ANALYSE OF INTEREST RATE – MODELING AND FORESTING TIME SREIES

STAT 790 Case Seminar--Spring 2015
Mentor: Prof. Subir Dhamoon
Student: Tim Xiaofu Lin

Contents

Abstract

This project is a study of time series forecast. Three methods of forecasting are modeled and compared to find the most accurate and appropriate model to forecast interest rates with different maturity years. These methods are Autoregressive Integrated Moving Average (ARIMA), Holt's Linear Trend Model, and Neural Networking model. In conclusion, Holt's Linear Trend Model is determined to be the best model based on measurement of accuracy.

Key Words: time series forecast, Autoregressive Integrated Moving Average, exponential smoothing, neural networking, interest rate

1. Introduction

Models are often inexpensive and efficient solution for making reliable predictions. A time series model is the study of observations based on time. Most often the measurements are made at regular time intervals. The purpose of time series model is to explain how the past affects the future and to forecast future values based on the series.

Important characteristics of time series are trend, seasonality and cyclic. A trend in time series exists when there is, on average, a long-term increase or decrease in the data. A seasonal pattern exists when there is a regularly repeating pattern of highs and lows according to fixed periods. A cyclic pattern exists when data, on long run, exhibits rises and falls unrelated to seasonally factors.

There are two basic types of time series models; autoregressive integrated moving average (ARIMA) and exponential smoothing, used to forecast future values. ARIMA models might include autoregressive terms, moving average terms and, and differencing operations to describe the autocorrelations in the data. Exponential smoothing often flattens out the irregular peaks of the data to illustrate a clearer pattern based on a description of trend and seasonality in the data. Both are famous method for forecast in time series.

In additional to ARIMA and exponential smoothing, neural networking model is a more advanced forecasting method based on mathematical model of the brain. It has a complex non-linear relationship between response variable and its predictors.

In this paper, we want to find the most accuracy model to predict interest rate using ARIMA, exponential smoothing, and neural network model using statistical software R. Then use the most accuracy-forecasted model to forecast interest rates for different years of maturity. We find a more advanced method of exponential smoothing, Holt's linear trend, is the most appropriate model for forecasting.

2. Data

The dataset is interest rate swaps on Federal Reserve website (http://www.federalreserve.gov/). The data is obtained from International Swaps and Derivatives Association (ISDA®) mid-market par swap rate. Rates are for a Fixed Rate Payer in return for receiving three month LIBOR, and are based on rates collected at 11:00 a.m. Eastern time by Thomson Reuters and published on Thomson Reuters Page ISDAFIX®1.

2.1. Data Exploratory

The data contain monthly interest rate from July 2000 to April 2015 with maturity of thirty, ten, five and two years. There are no missing observations in the data. The time series plot is shown in Figure 1. From the plot, we can see that interest rates with different maturity times behave the same. There is long-term fluctuation in the level of series that decreases from 2000 to 2003, increases from 2003 to 2005, and decreases again from 2007 to 2015. Interest rates stay relative constant between 2006 and 2007. We can also conclude that there is no seasonality for the data since there is not monthly pattern. A model will need to take all these features into account in order to effectively forecast future interest rate.

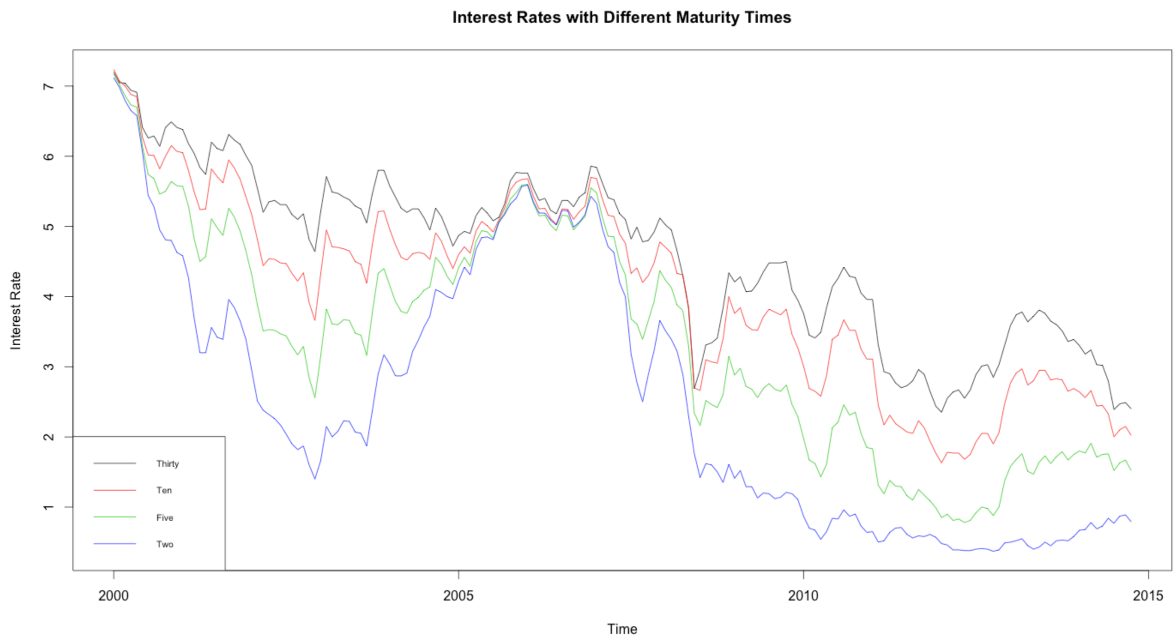

Figure 1. *Interest rate time plot from 2000 to 2015 with thirty (black), ten (red), five (green), and two (blue) years of maturity.*

3. Analysis & Modeling

3.1 ARIMA[4][5]

Autoregressive integrated moving average (ARIMA) might contains combinations of three components. Autoregressive term (AR) describes the linearly relationship with its own previous value. Moving average term (MA) uses past errors multiplied by a coefficient to forecast future value. Integrated term, also refer to differencing, compute the differences between consecutive observations. The model can be written as

$$y'_t = c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 e_{t-1} + \cdots + \theta_q e_{t-q} + e_t$$

where y'_t is the differenced series. The equation on the right side is predictors of differenced series contained both lagged values of y_t and lagged errors. We can also rewrite the whole equation in a clean version using backshift operators. The equivalent equation is

$$
\begin{array}{cccc}
(1-\phi_1 B - \cdots - \phi_p B^p) & (1-B)^d y_t & = & c + (1+\theta_1 B + \cdots + \theta_q B^q)e_t \\
\uparrow & \uparrow & & \uparrow \\
\text{AR}(p) & d \text{ differences} & & \text{MA}(q)
\end{array}
$$

This is an ARIMA (p, d, q) model, where p is the order of autoregressive term, q is the order of the moving average and d is the degree of first differencing involved.

