

The solution to the problem is as follows:

We have used the solver plugin for computation of the solution. The Assets column shows the Result and the maximized profit is shown in "Maximize Z".

A	B	C	D	E	F
Variables	**Asset**			**Liability**	
A1	1250			Current deposit (L1)	20000
A2	87500			Three month deposit (L2)	15000
A3	3750			Six month (L3)	5000
A4	0			One year deposit (L4)	15000
A5	0			Three year deposit (L5)	25000
A6	0			Five year deposit (L6)	20000
A7	0			**Total Liability**	**100000**
A8	3750				
A9	3750				
				Assets	**Interest Rates for Assets**
Objective				Cash	0
				Reserve againse deposit	1.725
Maximize Z	1948.125			Provision	1.725
				Deposit head office	4.5
Constraints				Three months loan	4.65
		InEquality		Six month loan	4.65
1	1250	>=	600	One year loan	4.875
2	1250	<=	1500	Three year loan	4.9
3	87500	>=	600	Five year loan	5.025
4	5000	>=	5000		
5	5000	>=	5000		
6	7500	<=	7500		
7	11250	<=	54000		
8	0	>=	0		
9	7500	<=	45000		
10	0	<=	52500		
11	0	<=	57500		
12	0	<=	72500		
13	0	<=	16250		
14	100000	>=	100000		
15	1250	>=	0		
16	87500	>=	0		
17	3750	>=	0		
18	0	>=	0		
19	0	>=	0		
20	0	>=	0		
21	0	>=	0		
22	3750	>=	0		
23	3750	>=	0		

Figure 3.2: LPP Problem Solution in Excel

3.4 Computational Complexity

The worst case complexity of Simplex Method is Exponential but in practice it's complexity is in polynomial time.

3.5 Conclusion

In this chapter we have addressed the problem of liquidity risk management using single objective optimization method.

Chapter 4

LSTM implementation for Prediction of Stock Price

4.1 Prior Work

Prediction of stock prices is the famous problem for several years now. In the era of deep learning various researchers have made the contributions to this topic. Hao Li et al. (2018)[] proposed a Attention based Multi Input LSTM for prediction of opening price given the historical values. It uses attention mechanism for noise filtering and extracting the related information from related stocks. Yao Qin et al. (2017)[] proposed a dual attention based RNN for prediction of treading index using the historical prices of 81 different stocks. The dual attention consists of the input attention mechanism for encoder and a temporal attention mechanism for decoder. Rather et al. (2015)[] proposed the hybrid approach containing RNN and statistical models for prediction of stock price.

4.2 Algorithm

The data is collected prior from 2013 to 2018 for the historical stock price of Infosys stock named INFY. The algorithm is trained on the 90% of the data and tested on the rest of the 10% of the data. The algorithm is as follows:

- **Task:** Prediction of Stock Price

- **Data:** $\{\mathrm{X_i} : \mathrm{Open_i}, \mathrm{High_i}, \mathrm{Low_i}\ ; \mathrm{Y_i} : \mathrm{Close_i}\}_{i=1}^{N}$

- **Model:** LSTM:

$$h_t, C_t = LSTM(h_{t-1}, C_{t-1}, X_t) \tag{4.1}$$

- **Parameters:** W_f, b_f, W_i, b_i, W_c, b_c, W_o, b_o

- **Loss Function:**

$$MeanSquaredError(MSE) = \frac{1}{n}\Sigma_{i=1}^{n}(Predicted\_Y_i - Y_i)^2 \tag{4.2}$$

- **Algorithm:** Adam is the algorithm used for learning the parameters for the LSTM model.

Algorithm 4 Learning Parameters of LSTM for Prediction of Stock Price

1: Parameter Initialization: {strategy = Uniform}
2: *epoch* $\leftarrow$ $Positive\_Integer$
3: *while(eopch > 0)*:
4:     Forward Propagation of Data in Model
5:     Compute Loss using Loss Function
6:     Compute Gradients with respect to Loss
7:     Update the Parameters in Backward Propagation
8:     *epoch* $\leftarrow$ *epoch* $- 1$

Algorithm 5 Forward Propagation

1: Compute Vactor Product of Parameters and Input.
2: Pass Output of step 1 to Activation Function.
3: Repeat step 1 and 2 until we reach Linear Output Activation Function.
4: Send the Predicted Value which we got as Output to Loss Function.

Algorithm 6 Compute Loss

1: The Loss is Computed using Loss Function described above.
2: The loss is computed using Predicted Output from step 4 of Algorithm 4 and Actual Output.

