Task-by-Task Guide

If you'd like a little more support while completing this project, explore this step-by-step resource to get additional hints and resources to help you along each task of this project.

Task 1 - Load and explore the data

For this project, we will be working with a built-in R data set containing the hourly and daily count of rental bikes between the years 2011 and 2012 in the Capital bike share system in Washington, DC, with the corresponding weather and seasonal information.

You may spend some time reading a well-detailed description of the data here.

In this first task, you will load data, install required packages and perform extensive data exploration by asking yourself some questions to uncover insights. The whole idea for this task is to get a solid understanding of the data. Don't forget to write out some insights that you have gained along with your analysis.

Note: I did not provide a CSV file containing this data set because the data set is an inbuilt data set in the timetk package. The data is called **bike_sharing_daily.**

Hint
To load the data in R, you will need to install the timetk package (if not already installed) and import the package using the **library()** function. Once the package is successfully imported, use the function **data("dataset name")** to load the built-in R dataset. After importing the dataset, you may use the **View()** function to see the content of the data. You may decide to rename the data after importing it into R as you proceed with your time series analysis.

Once the data is successfully imported in R, you can describe the time series data by answering some key questions about the data, such as Is there a correlation between the normalized temperature and normalized feeling temperature and the total count of bike rentals? What are the mean and median temperatures for different seasons (Winter, Fall, Summer, and Spring)? What are the mean temperature, humidity, wind speed, and total rentals per month? Is temperature associated with bike rentals (registered vs. casual)?

Also, you can create graphs to answer questions such as How do the temperatures change across the seasons?

Some useful base R functions are mean(), median(), cor(), boxplot(), etc.

Task 2 - Create interactive time series plots

In a bid to further understand the time series data, it will be very useful and informative to create (interactive) time series plots.

Hint
To complete this task, I strongly recommend using functions in the timetk package, such as plot_time_series().

This function allows for arguments to create interactive plots such as .interactive=TRUE, .plotly_slider=TRUE. For example, you can create a plot of the data and count of total rental bikes. In fact, you can group this plot by year (you can get the year from the data column using the **year()** function from the lubridate package).

Furthermore, the timetk package contains more functions to perform tasks like visualizing the time series by its seasonality (using plot_seasonal_diagnostics() function) and plotting anomalies in the time series data (using plot_anomaly_diagnostics() function).

There are hosts of different interesting explorations you can perform using the timetk package. Here are some useful resources to help you with this task:
About timetk package
Visualizing Time Series
Time Series Data Wrangling

Task 3 - Smooth time series data

After exploring the data, you can begin to make short-term forecasts using the time series data by smoothing the data.

Hint
Start this task by cleaning your time series data of any anomaly or outlier you may have found in task 2. You can start by putting the date and count of total bike variables as time series objects using the ts() function from the stats package. Then, use the tsclean() function from the forecast package to identify and replace outliers and missing values in a time series.

You can smooth the cleaned data to make forecasts using a Simple Exponential Smoothing and a

Simple Moving Average with order 10 using the HoltWinters() and SMA() functions, respectively.

Here are some useful resources to help you with this task:
Using R for Time Series Analysis
Time Series and Forecasting
Replace Outliers & Missing Values in a Time Series
Time Series Forecasting in R with Holt-Winters
Simple Moving Average (SMA)

Task 4 - Decompose and assess the stationarity of time series data

Ultimately, we usually want to fit more advanced ARIMA models to use for forecasting instead of short-term forecasts. Therefore, it is important to decompose the data and assess the stationarity of the data because if the time series data is not stationary, we cannot fit more advanced time series models.

Hint

You may use plots to answer questions such as Does the series count_value_ma appear to have trends or seasonality?

To decompose the data use the decompose() or stl() functions to examine and possibly remove components of the series. Then, create a time series data by removing the seasonal component.

For stationarity, ask yourself, Is the time series data stationary? If not, how to make stationary? Some functions to check stationarity are adf.test(), acf(), and pacf().

Suppose you find that the stationarity assumption is violated; you can make the time series stationary using *differencing*. Differencing is a process of subtracting each data point in the series from its successor. First, you will need to know how many differencing is needed. You can use the diff() function.

Here are some useful resources to help you with this task:
Using R for Time Series Analysis
Time Series and Forecasting
A Complete Tutorial on Time Series Modeling in R
Time Series Decomposition in R
Augmented Dickey-Fuller Test in R (With Example)

Task 5 - Fit and forecast time series data using ARIMA models

After decomposing and making sure that the time series is stationary, you can fit and forecast using ARIMA models.

Hint

You may fit a manual ARIMA model using the arima() function in R or an auto ARIMA model using the auto.arima() function in R.

Then you can check residuals, which should have no patterns and be normally distributed using Shapiro-Wilk normality test - shapiro.test(), for example. If not normally distributed, evaluate and iterate by:

- Checking if there are visible patterns or biases, plot ACF/PACF
- Fitting different manual ARIMA models. Compare model errors and fit criteria such as AIC or BIC
- Select the model order that gave the minimum AIC value

For forecasting:

- Forecast, for example, the next 25 observations using the manual and auto ARIMA models using the forecast() function. Make comments on the forecast using manual ARIMA and using auto ARIMA.

Here are some useful resources to help you with this task:

Using R for Time Series Analysis
Time Series and Forecasting
A Complete Tutorial on Time Series Modeling in R
Performing Time Series Analysis using ARIMA Model in R
Time Series Analysis Using ARIMA Model In R

Task 6 - Findings and Conclusions

Finally, we can wrap up the project. You can write a conclusion about your process and any key findings.

Hint

The main components that you will want to include:

- What did you learn throughout the process?
- Are the results what you expected?
- What are the key findings and takeaways?

Knit your R Markdown file into an HTML file that you can share.