For an effective ARIMA, we need to choose the appropriate values for p, q, and d. There are two methods we can choose these value in R. First method, we can use auto.arima() function in R to choose (p,d,q) automatically using unit root tests, minimization of AICc and MLE to obtain an ARIMA model. The function derivate from Hyndman-Khandakar algorithm as follow:

1) The number of differences *d* is determined based on repeated KPSS tests.
2) The value of p and q are chosen by stepwise search minimizes the AICc after differencing the data d times.
    a. The model with smallest AICc is selected from four models, ARIMA (2,d, 2), ARIMA (0,d, 0), ARIMA (1,d,0), and ARIMA (0,d,1). If d = 0 then constant c is included otherwise c is equals to zero.
    b. Then, vary p and/or q from current model by ± 1 and include/exclude c from the current model. The model with lowest AICc becomes the new model.
    c. Repeat part (b) until no lower AICc can be found.

The second method is to de-trend time series and use ACF and PACF to choose appropriate p and q.

1) Differencing is used to de-trend the data and make it stationary so a pattern can be found to predict de-trend time series. A time series is stationary if its underlying statistical structure does not evolve with time.
2) Sample autocorrelation function (ACF) for series gives correlation between the series x_t and lagged value of the series for lags of 1,2,3…. The ACF can be used to identify order (q) of autoregressive based on statistical significant lag q.
3) Partial autocorrelation function (PACF) is a conditional correlation. The PACF can be used to identify order (p) of moving average based on statistical significant lag p.
4) Then, examine the residuals of model by plotting ACF of the residuals, and perform a portmanteau test of the residuals. If the residuals are white noise then the model is appropriate.

However, in this paper, we are going to use second method to select an appropriate model and confirm with *auto.arima()* function.

Since the interest rate behave similarly for different maturity times (Figure 1). We can just use one set of the interest rate to build ARIMA model and apply the model to rest of the interest rates. We choose the interest rate with thirty years of maturity since it has the most variations on times (Figure 1). Frist, we take the first difference of the data. The differenced data are shown in Figure 2.

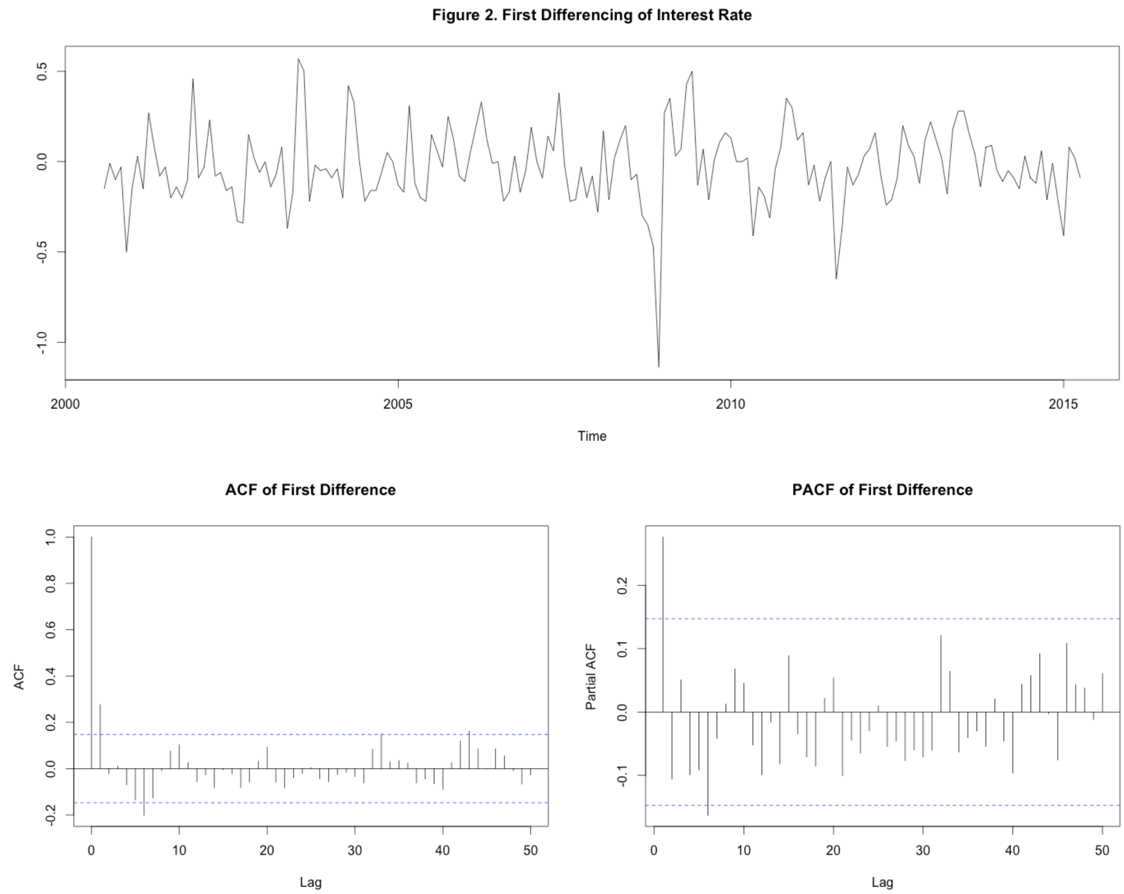

Figure 2. *The first difference of data (top) show there is no trends – possible stationary. Correlogram of First Difference ACF shows a significant peak at lag 1 and 6 (bottom left). Correlogram of First Difference PACF shows significant peak at lag 1 and 6 (bottom right).*

Both ACF and PACF plot have a few significant lags but these die out quickly, so we can conclude our first difference series is stationary.

From ACF correlogram, we see that the autocorrelation at lag 1 exceeds the significance bound and lag 6 also exceed the significant bound, which might be due to chance. We expect the two in fifty of the autocorrelations for the first fifty lags to exceed the 95% significance bound by chance.

From PACF correlogram, lag 1 is positive and exceeds the significant bound. The autocorrelation tail off to zero after lag 1 with exception for lag 6, this might also due to chance.

Since the correlogram tails to zero after lag 1 and the partial correlogram tails off to zero after lag 1, this mean that following ARIMA models are possible with first differences:

1) ARIMA (1,1,0) model, an autoregressive model of order p =1, since the partial autocorrelation after lag 1 tails off to zero.
2) ARIMA (0,1,1) model, a moving average model of order q = 1, since the correlogram is zero after lag 1 and the partial correlogram tails off to zero.
3) ARIMA (p,1,q) model, an mixed model with p and q greater than 1, since the autocorrelogram tails to zero after lag 1.

We want to choose a model that uses the least parameters. ARIMA (1,1,0) and ARIMA (1,1,0) models both have 1 parameter and ARIMA (p,1,q) have more than 1 parameter. Therefore, we choose either ARIMA (1,1,0) or ARIMA (0,1,1). Intuitively, it betters to choose ARIMA model with moving average because interest rate has a short-term dependencies on the previous successive observations. Therefore, we choose an ARIMA (0,1,1) model. This also confirmed by the *auto.arima()* function as the result following:

```
ARIMA(0,1,1)

Coefficients:
         ma1
      0.3357
s.e.  0.0751

sigma^2 estimated as 0.04209:  log likelihood=29.16
AIC=-54.32   AICc=-54.25   BIC=-47.97
```

The equation for model can be written as $y_t = y_{t-1} + \alpha e_{t-1}$ where α is determined by *ma1* in auto.arima() function.