Algorithm 7 Back propagation

1: Using Gradient computed in step 6 of Algorithm 4 update the parameters of the Output Layer.
2: while(reach Input Layer from Output Layer):
3:     Compute Gradients using chain rule.
4:     Update the parameters in Backward Propagation.

4.3 Illustration

The experimental setup was done in Google Colab, a facility provided by Google Inc. to students and researchers for training deep learning algorithms where they provide free GPU. The detailed setup is discussed in Appendix. The computational specification of Colab is as follows:

- CPU model name: Intel(R) Xeon(R) CPU @ 2.30GHz
- No. of Processors : 2
- CPU Cache Size : 46080 KB
- GPU Name : Tesla T4
- GPU Memory : 16 GB (15079MiB usble)
- RAM : 12.9 GB

The programming language used for performing experiment is Python 3.6. The packages used for the experiment include sklearn, keras, math, time, pandas, numpy, pandas_datareader, matplotlib, h5py,statsmodels, etc.
The hyper-parameters that were fine tuned during the experiment for better performance of the model are as follows with their fine tuned values:

- seq_len = 22 # Input Window Size
- shape = [4, seq_len, 1] # No. of Features, Window, Output
- epochs = 90 # No. of times the the forward and backward propagation happens on whole training data through the model
- dropout rate = 0.3 # that percent of neurons from the model will be dropped for each epoch randomly
- decay = 0.2 # parameter decay rate for Adam optimizer
- neurons = [512, 512, 64, 1] # the different combinations of neurons is tried for optimization

The architecture of the model used for experimentation is as follows:

Layer (type)	Output Shape	Param #
lstm_40 (LSTM)	(None, 22, 512)	1058816
dropout_40 (Dropout)	(None, 22, 512)	0
lstm_41 (LSTM)	(None, 512)	2099200
dropout_41 (Dropout)	(None, 512)	0
dense_39 (Dense)	(None, 64)	32832
dense_40 (Dense)	(None, 1)	65

Total params: 3,190,913
Trainable params: 3,190,913
Non-trainable params: 0

Figure 4.1: The architecture of the LSTM model

4.4 Computational Complexity

Computational complexity of LSTM is O(Z),

where Z = 4 * #IP * h + 4 * h^2 + 3 * h + h * #OP ,

#IP: Number of inputs

#h: Number of hidden layers

#OP: Number of outputs

The total time taken by the model to train is 1 min 9 sec.

4.5 Results

After training the LSTM model we have tested the model on unseen test data and the MSE is as follows:

- MSE on Train Data: 0.00128 MSE
- MSE on Test Data: 0.00230 MSE

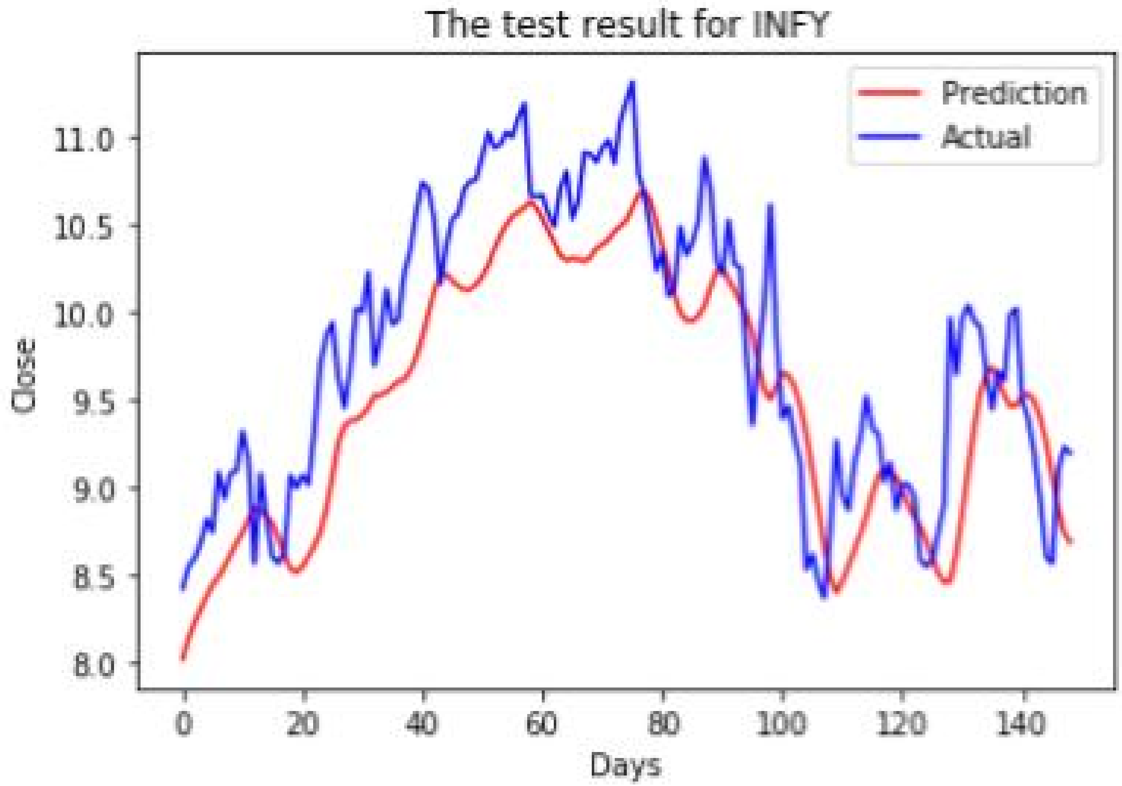

Figure 4.2: Plot showing of Actual and Predicted values

The plot and the results shows that the predicted sequence by the model is trying to catch the actual sequence. As the MSE is very small our predictions are closer to the actual outputs.

4.6 Comparative Analysis

The predictions of ARIMA and Auto-ARIMA models were taken into consideration for comparison with predictions of LSTM model.

- MSE on Test Data of ARIMA Model: 0.576 MSE
- MSE on Test Data of Auto-ARIMA Model: 0.695 MSE

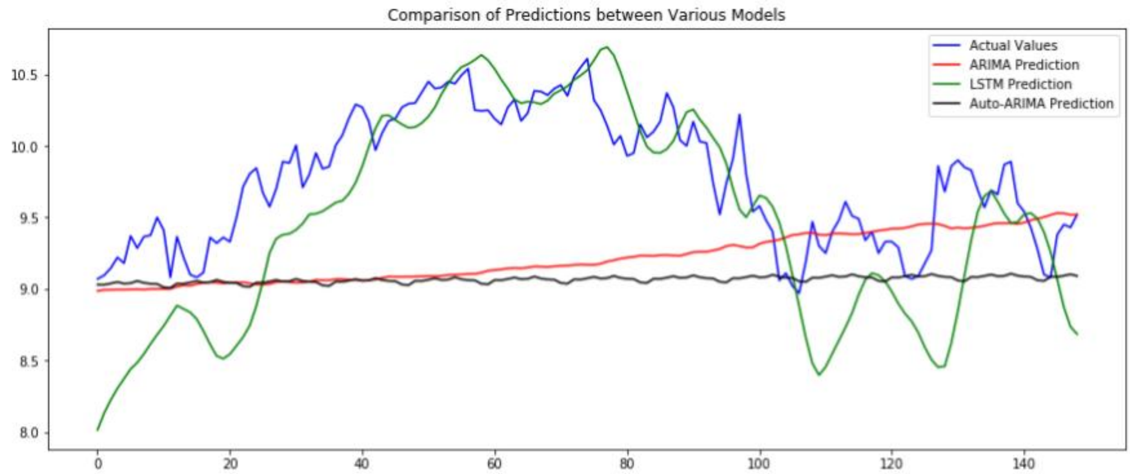

Figure 4.3: Comparison of Prediction between ARIMA, Auto-ARIMAl and LSTM Model with Actual Values

The plot and the results shows that the ARIMA and Auto-ARIMA models are performing worst as compared to LSTM model.

4.7 Conclusion

In this chapter we have discussed about the model architecture, its parameters, and tuning the parameters for optimization of performance of the model. The LSTM model performed far better than the statistical models.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

The work done in this thesis tries to solve ALM problem for bank. The second chapter simulates the ALM environment of the bank and gives the brief understanding of the concept. It also gives nice insights from the data which can be utilized for maximizing profit.

The third chapter discusses the single objective optimization method for liquidity risk management using Linear Programming Formulation using objective function as maximizing the profit of the bank.

The fourth chapter makes an attempt to predict the stock price of the particular stock based on it's historical values using the novelty of the deep learning techniques. The LSTM model trained on the data completely outperforms the traditional statistical models.

5.2 Future Work

The single objective linear optimization can be updated by multi objective optimization and non linear optimization. A fuzzy controller can be formulated for the same purpose.

The LSTM model can be replaced by attention based models, dual attention based models and also with Multi input based models. For creating features the stastical approaches can be used as Moving Average, moving average convergence divergence (MACD), Momentum, Relative Strength Index, Average Directional Index, etc.