3.1.2 Model Diagnosis

For a good ARIMA model, time series residuals (ε_i) should behave like white noise. This means the residuals should be uncorrelated random variables with constant variance and mean. ACF of residuals is used to check all autocorrelations for residuals series are uncorrelated (if lags are between significant lines). The Ljung-Box statistic is used to exam if correlations for the errors may be 0 for given lag. A small p-value indicates the possible of correlation. The result of the Ljung-Box test of first fifty lags is:

```
                Box-Ljung test

             data:  residuals(fit)
   X-squared = 38.3755, df = 46, p-value = 0.7803
```

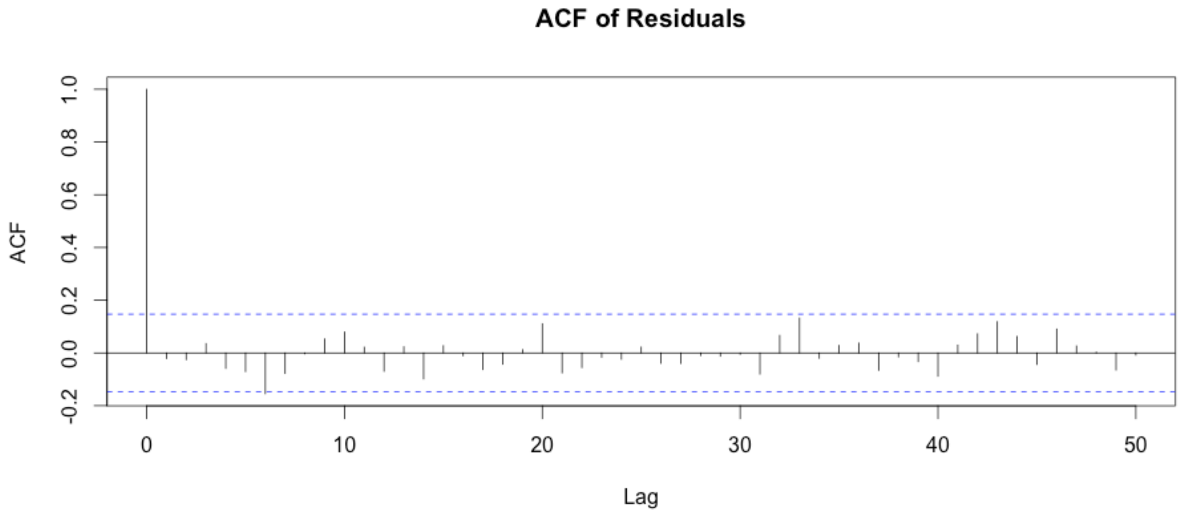

Figure 3. *ACF of residuals of ARIMA (0,1,1) model.*

Ljung-Box statistic test returns a large p-value (0.78) suggests that residuals are not correlated, they are all independent. This can also be seen from ACF of Residual (Figure 3) with slight significant spike at lag 6 which it might be ignored. Therefore, we can conclude that residuals are white noise by Ljung-Box test and ACF of residuals. Thus, ARIMA(0,1,1) is a good model for interest rate dataset. This is confirmed by the time series plot of fitted values by ARIMA(0,1,1) shown in Figure 4 that autoregressive integrated moving average model with d =1 and q = 1 fits well with the original data.

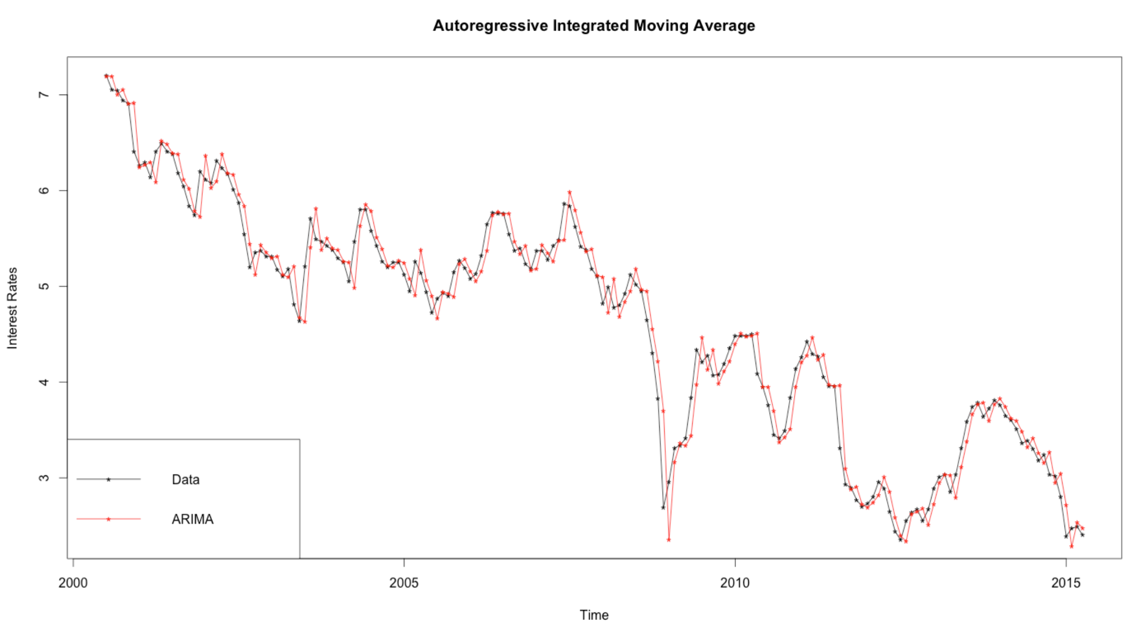

Figure 4. *The black line is the original data and the red line is points fitted by autoregressive integrated moving average model.*

3.2 Exponential Smoothing [4][6]

Smoothing is a method to smooth out the irregular roughness of the pattern in order to see a clear pattern. Simple exponential smoothing is only suitable for forecasting data with no trend; therefore, we use a more advance extension of simple exponential model based on Charles C. Holt's (1959). The method is also known as Holt's linear trend method. This method contains two smoothing parameters, one for trend and one for levels, and a forecasting equation:

$$\begin{aligned} &\text{Forecast equation} && \hat{y}_{t+h|t} = \ell_t + hb_t \\ &\text{Level equation} && \ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\ &\text{Trend equation} && b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \end{aligned}$$

Denotation:

- ℓ_t – an estimate of level of the series at time t
- b_t – an estimate of slope of the series at time t
- α – smoothing parameter for level
- β^* – smoothing parameter for trend which is only between 0 and 1.

Similarly to simple exponential smoothing, level equation is a weighted average of observation y_t and within sample one-step-ahead forest for time t. The trend equation is a weight average of the estimated trend at time t based on the previous estimated of the trend. The forecasting function is trended for h-steps equal to the last estimated level plus the h times the last estimated trend value. It can also be seen as a linear function of h.

3.2.2 Optimization of Holt's Linear Trend Model

For exponential smoothing method, values for smoothing parameters are needed (α and β^*) to make a model sufficient. As now, there is not a function in R that automatic return these values. Therefore, a way to solve this problem is to set up a simulation with different values of smoothing parameters and chooses values that minimized sum of the squared errors (SSE). Sum of the squared errors can be seen as a way to see the fit of the model. A small sum of squared error can suggest that a model fit adequately. The errors are the observed minus the estimated: $e_t = y_t - \hat{y}_{t|t-1}$ for t = 1…T. We want to find the values that minimize

$$SSE = \sum_1^T (y_t - \hat{y}_{t|t-1})^2 = \sum_1^T e_t^2.$$

In R, *holt()* function can be used to build exponential smoothing model with specified alpha and beta. Then, we can calculate sum of squared errors using excel. We use the last 12 observations in the original data as a control in the simulation and create a new dataset without 12 recent observations. Then, we build models use new dataset with various alpha and beta and forecast 12 new interest rates based to calculate the sum of squared errors. We first choose α from 0.75, 0.5, and 0.25 with fixed $\beta^* = 0.2$ that minimized sum of squared errors and test β^* with 0.4, 0.6, and 0.8 to find the best value for β^*. In Table 1, there is summary of simulations used to find best values for α and β^*.

Date	Observed	Predicted/ Estimated values		
		$\alpha = 0.75,$ $\beta^* = 0.20$	$\alpha = 0.50$ $\beta^* = 0.20$	$\alpha = 0.25$ $\beta^* = 0.20$
2014-05	3.36	3.53	3.623	3.959
2014-06	3.39	3.52	3.621	4.004
2014-07	3.3	3.51	3.618	4.049
2014-08	3.18	3.504	3.616	4.095
2014-09	3.24	3.495	3.614	4.141
2014-10	3.03	3.485	3.612	4.188
2014-11	3.02	3.476	3.611	4.236
2014-12	2.8	3.466	3.608	4.283
2015-01	2.39	3.457	3.607	4.332
2015-02	2.47	3.448	3.605	4.381
2015-03	2.49	3.438	3.602	4.431
2015-04	2.4	3.429	3.6	4.481
SSE		5.170936	7.340353	23.48606

Date	Observed	Predicted/ Estimated values		
		$\alpha = 0.75,$ $\beta^* = 0.40$	$\alpha = 0.75,$ $\beta^* = 0.60$	$\alpha = 0.75,$ $\beta^* = 0.80$
2014-05	3.36	3.466	3.441	3.429
2014-06	3.39	3.407	3.365	3.346
2014-07	3.3	3.349	3.291	3.265
2014-08	3.18	3.292	3.218	3.186
2014-09	3.24	3.236	3.147	3.109
2014-10	3.03	3.181	3.078	3.033
2014-11	3.02	3.127	3.011	2.96
2014-12	2.8	3.074	2.944	2.888
2015-01	2.39	3.022	2.879	2.818
2015-02	2.47	2.97	2.815	2.751
2015-03	2.49	2.92	2.753	2.684
2015-04	2.4	2.871	2.693	2.619
SSE		1.191977	0.553645	0.384214

Table 1. *Sum squared error of interest rates for three level parameters and four trend parameters.*

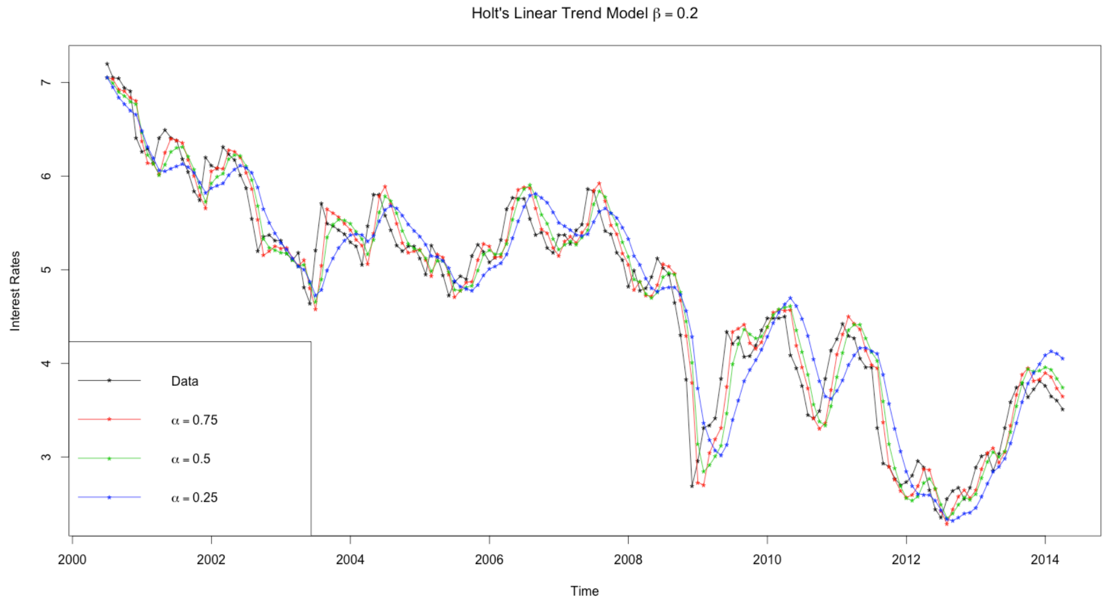

Figure 5a. *Fitted points of Holt's linear trend model with fixed beta = 0.2 and alpha = 0.75, 0.5 and 0.25.*

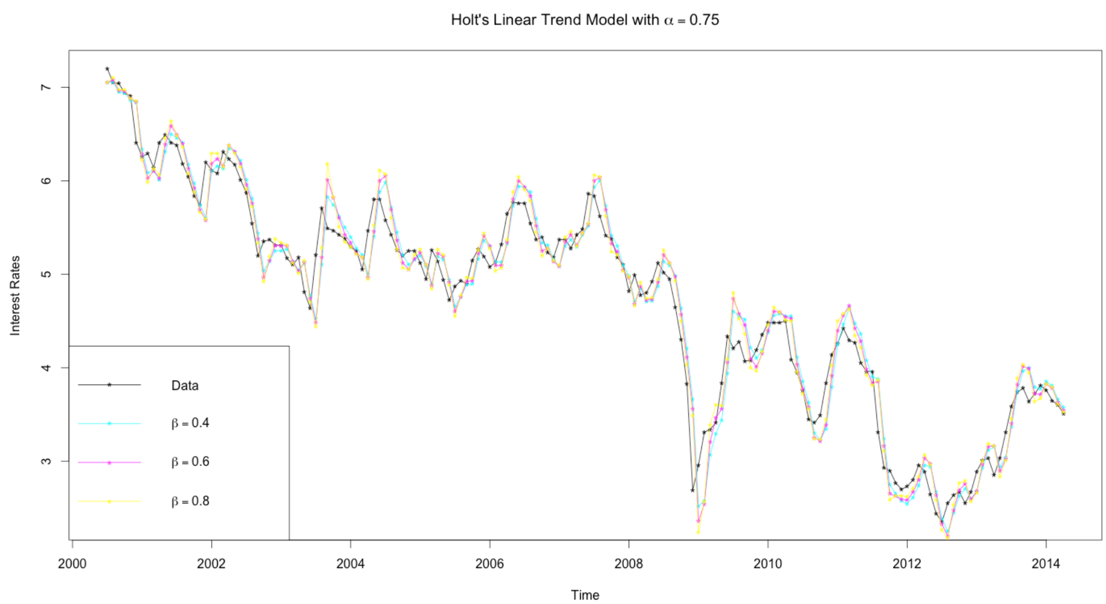

Figure 5b. *Fitted points of Holt's linear trend model with fixed alpha = 0.75 and beta = 0.4, 0.6 and 0.8.*

The sum of squared error is minimized with alpha equals to 0.75 based the table 1. From the time series of fitted values (Figure 5a), we can also see that the red line ($\alpha = 0.75$) is much closer the black line (original data) than other lines. After selecting the proper alpha, sum of squared error is minimized when beta is equal to 0.8. We can also observe that the golden line ($\beta^* = 0.80$) is the closest line to the black line (original data) when time proceeds. Therefore, we can conclude that $\alpha = 0.75$ and $\beta^* = 0.80$ are the best smoothing parameters for Holt's linear trend model.

3.3 Neural Networking Models[3][4]

Neural network models were first developed to simulate human brains. In a brain, there are millions of neurons connected by ten and thousands of synapses. The new information is received by preceptors and is passed down from one neuron to another neuron in an incredible speed to the brain where prior information is stored and simulates a human response. However, we don't have advanced computer to mimic human brain yet. But this idea is applied to data analysis such as regression, classification and clustering.

In statistical model, a neural networking consists of inputs, known as predictors, from the bottom layer, and output, known as forecasts, from the top layer. There could be few or none intermediate hidden layers. A simple neural network model with no hidden layers are equivalent to linear regression (shown in Figure 6).

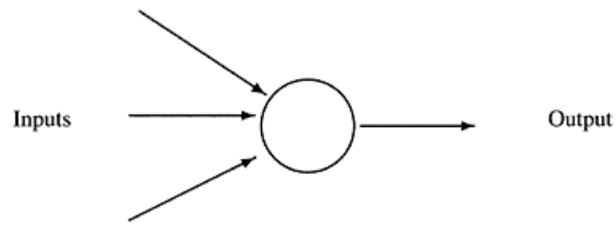

Figure 6. *A simple neural network model with three inputs (predictors in linear model) and one output (response in linear model).*

A more complex neural networking model might contain few intermediate layers in additional to bottom levels. When one intermediate layer is added, neural networking model might be equivalent to multivariate linear regression or polynomial regression (shown in Figure 7).

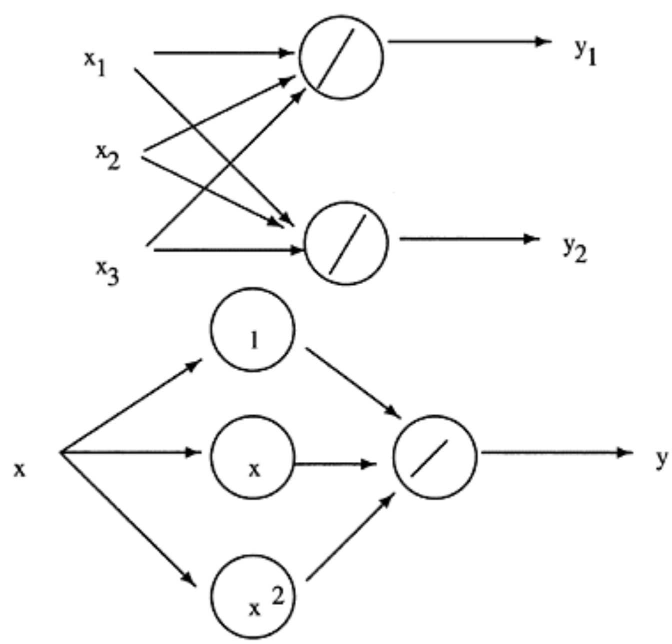

Figure 7. *(Top) Neural networking model is in a form of a multivariate model with two response variables and three predictors. (Bottom) Neural networking model is in a form of second-degree polynomial regression.*

When multiple intermediate layers are added, neural model might become non-linear. This is known as *feed-forward network* (shown in Figure 8) where each layer of node receive input from pervious layers. The inputs of each node are combined using a weighed linear combination. Then, the result is modified by nonlinear function before output.

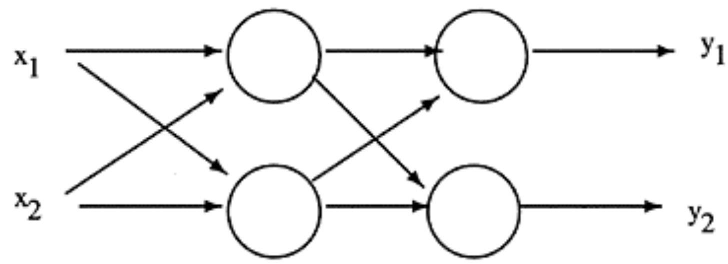

Figure 8. *Feed-forward network model with one intermediate layer, two inputs and two outputs.*

The inputs into intermediate level neuron j are linearly combined to give

$$z_j = b_j + \sum_{all\ i,j} w_{i,j}\, x_i$$

where $w_{i,j}$ is the weights selected in the neural network framework using a criterion that minimize a cost function such as mean squared errors. The weights take on random value at beginning and updated using the observed data. The model usually trained several times using different random starting points, and the results are averaged.

In the intermediate layer, this is then modified using a nonlinear function such as sigmoid,

$$s(z) = \frac{1}{1 + e^{-z}}$$

to give input for the next layer. The number of intermediate layer and the number of nodes are needed to be self-specified.

Neural networking auto regression (NNAR) is often used with time series where lagged values are used as inputs. In R, nnetar() function is used to fits an NNAR(p, k), where p is the number of lagged inputs and k is the number of nodes in intermediate layer. For the interest rate time series data, we are going to let the function to select values of p and k for us. The result is following:

```
Series: irts
Model:  NNAR(1,1)
Call:   nnetar(x = irts)
```

```
Average of 20 networks, each of which is
a 2-2-1 network with 9 weights
options were - linear output units

sigma^2 estimated as 0.04476
```

The nnetar() function fits a model with one lagged input and one node in the hidden layer used an average of 20 networks with 9 weights each. The model fits relative well except at the end where the original data tends to decrease and neural networking model tends to increase (Figure 8). This might be problematic when forecasting for future interest rates.

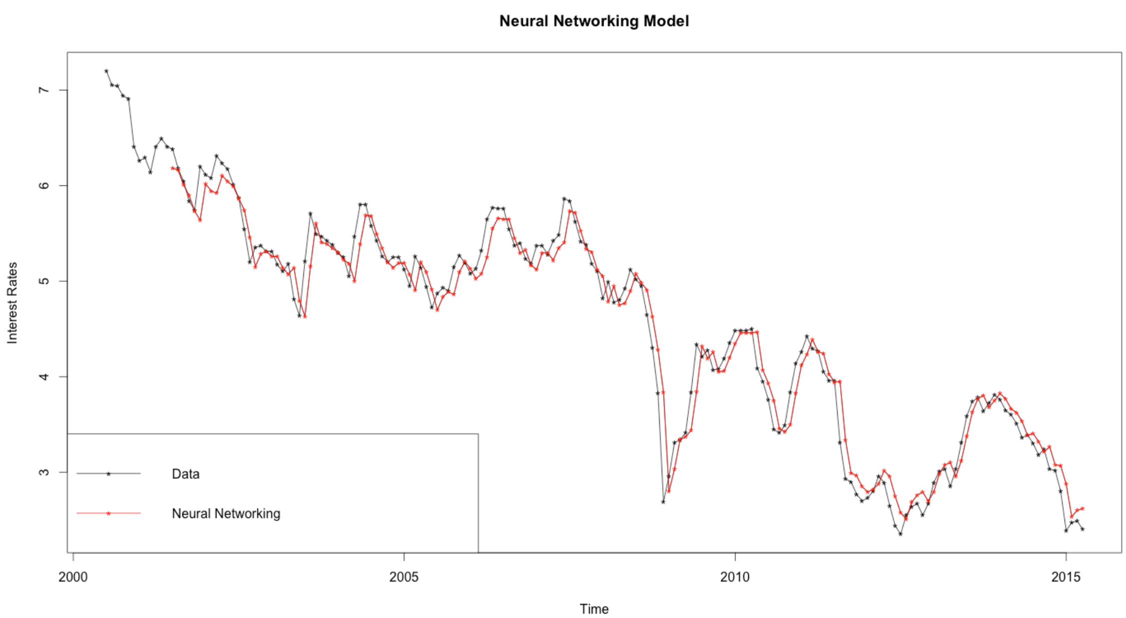

Figure 8. *The fitted values of neural networking model (in red).*

4. Measure of Accuracy[4]

The accuracy of the model is measured by the error (e_i) of observed (y_i) from the forecasted ($\hat{y}_i$) of observation *i*. We can measure the error in following ways:

$$\text{MEAN EEROR: ME} = mean\ (e_i)$$
$$\text{MEAN ABSOULTE ERROR: MAE} = mean\ (|e_i|)$$
$$\text{ROOT MEAN SQAURE ERROR: RMSE} = \sqrt{mean\ (e_i^2)}$$

where $e_i = y_i - \hat{y}_i$.

We can also measured the percentage error where $p_i = \frac{e_i}{y_i} \times 100$. Common measures are:

MEAN PERCENTAGE ERROR: MPE = $mean\ (p_i)$
MEAN ABSOLUTE PERCENTAGE ERROR: MAPE = $mean\ (|p_i|)$

This measure is a little problematic when observation (y_i) is zero because percentage error would be undefined. When the observation is a meaningful zero then information might be lost. However, the interest dataset contains no zero observations, percentage errors can be used in this situation to reflect the accuracy of model.

In additional to percentage error, accuracy can also measured by scaled error. The error is scaled based on the training MAE from a simple forecast method:

$$q_j = \frac{e_j}{\frac{1}{n-1} \sum_{i=2}^{n} |y_i - y_{i-1}|}$$

where q_j is independent of the scale of the data since both numerator and denominator are values on the scale of the original data. The scaled error is measured as

MEAN ABSOULTE SCALED ERROR: MASE = $mean\ (|q_i|)$

comparing forecast accuracy across series on different scales. For measurement of accuracy, the desired result would be smallest possible value for absolute of errors and near zero for average of errors.

In R, the function *accuracy()* in forecast package returns summary of the forecast accuracy. The measures calculated include mean error, root mean absolute error, mean percentage error, mean absolute percentage error, and mean absolute scaled error. The function used cross validation method to measure the accuracy of the prediction a model performed. For cross validation, one dataset is randomly divided into two datasets (one for training and one for testing). The model is built based on the training dataset and forecasts are using the predictors of testing dataset. Then the error is obtained from the difference of observed and forecasts based predictors of the testing dataset.

In addition to function *accuracy()* , we also cut off the last 12 observations to compare to the forecast values make by each models similarly to simulation in finding smoothing parameters for Holt's linear trend model. Therefore, we can calculate the sum of squared errors to determine the accuracy of prediction for each model.

5. Model Selection

From the analysis, we find that interest rate with maturity of thirty years can be fit with

1. Autoregressive integrated moving average with parameter d=1, and q =1.
2. Holt's linear trend model with $\alpha = 0.75$ and $\beta^* = 0.8$.
3. Neural networking model with p = 1, and k =1.

The forecast of 12 observations and the summary models' accuracy is shown in table 2.

Autoregressive integrated moving average model fit the model most adequately since it has the smallest least root mean square error, mean absolute error, and mean absolute percentage error, However, the forecasts are the same (Figure 9) and sum of square error is large (Table 2). Therefore, it is not considerate a appreciate model for prediction future values.

Holt's linear trend model predicts the last 12 times most precisely with least sum of squared errors (Table 2). It has a decreasing trend with values closest to last 12 observations (Figure 9). It has the lowest accuracy of all three models, however we can ignore that because measurements of error don't varies much from other two models (Table 2). Therefore, it is consider as a possible model for forecasting.

Neural Networking model has the lowest mean error. The other measurements of error are similar to autoregressive integrated moving average model, but the sum of squared error is the greatest of all three models (Table 2). From the plot, we can see the forecasts follow a cyclic pattern from pervious time. Since our time series is not seasonal and the sum square error is large, neural networking is no an adequate model for forecasting.

Considering measurements of error and sum of squared errors Holt's linear trend model is the most appropriate model used to forecast interest rate with different maturity years.

Table 2. *(Top) The measure of accuracy for autoregressive integrated moving average model, Holt's Linear Trend Model, and Neural Networking Model.* (Bottom) *Observed values and forecasted values of for a year starting from May 2014.*

Measurement of Accuracy for Interest Rate with Maturity of 30 years.						
Models	*ME*	*RMSE*	*MAE*	*MPE*	*MAPE*	*MASE*
ARIMA	-0.0202	0.204	0.147	-0.557	3.546	0.933
Holt's Linear Trend	-0.0132	0.249	0.192	-0.275	4.476	0.318
Neural Networking	0.0005	0.218	0.155	-0.651	3.885	0.986

Time	Observed Interest Rates	Forecast Interest Rates With		
		ARIMA	Holt's Linear Trend Model	Neural Networking

2014-05	3.36	3.48	3.42	2.96
2014-06	3.39	3.48	3.34	3.21
2014-07	3.30	3.48	3.26	3.52
2014-08	3.18	3.48	3.18	3.70
2014-09	3.24	3.48	3.10	3.80
2014-10	3.03	3.48	3.03	3.59
2014-11	3.02	3.48	2.96	3.66
2014-12	2.8	3.48	2.88	3.80
2015-01	2.39	3.48	2.81	3.77
2015-02	2.47	3.48	2.75	3.64
2015-03	2.49	3.48	2.68	3.59
2015-04	2.4	3.48	2.61	3.68
SSE		**5.4337**	**0.3723**	**8.669**

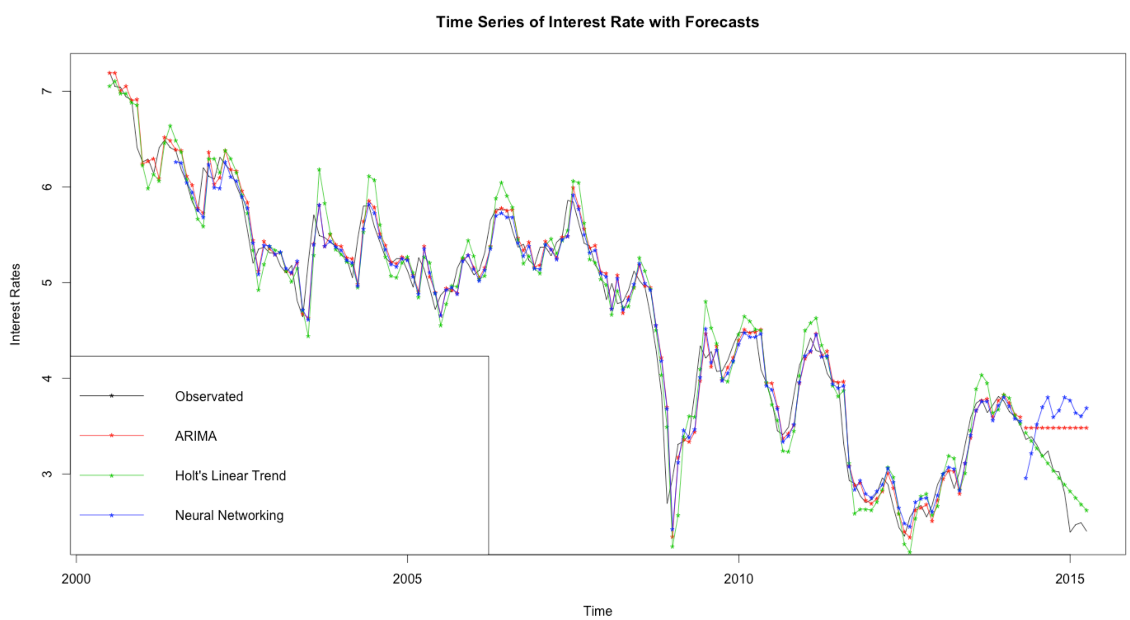

Figure 9. *Observed values and forecasted values of for a year starting from May 2014.*

6. Results and Discussion

Forecasts of interest rate with maturity years of thirty, ten, five and two years are shown in Table 3. From the time series plot (Figure 9), we can see that the interest rates have a decreasing trend. The results are as expected since Holt's linear trend model use pervious observations to forecast new prediction. From the time series plot, we can see that interest rates started to decrease since the year of 2012, so it make sense that the interest rates for 2015 are also like to be decreasing.

However, there are many factors that can affect interest rates at a time. These factors include inflationary pressures, actions of federal government, and foreign exchange of US dollars[2]. From time series plot (Figure 10), we can see that that there is a sharp decrease in interest rate due to Federal regulation to fight the recession [1]. These factors greatly influence the interest rate. Therefore, we can only forecast interest rate without the influence of these factors. We can only say that under no other external influence the interest rate for next year for any maturity year is likely to decrease.

From the Holt's linear model, the interest rate follows a decreasing trend. If we want to forecast the interest rate future, such as ten years from 2015, the interest rates are likely to converge to zero. Therefore, Holt's linear trend model is only appreciated for short term forecasting. Other methods are need for long-term forecast interest rate.

In conclusion, we find that Holt's linear trend model is best to forecast interest rates. However, there are some disadvantages such as that the interest rates might converge to zero for long-term forecasting. There also external factors that might influence interest rates that we need to taken into consideration when predicting future rates.

Time	**Forecast Interest Rates With**			
	Thirty	**Ten**	**Five**	**Two**
2015-05	2.39	2.02	1.50	0.785
2015-06	2.37	1.98	1.46	0.746
2015-07	2.35	1.95	1.41	0.709
2015-08	2.34	1.92	1.36	0.673
2015-09	2.32	1.89	1.31	0.641
2015-10	2.30	1.85	1.27	0.608
2015-11	2.28	1.83	1.22	0.578
2015-12	2.26	1.80	1.18	0.549
2016-01	2.24	1.77	1.14	0.521
2016-02	2.22	1.74	1.10	0.496
2016-03	2.21	1.71	1.06	0.471
2016-04	2.19	1.68	1.04	0.447

Table 3. *Forecasts for a year starting from May 2015 for different maturity times.*

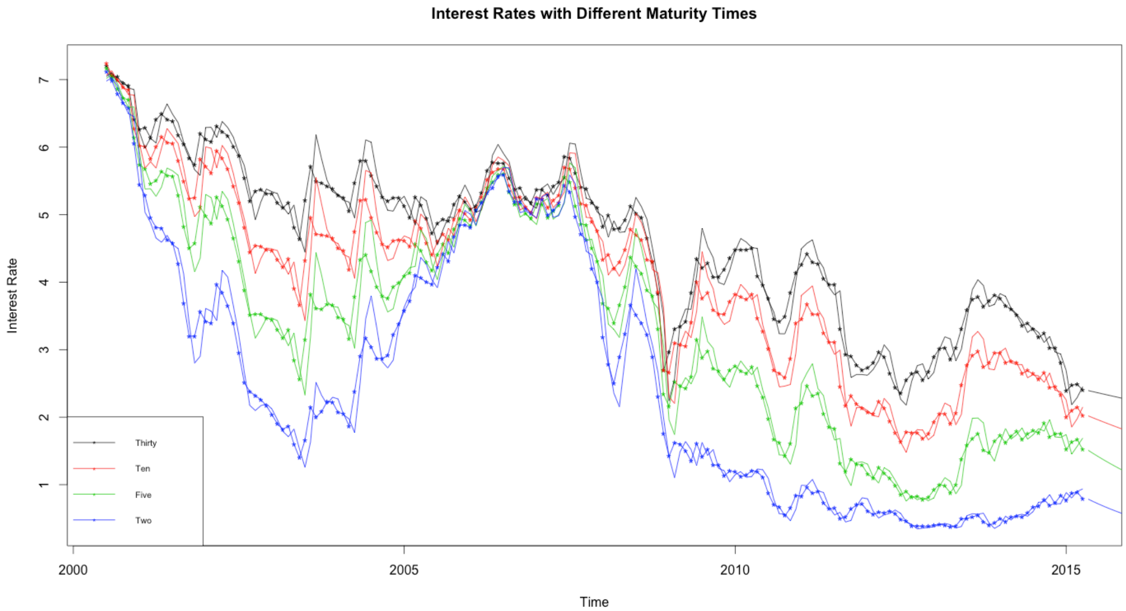

Figure 10. *Forecasted values of for a year starting from May 2015 for different maturity times.*

Reference

1. Andrews, Edmund L., and Jackie Calmes. "Fed Cuts Key Rate to a Record Low." *The New York Times*. The New York Times, 16 Dec. 2008. Web. 09 July 2015.

2. "Factors Influencing Interest Rates." - *Financial Web*. 09 July 2015.

3. Faraway, Julian James. *Extending the Linear Model with R: Generalized Linear, Mixed Effects and Nonparametric Regression Models*. Boca Raton: Chapman & Hall/CRC, 2006. Print.

4. Hyndman, Rob J., and George Athanasopoulos. *Forecasting: Principles and Practice*. Web.

5. "Lesson 3: Identifying and Estimating ARIMA Models; Using ARIMA Models to Forecast Future Values." *Lesson 3: Identifying and Estimating ARIMA Models; Using ARIMA Models to Forecast Future Values*. Web. 09 July 2015.

6. "Lesson 5: Smoothing and Decomposition Methods and More Practice with ARIMA Models." *Lesson 5: Smoothing and Decomposition Methods and More Practice with ARIMA Models*. N.p., n.d. Web. 09 July 2015.

Appendix A. Related R Codes

```
#Reading Data into R
interest <- read.csv("~/Desktop/interest.csv")
#Create and Plot Time series
ts = ts(interest[,-1], start = c(2000,7), frequency = 12)
plot(ts, plot.type = "single", col = 1:ncol(ts), main = "Interest Rates with Different
Maturity Times", ylab = "Interest Rate")
legend("bottomleft", colnames(ts), col=1:ncol(ts), lty=1, cex=.65)

## Create Time Series with Thirty Years of Maturity
irts = ts(interest[,2], start = c(2000,7), frequency=12)
plot(irts)

#ACF of Time Series with Thirty Years of Maturity
par(mfrow =c(1,1))
acf = acf(irts, plot = FALSE, lag.max = 50)
acf$lag = acf$lag*12

plot(acf, main = "ACF of Interest Rate")
#PACF of Time Series with Thirty Years of Maturity
pacf = pacf(irts, plot = FALSE,lag.max = 50)
pacf$lag = pacf$lag*12
plot(pacf, main = "PACF of Interest Rate")

#First Difference of Time Series with Thirty Years of Maturity
diff <- diff(irts, differences=1)
plot(diff, main = "Figure 2. First Differencing of Interest Rate", ylab = "")

#ACF of First Difference
par(mfrow =c(1,2))
acf = acf(diff, plot = FALSE,lag.max=50 )
acf$lag = acf$lag*12
plot(acf, main = "ACF of First Difference")

#PACF of First Difference
pacf = pacf(diff, plot = FALSE,lag.max=50)
pacf$lag = pacf$lag*12
plot(pacf, main = "PACF of First Difference")

# ARIMA model
arima1 = Arima(irts, order = c(0,1,1))
plot(irts, main = "Autoregressive Integrated Moving Average", ylab = "Interest Rates",
type = "o", pch = "*")
lines(fitted(arima1), col = 2, type = "o", pch ="*")
legend("bottomleft",lty=1, col=1:2, c("Data", "ARIMA"),pch="*")

#Auto Arima
auto.arima(irts,stepwise = FALSE)

#ARIMA Model Diagnosis
acf = acf(residuals(fit), plot = FALSE, lag.max=50 )
acf$lag = acf$lag*12
plot(acf, main = "ACF of Residuals")

Box.test(residuals(fit), lag=50, fitdf=4, type="Ljung")

# Fitting Holt's Linear Trend models
ts1  = ts(irts[-c(167:178)], start = c(2000,7), frequency=12)
```

```
fit = holt(ts1, alpha=0.75, beta=0.2, initial="simple", exponential=TRUE, h =12)
summary(fit)
fit2 = holt(ts1, alpha=0.50, beta=0.2, initial="simple", exponential=TRUE, h =12)
summary(fit2)
fit3 = holt(ts1, alpha=0.25, beta=0.2, initial="simple", exponential=TRUE, h =12)
summary(fit3)
fit4 = holt(ts1, alpha=0.75, beta=0.4, initial="simple", exponential=TRUE, h =12)
summary(fit4)
fit5 = holt(ts1, alpha=0.75, beta=0.6, initial="simple", exponential=TRUE, h =12)
summary(fit5)
fit6 = holt(ts1, alpha=0.75, beta=0.8, initial="simple", exponential=TRUE, h =12)
summary(fit6)

# Finding Opitmal Holt's Model
plot(ts(irts[-c(167:178)], start = c(2000,7), frequency=12), main =
expression(paste("Holt's Linear Trend Model ",beta == 0.2)), ylab = "Interest Rates", type
= "o", pch = "*")
lines(fitted(fit), col = 2, type ="o", pch = "*")
lines(fitted(fit2), col = 3, type ="o", pch = "*")
lines(fitted(fit3), col = 4, type ="o", pch = "*")
legend("bottomleft",lty=1, col=1:4, c("Data", expression(alpha == 0.75), expression(alpha
== 0.5),expression(alpha == 0.25)),pch="*")
plot(ts(irts[-c(167:178)], start = c(2000,7), frequency=12), main =
expression(paste("Holt's Linear Trend Model with ",alpha == 0.75)) , ylab = "Interest
Rates", type = "o", pch = "*")
lines(fitted(fit4), col = 5, type ="o", pch = "*")
lines(fitted(fit5), col = 6, type ="o", pch = "*")
lines(fitted(fit6), col = 7, type ="o", pch = "*")
legend("bottomleft",lty=1, col=c(1,5:7), c("Data", expression(beta == 0.4),
expression(beta == 0.6),expression(beta == 0.8)),pch="*")

## Neural Networking Model
library(caret)
nn = nnetar(irts)
plot(irts, main = "Neural Networking Model", ylab = "Interest Rates", type = "o", pch =
"*")
lines(fitted(nn), col = 2, type = "o", pch = "*")
legend("bottomleft",lty=1, col=1:2, c("Data", "Neural Networking"),pch="*")


## Accuracy of Model without last 12 observations
a = forecast(Arima(ts1, order = c(0,1,1)), h =12)
b = forecast(fit6, h =12)
c = forecast(nnetar(ts1), h =12)
accuracy(arima1)
accuracy(fit6)
accuracy(nn)

## Plot of last 12 forecast
plot(irts, main = "Time Series of Interest Rate with Forecasts", ylab = "Interest Rates",
pch = "*")
lines(fitted(a), col = 2, type ="o", pch = "*")
lines(fitted(b), col = 3, type ="o", pch = "*")
lines(fitted(c), col = 4, type ="o", pch = "*")
lines(a$mean, col = 2,type ="o" ,pch ="*")
lines(b$mean, col = 3,type ="o" ,pch ="*")
lines(c$mean, col = 4,type ="o" ,pch ="*")
legend("bottomleft",lty=1, col=1:4, c("Observated", "ARIMA","Holt's Linear Trend" ,"Neural
Networking"),pch="*")
```

```
## Forecast Interest Rates with Different Maturity Times
plot(ts, plot.type = "single", col = 1:ncol(ts), main = "Interest Rates with Different
Maturity Times", ylab = "Interest Rate",pch ="*", type ="o")
legend("bottomleft", colnames(ts), col=1:ncol(ts), lty=1, cex=.65, pch = "*")
for (i in 1:4) {
  lines(fitted(holt(ts[,i], alpha=0.75, beta=0.8, initial="simple", exponential=TRUE, h
=12)), col =i)
  lines((holt(ts[,i], alpha=0.75, beta=0.8, initial="simple", exponential=TRUE, h
=12)$mean), col =i)
  print(holt(ts[,i], alpha=0.75, beta=0.8, initial="simple", exponential=TRUE, h =12))}
```